

M Ű E G Y E T E M 1 7 8 2

Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Híradástechnikai Tanszék

Mérési útmutató a „WIFI 2: Vállalati megoldások elleni támadások” című méréshez



Hírközlő rendszerek biztonsága szakirány
Hírközlő rendszerek biztonsága laboratórium I.
(BMEVIHIM220)

A mérést kidolgozták:
BUTTYÁN LEVENTE, DÓRA LÁSZLÓ
meres@crysys.hit.bme.hu



CrySyS – Adatbiztonság Laboratórium

2009. december 2.

Tartalomjegyzék

1. Mérés célja	1
2. Elméleti összefoglaló	1
2.1. IEEE 802.11i	1
2.1.1. IEEE 802.1X	1
2.1.2. EAP	2
2.1.3. EAP-TTLS	3
2.1.4. PKI gyengeségéből adódó támadás	4
2.2. Captive portal	4
2.2.1. Ismertető	4
2.2.2. Gyengeségek	5
3. Mérési környezet	7
4. Feladatok	8
4.1. Megszemélyesítéssel támadás kivitelezése RADIUS-os hitelesítés mellett	8
4.2. Captive portal elleni támadás DNS tunnelezéssel	8
5. További információk	9
5.1. Hostapd — PC-ből AP	9
5.2. Freeradius WPE patch-csel	9
5.3. DNS tunneling	11
5.3.1. DNS tunnel szerver	11
5.3.2. DNS tunnel kliens	12
5.3.3. Szükséges alkalmazások telepítése	12
6. Beugró kérdések	12

1. Mérés célja

Folytatva az Wifi 1 mérést a központosított hitelesítési megoldásokat tekintjük át. Itt alapvetően kétféle megoldás létezik: a biztonságosabb IEEE 802.11i-ben definiált RADIUS-os megoldás, valamint a HotSpot megoldásként elhíresült captive portal-os hitelesítés. A mérési útmutatóban ezen megoldásokat technikai részletességgel ismertetjük, és bemutatjuk főbb gyengeségeit. Ezeket a mérés során a hallgató láthatja, miként lehet kihasználni. A RADIUS-os támadás kivitelezése során a hallgató azt is elsajátíthatja, miként lehet AP-t, illetve RADIUS szervert beüzemelni.

2. Elméleti összefoglaló

Azt feltételezve, hogy a hallgató a Wifi 1 mérés ismeretének birtokában van, ebben a fejezetben bemutatjuk a RADIUS-os, illetve a captive portal-os hitelesítés technikai hátterét.

2.1. IEEE 802.11i

2.1.1. IEEE 802.1X

A 802.11i-ben a hitelesítés és hozzáférés-védelem modelljét a 802.1X szabványból kölcsönözték [1]. Ezt a szabványt eredetileg vezetékes LAN-ok számára tervezték, de az elvek végülis vezeték nélküli WiFi hálózatokban is ugyanúgy alkalmazhatóak. A 802.1X modell három résztvevőt különböztet meg a hitelesítés folyamatában: a hitelesítendő felet (supplicant), a

hitelesítőt (authenticator), és a hitelesítő szerveret (authentication server). A hitelesítendő fél szeretne a hálózat szolgáltatásaihoz hozzáférni, és ennek érdekében szeretné magát hitelesíteni, azaz kilétét bizonyítani. A hitelesítő kontrollálja a hálózathoz történő hozzáférést. A modellben ez úgy történik, hogy a hitelesítő egy ún. port állapotát vezérli. Alapállapotban a porton adatforgalom nincs engedélyezve, ám sikeres hitelesítés esetén a hitelesítő „bekapcsolja” a portot, ezzel engedélyezve a hitelesítendő fél adatforgalmát a porton keresztül. A hitelesítő szerver az engedélyező szerepet játsza. Tulajdonképpen a hitelesítendő fél hitelesítését nem a hitelesítő, hanem a hitelesítő szerver végzi¹, és ha a hitelesítés sikeres volt, engedélyezi, hogy a hitelesítő bekapcsolja a portot.

WiFi hálózatok esetében a hitelesítendő fél a mobil eszköz, mely szeretne a hálózathoz csatlakozni, a hitelesítő pedig az AP, mely a hálózathoz történő hozzáférést kontrollálja. A hitelesítő szerver egy program, mely kisebb hálózatok esetében akár az AP-ben is futhat, nagyobb hálózatoknál azonban tipikusan egy külön erre a célra dedikált hoszton futó szerver alkalmazás. WiFi esetében a port nem egy fizikai csatlakozó, hanem egy logikai csatlakozási pont, amit az AP-ben futó szoftver valósít meg.

Vezetékes LAN esetében, a hitelesítendő fél egyszer hitelesíti magát, mikor fizikailag csatlakozik a hálózathoz. További védelmi lépésekre nincsen szükség (legalábbis hozzáférésvédelem tekintetében), hiszen a használatba vett portot más úgyszemint használhatja. Ahhoz ugyanis a hitelesítendő fél és a hitelesítő között létrejött fizikai kapcsolatot kellene megbontani (pl. a hálózati csatlakozót kihúzni egy Ethernet hub-ból). Ezt a hitelesítő hardvere detektálja és a portot azonnal letiltja. WiFi esetében más a helyzet, mert nincsen fizikai kapcsolat a STA és az AP között. Ezért a 802.11i azzal a követelménnyel egészíti ki a 802.1X modellt, hogy a hitelesítés során létre kell jöjjön egy titkos kulcs a STA és az AP között, melyet azok a további kommunikáció kriptográfiai védelmére használhatnak. Ezen kulcs hiányában egy támadó nem tudja a STA és az AP között kialakult logikai kapcsolatot csalárd módon saját céljainak megvalósítására felhasználni.

2.1.2. EAP

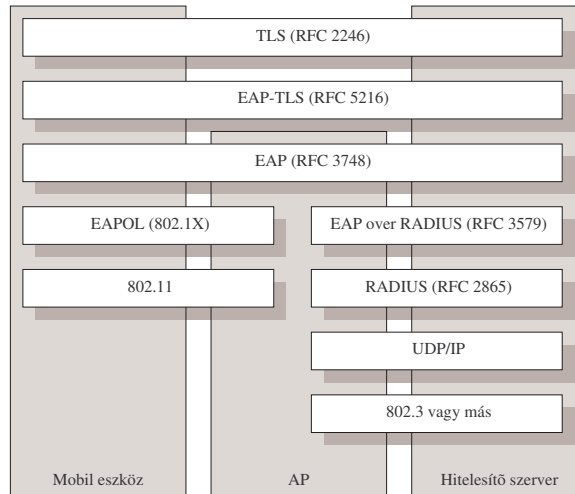
Maga a hitelesítés az EAP (Extensible Authentication Protocol) segítségével történik [2]. Az EAP egy igen egyszerű protokoll, aminek az az oka, hogy nem maga az EAP végzi a hitelesítést. Az EAP csak egy illesztő-protokoll, amit arra terveztek, hogy tetszőleges hitelesítő protokoll üzeneteit szállítani tudja (ezért „extensible”). Egy adott hitelesítő protokoll EAP-ba történő beágyazásának szabályait külön kell specifikálni. Több elterjedt hitelesítő protokollra létezik már ilyen specifikáció (pl. EAP-TLS, LEAP, PEAP, EAP-SIM).

Négy fajta EAP üzenet létezik: request, response, success, és failure. Az EAP request és response üzenetek szállítják a beágyazott hitelesítő protokoll üzeneteit. Az EAP success és failure speciális üzenetek, melyek segítségével a hitelesítés eredményét lehet jelezni a hitelesítendő fél felé.

Ahogy azt fentebb említettük, a 802.1X modellben, a hitelesítendő fél hitelesítését a hitelesítő szerver végzi. WiFi esetében ez azt jelenti, hogy az EAP protokollt (és az abba beágyazott tényleges hitelesítő protokollt, például a TLS-t) lényegében a mobil eszköz és a hitelesítő szerver futtatják. Az AP csak továbbítja az EAP üzeneteket a mobil eszköz és a hitelesítő szerver között, de nem érti azok tartalmát. Az AP csak az EAP success és failure üzeneteket érti meg, ezeket figyel, és ha success üzenetet lát akkor engedélyezi a mobil eszköz csatlakozását a hálózathoz.

Az EAP üzeneteket a mobil eszköz és az AP között a 802.1X-ben definiált EAPOL (EAP over LAN) protokoll szállítja. Az AP és a hitelesítő szerver között a WPA a RADIUS protokoll [3] használatát írja elő. A RADIUS-t az RSN opcióként ajánlja, de más alkalmas protokoll használatát is lehetővé teszi a specifikáció. Végülis tetszőleges protokoll használható, amely az

¹Ebből a szempontból az elnevezések kicsit szerencsétlenek.



1. ábra. Hitelesítési protokoll-architektúra a 802.11i-ben TLS használata esetén

EAP üzenetek szállítására alkalmas. Elterjedtsége miatt azonban várhatóan a legtöbb hálózat RADIUS-t használ majd. Az így kialakuló protokoll architektúrát a 1. ábra szemlélteti.

Ahogy azt korábban említettük, a hitelesítés eredményeként nemcsak a hálózathoz való hozzáférést engedélyezi a hitelesítő szerver, hanem egy titkos kulcs is létrejön, mely a mobil eszköz és az AP további kommunikációját hivatott védeni. A továbbiakban ezt a kulcsot PMK-ként fogják használni, ami a helyi hitelesítés esetén a mindenki számára szétszotott egyetlen kulcsot jelentette. Mivel a központi hitelesítés során — biztonságos hitelesítő eljárást feltételezve — csak elhanyagolható valószínűséggel generálják a résztvevők többször ugyanazt a kulcsot, ezért az azonos hálózathoz kapcsolódó mobil eszközök akkor sem tudják dekódolni a többiek üzeneteit, ha sikerül a négy-utas kézfogás minden üzenetét lehallgatni.

Mivel a hitelesítés a mobil eszköz és a szerver között folyik, ezért a protokoll futása után ezt a kulcsot csak a mobil eszköz és a hitelesítő szerver birtokolja, és azt még el kell juttatni az AP-hez. A RADIUS protokoll biztosít erre használható mechanizmust az MS-MPPE-Recv-Key RADIUS üzenet-attribútum formájában, mely kifejezetten kucsszállítás céljára lett specifikálva. A kulcs rejtjelezett formában kerül átvitelre, ahol a rejtjelezés egy a hitelesítő szerver és az AP között korábban létrehozott (tipikusan manuálisan telepített) kulcs segítségével történik.

2.1.3. EAP-TTLS

Amikor TLS-t (Transport Layer Security) —, azaz az SSL továbbfejlesztett verzióját — ágyaznak EAP üzenetbe, kétféleképp is létrejöhet a kölcsönös hitelesítés attól függően, hogy a mobil állomás rendelkezik-e tanúsítvánnyal. Amennyiben rendelkezik vele, EAP-TLS is [4] használható, egyébként EAP-TTLS (Tunneled TLS) [5].

Mindkét esetben a hitelesítő szerver és a mobil állomás létrehoz egy biztonságos kapcsolatot a TLS segítségével. Pazarló módon ezt a kapcsolatot EAP-TLS esetén nem használják semmire, hanem kihasználják, hogy egyrészt a TLS handshake biztosítja mindkét fél számára a PMK-hoz szükséges egyedi titkos kulcs generálását, másrészt a TLS handshake protokoll csak akkor fut le sikeresen, ha mindkét fél hitelesítette magát. Ilyen esetben a hitelesítő szerver a handshake során kéri a mobil állomást, hogy tanúsítvánnyal hitelesítse magát a TLS-ben opcionális `certificate request` típusú üzenettel.

EAP-TTLS esetén a mobil állomás hitelesítése némiképp elválik a hitelesítő szerver hitelesítésétől oly módon, hogy a hitelesítő szerver hitelesítése során létrejövő TLS kapcsolaton (tunnel-en) belül hitelesíti magát a mobil állomás. Ebben az esetben a TLS kapcsolaton

belül szintén EAP üzeneteket cserélnek a felek, aminek köszönhetően az egész rendszer rugalmassága megmarad. Így ugyanis, a mobil állomás hitelesítését az EAP-TTLS-től függetlenül lehet megválasztani. Pontosabban szólva nem teljesen függetlenül, hiszen ki lehet használni, hogy egy rejtjelezett csatornán hitelesített féllel kommunikál a mobil állomás. Tehát akár a PAP (Password Authenticaion Protocol) hitelesítő protokollt is lehet nyugodtan használni, amely protokoll a jelszót, mint hitelesítő kódot, nyíltan küldi át a hitelesítő szerver számára.

Ráadásul az EAP-TTLS belső protokolljának megválasztásakor nem szempont, hogy képes legyen a PMK-hoz szükséges kulcs generálásához inputot adni. Ugyanis a külső hitelesítő protokollban, a TLS kapcsolat létrehozásakor már elegendő entrópiájú kulcs generálódik, ami ráadásul mindkét féltől függ, hiába csak a hitelesítő szerver kerül hitelesítésre. Ugyanakkor ez a fajta függetlenség biztonsági problémákat is rejt magában. Mivel a mérésben nem kerül ez a fajta sérülékenység bemutatásra, ezért ennek részletezésétől el is tekintünk.

2.1.4. PKI gyengeségéből adódó támadás

A mérés során bemutatott sérülékenység alapvetően a PKI hiányosságaiból adódik. Pontosabban abból, hogy a hitelesített tanúsítványok használata nem terjedt el mind a mai napig. Emiatt sok 802.1X kliens nem kötelezi a felhasználót a root CA tanúsítványának megadására. Így természetesen bármilyen önálló tanúsítvánnyal lefut a TLS handshake, de valódi hitelesítés nem történik. Hiszen hiába küld át a hitelesítő szerver egy tanúsítványt, ha a mobil állomás, nem tudja ellenőrizni a tanúsítvány láncot egy olyan elemig, amelyiket ismert, nem tudja ellenőrizni, hogy a másik fél valóban az-e, akinek a tanúsítványában állítja magát.

Ha valóban megengedjük, hogy root CA tanúsítványának megadása nélkül is a mobil állomás folytassa a hitelesítést, akkor a belső hitelesítő protokollban olyan információt is megadhat, amivel a támadó a későbbiekben képes lesz megszemélyesíteni a mobil állomást. Ilyen eset, amikor a belső protokoll EAP-PAP, ahol is a mobil állomás a jelszavát nyíltan küldi el a másik fél számára, de több challenge-response típusú hitelesítő protokoll ellen létezik már hatékony megoldás a jelszó visszafejtésére.

A sikeres támadáshoz pedig nem kell mást tenni, mint felállítani egy ál AP-t és ahhoz egy ál hitelesítő szervert. Az ál AP-nak, mint hitelesítőnek meg kell adni az ál hitelesítő szerver elérését, így a támadó AP-ja nem a legális hitelesítő szerverhez fog fordulni, hanem a sajátjához. Ugyanis a hitelesítő szervert nem a kliens adja meg, hanem az AP-ban kell beállítani. Mivel a támadó minden eszközt maga állított be, így a hitelesítő és a hitelesítő szerver közötti megosztott titkot is maga adja meg.

Nyilván a támadó az eredeti AP SSID-jét és BSSID-jét is tudja hamisítani, és ezt akár egy olyan helyen is megteheti, ami helyben távol van az eredeti AP-tól vagy egyszerűen egy másik csatornát állít be, miközben az eredeti AP csatornáját jammaléssel használhatatlanná teszi. De a támadó azt is kihasználhatja, hogy azonos SSID-jű eszközök közül a mobil állomás hitelesítő alkalmazása a nagyobb jelerősségű AP-t választja, így a támadónak csupán közelebb kell kerülnie a mobil állomáshoz vagy nagyobb teljesítménnyel kell adnia.

2.2. Captive portal

2.2.1. Ismertető

Kávézókban, reptereken, szállodákban a legelterjedtebb hitelesítő megoldás a publikus Wi-fi hálózatokban, ún. HotSpotokban a captive portal.

Captive portalos rendszerek esetén a mobil állomás, a hitelesítő és a hitelesítő szerver mellett egy újabb hálózati eszközt érdemes megemlíteni, az átjárót. Az eddig vizsgált esetekben az átjáró maga az AP volt, itt viszont ez a szolgáltatás külön válik/válhat. Itt ugyanis az AP minden forgalmat átenged, és az átjáróra bízta, hogy az szűrje a jogosulatlan forgalmat. Az átjáró egyik legfontosabb eleme a tűzfal, mely megvalósítja a szűrést, valamint bizonyos

esetekben szükség lehet DNS szolgáltatás biztosítására is. A captive portal-t egy web szerver szolgáltatja, amely tipikusan ugyanabban az eszközben fut, mint az átjáró és a hitelesítő szerver. Természetesen lehetőség van minden szolgáltatás külön eszközön való futtatására is, de mindent biztosíthat egyedül az AP is kisebb hálózatokban.

A gyakorlatban a captive portal-os rendszer úgy működik, hogy amikor a mobil állomás először kér le egy honlapot, a lekért honlap helyett egy bejelentkező ablak jön be. Itt megadhatja felhasználó nevét, jelszavát, vagy vásárolhat jogot a használatra, esetleg csak el kell fogadnia a használati rendet. Miután a mobil állomás eleget tett az AP igényeinek, jogot kap minden port használatára (az adott hálózat policy-jének megfelelően). A megoldás fő előnye, hogy rendkívül intuitív, ugyanis automatikusan történik a felhasználó hitelesítése, ahogy egy honlapot lekér. Ráadásul mivel a mobil állomás csatlakozhat a hálózathoz, és eléri a szolgáltató által biztosított weblapokat, ezért könnyű segítséget nyújtani a kezdő felhasználók számára.

Technikailag a következő lépéseken át jut el a mobil állomás a hálózat jogos hozzáféréséhez. Először is, a mobil állomás kapcsolódik a nyílt hitelesítéssel bíró AP-hoz. Itt még nem történik valós hitelesítés. Ezután a mobil állomás DHCP-n keresztül kap IP címet. Ekkor még a forgalom nagy része korlátozva van. Amikor a mobil állomás egy honlapot kíván letölteni, akkor a kérést az átjáró átírányítja a captive portal felé. Azt, hogy átírányításra van szükség egy tűzfal veszi észre, amely detektálja, hogy az adott MAC és/vagy IP címmel rendelkező mobil állomás nem hitelesítette még magát. Az átírányítás történhet HTTP-n, IP-n vagy DNS-en keresztül:

- **HTTP:** A honlap lekérésekor ugyanúgy, mint normális esetben, a böngésző DNS kéréssel megszerezni az adott URL-hez tartozó IP címet. Amikor a konkrét HTTP kérést kiküld, az AP tűzfala 302-es HTTP kóddal átírányítja azt a saját captive portal-jára.
- **IP:** Az első kérést hasonló módon IP rétegben is át lehet irányítani, de ilyenkor a felhasználó számára úgy tűnik, mintha a lekért tartalom a captive portal lenne.
- **DNS:** Harmadik megoldás a DNS átírányítás. Ebben az esetben az AP saját DNS szerveret ad meg a mobil állomás számára, és ilyenkor csak ezt engedi használni. Ebben a megoldásban, már a DNS lekéréskor közbelép az AP, és a DNS kérésre a lekért honlap IP címe helyett a captive portal IP címét adja meg.

A captive portalt általában SSL-lal vagy TLS-sel védett csatornán keresztül éri el a mobil állomás. Itt ez a védelem annyit ér, mint a korábban bemutatott EAP-TTLS esetén, ráadásul, az emberek a böngésző világában talán még jobban hozzá vannak szokva ahhoz, hogy önáláért tanúsítványt elfogadjanak, ami a támadók számára könnyebbé teszi a támadást. Mivel ezt a problémakört már tárgyaltuk, és a captive portal-os megoldásnak további figyelemre méltó gyengeségei is vannak, ezt a sérülékenységet most nem vizsgáljuk. Amennyiben a mobil állomás helyesen megadta a hitelesítéséhez szükséges adatokat, a tűzfal beállításait megváltoztatják, és a továbbiakban minden az adott MAC vagy IP címhez tartozó forgalmat kienged.

2.2.2. Gyengeségek

A már említett captive portal SSL vagy TLS alapú hitelesítésének sérülékenysége mellett, továbbiakat is könnyen lehet találni. Ezek közül a legfontosabb, hogy a hozzáférést a MAC vagy IP cím alapján végzi el az átjáró. Amikor a mobil állomás csomagot küld az AP felé, a csomag tartalmazza mind a MAC, mind az IP címet. Mivel a csomag teljesen nyílt ezért ezeket könnyen ki tudja olvasni bárki. MAC és/vagy IP spoofinggal könnyen lehet hozzáférést szerezni a hálózathoz. Ebben az esetben azonban számítani kell arra, hogy a MAC és IP cím ütközés gondokat okozhat (pl. A másik mobil állomás a TCP kapcsolat felépítését reseteli).

Ez a megoldás legkönnyebben úgy használható, ha a másik eszköz távozásával a helyébe lép a támadó.

Ráadásul egy támadó nem csak az IP és MAC címet tudja lehallgatni, hanem minden csomagot értelmezni tud, amit felsőbb rétegbeli eljárások nem védenek, mivel adatkapcsolati rétegben nincs kriptográfiai védelem.

Egy másik érdekes támadási felület a DNS forgalom engedélyezéséből adódik. Ugyanis mind a HTTP, mind az IP alapú átirányítás esetén a DNS forgalmat ki kell engedni, ráadásul a DNS átirányítás esetén elég egy apró figyelmetlenség (lustaság, ismeret hiánya) a rendszergazda részéről, és kiengedi a DNS forgalmat is. A DNS forgalomba viszont akár IP csomagot is lehet beágyazni. Így ha a támadó rendelkezik egy DNS szerverrel, amelyikhez majd végül továbbítja a captive portalos rendszer a DNS lekérő csomagokat (pl. `dnstunnel.crysys.hu`), akkor ezen tud futtatni egy olyan alkalmazást, ami képes átalakítani a DNS-be ágyazott forgalmat IP forgalommá és vissza.

De először is tekintsük át a vezeték nélküli környezettől függetlenül, hogy hogyan működik a DNS lekérés. Egy kliens észreveszi, hogy egy névhez nem ismeri a hozzátartozó IP címet. Ilyenkor egy úgynevezett resolver-hez fordul, amely a névszerverekhez kapcsolódva feloldja a nevet. Bizonyos névszerverek egy vagy több zónáért felelnek. Egy ilyen zóna pl. a `*.crysys.hu`. Ha tehát valaki a `www.crysys.hu` IP címét akarja megtudni, akkor azt az előbbi zónáért felelő névszervertől lehet elsődlegesen megtudni. Azonban könnyen megeshet, hogy a resolver nem ismeri azt sem, hogy ki felel a fenti zónáért. Ezt az eggyel külsőbb zónáért felelő névszerver tudja megmondani, jelen esetben a `*.hu` zónáért felelős névszerver. És a mostani példánkban el is érkeztünk egy speciális esethez, a legkülső zónáért felelős névszervert ugyanis minden névszerver és resolver ismeri.

Anélkül, hogy mélyrehatóan vizsgálná a DNS rendszerét, meg kell említeni egy fontos részletet, mely a mérés szempontjából sem elhanyagolható. Hogy kevésbé legyenek a legfelsőbb szintű névszerverek terhelve, a névszerverek úgy tudják segíteni egymást, hogy a már lekért név és IP megfeleltetéseket maguk is egy időre megjegyzik, és kérés esetén maguk megválaszolják. A megfeleltetés érvényességi idejét az elsődleges névszerver határozza meg. Minden lekéréshez küld egy TTL (Time To Live) értéket is. Világossá válik az olvasó számára, hogy miért fontosak ezek a részletek, amikor a DNS tunnel-ezést bemutatjuk.

A DNS tunnelezés megértéséhez a következő kérdések megválaszolásán keresztül juthatunk el: 1) Hogyan épül fel a DNS tunnel, azaz hogyan tud a DNS tunnel kliens oldala kapcsolatot létesíteni a szerverrel? 2) Hogyan lehet ezt a kapcsolatot fenntartani? 3) Hogyan küld üzenetet a kliens és miként válaszol a szerver?

A DNS tunnel felépítéséhez, pontosabban az első üzenet elküldéséhez arra van szükség, hogy a DNS kérést tudjon a kliens küldeni a szerver felé. Ehhez egy olyan hoszt nevet kell feloldatni, amelyik egy olyan zónába tartozik, amelyikért éppen az a szerver felel, amelyiken a DNS tunnel szerver oldali implementációja fut. Folytatva az eddigi példát, tehát tegyük fel, hogy a `crysys.hu` alatt található szerver felel a `*.crysys.hu` zónáért. Itt meghatározzuk, hogy a `dnstunnel.crysys.hu` felel a `*.dnstunnel.crysys.hu` zónáért. Értelemszerűen a `dnstunnel.crysys.hu` alatt fut a DNS tunnel szerver oldala. Így, ha a kliens pl. a `xyz.dnstunnel.crysys.hu` cím feloldását kéri bármilyen resolver-től, az a `dnstunnel.crysys.hu`-hoz fog fordulni. Értelemszerűen a DNS tunnel szerver oldala tudni fogja, hogy ezt a kérést nem névszerverként kell feloldania, hanem a kérésben rejlő csomagot kell kifejtenie. Minden esetben a DNS tunnel szerver oldala az 53-as porton figyel. Ezért egy szerveren nem futhat névszerver és DNS tunnel szerver oldali implementációja egyszerre.

A kapcsolat fenntartásához az szükségeltetik, hogy minden újabb kéréssel a resolver a DNS tunnel szerverhez forduljon. Ezt a DNS tunnel szerver azzal érheti el, hogy a TTL-t 0-ra állítja, tehát gyakorlatilag nem kerül be a feloldandó név a cache-be. Ennek egy másik jótékony hatása, hogy a resolver memóriája nem telik meg értelmetlen DNS kérésekkel és az azokra adott válaszokkal.

Egy másik lehetőség az DNS tunnel szerver eléréséhez, hogy minden alkalommal újabb és újabb a DNS tunnel szerver zónájához tartozó kérésekkel bombázza a kliens a resolvert. Ez annak a mellékhatásaként is teljesül, hogy a kliens a szerver felé az üzeneteket a feloldandó névbe illeszti be. Tehát a feloldandó név formátuma a következőképp alakul (követve az eddigi példákat): `üzenet.dnstunnel.crysys.hu`.

A DNS protokoll [6] megengedi, hogy szöveg elemet is beillesszen a resolver a válaszbába (TXT). A szerver a kliens felé küldött üzenet esetén ezt használja ki, és így illeszti a DNS válaszbába. Amennyiben a küldendő üzenet nem fér egészben a DNS válaszbába, a kliens üzenet nélküli lekérést küld a szerver felé, és így a szerver az üres kérésre küldött válaszában tudja folytatni az üzenetet.

Hivatkozások

- [1] IEEE Std 802.1X-2001. IEEE Standard for Local and Metropolitan Area Networks - Port-Based Network Access Control, June 2001.
- [2] B. Aboba, L. Blunk, J. Vollbrecht, J. Carlson, and H. Levkowetz. Extensible Authentication Protocol (EAP). RFC 3748 (Proposed Standard), June 2004. Updated by RFC 5247.
- [3] C. Rigney, S. Willens, A. Rubens, and W. Simpson. Remote Authentication Dial In User Service (RADIUS). RFC 2865 (Draft Standard), June 2000. Updated by RFCs 2868, 3575, 5080.
- [4] D. Simon, B. Aboba, and R. Hurst. The EAP-TLS Authentication Protocol. RFC 5216 (Proposed Standard), March 2008.
- [5] P. Funk and S. Blake-Wilson. Extensible Authentication Protocol Tunneled Transport Layer Security Authenticated Protocol Version 0 (EAP-TTLSv0). RFC 5281 (Informational), August 2008.
- [6] P.V. Mockapetris. Domain names - implementation and specification. RFC 1035 (Standard), November 1987. Updated by RFCs 1101, 1183, 1348, 1876, 1982, 1995, 1996, 2065, 2136, 2181, 2137, 2308, 2535, 2845, 3425, 3658, 4033, 4034, 4035, 4343.
- [7] CoovaAP. <http://coova.org/CoovaAP>. CoovaAP is an OpenWRT-based firmware designed especially for HotSpots. It comes with the CoovaChilli access controller built-in and makes it easily configurable.
- [8] Jouni Malinen. WPA/RSN Supplicant (wpa_supplicant) and WPA/RSN/EAP Authenticator (hostapd). <http://hostap.epitest.fi/hostapd>.
- [9] FreeRADIUS: The world's most popular RADIUS Server. <http://www.freeradius.org>.

3. Mérési környezet

Hasonlóan a Wifi 1 méréshez, a mérést a Backtrack 3 Live CD segítségével lehet elvégezni.

A mérés során két AP áll rendelkezésre: egy a RADIUS-os hitelesítés számára, egy a captive portal-os hitelesítés számára. Mindkét AP firmware-e OpenWRT alapú, és mindkét esetben lokálisan fut minden szükséges alkalmazás.

A RADIUS hitelesítő szervert az AP-n futó FreeRADIUS v1.1.8 biztosítja. Ez támogatja az EAP-TLS, EAP-TTLS, PEAP, EAP-PAP, EAP-CHAP és EAP-MSCHAPv2 típusú EAP hitelesítő módokat.

A kliens egy PC, amely `wpa_supplicant`-ot futtat. Ez bizonyos időközönként kapcsolódik a `wifi.ttls` SSID-jű AP-hoz, ahol hitelesíti magát EAP-TTLS/EAP-PAP módszerrel.

A captive portal-os AP a Coova Chilli [7] alkalmazást futtatja. Ez egy rendkívül rugalmas alkalmazás, amely webes felületen keresztül engedi a HotSpot széleskörű konfigurálását. A mérés során az AP felhasználónév-jelszó párokkal hitelesíti a mobil állomást, és lokálisan egy fájlból kiolvastva a jelszavakat dönti el a hozzáférés jogosultságát. A hitelesítést HTTP alapú átirányítással kényszeríti ki a felhasználóból.

A captive portal-os AP-hez a mérés során nem csatlakozik kliens, mivel a DNS tunnel támadás kivitelezhető anélkül is.

A DNS tunnel kliens alkalmazása a mérő gépen, a szerver egy különálló gépen fog futni úgy, hogy minden mérőcsoport ugyanazt a PC használja. Virtuális interfészek és minden interfész számára egyedi IP cím biztosítja, hogy a mérőcsoportok ne akadályozzák egymás munkáját. Az `X`. mérőcsoporthoz az `10.105.1.23X` IP címmel rendelkező virtuális interfész tartozik, és ehhez a `dnstunnelX` felhasználónévvel és `mereslabor` jelszó párral tud csatlakozni.

A DNS tunnel neve nagyon hasonlít az elméleti összefoglalóban példaként említetthez, azonban további egy szintet be kellett iktatni a több mérőcsoport kezelése miatt. Így az `X`. mérőcsoport DNS tunnel szervere a `*.merohelyX.dnstunnel.crysys.hu` zónáért felel. A DNS beállítások a `crysys.hu`-n a mérésvezető engedélyével megtekinthetők.

Hasonlóan a Wifi 1 méréshez, a mérési jegyzőkönyvet a mérés során kell kitölteni. A kitöltéshez itt is a Google docs webes szövegszerkesztőt ajánljuk. Belépni a `wifimérésX@crysys.hu` (`X` a mérőcsoport száma) felhasználónévvel és a mérés elején megadott jelszóval lehet.

4. Feladatok

4.1. Megszemélyesítéses támadás kivitelezése RADIUS-os hitelesítés mellett

- Indítson el a mérőgépen egy RADIUS szervert WPE patch-csel!
- Fordítson `hostapd`-t madwifi driver támogatással!
- Konfigurálja be a `hostapd`-t úgy, hogy az a mérőgépen futó RADIUS szerverhez forduljon a hitelesítés elvégzéséért!
- Az `hostapd` elindítása mellett hallgassa le a mérőgépet érintő RADIUS és EAPOL üzeneteket!
- Szerezze meg az automata kliens jelszavát!
- Csatlakozzon az eredeti AP-hez!

4.2. Captive portal elleni támadás DNS tunnelezéssel

- Csatlakozzon a captive portallal védett AP-hoz, és próbáljon elérni külső szervert!
- Vizsgálja meg, hogy az AP védekezik-e a DNS tunnelezés ellen!
- Ellenőrizze a `crysys.hu` zónabeállításait a mérésvezető segítségével!
- Állítsa be és indítsa el a DNS tunnel szerver oldalát!
- A kliens oldal felparaméterezésével és SSH kapcsolat felépítésével tesztelje a DNS tunnelt a kliens!
- A kliens oldalon indítsa a DNS tunnelt úgy, hogy Socks proxy-ként lehessen használni az SSH kapcsolatot!

- A webböngészőt állítsa be úgy, hogy minden forgalmat a DNS tunnelen keresztül bonyolítson le! Hallgasson le egy ilyen forgalmat is!

5. További információk

A Wifi 1 mérésből már megismert `wlanconfig` parancs mellett egy `hostapd` és egy `FreeRADIUS` nevű alkalmazásra lesz szükség a mérés első felében, valamint a `DNStunnel` nevű alkalmazásra a második felében. Alább ismertetjük ezen alkalmazások telepítését konfigurálását, használatát.

5.1. Hostapd — PC-ből AP

A `hostapd` [8] egy implementáció, amely megvalósítja az AP-hoz tartozó funkciókat. Ennek segítségével egy PC-be illesztett hálózati kártyából könnyen lehet AP-t készíteni. Mindameltett hitelesítő szerver funkcióját is képes ellátni, de a mostani mérés során ezt a tulajdonságát nem részletezzük, mivel a hitelesítő szerver funkcióját a `Freeradius` fogja biztosítani később ismertetett okok miatt. Ugyanakkor, ha a hallgató kizárólag `hostapd`-t használva kívánja végrehajtani a mérési feladatot, mi nem tántorítjuk el ettől, de ebben a mérési útmutatóban csak a `freeradius`os megoldást részletezzük.

Ugyan a mérés során használt `Backtrack 3 Live CD`-n van előre telepített `hostapd`, mégis fordítani kell, mivel az nem tartalmaz `madwifi` drivert. A forrás letölthető a következő címről: <http://www.crysys.hu/netsec/wifimeres/hostapd-0.7.0.tar.gz>

A fordítás előtt be kell állítani, hogy a `madwifi` drivert is fordítson hozzá. Ehhez a `hostapd` könyvtáron belül a `defconfig` fájlt kell átnevezni `.config` nevére, és itt lehet megadni, hogy mely modulok kerüljenek fordításra. Ugyanitt a `madwifi` driver forráskódjának elérhetőségét is meg kell adni. A `madwifi` driver forráskódja az alábbi címen érhető el: <http://www.crysys.hu/netsec/wifimeres/madwifi-0.9.4.tar.gz>.

A fordítás befejezéséhez egyszerűen a `make` parancsot kell kiadni, ami jó konfigurációs fájl esetén sikeresen le is kell, hogy fusson.

A `hostapd`-t a `hostapd.conf` konfigurációs fájlban keresztül lehet felparaméterezni. A központi hitelesítéshez állítandó vagy ellenőrizendő paramétereket a 1. táblázatban soroljuk fel.

A `hostapd`-t a `./hostapd hostapd.conf` paranccsal lehet futtatni. Említést érdemel a `-d` kapcsoló, ami a `debug` üzenetek megjeleníti, valamint a `-B`, aminek köszönhetően az alkalmazás háttérben fog futni.

5.2. Freeradius WPE patch-csel

A `FreeRADIUS` [9] egy nyílt forráskódú `RADIUS` szerver implementáció. Amellett, hogy ingyenes több hitelesítő módszert és adatbázis típust támogat, mint a legtöbb kereskedelmi termék. A mérés során vizsgált `EAP-TTLS/EAP-PAP`-ot is képes kezelni. Szolgáltatásai között lehet felsorolni a hozzáférés jogosultság vizsgálatot, hitelesítést, számlázást és szolgáltatás kihelyezést. A jogosultság vizsgálatot képes saját formátumú fájlból, de adatbázisból is végrehajtani. Szolgáltatás kihelyezés alatt azt értjük, hogy képes arra, hogy egy másik `RADIUS` szerverhez forduljon, mint egy kliens, hogy attól szolgáltatást kérjen. A fent felsorolt szolgáltatások közül a mérés során nem foglalkozunk sem a szolgáltatás kihelyezéssel, sem a számlázással.

A `FreeRADIUS` konfigurálása viszonylag bonyolult dolog, mivel több komponensből áll. Minden konfigurációs fájl a `/usr/local/etc/raddb` elérési útvonalon található a `Backtrack 3 Live CD`-n. Amennyiben saját formátumú fájlból kell kiolvasni a mobil állomások hitelesítéséhez információt, ezt a `users` nevű fájlból teszi meg. A hitelesítők (Wi-fi esetén AP-k) listája, IP címe, megosztott titkok a `clients.conf` fájlban kerülnek felsorolásra.

1. táblázat. Hostapd konfigurálásához szükséges paraméterek listája a különálló RADIUS szerver megadásához

Név	Magyarázat	Lehetséges értékek
<code>interface</code>	A kezelni kívánt interfész beállítása	pl. ath0
<code>driver</code>	Az interfészt kezelni tudó driver megnevezése	pl. madwifi
<code>ssid</code>	A beacon üzenetben megjelenő SSID-t lehet itt állítani	-
<code>wpa</code>	Ezzel lehet beállítani, hogy milyen típusú WPA-t használjon a hitelesítésnél	1-WPA, 2-WPA2
<code>wpa_key_mgmt</code>	Azt lehet beállítani, hogy helyi vagy központi hitelesítés történjen	WPA-PSK, WPA-EAP
<code>ieee8021x</code>	Ezzel lehet beállítani, hogy IEEE 802.1X típusú hitelesítést kell-e végezni	0-nem, 1-igen
<code>eapol_version</code>	Az EAPOL verziószámát lehet állítani. Erre azért van szükség, mert néhány mobil állomás eldobja, a 2-es verzióhoz tartozó EAPOL üzeneteket	1, 2
<code>auth_server_addr</code>	RADIUS szerver IP címe	-
<code>auth_server_port</code>	RADIUS szerver portszáma	általában 1812
<code>auth_server_shared_secret</code>	RADIUS szerver és az AP között megosztott titok	-

Általános konfigurációs paramétereket a `radiusd.conf` fájlban adhatunk meg. A jogosultság vizsgálatához és a hitelesítéshez szükséges modulok betöltését a `sites-enabled/default` fájlban lehet meghatározni. Az EAP-hoz kapcsolódó paraméterek az `eap.conf` fájlban kerülnek definiálásra. Itt lehet megadni pl. az EAP-TTLS-hez szükséges publikus-privát kulcsot, tanúsítványt.

A FreeRADIUS 2.0.2-es verziójához készült egy patch, amelyben a szerzők a RADIUS-ban megvalósítható megszemélyesítéses támadást kívánták bemutatni. Azóta a FreeRADIUS újabb verziókkal számos ponton továbbfejlesztésre került, azonban az PKI-ból adódó problémákat azóta sem sikerült elhárítani. Tehát a patch lényegét tekintve ugyanúgy működne a legfrissebb verzióban is, azonban az egyszerűség kedvéért a 2.0.2 verziószámú FreeRADIUS-t használjuk a mérés során (Elérhetőség: <http://www.crysys.hu/netsec/wifimeres/freeradius-server-2.0.2.tar.gz>). A patch neve Wireless Pwnage Edition (WPE). Elérhetőség: <http://www.crysys.hu/netsec/wifimeres/freeradius-wpe-2.0.2.patch>.

Ebben a patchben a szerzők úgy módosították a FreeRADIUS-t, hogy minden EAP hitelesítésnél, ahol a kliens küld hitelesítő információt az adatbázissal való egyeztetés helyett mindent elfogadjon és azt kiírja egy log fájlba.

A patch-elést és a fordítást a következő parancsokkal lehet végrehajtani:

```
$ cd freeradius-server-2.0.2/
$ patch -p1 < ../freeradius-wpe-2.0.2.patch
$ ./configure && make && sudo make install && sudo ldconfig
```

Azonban a Backtrack 3 tartalmazza ezt a módosított FreeRADIUS-t, ezért ott csak a futtatással kell foglalkozni.

Amennyiben nincsenek érvényes tanúsítványok, újakat a következő parancsokkal lehet generálni:

```
$ cd freeradius-server-2.0.2/raddb/certs
$ ./bootstrap
$ sudo cp -r * /usr/local/etc/raddb/certs
```

A FreeRADIUS-t a `radiusd` paranccsal lehet futtatni. Amennyiben debugolni kell, a `-X` kapcsolót érdemes használni.

Az elfogott hitelesítő adatokat a következő paranccsal követhetjük figyelemmel:

```
$ tail -f /usr/local/var/log/radius/freeradius-server-wpe.log
```

5.3. DNS tunneling

A mérés során Dan Kaminsky által fejlesztett OzymanDNS DNS tunnel szerver és kliens oldali implementációjának egy tovább fejlesztett változatának használatát javasoljuk. A további fejlesztések első sorban programozási hibák, hanyagságok kijavítására irányultak. A kód tisztán perl alapú, ezért lényegében fordításra, telepítésre nincs szükség, csupán konfigurálásra.

Az SSH port forwarding mechanizmusával maga is támogatja a tunnelezést. Ezt a fejlesztők kihasználva úgy valósították meg a DNS tunnelt, hogy a DNS üzenetekbe csupán SSH-t ágyaztak be, az általános tunnel megvalósítását már rábízták a régóta fejlesztett és sokat tesztelt SSH-ra.

Az implementáció mérésre szabott változata letölthető a következő helyről: <http://www.crysys.hu/netsec/wifimeres/dnstunnel-wifimeres.tar.gz>. Változásokat azért kellett eszközölni, hogy több ilyen alkalmazás is tudjon futni egy számítógépen, hogy a mérőcsoportok ne akadályozzák egymást.

5.3.1. DNS tunnel szerver

A szerver oldalon legegyszerűbben a `dnstunnel.wrapper` script editálásval tudjuk beállítani a szerveret. Itt lehetőség van az IP cím megadására, ahol hallgatónia kell a szervernek, a valós DNS lekérések esetén a válasz megadására, valamint definiálni lehet, hogy milyen felhasználó és csoport joggal fusson a DNS tunnel. Az alábbi példa jól mutatja a beállítás módját:

```
DNSHOST="example.dnstunnel.crysys.hu"
REPLYIP="127.0.0.1"
OPTIONS="-l 10.105.1.230 -u dnstunnel0"
```

A DNS tunnel szerver futását a következő paranccsal lehet elindítani:

```
$ ./dnstunnel.init start
```

Fontos, hogy futtatáskor root jogot adjunk az alkalmazásnak, különben nem tud az 53-as porton hallgatónia.

Hibákat a `dnstunnel.log` fájlban sorolja föl. Ezt érdemes a következő paranccsal felügyelni:

```
$ tail -f dnstunnel.log
```

5.3.2. DNS tunnel kliens

Kliens oldalon nincs más teendő, mint az SSH proxy megfelelően paraméterezett futtatása. Az SSH ugyanis elég flexibilis ahhoz, hogy a proxy megvalósítását más alkalmazásra bízza rá.

A kapcsolat felépíthetőségének tesztelését a legegyszerűbben a következő SSH kapcsolat felépítésével lehet demonstrálni:

```
$ ssh -C -o ProxyCommand="./dnstunnelc -v sshdns.example.dnstunnel.crysys.hu"
  user@localhost
```

Nagyon fontos itt az `sshdns` előtét, mivel a DNS szerver csak az ilyen címekre válaszol DNS tunnelként. Egyéb esetben DNS kérésnek tekinti és úgy válaszol, ahogy a `REPLYIP` változóban megadtunk.

SOCKS proxyt a következő paranccsal lehet nyitni:

```
$ ssh -D 8000 -N -C -o ProxyCommand="./dnstunnelc sshdns.example.dnstunnel.crysys.hu"
  user@localhost
```

5.3.3. Szükséges alkalmazások telepítése

A DNS tunnel szerver és kliens oldalon is szükség van a következő csomagok meglétére: `libnet-dns-perl` és `libmime-base32-perl`. Ezeket a szerver oldalon biztosítjuk, de kliens oldalon a Live CD miatt ez csak nagyon bonyolult módon kivitelezhető, ezért ezt a hallgatóknak kell telepíteni.

A két csomag forráskódja elérhető az alábbi címről:

http://www.crysys.hu/netsec/wifimeres/libnet-dns-perl_0.65.tar.gz

http://www.crysys.hu/netsec/wifimeres/libmime-base32-perl_1.01.tar.gz

A két csomagot telepíteni Backtrack esetén a kicsomagolást követően a következő parancsokkal lehet:

```
$ perl Makefile.pl
$ make
$ make test
$ make install
```

6. Beugró kérdések

A beugró két kérdésből áll, egy nagy és egy kis kérdésből. A két kérdés minden esetben különböző tématerületek ismeretanyagára kérdez rá. A kis kérdésre általában egy rövid mondatos választ várunk, de a nagy kérdés esetén sem kell több 5-6 mondatnál.

Kis kérdések:

- Mire használják az AP és a hitelesítő szerver között megosztott titkot IEEE 802.11i esetén?
- Mennyiben különbözik az EAP-TLS az EAP-TTLS-től?
- Milyen feltételnek kell teljesülnie a mobil állomás elleni sikeres támadással szemben EAP-TTLS alapú hitelesítés esetén?
- Miért lehet használni az EAP-PAP-ot EAP-TTLS-en belül?
- Mivel védik a captive portal esetén a mobil állomás által küldött hitelesítő adatokat?
- Miért és milyen esetben kell DNS forgalmat kiengedni a captive portal-os hitelesítés előtt?

- Hogyan lehet átirányítani a hitelesítetlen mobil állomás forgalmát a captive portal felé?
- Hogyan dönti el a tűzfal, hogy egy mobil állomás jogosult-e a hálózat használatára captive portal-os hitelesítés esetén?

Nagy kérdések:

- Írja le a hitelesítés menetét EAP-TTLS/EAP-PAP esetén!
- Írja le a támadás menetét EAP-TTLS alapú hitelesítés esetén!
- Írja le a hitelesítés menetét captive portal esetén!
- Sorolja föl a captive portal gyengeségeit!