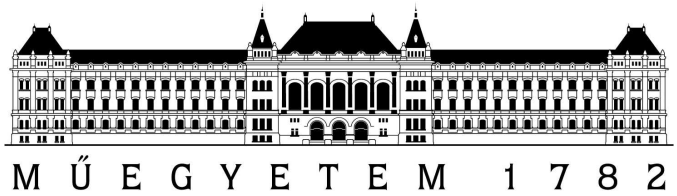


Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Híradástechnikai Tanszék

Biztonságos útvonalválasztás ad hoc hálózatokban

Ács Gergely

Konzulens:

Dr. Buttyán Levente

Híradástechnikai Tanszék

Diplomamunka

2005.

Hallgatói nyilatkozat

Alulírott *Ács Gergely*, szigorló hallgató kijelentem, hogy ezt a diplomatervet meg nem engedett segítség nélkül, saját magam készítettem, csak a megadott forrásokat (szakirodalom, eszközök, stb.) használtam fel. Minden olyan részt, amelyet szó szerint, vagy azonos értelemben de átfogalmazva más forrásból átvettem, egyértelműen, a forrás megadásával megjelöltem. Tudomásul veszem, hogy az elkészült diplomatervben található eredményeket a Budapesti Műszaki és Gazdaságtudományi Egyetem, a feladatot kiíró egyetemi intézmény saját céljaira felhasználhatja.

Budapest, 2005. május 18.

.....

Ács Gergely

Abstract

In this thesis I present a new formal framework that can be used for analyzing the security of on-demand source routing and distance vector routing protocols proposed for wireless mobile ad hoc networks. My approach is based on the simulation paradigm which is a well-known and general procedure to prove the security of cryptographic protocols. I give the formal definition of secure ad hoc routing in a precise and rigorous manner using the concept of statistical indistinguishability. I illustrate the usage of the model on real examples. Several "secure" ad hoc routing protocols have been proposed so far, but their security have been mainly analyzed by informal means only. I show that the informal reasoning is not sufficient to guarantee security in ad hoc routing. I describe as yet unknown, subtle attacks against Ariadne and SAODV, and I prove that ARAN is a secure ad hoc routing protocol in my model. I also introduce new ad hoc routing protocols, called endairA and ASAR, and I show that they are also provable secure in this model.

Kivonat

Ebben a munkában bemutatok egy olyan formális módszert, amivel biztonsági szempontból lehet elemezni az igény szerinti forrás alapú útvonalválasztó (on-demand source routing) valamint az igény szerinti útvonalválasztó tábla alapú (on-demand distance vector routing) protokollokat vezetéknélküli ad hoc hálózatokban. A módszer alapját a szimulációs paradigma adja, melyet kriptográfiai protokollok biztonságának a bizonyítására javasoltak. A dolgozatban részletesen ismertetem a szimulációs paradigma adaptációját ad hoc útvonalválasztó eljárásokra. Formálisan megfogalmazom, hogy mit értek biztonságos útvonalválasztás alatt, melyhez felhasználom a számításelméleti megkülönböztethetetlenség fogalmát. A módszer lényegét valós példákon szemléltetem, elemzem az Ariadne, SAODV és ARAN útvonalválasztó protokollokat. A dolgozat során megmutatom, hogy rendkívül szövevényes támadások konstruálhatóak olyan protokollokkal szemben, melyek biztonsága csak informális érveléssel igazolt. Rámutatok eddig még nem ismert támadásokra az Ariadne és SAODV protokollok ellen, majd formálisan igazolom az ARAN biztonságosságát a bemutatott modellben. Végül ismertetek két új protokollt, amelyek szintén bizonyíthatóan biztonságosak a definiált modellben.

Tartalomjegyzék

1. fejezet

Bevezetés

"A proof is whatever convinces me."

Shimon Even (1935-2004)

Az útvonalválasztó protokollok biztonsága kritikus fontossággal bír bármilyen számítási képességgel rendelkező entitások (csomópontok) alkotta hálózatban. Ha egy támadónak sikerül valamilyen módon az útvonalválasztást meghiúsítania, akkor már képes lehet akár az egész hálózat működését megbénítani. Ennek vezetéknélküli ad hoc[1] hálózatokban különös jelentősége van azok speciális struktúrája és működése miatt.

Az utóbbi években több „biztonságos" útvonalválasztó protokollt is javasoltak vezetéknélküli ad hoc hálózatokra. Ugyanakkor ezen protokollok biztonságát vagy csak informálisan, vagy pedig olyan formális módszerekkel bizonyították, amelyek erre a célra nem bizonyultak alkalmasnak. Ennek két fontos következménye van. Az egyik, hogy nincs egy precízen, jól definiált formális szabálya a „biztonságos útvonalválasztásnak". Így a különböző szerzők különbözőképpen értelmezik a biztonság fogalmát, és más követelményeket is támasztanak egy protokollal szemben. Ez megnehezíti a különböző protokollok összehasonlítását is. A másik fontos következmény, hogy nincs matematikai szempontból jól definiált precíz módszer a javasolt protokollok biztonságának a bizonyítására. A dolgozat során látható lesz, hogy az informális érvelés egy protokoll biztonsága mellett nem elégséges a legtöbb esetben, hiszen minden később felfedezett hiba a protokoll iteratív „foltozását" vonja maga után. Ezzel szemben egy formális, precíz matematikai érveléssel már garantálni lehet egy protokoll biztonságát az adott modellben.

A dolgozatban bemutatok egy új formális modellt, amelyben precízen megfogalmazható, hogy mit értünk biztonságos útvonalválasztás alatt egy adott támadómodell esetén. A támadómodell egy általános esetet tételez fel, amikor egy támadó több csomópontot kontrollál és több csomópontazonosítót is használhat (Aktív-y-x modell). Ebben a modellben lehetőség nyílik egy protokoll biztonságosságának (vagy éppen nem biztonságosságának) a bizonyítására ezen általános támadómodellt feltételezve. A biztonságosság bizonyítása a már létező kriptográfiai primitívek (digitális aláírás, hitelesítő kód, stb.) biztonságára építenek, míg a

[1] az angol *ad hoc* szó jelentése: ideiglenes, alkalmi. Jelen esetben az ad hoc hálózatok alatt ideiglenes jelleggel létrejött önszerveződő hálózatot kell érteni. A dolgozat hátralevő részében én az ad hoc jelzőt használom.

biztonságosság cáfolata általában valamilyen szövevényes támadás konstruálását jelenti a kérdéses protokoll ellen. A dolgozat a forrás alapú (*source routing*) és az útvonalválasztó tábla, vagy más nevén távolsági vektor (*distance vector routing*) alapú ad hoc útvonalválasztó protokollokat, azon belül is magának az útvonalfelderítési szakasznak a biztonságát tárgyalja.

A dolgozat felépítése a következő: A 2. fejezetben bevezetem az ad hoc hálózatok és az ad hoc útvonalválasztás fogalmát. Röviden bemutatok egy forrás és egy tábla alapú útvonalválasztó protokollt, mivel a később tárgyalandó protokollok is ezeket a működési elveket követik. Ezután kategorizálom az ad hoc hálózatok elleni lehetséges ismert támadásokat, bemutatok egyszerűbb támadási konstrukciókat a kriptográfiát nem használó protokollokkal szemben, bevezetem a különböző támadómodellek fogalmát, majd intuitív módon megfogalmazom a számomra minimálisan elvárt követelményeket a biztonságos útvonalválasztó protokollokkal szemben. A 3. fejezetben leírom annak a formális modellnek az alapját, amelyre alapozva fogom felépíteni az útvonalválasztó protokolloknak az általam definiált biztonsági modelljeit a 4. és a 7. fejezetekben forrás illetve tábla alapú protokollokra. Mindegyik modell gyakorlati jelentőségét valóságos példákon szeretném szemléltetni. A 5. fejezetben az Ariadne protokoll gyengeségeit mutatom be, majd a 6. fejezetben leírom egy általunk tervezett forrás alapú útvonalválasztó protokoll, az endairA működését, amelyről bebizonyítom, hogy bizonyíthatóan biztonságos az általam definiált modellben. A tábla alapú protokollok modelljének gyakorlati hasznát is valóságos példákon szeretném megmutatni: a 8. fejezetben rámutatok arra, hogy az SAODV egy nem biztonságos protokoll a modellemben, míg a 9. fejezetben formálisan igazolom ennek ellenkezőjét az ARAN protokollról. Végezetül a 10. fejezetben az ASAR mint hibrid útvonalválasztó protokoll kerül ismertetésre, amelynek biztonságosságát szintén formálisan igazolom. A 11. fejezetben bemutatom az eddigi munkák és a saját munkám közötti különbségeket, végül a 12. fejezetben összefoglalom és értékelem a munkám eredményeit, és ismertetek jövőbeli lehetséges kutatási irányvonalakat.

2. fejezet

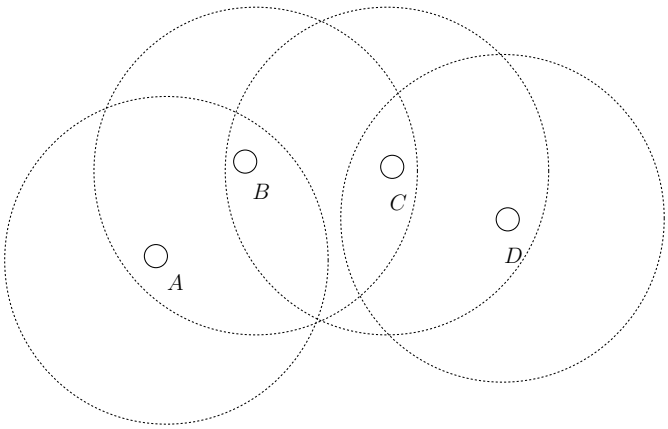
Ad hoc hálózatok

Ad hoc hálózatokban az egyes mobilis csomópontok képesek rögzített infrastruktúra és centralizált adminisztráció nélkül egymással kommunikálni, tehát nincs szükség bázisállomásokra vagy hozzáférési pontokra. Ezen tulajdonságok miatt alkalmazásuk rendkívül széleskörű és sokrétű. Önszerveződő jellegükből adódóan elsősorban olyan helyen alkalmazzák őket, ahol nem cél vagy nem lehetséges infrastruktúrával rendelkező hálózat kiépítése; például katonaság esetében harcmezőn a mobilitás miatt, katasztrófa súlytotta területeken, illetve egyéb nehezen megközelíthető helyeken, ahol egyébként túl költséges lenne egy központosított hálózat kiépítése.

Az ilyen hálózatot tipikusan egyenrangú csomópontok alkotják, amelyek vezetéknélküli adatkapcsolatokon tartják egymással a kapcsolatot egy központi vezérlőelem irányítása nélkül. Léteznek speciális hibrid ad hoc hálózatok, mint például a szenzorhálózatok, amelyek monitorozás céljából egy monitorozó központhoz kapcsolódnak. Az egyes eszközök korlátozott számítási kapacitással és energia-ellátással rendelkeznek, vagyis a kommunikációs képességük is behatárolt.

A távoli csomópontok kommunikációja *multihop* módon történik, hiszen egy single-hop (közvetlen) kapcsolat az egyes eszközök korlátozott energia-ellátása valamint a mobilitás és esetleges interferencia miatt nem lehetséges. A multihop kommunikáció során minden csomópont csak olyan csomópontokkal tartja a közvetlen kapcsolatot a hálózaton belül, amelyek adását képes értelmezni (pl. rádióadását venni). Ezek számára a szomszédos csomópontok. Minden más – számára nem szomszédos – csomópontot csak közvetetten a saját szomszédai által képes elérni úgy, hogy azok továbbítják a nem szomszédos célcsomópont felé elküldött üzeneteket valamilyen mechanizmus alapján (amit az útvonalválasztó protokoll specifikál). A 2.1. ábrán látható egy példa multihop kommunikációra. Látható, hogy multihop esetben minden csomópont egyben útvonalirányító (router) funkciót is ellát. Ha szükséges, akkor a kapott csomagot továbbítják a cél felé, másrészről útvonalkarbantartást is végeznek, és útvonalinformációkkal látják el a többi csomópontot.

A 2.1. részben bemutatom az ad hoc útvonalválasztás feladatát, majd a 2.2. és a 2.3. részekben bemutatok két reprezentatív példát mobil ad hoc útvonalválasztó protokollokra, áttekintem a DSR és az AODV működését. A 2.4. részben leírom a dolgozat során használt támadómodellt, majd egyszerű támadásokkal szemléltetem a védelmi mechanizmusok nélkül használt protokollok gyengeségét. Végül a 2.5. részben szemléletesen megfogalmazom, hogy



2.1. ábra. Négy csomópont, amelyek multihop módon képesek kommunikálni egymással. A pontozott kör jelöli az egyes csomópontok adásának hatósugarát. A és B csomópont szomszédos, mivel képesek kölcsönösen értelmezi egymás adását. Ugyanígy B és C valamint C és D is szomszédosak. Ha A küld egy üzenetet D számára, akkor azt először B kapja meg, majd továbbítja C-nek, amely pedig eljuttatja D-nek. A csomóponti kapcsolatok kétirányúak.

számomra mit jelent a *biztonságos* útvonalválasztás.

2.1. Ad hoc útvonalválasztás

Az útvonalválasztás akkor szükséges, amikor egy csomópont (forrás) kommunikációt kíván folytatni egy másik csomóponttal (cél). Az útvonalválasztó protokoll feladata az útvonalválasztás eredményének visszaadása a forrásnak. Ez az eredmény lehet egy vagy több útvonal a forrás és cél között, vagy valamilyen alkalmas sikertelenséget jelző objektum, ha például nem sikerült útvonalat találni a cél és a forrás között.

Az útvonalválasztás lehet *reaktív* (igény szerinti, on-demand) vagy *proaktív*. Reaktív útvonalválasztás során csak akkor történik egy tényleges útvonalnak a meghatározása, ha arra szükség van, vagyis amikor egy csomópont (küldő) adatcsomagot kíván küldeni egy másik csomópontnak (fogadó). Proaktív esetben periodikus üzenetekkel állapítják meg az egyes csomópontok a lehetséges útvonalakat. A gyakorlati tapasztalatok azt mutatják, hogy a legtöbb helyzetben a reaktív útvonalválasztó eljárások lényegesen jobban teljesítenek kevesebb többletterheléssel mint proaktív társaik. A proaktív protokollok legtöbb esetben gyorsabban reagálnak a csomópontnál történő sok kapcsolatváltozásra, viszont azon a hálózatrészeken, ahol a topológiaváltozás ritka, a reaktív eljárások eliminálják a periodikus üzenetek jelentette többletterhelést.

Az igény szerinti útvonalválasztó protokolloknak is két fő csoportjuk létezik. Az egyikbe a *forrás alapú* (source routing), míg a másikba az *útvonalválasztó tábla (vagy távolsági vektor) alapú* (distance vector routing) protokollok tartoznak. A forrás alapú protokollok esetében minden üzenet explicit módon tartalmazza azt az útvonalat, amelyen az üzenetnek haladnia kell. Így az üzenetek hossza a változó hosszúságú útvonalak miatt nem állandó. Ide tartozik pl. a DSR, Ariadne, SRP. A tábla alapú protokollok esetében az egyes csomópontok az útvonalinformációkat lokálisan tárolják (pl. táblában), így az egyes hálózatbeli üzenetek nem tartalmazzák explicit módon az útvonalat amelyen az üzenetnek haladnia kell, hanem min-

den csomópont lokálisan dönt arról, hogy melyik szomszédja felé kell továbbítani az üzenetet. Ebbe a csoportba tartozik pl. az AODV, SAODV, ARAN.

A dolgozat hátralevő részében a reaktív, azon belül is először forrás és végül az útvonalválasztó tábla[1] alapú protokollokkal szeretnék foglalkozni. Ahhoz, hogy megértsük a később tárgyalandó „biztonságos" útvonalválasztó protokollok működését szükséges ismertetni mindkét családból egy-egy jelentős képviselőt, mivel a később tárgyalandó „biztonságos" protokollok is ezeknek a működési elveit követik. A 2.2. részben röviden ismertetem a DSR-t, ami egy forrás alapú, míg a 2.3. részben az AODV-t, ami pedig egy tábla alapú útvonalválasztó protokoll. Minden ad hoc útvonalválasztó protokoll két részre osztható: *Útvonalfelderítés* (Route Discovery) és *Útvonalkarbantartás* (Route Maintenance). Én az útvonalfelderítés biztonságával fogok a továbbiakban foglalkozni.

2.2. A DSR protokoll

A DSR (Dynamic Source Routing Protocol) egy egyszerű de ugyanakkor rendkívül hatékony forrás alapú ad hoc útvonalválasztó protokoll [14], amelyet David B. Johnson, David A. Maltz és Josh Broch tervezett, és első verzióját 1994-ben publikálták. A DSR figyelembe veszi a hálózatok változó és alkalmi jellegét, és sikeresen birkózik meg az egyes csomópontok mobilitása és interferencia miatt fellépő problémákkal. Mivel az útvonalat a protokollüzenetek explicite tartalmazzák, így nem szükséges az egyes csomópontoknak állandóan friss útvonalinformációt nyilvántartani. A DSR kis többletterheléssel, gyorsan, reaktív módon reagál a csomópontok állandó mozgására, és biztosítja az elküldött adatok célbaérkezését az állandó mobilitás ellenére is. A DSR jelenleg IETF (Internet Engineering Task Force) elfogadás alatt áll, és esélyes arra, hogy az IETF MANET (Mobile Ad hoc Networks) csoportja által az Internet sztenderd útvonalválasztó protokollja legyen ad hoc hálózatokban. A következőkben röviden áttekintem az alapműködését. Részletes ismertetése megtalálható [14]-ben.

Amikor egy csomópont egy csomagot akar küldeni bizonyos célcsomópont(ok)nak, akkor először ellenőrzi, hogy a kérdéses célcsomópont(ok) szerepelnek-e az átmeneti gyorsító tárban (Route Cache). Ha nem, akkor a küldő (*initiator*) kezdeményez egy útvonalfelderítést a *cél* felé. A kezdeményező kezdetben küld egy rreq (*Route Request*) üzenetet többesszórással az összes szomszédjának, amely üzenet tartalmazza a célcsomópont azonosítóját valamint egy minden rreq üzenetre jellemző egyedi azonosítószámot. Minden egyes csomópont, amely megkapja ezt az rreq üzenetet, először ellenőrzi, hogy nem kapott-e már egy ilyen azonosítójú rreq üzenetet. Ha igen, akkor eldobja, ha nem, akkor hozzáfűzi az rreq-hez a saját azonosítóját, majd továbbítja azt minden szomszédjának. Amikor az rreq eléri a kívánt célcsomópontot, akkor az válaszul küld egy rrep (*Route Reply*) üzenetet, amely a fogadott rreq üzenetben összegyűjtött csomópontazonosítókat tartalmazza. Az rrep üzenet az összgyűjtött azonosítók útvonalán visszajut a kezdeményezőhöz, amely ezután képes további adatcsomagokat küldeni a már ismert útvonalon. Az útvonalat a forráscsomópont eltárolja a saját gyorsító tárában. Minden később küldendő adatcsomag tartalmazza a fejlécében a teljes útvonalat, amelyen a csomagnak haladnia kell.

[1] röviden: tábla

Az *útvonalkarbantartás* az a mechanizmus, amely során a küldő detektálja, hogy egy adott útvonal sérült-e (pl. amikor két csomópont kikerül egymás hatótávolságából). Mikor a DSR elküld egy adatcsomagot, minden adatcsomag fejlécében rögzíti a már ismert útvonalat alkotó csomópontok listáját. Az útvonalat alkotó csomópontok továbbítják ezt a csomagot az útvonalban szereplő következő csomópont (hop) felé (*source routing*). Két szomszédos hop mindig megpróbálja valamilyen módszerrel megerősíteni a sikeres csomagküldést. Ha ez sikertelen bizonyos számú próbálkozás után, akkor a küldő csomópont egy rerr üzenetet küld az eredeti forráscsomópontnak, amelyben specifikálja a hibás kapcsolatot. Ezekután a csomag eredeti küldője az összes általa nyilvántartott útvonalat módosíthatja (eldobja vagy újat kezdeményez) a hibás kapcsolat alapján.

A DSR-nek még számos optimalizált változata létezik, és sok más eddigi ad hoc útvonalválasztó protokoll alapját képezi. Ide tartozik a ZRP [25] (Zone Routing Protocol) vagy a LAR [24] (Location-Aided Routing), amelyek az rreq csomagok többesszórása miatt fellépő hálózati terheltséget próbálják minimalizálni különböző technikákkal, figyelembe véve a csomópontok fizikai elhelyezkedést pl. GPS (Global Positioning System) segítségével. Fontos tulajdonsága a DSR-nek, hogy nem tételez fel kétirányú kapcsolatokat az egyes szomszédos csomópontok között.

2.3. Az AODV protokoll

Az AODV (Ad hoc On Demand Distance Vector) [16] szintén egy reaktív útvonalválasztó protokoll ad hoc és mobil hálózatok számára, amelyet Charles E. Perkins és Elizabeth M. Royer tervezett 1999-ben. Az egyes csomópontok lokálisan útvonalválasztó táblák formájában tárolják az útvonalválasztáshoz feltétlen szükséges információkat. A protokoll a szomszédos csomópontok között kétirányú kapcsolatot tételez fel, viszont nagyon robusztus, és gyorsan reagál a csomópontok közötti kapcsolatok hibáira. Sorozatszámokat (sequence number) használ az útvonal frissességének megállapítására. Ezek a csomópont-azonosítószámok egyben ciklusmentességet is garantálnak a hálózatban. Az AODV nagy mértékben skálázható és optimalizált, mivel egy útvonalat a célcsomópontnak bármely az útvonalon közbenső csomópont visszaadhat az útvonalfelderítés során, valamint az inaktív, nem használt útvonalak viszonylag gyorsan eliminálásra kerülnek. Ugyanakkor ezen optimalizációk megnehezítik a protokoll biztonságossá tételét. Az AODV – ugyanúgy, mint a forrás alapú protokollok – az rreq csomagok többesszórásával árasztja el a hálózatot, de az egyes csomópontok lokális útvonalválasztó táblákban tárolják az útvonalválasztáshoz szükséges információkat, amelyeket a közbenső csomópontok a kapott rreq alapján, ha szükséges, állandóan frissítenek. Ahhoz, hogy a táblában nyilvántartott információk mindig a lehető legfrissebbek legyenek az AODV célsorozatszámokat (destination sequence number) használ. Minden egyes ad hoc csomópont nyilvántart egy monoton módon növekedő sorozatszám-számlálót a célból, hogy elnyomják a nem friss rreq és rrep üzeneteket.

Az útvonalfelderítési szakasz során kezdetben a forrás egy rreq üzenetet küld el a szomszédjainak többesszórással, ha még az nem rendelkezik a szükséges útvonalinformációkkal a táblájában a célcsomópontot illetően. Minden közbenső csomópont, amely megkapja ezt az rreq üzenetet eltárolja annak a szomszédos csomópontnak a címét – mint *következő csomópont*

(next hop) a forráscsomópont felé –, akitől kapta ezt az rreq üzenetet. E mellett szintén tárolja a *forrás címét*, sorozatszámát valamint a forrás és közte levő *út hosszát*, ami jelen esetben a köztük levő csomópontok száma (hop count). Ezzel a mechanizmussal nagy mértékben optimalizálni tudják a csomópontok az útvonalfelderítési szakaszt. Ha egy közbenső csomópont (esetleg maga a cél), amely megkapja az rreq üzenetet és rendelkezik a kért útvonalinformációval, ellenőrzi annak frissességét azzal, hogy összhasonlítja az általa tárolt és a kapott rreq üzenetben szereplő célsorozatszámokat. Csak akkor válaszol a közbenső csomópont (amely nem a cél) a kérésre egy válaszüzenettel (rrep), ha az rreq-ben szereplő célsorozatszám kisebb vagy egyenlő mint a tárolt célsorozatszám, de nagyobb az rreq-ben szereplő úthossz értéke (hop count) mint a tárolt úthossz értéke. Más esetekben a közbenső csomópont (ha az nem maga a célcsomópont) újraküldi többesszórással az összes szomszédjának a kérést. A célcsomópont mindenképpen válaszol a kérésre egy rrep üzenettel, ha a kérés eljut hozzá. Ha a közbenső csomópontok nem kapnak egy meghatározott lejárati időn belül választ a továbbított kérésre, akkor törlik a kéréshez tartozó bejegyzést a táblájukból. Minden rrep pont-pont (unicast) címzéssel továbbítódik annak a közbenső csomópontnak, amelytől a küldő korábban az rrep üzenethez tartozó rreq üzenetet kapta. Ha esetleg több rrep üzenetet kapnak egy kérésre, akkor csak akkor továbbítják a később érkező rrep üzenetet, ha az nagyobb célsorozatszámmal rendelkezik, vagy az ugyanakkora, de a később érkező rrep kisebb úthosszal rendelkezik. Egy közbenső csomópont eltárolja az rrep üzenet alapján a *célcsomópont címét*, sorozatszámát, a szomszédos mint *következő csomópont címét* a cél fele amelytől a választ kapta, valamint a cél és a közte levő *út hosszát*, ami jelen esetben a köztük levő csomópontok száma (hop count). A forrás akkor kezdi meg az adatok küldését a cél felé, ha megérkezik az első rrep üzenet, és később, ha esetleg egy kedvezőbb értékekkel rendelkező rrep-et kap, akkor annak megfelelően módosíthatja az útvonalinformációit.

Látható, hogy a forrástól a célig terjedő útvonal kiépítése az rreq üzenetek küldésével, míg a céltól a forrásig terjedő ugyanazon de ellentétes irányú útvonal kiépítése az rrep üzenetek küldésével történik meg. Minden csomópont mihelyt megkap egy rreq vagy rrep üzenetet, képessé válik válaszolni azon későbbi kérésekre, amely a kapott rreq vagy rrep küldőjére vonatkoznak, hiszen mindegyik sorozatszámát megtudja a megfelelő üzenetből.

2.4. A támadó modellezése és támadási típusok

A következőkben megadom az általam használt támadómodellt, majd informálisan felsorolok néhány lehetséges támadási típust az AODV és az DSR ellen. A támadások részletes leírásai megtalálhatóak a [4], [15] és [17] munkákban.

Az információs biztonságban megkülönböztetünk *aktív* és *passzív* támadókat. Passzív támadó csak lehallgatni képes bizonyos információs csatornákat, és nem megfelelő rejtjelezés esetén értelmezni azokat, tehát csak a titkosságot és anonimitást veszélyezteti. Ezzel szemben egy aktív támadó képes a lehallgatáson túl az egyes üzeneteket módosítani, törölni, késleltetni, vagy akár tetszőleges üzeneteket beszúrni. Mivel az útvonalválasztásra csak az aktív támadó jelenthet effektíven veszélyt, így a továbbiakban csak ezzel foglalkozunk.

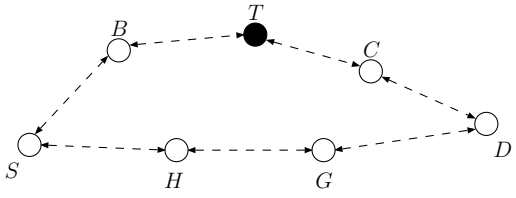
Egy aktív támadó erejét két számmal jellemezhetjük. Egyrészről azon csomópontok (eszközök) száma, amelyek a támadó birtokában vannak, vagy képes azokat kontrollálni. Más-

részről azon kompromitált csomópontazonosítók száma, amelyeket képes felhasználni, és így más csomópontoknak kiadnia magát. Feltételezzük, hogy a támadó rendelkezik minden kriptográfiai információval az egyes kompromitált azonosítókat illetően, valamint ezen azonosítókat terjeszti a hálózatban. Jelöljük ezen támadót *Aktív-y-x*-nek, ahol x jelenti az irányított csomópontok számát, valamint y a támadó birtokában levő azonosítók számát. A két szám szemléletesen alkalmas egy támadó erejének kifejezésére. Például egy *Aktív-0-x* támadó x darab csomópontot vezérel, de nem tud felhasználni egyetlen csomópontazonosítót sem, vagyis nem tudja magát kiadni más csomópontnak. Egy *Aktív-1-1* támadó képes egy kompromitált azonosítót felhasználni, és egy eszközt irányítani a hálózatban. Az *Aktív-VC* támadónak irányítása alatt vannak olyan csomópontok, amelyek a hálózatban egy csomópontvágatot alkotnak, tehát a hálózatot olyan diszjunkt partíciókra bontják, amelyek között a megbízható csomópontok csak egy támadó csomóponton keresztül képesek kommunikálni. Ezen támadó képes a partíciók közötti összes adatforgalmat támadni.

Általánosan kétféle támadási kategóriát különböztetünk meg ad hoc hálózatokban az útvonalválasztás ellen. *Útvonalbomlasztás* esetén a támadó az érvényes adatcsomagokat nem a rendeltetési útvonalon továbbítja, míg *erőforrásleterhelés*nél a támadó értékes hálózati erőforrásokat foglal le – mint pl. hálózati sávszélesség, csomópont-erőforrások (memória, processzálási idő, stb.) – megbénítva így a hálózati forgalmat. A továbbiakban én az útvonalbomlasztással fogok foglalkozni.

Az AODV-t vizsgálva látható, hogy sok egyszerű támadás megvalósítható ellene, hiszen az nem használ semmilyen kriptográfiai eszközt az üzenetek integritásának a védelmére. Már önmagában a cél- és forráscsomópontnak a címe is megváltoztatható például egy rreq üzenetben, amivel így akár más csomópontokat személyesíthet meg, vagy a saját csomópontját tüntetheti fel mint cél- vagy forráscsomópont. Mindehhez csak annyi szükséges, hogy egy támadó csomópont a kérdéses útvonalban szerepeljen mint köztes csomópont, vagy szomszédos legyen egy az útvonalban szereplő köztes csomóponttal.

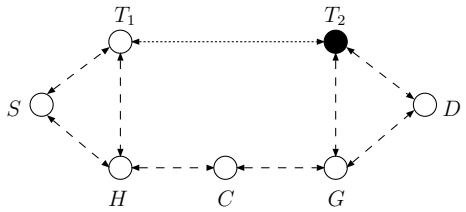
A sorozatszámok manipulálásával az egyes kérések frissességét képes befolyásolni, míg a hop szám változtatásával növelheti vagy csökkentheti egyes útvonalak kiválasztásának a valószínűségét. Ezt szemlélteti a 2.2. ábra.



2.2. ábra. S csomópont útvonalkeresést kezdeményez D felé. AODV protokollt futtatva a közbenső T támadó csomópont például válaszolhat D nevében a valóságos értékeknél egy sokkal nagyobb célsorozatszámú vagy akár egy sokkal kisebb hop számú rrep üzenettel a kérésre. Így akár azt is elérheti, hogy az (S, B, T, C, D) útvonalon kommunikáljon S és D az (S, H, G, D) útvonal helyett.

A 2.3. ábra egy csatornás támadást (tunneling attack) szemléltet. Itt ugyan T_1 és T_2 támadó csomópontok nem szomszédosak, mégis T_1 képes egy fabrikált kérést egy másik legális kérésben csatornázva eljuttatni T_2-nek, és így közvetetten D-nek. Ha hop számot használunk az útvonalhossz mérésére, akkor ezt a támadást nem lehet kivédeni. Ha viszont valamely

az adott protokoll működésétől független metrikát használunk (pl. fizikai idő), akkor ez a támadás megakadályozható (lásd: ARAN a 9. fejezetben).



2.3. ábra. S csomópont útvonalkeresést kezdeményez D felé. A közbenső T_1 támadó csomópont (S, T_1, T_2, D) útvonalat (T_1 és T_2 nem szomszédosak) mint adatot beleágyazza egy S által indított rreq üzenetbe (pl. rejtjelezve). T_2 a beágyazott kérést továbbítja D felé. A választ szintén beágyazva T_2 eljuttatja T_1-nek. Az rövidebb hossza miatt (S, T_1, T_2, D) nem létező útvonal lesz kiválasztva (S, H, C, G, D) helyett.

A fenti támadásokon kívül még sok más egyszerűbb támadás is létezik. Például az útvonalválasztási ciklus (routing loop) esetén a támadó olyan érvényes rreq üzeneteket konstruál, amelyek eredményeképp ciklikusan keringenek az adatcsomagok a hálózatban, soha nem érve el a kívánt célt, miközben értékes erőforrásokat foglalnak le. Egy támadó létrehozhat fekete lyukat (black hole), amelynél a támadó eléri, hogy egy adott csomópontnak küldött üzenetek felé továbbítódjanak, majd eldobja ezeket. Ennek egy speciális esete a szürke lyuk (grey hole), amelynél csak szelektált csomagok kerülnek eldobásra, a többi továbbítódik. Lehetséges cél lehet nem optimális útvonalak, ú.n. kerülők visszaadása (suboptimal routes, detours), vagy a hálózat particionálása, így megakadályozva a különböző partíciókba eső csomópontok kommunikációját. Egy speciális kiterő útvonal, mikor a visszaadott útvonal nem létező, virtuális csomópontokat tartalmaz (gratuitous detour).

Egy speciális de fontos támadási típus a féregjárat (wormhole). Ebben az esetben egy A és egy B támadó csomópont privát csatornát létesítenek a kettőjük közötti kommunikációhoz. Így minden csomag, amely eléri A-t, eléri B-t is a „féregjáraton" keresztül. Ez a kapcsolat fordított irányban is lehetséges. Ilyen módon a hálózatban a két támadó virtuális kapcsolatokat építhet ki csomópontok között, hiszen két csomópont a féregjárat miatt hiheti azt, hogy szomszédosak, valójában fizikailag távol helyezkednek el egymástól. Ezt a virtuális kapcsolatot persze a két támadó csomópont vezérli.

Rushing támadás célja lehet érvényes rreq csomagok eldobása, amely bizonyos optimalizált protokollok esetén léphet fel. Duplikált üzenetek elnyomását alkalmazzák a felesleges rreq üzenetekkel történő hálózat-elárasztás ellen, amikor minden rreq üzenetet csak egyszer továbbít minden csomópont. Ekkor elérheti egy támadó, hogy rreq elárasztást megelőzve gyorsan elterjeszti az általa megkonstruált rreq üzeneteket, ami után az érvényes rreq üzenetek már eldobásra kerülnek a megbízható csomópontok által.

A DoS támadások hatása csökkenthető, ha minden rreq üzenet élettartamára bizonyos korlátot szabunk (pl. csak n darab hopon mehet keresztül), vagy ha a visszaadott útvonal hosszát maximalizáljuk. DoS támadásnak az minősül, ha a csomópontok valamint a támadó által végzett műveletek száma a hálózat méretének a nagyságrendjében mozog.

A fentiekben a teljesség igénye nélkül soroltam fel néhány támadást az útvonalválasztás

ellen ad hoc hálózatokban ([15], [17], [4]). Ezek nagy része elkerülhető vezetékes hálózatokban, hiszen ott egy központi elem (útvonalirányító) képes ellenőrizni az egész adatforgalmat. Ugyanakkor vezetéknélküli ad hoc hálózatokban a megfelelő kriptográfiai primitívek alkalmazásával a legtöbb támadás sikerének a valószínűsége elhanyagolhatóvá válik.

2.5. Az útvonalválasztás biztonsága

Az eddig javasolt útvonalválasztó protokollok nagy része megbízható csomópontokból álló ad hoc hálózatot tételezett fel, ugyanakkor a valós életben ez nem mindig teljesül mint azt az előbb megmutattuk. A kívánt biztonsági szint elérése megfelelő kriptográfiai primitívek alkalmazásával történhet. Sajnos a legtöbb kriptográfiai primitív meglehetősen számításigényes, ami ezen hálózattípusok hatékonyságát a korlátolt számítási kapacitás miatt erősen befolyásolja. A mobil ad hoc hálózatok többsége dinamikus viselkedésű; a hálózat topológiája időben gyakran változik, új kapcsolatok épülhetnek ki és tűnhetnek el rövid időn belül.

Az alapvető kérdés amit feltehetünk, hogy mikor biztonságos egy ad hoc útvonalválasztó protokoll. A kérdés megválaszolása azért is fontos, mert csak egy konkrétan megfogalmazott cél köré lehet precíz, pontos formális modellt építeni. Az egyes protokollok bemutatása során látható volt, hogy mindegyik működési mechanizmusa eltérő, és magát az eredményt is (ami legtöbbször egy útvonal) más formában adják vissza, gondoljunk csak a forrás és tábla alapú protokollokra. Így bizonyos, hogy a biztonsági követelmények a két esetben különbözni fognak. Az viszont nyilvánvaló, hogy magának az eredménynek, vagyis a visszaadott útvonalnak a valóságot hűen tükröznie kell, és a protokoll futása során a hálózat nem kerülhet olyan állapotba, ami nem felel meg a valóságnak. Részletesebben kifejtve:

- Forrás alapú protokollok esetén a visszaadott útvonal létezzen a hálózatban. Az egyszerűség kedvéért most feltételezhetjük, hogy a csomópontok lokálisan nem tárolnak útvonalinformációt, vagyis eltekintünk az optimalizációktól. Ekkor a hálózat állapotát egyedül az az üzenet jellemzi, amely explicit módon tartalmazza az útvonalat. Egy nem létező útvonal akkor kerülhet visszaadásra, ha például egy támadó csomópont a válaszüzenetben szereplő útvonalba valójában nem létező csomópontot szúr be, vagy egy olyan meglévő és létezőt távolít el, amely nélkül a megmaradt csomópontok nem képesek a végpontok üzeneteinek továbbítására.

- Tábla alapú protokollok esetén az egyes csomópontok táblabejegyzéseinek az együttese írja le a visszaadott útvonalat. Így itt ésszerű azt megkövetelni, hogy a protokoll futása során egyik csomópont táblájába se kerüljön olyan bejegyzés, amely nem felel meg a valóságnak. Ha ez előfordulhatna, akkor a protokoll további futása során újabb valótlan bejegyzések kerülhetnek a többi táblákba is, ami a futás végére kialakult táblabejegyzések és így magának a visszaadott útvonalnak a valótlanságához vezetne. Egy valótlan bejegyzés például, ha nem létezik egy bejegyzésen belül a bejegyzett hosszal a szomszédként bejegyzett csomóponton keresztül egy másik célként bejegyzett csomópont fele útvonal a hálózatban.

Mivel az ad hoc hálózatok dinamikusan állandóan változnak, vagyis a csomópontok mozognak, és így új kapcsolatok épülnek fel és meglévők tűnnek el, ezért nem egyértelmű, hogy

egy útvonal létezését valamint az állapot valódiságát milyen időtartamra levetítve vizsgáljuk. Én egy protokoll elemzése során feltételezem, hogy az analízis ideje alatt a hálózat nem változik (statikus), vagyis mintha „befagyasztanánk” a hálózatot a kérdéses protokoll analízise alatt.

Ezen intuitív fogalmak köré fogom felépíteni a formális modellt, amelyhez a *szimulációs paradigmát* fogom felhasználni.

3. fejezet

Ad hoc hálózatok modellezése és a szimulációs paradigma

Ebben a fejezetben leírom az általam definiált és használt ad hoc hálózati modellt. A 3.1. részben felvázolok egy statikus modellt, vagyis itt még nem modellezem a dinamikus működést, tehát a protokollok működési mechanizmusát. A dinamikus viselkedés modellezését egy alkalmas szimulációs modell kialakításával valósítom meg, melynek egy ad hoc hálózatokra generalizált változatát írom le a 3.4. részben. A 3.2. részben kerül bemutatásra a támadó statikus modellje. A 3.3. rész részletesen leírja az egész hálózat felépítését egyértelműen jellemző konfigurációt illetve a konfiguráció redukció fogalmát. Az egyes útvonalválasztó protokollok eltérő működési mechanizmusuk miatt eltérő szimulációs modellel rendelkeznek. Így ezek külön ismertetésre kerülnek mind a forrás mind pedig a tábla alapú protokollok esetében. A 3.5. részben ismertetem a kriptográfiai protokollok biztonságának bizonyítására széles körben alkalmazott szimulációs paradigmát.

3.1. Hálózati modell

A következőkben bemutatom az egyes absztrakciós szinteknek megfeleltetett gráfokat. A biztonság fogalmát az utolsó, L_3 gráfon fogjuk értelmezni.

L_2 gráf. Az adatkapcsolati szinten – ami a fizikai réteg felett elhelyezkedő 2. szint az ISO OSI hivatkozási modellben – egy ad hoc hálózatot vezetéknélküli eszközök alkotják. Ebben a rétegben a hálózatot egy $G_{L_2}(E, V)$ gráf reprezentálja, ahol E a gráf éleinek a halmaza, míg V a csúcsok halmaza. $G_{L_2}(E, V)$ gráfot hívjuk röviden L_2 gráfnak.

L_2 gráf csúcspontjai az egyes eszközöknek felelnek meg. Feltételezzük, hogy minden eszköz vagy a támadó vagy egy megbízható felhasználó irányítása alá esik. A megbízható felhasználó irányította csomópontokat röviden *megbízható csomópont*oknak, a támadó irányítása alatt levőket pedig *támadó csomópont*oknak fogjuk hívni. A támadó modellezését lásd a 3.2. részben.

L_2 gráf élei jelentik a kommunikációs csatornát az egyes eszközök között. Vagyis $(v, v') \in E$ akkor és csak akkor, ha v és v' eszközök képesek venni egymás adását. Feltesszük, hogy L_2 gráf irányítatlan, tehát ha v hallja v'-t, akkor v' is hallja v-t. Mivel a támadó képes

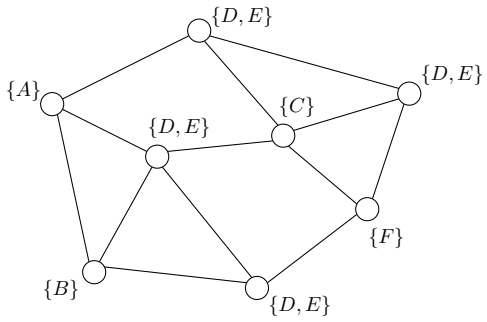
*féregjárat*okat létrehozni a hálózatban, ezért L_2 egy éle értelmezhető rádiókapcsolatként is és féregjárat okozta kapcsolatként is.

Itt fontos megemlíteni, hogy a féregjáratot alapvetően a szomszédfelderítési eljárás (*neighbor discovery mechanism*) elleni támadásnak veszem, nem pedig az útvonalválasztás elleninek, ezért a továbbiakban nem foglalkozom vele.

L_3 **gráf.** Ezen gráf írja le a hálózati szintű kapcsolatokat az egyes csomópontok között. Hálózati szinten – ami az adatkapcsolati szint fölötti 3. szint az ISO OSI hivatkozási modellben – az eszközök valamilyen hálózati szintű azonosítót használnak, amely egyértelműen azonosítja magát a csomóponti eszközt (pl. hálózati cím, publikus kulcs). A lehetséges azonosítók halmazát jelöljük L-el. Feltesszük, hogy a megbízható csomópontok és L között kölcsönösen egyértelmű hozzárendelés van.

A továbbiakban feltételezem, hogy minden eszköz használ valamilyen eljárást szomszédjai azonosítóinak a megállapításához (neighbor discovery mechanism). Itt szomszéd alatt az L_2 gráfban szomszédos csúcsoknak megfelelő eszközöket értjük. A szomszédos kapcsolatok kialakítása a saját azonosítók alapján történik.

Hálózati szinten a hálózat leírható egy címkézett $G_{L_3}(E, V, \mathcal{L})$ gráffal, ahol E és V ugyanazon élek és csúcsok halmazát jelentik mint L_2 gráf esetében, $\mathcal{L} : V \to 2^L$ pedig egy címkéző függvény, amely minden egyes csúcshoz hozzárendeli L egy részhalmazát, ami azon azonosítókat reprezentálja, amelyek az egyes csúcsoknak megfelelő eszközhöz lettek hozzárendelve. Ha v és v' két különböző megbízható csomópont, akkor $\mathcal{L}(v)$ és $\mathcal{L}(v')$ egyelemű halmazok (szingletonok), és $\mathcal{L}(v) \neq \mathcal{L}(v')$. $G_{L_3}(E, V, \mathcal{L})$ gráfot hívjuk L_3 gráfnak, az L_3 gráfok halmaza pedig legyen $\mathbb{G}$. Egy L_3 gráf látható a 3.1. ábrán.



3.1. ábra. Az L_3 gráfban két csúcs között pontosan akkor megy él, ha a megfelelő csomópontok hallják egymás adását. Minden csúcs mellett jelöltem a csúcs azonosítóit, ami megbízható csomópont esetén egy szingleton, míg támadó esetén az összes általa használt azonosítók halmaza. Viszont itt még pontosan nem tudjuk, hogy melyek a támadó csomópontok.

Definiálok egy $\mathcal{N} : V \to 2^L$ függvényt, amely visszaadja azon csúcsok azonosítóit, amelyek szomszédosak az argumentumként definiált csúccsal. Formálisan:

$$\mathcal{N}(v) = \bigcup_{\forall w:(v,w)\in E} \mathcal{L}(w)$$

Szemléletesen ha v egy megbízható csomópont, akkor $\mathcal{N}(v)$ jelöli v feltételezett szom-

szédjait G_{L_3} gráfban.

3.2. A támadó statikus modellezése

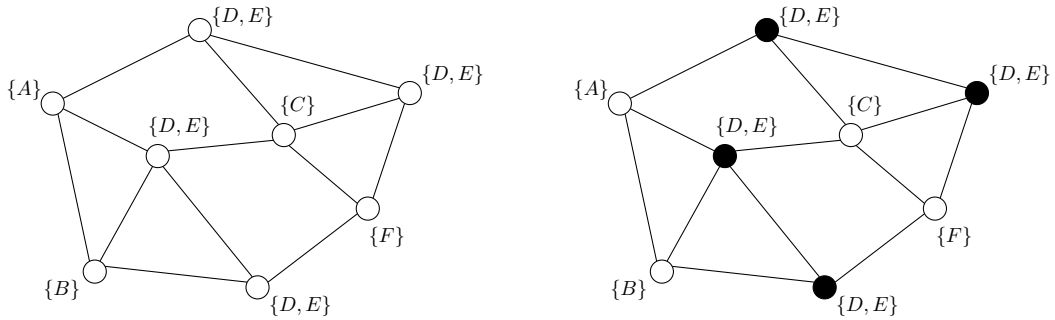
Feltételezem, hogy a támadónak – akit $\mathcal{A}$-val jelölünk – nincs teljes ellenőrzése a hálózat felett, de irányítani képes x csomópontot és hitelesíteni tudja magát y csomópontazonosító által (*Aktív-y-x* támadó).

Jelölje V^* azon csúcsok halmazát, amelyek a támadó irányítása alá esnek. Látható, hogy $V^* \subset V$, valamint Aktív-y-x támadót feltételezve $|V^*| = x$. Feltesszük még, hogy a támadó rendelkezik L^* kompromitált azonosítókkal (pl. publikus kulcsú azonosítás esetén a megfelelő privát kulcsokkal), ahol $L^* \subset L$. Ez azt is jelenti, hogy ezen azonosítókat használhatja a szomszédok megállapítása során (spoofing), és így bármely irányítása alatt levő eszköz használhat bármely L^*-ban levő azonosítót, formálisan minden $v \in V^*$-ra $\mathcal{L}(v) = L^*$. Ebből következik, hogy Aktív-y-x támadó esetén $|L^*| = y$.

Én most azokkal az esetekkel foglalkozom, amikor L^* nem tartalmazhat már kirendelt (más megbízható csomópont által használatban levő) azonosítót, vagyis egy támadó csomópont nem adhatja ki magát a rendszerben már létező más megbízható csomópontnak. A továbbiakban azt is felteszem, hogy egy támadó csomópont minden szomszédja felé ugyanazzal az azonosítóval jelenik meg mint szomszéd. Ha a támadó és a megbízható csomópontok birtokában levő azonosítók halmaza nem diszjunkt, akkor vagy a szomszédok felderítése során a felhasznált azonosítók nincsenek hitelesítve, vagy hitelesítve vannak, de a felhasznált hitelesítő (titkos) kulcs kompromitálódott, és a kulcs tulajdonosa ennek még nincs tudatában. A továbbiakban feltételezem, hogy ilyen eset nem fordul elő.

3.3. A konfiguráció

Az eddigiek alapján látható, hogy egy ad hoc hálózatot egyértelműen leír egy $(G_{L_3}(E, V, \mathcal{L}), V^*)$ kettős, ahol $G_{L_3}(E, V, \mathcal{L}) \in \mathbb{G}$. Legyen $conf = (G_{L_3}(E, V, \mathcal{L}), V^*)$ egy ad hoc hálózatot egyértelműen leíró *konfiguráció*, ezek halmaza pedig legyen $\mathbb{K}$. Jelöljük $\mathbb{K}^{dj} \subset \mathbb{K}$-vel azon konfigurációk halmazát, ahol a támadó és a megbízható csomópontok által használt azonosítók halmaza diszjunkt. Formálisan megfogalmazva: minden $conf = (G_{L_3}(E, V, \mathcal{L}), V^*) \in \mathbb{K}^{dj}$ konfigurációra minden $v^* \in V^*$ csúcs esetén igaz, hogy minden $v \in V \setminus V^*$ csúcsra $\mathcal{L}(v^*) \cap \mathcal{L}(v) = \varnothing$. Ilyen konfigurációra mutat példát a 3.2. ábra. Ezen konfigurációk között viszont még mindig vannak olyanok, amelyek L_3 gráfjaiban vannak szomszédos támadó csomópontoknak megfeleltetett csúcsok. Ezzel a modellezési feltételezéssel olyan támadásokat is megengednénk (lásd később 4. és 7. fejezetekben), amelyek ellen nem tudunk védekezni, hiszen a szomszédos támadó csomópontok közötti szabad információcserét nem tudjuk megakadályozni és ellenőrizni. Ezért definiálunk $conf \in \mathbb{K}^{dj}$ konfigurációk között egy $\triangleright \subseteq \mathbb{K}^{dj} \times \mathbb{K}^{dj}$ relációt. Mielőtt ezt megtennénk, bevezetünk egy segédfüggvényt. A támadó csomópontok alkotta blokk egy konfiguráció L_3 gráfjának olyan összefüggő maximális méretű részgráfja, amelyben csak támadó csomópontoknak megfeleltetett csúcsok találhatóak. $\mathcal{K} : V^* \to 2^{V^*}$ függvény visszaadja annak a támadó csomópontok alkotta blokknak az összes csúcsát, amelynek tagja a függvény argumentumaként megadott csúcs.



3.2. ábra. A jobb oldalon maga a konfiguráció, míg bal oldalon annak L_3 gráfja látható. A konfiguráció esetén tudjuk, hogy a támadó pontosan melyik azonosítókat melyik csomópontoknál használja. A támadó csomópontoknak megfeleltetett csúcsokat telített körök jelzik. Látható, hogy a támadó nem használhat már kirendelt azonosítót.

1. definíció (Konfiguráció redukció). *Legyenek adottak* $conf = (G_{L_3}(E, V, \mathcal{L}), V^*) \in \mathbb{K}^{dj}$ *és* $conf' = (G_{L_3}(E', V', \mathcal{L}'), V'^*) \in \mathbb{K}^{dj}$ *konfigurációk.* $conf \triangleright conf'$ *akkor és csak akkor, ha igaz, hogy* $v' \in V'$ *pontosan akkor, ha létezik olyan* $v \in V$ *csúcs, hogy* $\mathcal{L}'(v') = \mathcal{L}(v)$ *és*

- *ha* $\mathcal{L}(v) = L^*$, *akkor* $\mathcal{N}(v') = \bigcup_{\forall w : w \in \mathcal{K}(v)} \mathcal{N}(w) \setminus L^*$,
- *ha* $\mathcal{L}(v) \neq L^*$, *akkor* $\mathcal{N}(v) = \mathcal{N}(v')$. ♣

A definíció következménye, hogy a redukált konfiguráció gráfjában a csúcsok száma megegyezik az eredeti gráfban található támadó csomópontok alkotta blokkok minimális számának és a megbízható csomópontoknak megfeleltett csúcsok számának az összegével. A definíció azt mondja, hogy egy redukált konfigurációnak megfeleltetett hálózatban nem léteznek olyan szomszédos csomópontok, amelyek kompromitált azonosítót használnak, ami jelen esetben ekvivalens azzal az állítással, hogy nem léteznek szomszédos támadó csomópontok ebben a hálózatban.

A fenti redukcióra egy egyszerű algoritmus is adható. Módosítsunk egy $conf$-beli L_3 gráfot az alábbi módon. Alkalmazzunk $conf \in \mathbb{K}^{dj}$ konfiguráción egy $\mathcal{G} : \mathbb{K} \to \mathbb{K}$ függvényt, amely a $conf$-beli L_3 gráfban minden szomszédos támadó csomópontnak megfeleltethető szomszédos csúcsot összevon egy csúcsba, és a kapott konfigurációt adja vissza értékül.

$$\mathcal{G}\left((G_{L_3}(E, V, \mathcal{L}), V^*)\right) = (G_{L_3}(E', V', \mathcal{L}'), V'^*)$$

$\mathcal{G}$ függvény tekinthető egy speciális gráfalgoritmusnak, amelynek egy lehetséges definícióját mutatják pszeudónyelven az 1. és 2. algoritmusok.

Az új csúcs szomszédai az összevont csúcsok szomszédai lesznek. A 1. algoritmus futása során visszaad egy olyan konfigurációt, amelyhez tartozó L_3 gráfban nincs szomszédos támadó csomópont. A 2. algoritmus egy rekurzív segédfüggvény, ami visszaadja a paraméterként kapott támadó csomópontnak megfelelő csúcshoz szomszédos szintén támadó csomópontnak megfelelő csúcsok listáját és a módosított élhalmazt. A $\leftarrow$ mint halmaz operátor szintaktikailag az uniót jelenti, vagyis $V \leftarrow v$ ekvivalens a $V = V \cup \{v\}$ kifejezéssel, ahol v egy elem vagy egy halmaz.

Algoritmus 1 $\mathcal{G}(G_{L_3}(E,V,\mathcal{L}),V^*)$

let $V' = E' = A = V'^* = \varnothing$
for all $v \in V$ **do**
  if $v \in V^* \wedge v \notin A$ **then**
    $(A, E') \leftarrow$ GetAdversaryNeighbors(v, V^*, E, v)
    $V'^* \leftarrow v$
  else if $v \in V \setminus V^*$ **then**
    $V' \leftarrow v$
    for all $(v, v') \in E$ **do**
      if $v' \in V \setminus V^*$ **then**
        $E' \leftarrow (v, v')$
      end if
    end for
  end if
end for
$V' \leftarrow V'^*$
return $(G_{L_3}(E', V', \mathcal{L}), V'^*)$

Algoritmus 2 GetAdversaryNeighbors(v, V^*, E, a)

let $A = T = \varnothing$
for all $(v, v') \in E$ **do**
  if $v, v' \in V^*$ **then**
    $A \leftarrow v'$
    $(A, T) \leftarrow$ GetAdversaryNeighbors(v', V^*, E, a)
  else
    $T \leftarrow (a, v')$
  end if
end for
return (A, T)

1. állítás. *Legyen adott egy* $conf = (G_{L_3}(E,V,\mathcal{L}),V^*)$ *konfiguráció. Ha* $conf \triangleright conf'$*, akkor* $\mathcal{G}(conf) = conf'$*, ahol* $conf' = (G_{L_3}(E',V',\mathcal{L}'),V'^*)$ ♢

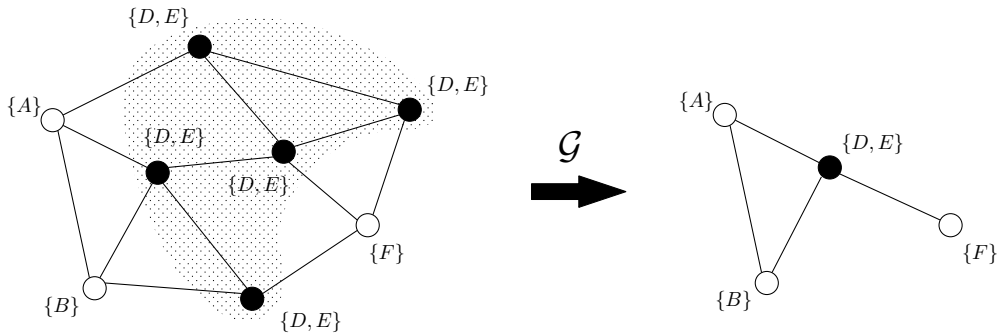
Bizonyítás Az igazolás 1. algoritmus és a konfiguráció redukciója alapján egyértelmű. Az ellenkező irány könnyen belátható, hogy nem igaz, hiszen 1. algoritmussal megvalósított $\mathcal{G}$ alkalmazható olyan $conf$ konfigurációra, amelyre $conf \notin \mathbb{K}^{dj}$. ■

A továbbiakban ha külön nem említjük meg, akkor egy ad hoc hálózat konfigurációján mindig egy *redukált* $conf = (G_{L_3}(E,V,\mathcal{L}),V^*)$ konfigurációt értünk.

Megjegyzés: A modell eddigi leírásából látható, hogy a megbízható csomópontok, azok azonosítói valamint a gráfban a nekik megfeleltetett csúcsok között kölcsönösen egyértelmű megfeleltetés van. Így a dolgozat hátralevő részében ezen fogalmak intuitív jelentése ugyanaz, és csak formális jelölésben térnek el egymástól.

3.4. Ad hoc hálózatok szimulációs modellje

Az itt bemutatandó modell az [3]-ben leírt ad hoc protokollok számára javasolt modell egy kibővített változata, amellyel leírhatók az Aktív-y-x támadót is tartalmazó ad hoc hálózatok.



3.3. ábra. A konfiguráció redukció során összevonjuk a szomszédos támadó csomópontoknak megfelelő csúcsokat egy csúccsá, mivel a szomszédos támadó csomópontok szabadon cserélhetnek egymás között információt, ami ellen nem tudunk védekezni.

Kommunikációs modell. Mivel az ad hoc hálózatok multi-hop jellegűek, ezért nem modellezhetők egyetlen átmeneti tárral (single buffer), amelybe a protokoll résztvevők üzeneteket helyeznek el, majd ezen üzeneteket a hálózat kézbesíti a megfelelő címzettekhez. Továbbá figyelembe kell vennünk a vezetéknélküli hálózatok többesszórás jellegét, vagyis hogy egy csomópont által küldött üzenetet minden olyan csomópont képes venni, amely a küldő hatótávolságán belülre esik, annak ellenére, hogy azt nem feltétlen neki címezték.

Támadómodell. Az eddig javasolt formális modellekben a Dolev és Yao által javasolt támadómodellt[1] [18] használták a támadó modellezésére, ami a valóságban egy erős támadót tételez fel (a gyakorlatban ennél gyengébb fordul elő). Ezen támadó az összes protokollüzenetet képes értelmezni, késleltetni, módosítani, átirányítani, újakat beszúrni, viszont a kriptokulcs hiányában nem képes a rejtjelezett üzenetet megfejteni. Kriptoanalízist és statisztikai támadásokat nem tud végezni, és csak a birtokában levő üzenetekkel és kulcsokkal operál. Itt a támadónak teljes ellenőrzése lehet a hálózati átmeneti tár (buffer) felett. Mivel a strukturált, centralizált hálózatokban egy támadó átvehette az irányítást speciális szerepű csomópontok felett (útvonalirányítók), így ebben az esetben ez egy helyénvaló modell. Ugyanakkor egy vezetéknélküli hálózat esetén egy támadó csak akkor valósíthat meg egy ilyen támadást, ha fizikailag mindenhol jelen van a hálózatban. Ez a legtöbb esetben túl költséges lenne, így nem jelenthet egy reális célt a támadónak. Ezekből adódóan feltételezzük, hogy az *ad hoc hálózati modellben egy támadó képességei megegyeznek egy átlagos csomópont képességeivel kommunikációs szempontból.* Jelen esetben – ahogy azt a 3.1. részben már leírtuk – a hálózat reprezentálható egy gráffal, ahol minden csúcsnak megfeleltetünk egy csomópontot. A csúcsok között akkor megy él, ha a csúcsoknak megfeleltetett csomópontok (eszközök) képesek venni egymás üzeneteit. Mivel a támadó csomópont is egy átlagos képességű csomópont a fentiek alapján, így ő is csak a szomszédos csúcsoknak megfeleltetett eszközök üzeneteit képes venni, és hasonlóan az ő üzeneteit is csak a szomszédos csomópontok képesek közvetlenül olvasni. A mi modellünkben a támadó *nem adaptív*, tehát egy időben képes több

[1] Angol elnevezései: eavesdropper, penetrator, attacker, spy, intruder, enemy, vagy egyszerűen: Eve, Mallory, Trudy

egymástól független támadást is indítani, viszont ezeknek teljesen függetleneknek kell lenniük egymástól.

Működési modell. A Dolev-Yao modell szerint egy támadó képes volt ütemezni a megbízható résztvevők működését, mivel azok üzenetvezéreltek. Vagyis akkor válnak aktívvá, amikor egy üzenetet kapnak, feldolgozzák az üzenetet, esetleg generálnak valamilyen választ, végül visszatérnek inaktív módba újabb bemenő üzenetre várakozva. Mivel egy aktív támadó képes az üzenetforgalmat a fent említett módon irányítani, ezért képes indirekt módon vezérelni a megbízható csomópontok működését. Ad hoc hálózatok esetében egy támadónak nincs teljes irányítása az egész hálózat felett, tehát vannak olyan események, amelyeket nem képes befolyásolni. Ezért az egyes csomópontok nemcsak általa kerülhetnek aktív állapotba. Így a jelen modellben minden egyes csomópont egy feltételezett (hipotetikus) ütemező által kerülhet aktív állapotba. Ezen aktiválások menetekben történnek, minden menetben minden csomópont egyszer kerül aktív állapotba az ütemező által. Első ránézésre ez egy szinkron hálózati modellt jelent, ugyanakkor az ad hoc hálózatok lehetnek aszinkron jellegűek is. Viszont fontos megjegyezni, hogy ha az aktuális menetszámot az egyes csomópontok nem ismerik, akkor ezen rendszerről megállapított jellemzők nemcsak szinkron ad hoc hálózatok esetén lesznek igazak.

3.5. A szimulációs paradigma

A kriptográfiai protokollok bizonyításához a szimulációs paradigmát fogjuk felhasználni [9]. Itt kétféle modellt fogunk definiálni a vizsgálandó protokoll számára. A *valós világ modell* leírja a protokoll működését minden részletében számításelméleti eszközökkel. Az *ideális világ modell* absztrakt módon specifikálja a protokoll ideális működését, vagyis az elvárásainkat a vizsgálandó protokollal szemben. Más szóval a valós világ modell tekinthető az implementációnak, míg az ideális világ modell a specifikációnak. Mindkét modell tartalmazhat támadót. A valós világbeli támadó egy tetszőleges képességű entitás (processz, program, személy, stb.) figyelembe véve a vezetéknélküli hálózatok korlátait, míg az ideális világbeli támadó képességei korlátozottak. Az ideális világbeli támadó reprezentálja a rendszerrel szembeni *elviselhető támadásokat* (*tolerable imperfections*). Ezen támadások ellen nem tudunk vagy túl költséges védekezni, ezért inkább megengedjük (toleráljuk) őket. *A protokoll akkor biztonságos, ha a valós és ideális világ modellek kimenetei megkülönböztethetetlenek a megbízható protokollrésztvevők szemszögéből.* Úgy is fogalmazható, hogy a biztonságosság bizonyítása során megmutatjuk, hogy bármely valós világbeli támadó hatása a protokoll végrehajtása során *szimulálható* egy megfelelően kiválasztott ideális világbeli támadóval. Az ideális világ modell definiálása során megfogalmazzuk, hogy egy protokollnak *mit* kell megvalósítania, és azzal nem foglalkozunk, hogy ezt *hogyan* teszi.

4. fejezet

A forrás alapú ad hoc útvonalválasztás modellje

Elsőként a forrás alapú ad hoc útvonalválasztó protokollokat fogom elemezni. Először a 4.1. részben bevezetem a plauziblis útvonal fogalmát, informális módon megfogalmazom a biztonsági követelményemet, amelyet azután a szimulációs paradigma segítségével formálisan is leírok a 4.2. és 4.3. részekben.

4.1. A plauziblis útvonal

Ad hoc hálózatok esetén is meg kell fogalmaznunk azokat a követelményeket, amiket egy útvonalválasztó protokollnak teljesítenie kell. Ez viszont jelen esetben nem egy egyszerű feladat, aminek számos oka van. Az egyik ilyen, hogy a legtöbb útvonalválasztó protokoll elég összetett, és általában tartalmaz valamiféle optimalizációt, mely szinte lehetetlenné teszi a kimenet pontos leírását. Például egy olyan egyszerű követelményt, hogy mindig a legrövidebb utat adja vissza, egy létező protokoll sem elégíti ki. Egyes protokollok például átmenetileg tárolják az útvonalakat (pl. DSR), viszont ekkor már nem feltétlen az aktuális topológiai felépítéssel számolnak. Más protokollok szabályozzák a kérések küldésének gyakoriságát, amivel a hálózat felesleges elárasztását korlátozzák (pl. SRP). Így a kérések terjedése függ attól is, hogy egy csomópont milyen gyakorisággal bocsát ki kérésüzeneteket. Ezeken kívül még azt is figyelembe kell venni, hogy az egyes csomópontoknak eltérő lehet a hardware felépítésük, és így az üzeneteket is eltérő késleltetéssel bocsátják ki. Ezért szintén előfordulhat, hogy nem az optimális útvonalat adja vissza a protokoll. Összegezve látható, hogy az eltérő csomóponti késleltetések, a generált forgalom mennyisége valamint az egyes protokollok összetettsége szinte lehetetlenné teszi az ideális világbeli protokoll kimenetének valószínűségi eloszlásának pontos meghatározását.

A következőkben bevezetem a *plauzibilis útvonal* fogalmát, amit felhasználva megfogalmazom az ideális világ modell elvárásait, tehát felállítom a biztonsági modellemet.

Egy útvonalválasztó protokoll egy hálózati szintű elosztott algoritmus, amely egy redukált konfiguráció L_3 gráfja alapján működik. Az algoritmust az egyes eszközök futtatják azzal a céllal, hogy útvonalakat találjanak. Egy útvonal megfelel egy hálózati azonosítókból álló sorozatnak, amely rendelkezik bizonyos tulajdonságokkal. Legyen $(L_1, L_2, \ldots, L_k)$ egy

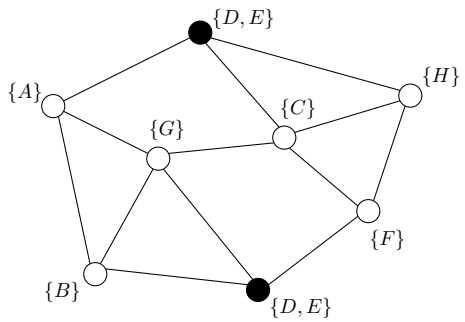
ilyen azonosítókból álló sorozat, vagyis útvonal. Ha nincs jelen a hálózatban támadó, akkor $L_1, L_2, \ldots, L_k$, ahol $k \geq 2$, egy érvényes útvonal $G_{L_3}(E, V, \mathcal{L})$ gráfban, ha $L_1, L_2, \ldots, L_k$ azonosítók mindegyike különböző, és létezik egy olyan $v_1, v_2, \ldots, v_k$ csúcsok sorozata V-ben, hogy $(v_i, v_{i+1}) \in E$ $(1 \leq i < k)$ és $\mathcal{L}(v_i) = \{L_i\}$.

Mivel V^* és L^* nem feltétlen szingleton, ezért figyelembe kell vennünk, hogy a támadó képes emulálni az útvonalkereső protokoll működését kompromitált azonosítók között, hiszen egy útvonalba tetszőleges számszor beszúrhat kompromitált azonosítókat.

2. definíció (Plauzibilis útvonal). *Vegyünk egy redukált* $conf = (G_{L_3}(E, V, \mathcal{L}), V^*)$ *konfigurációt.* $(L_1, L_2, \ldots, L_n)$ *egy plauzibilis útvonal* $conf$ *konfigurációban, ha* $L_1, L_2, \ldots, L_n$ *azonosítók mindegyike különböző, és létezik* $v_1, v_2, \ldots, v_k$ $(2 \leq k \leq n)$ *csúcsok egy olyan sorozata* V*-ben, valamint létezik* $j_1, j_2, \ldots, j_k$ *pozitív egész számok egy olyan sorozata, hogy*

1. $j_1 + j_2 + \ldots + j_k = n$,
2. $(v_i, v_{i+1}) \in E$ $(1 \leq i < k)$,
3. $\{L_{J_i+1}, L_{J_i+2}, \ldots, L_{J_i+j_i}\} \subseteq \mathcal{L}(v_i)$ $(1 \leq i \leq k)$, *ahol* $J_i = j_1 + j_2 + \ldots + j_{i-1}$, *ha* $i > 1$ *és* $J_i = 0$ *ha* $i = 1$. ♣

Szemléletesen a fenti definíció jelentése a következő. $L_1, L_2, \ldots, L_n$ útvonal particionálható k darab részsorozatokra olyan módon (1. feltétel), hogy az így kapott részsorozatok elemeinek halmaza egy-egy csúcshoz hozzárendelt azonosítók egy részhalmaza a gráfban (3. feltétel), és ezen csúcsok egy útvonalat alkotnak ebben a gráfban (2. feltétel). Triviálisan látható az is, hogy ha egy útvonal plauzibilis, akkor annak nyilván minden szomszédos partíciójához szükségszerűen olyan csúcsok rendelhetőek, amelyek közül legfeljebb csak egy tartozhat támadó csomóponthoz. Plauziblis és nem plauzibilis útvonalakat szemléltet a 4.1. ábra.



4.1. ábra. Redukált konfiguráción értelmezett plauzibilis útvonalak. Látható, hogy a (A, G, C, F), (A, E, C, F), (A, B, D, E, F) útvonalak plauzibilis útvonalak, amelyek particionálásai rendre $(A)|(G)|(C)|(F)$, $(A)|(E)|(C)|(F)$, $(A)|(B)|(D, E)|(F)$. Nem plauzibilis útvonalakra példa a (A, D, C, G, D), (A, G, F) vagy az (A, E, F) sorozatok.

Fontos megfogalmazni, hogy pontosan mit jelent az, hogy egy útvonal nem plauzibilis, mivel az egyes protokollok biztonságának és esetleg nem biztonságosságának a bizonyításánál ezt fogjuk felhasználni.

2. állítás. *Egy* $(\ell_1, \ldots, \ell_p)$ *útvonal, ahol* ℓ_1, ℓ_p *rendre a forrás és cél azonosítói, nem plauzibilis* $(G_{L_3}(E, V, \mathcal{L}), V^*)$ *redukált konfigurációban, ha az alábbi állítások valamelyike teljesül.*

1. *Létezik* j $(1 \leq j \leq p-1)$, *amelyhez nem létezik* $v, v' \in V \setminus V^*$, *hogy* $\ell_j \in \mathcal{L}(v)$, $\ell_{j+1} \in \mathcal{L}(v')$ *és* $(v, v') \in E$.

2. *Létezik* j $(1 \leq j \leq p-2)$ *és* q $(1 \leq q \leq p-j-1)$, *hogy* $\ell_j \in \mathcal{L}(v)$, $\{\ell_{j+1}, \ldots, \ell_{j+q}\} \subseteq L^*$, $\ell_{j+q+1} \in \mathcal{L}(v')$, *ahol* $v, v' \in V \setminus V^*$, *és nem létezik* $v^* \in V^*$, *hogy* $(v, v^*) \in E$ *és* $(v^*, v') \in E$. ♢

Bizonyítás Az 1. állítás szerint létezik két olyan az útvonalban szomszédos azonosító, amelyek megbízható csomópontoknak megfeleltetett csúcsokhoz tartoznak, viszont ezen csúcsok nem szomszédosak a gráfban. A két azonosító külön-külön egy-egy partíciót alkot.

A 2. állítás viszont azt mondja, hogy létezik egy olyan azonosító az útvonalban, ami megbízható csomóponthoz tartozó csúcshoz tartozik, azt követően viszont kompromitált azonosítók következnek egészen ℓ_{j+q}-ig, de ℓ_{j+q+1} ismét megbízható csomóponthoz tartozó csúcs azonosítója. Ennek ellenére nem létezik a két csúcsnak egy olyan közös szomszédja, amely támadó csomóponthoz tartozik. Itt ℓ_j és ℓ_{j+q+1} két külön partíciót alkot, míg $\{\ell_{j+1}, \ldots, \ell_{j+q}\}$ egy köztük levő másik partíciót. ■

Felmerülhet a kérdés, hogy miért definiáltuk a plauzibilis útvonalat és miért pont egy redukált konfiguráción. Ahhoz, hogy a szimulációs modellben definiálhassuk a valós világ modellt, szükséges az elviselhető támadások (tolerable imperfections) specifikációja. Természetesen léteznek már Aktív-1-1 támadó jelenlétében is olyan támadások, amelyek ellen nem tudunk vagy túl költséges védekezni. Például egy támadó képes minden rajta áthaladó üzenetet explicit vagy implicit módon eltávolítani, valamint képes ezeket módosítani, késleltetni, és újakat beszúrni. Ezen tulajdonságokkal tehát bármely ideális világbeli támadó rendelkezik, csakúgy mint bármely valós világbeli is. Aktív-y-x támadómodell esetén ezeken kívül még fellép két olyan támadási lehetőség, amelyeket szintén tolerálhatóaknak célszerű definiálni:

- Egy *nem* redukált konfiguráció esetén lehetséges olyan gráftopológia, hogy több támadó csomópont szomszédos egymással. Ekkor intuitíve látható, hogy a szomszédos támadók kooperatíven képesek olyan támadást véghez vinni, amit nem tudunk kivédeni (például ha kevesebb kompromitált azonosítót tesznek be az rreq és rrep üzenetekbe mint ahány közbenső támadó csomóponton a kérés valójában átmegy, feltéve, hogy ezen támadó csomópontok az útvonal szerint is szomszédosak).

- Egy támadó a birtokában levő bármely azonosítót tetszőleges sorrendbe beszúrhatja egy kérésbe, melyet hitelesíteni is tud, mivel feltevésünk szerint rendelkezik az azonosítóval kapcsolatos minden információval. Így képes az álcázásra, ami ellen az útvonalválasztó protokoll szintjén nem tudunk védekezni.

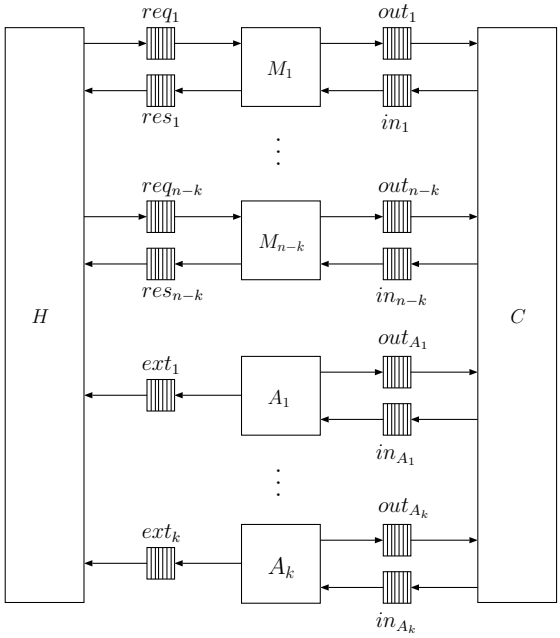
A fentiek tükrében látszik a plauzibilis útvonal definíciójának intuitív értelme.

4.2. A szimulációs paradigma adaptációja

A következőkben részletesebben leírom a forrás alapú útvonalválasztó protokollok valós és ideális világ modelljeit, amelyeknek alapjai megtalálhatóak [3]-ban.

4.2.1. A valós világ modell

A valós világ modell megfelel egy redukált $conf = (G_{L_3}(E, V, \mathcal{L}), V^*)$ konfigurációnak és egy $\mathcal{A}$ támadónak, ahol $\mathcal{A}$ egy Aktív-*y*-*x* támadó. Jelöljük ezt a rendszert $sys^{\mathsf{real}}_{conf,\mathcal{A}}$-val. Továbbá legyen $k = |V^*|$ és $n = |V|$. $sys^{\mathsf{real}}_{conf,\mathcal{A}}$ a $\{M_1, M_2, \ldots, M_{n-k}, H, A, \ldots, A_k, C\}$ halmazból áll, ahol az egyes halmazelemek egy-egy Turing-gépet jelentenek, amelyek átmeneti tárakon (buffer) keresztül kommunikálnak egymással. Minden egyes M_j megfelel egy megbízható csomópontnak, ami megfelel $G_{L_3}(E, V, \mathcal{L})$ gráf egy csúcsának. H a megbízható felhasználó által futtatott protokoll egy absztrakciója, A_i $(1 \leq i \leq k)$ jelenti a támadó csomópontokat, amelyek szintén megfelelnek G_{L_3} gráf csúcsainak. C Turing-gép reprezentálja a rádiókapcsolatokat, vagyis G_{L_3} éleit. Feladata a hozzá kapcsolódó átmeneti tárak közötti üzenetcsere megvalósítása. Minden gép probabilisztikus működésű. A valós világbeli kapcsolatokat ábrázolja az egyes gépek között a 4.2. ábra.



4.2. ábra. Kapcsolatok a valós világ modellben ($sys^{\mathsf{real}}_{conf,\mathcal{A}}$) forrás alapú útvonalválasztásnál.

A gépek kezdetben bemeneti adatokkal inicializálódnak, amelyek meghatározzák a gépek kezdeti állapotát. Ezenkívül mindegyikük bemenetként kap véletlen értékeket (pl. egy pénzérme feldobásainak eredményét). Az inicializálódás után megkezdődhet a számítási szakasz. Az egyes gépek reaktív működésűek, vagyis aktiválni kell őket ahhoz, hogy számításokat végezzenek. Mikor egy gép aktiválódik, beolvassa a hozzá tartozó bemeneti tár tartalmát, feldolgozza azt, frissíti a belső állapotát, generál valamilyen kimeneti adatot, amit elhelyez a hozzá tartozó kimeneti tárban, majd visszatér inaktív módba várakozni a következő aktiválásig. A bemeneti tárból történő olvasás eltávolítja az üzenetet a bemeneti tárból, míg a kimeneti tárba írás hozzáfűzi az üzenetet a tár aktuális tartalmához. A gépeket *menetek*ben (rounds) aktiválja egy feltételezett *ütemező*. Minden menetben az ütemező az egyes gépeket a következő sorrend szerint aktiválja: $H, M_1, \ldots, M_{n-k}, A_1, \ldots, A_k, C$. Ez azt jelenti, hogy először H kerül aktív állapotba, majd minden gép akkor, amikor az őt megelőző visszakerült

inaktív állapotba. A menetnek akkor van vége, amikor C is visszakerült inaktív állapotba.

A következőkben leírjuk részletesebben az egyes gépek működését:

- C **gép.** Ez a gép modellezi a rádiókommunikáció többesszórás jellegét. Aktiválása során először meghatároz egy véletlen sorrendet a hozzá tartozó bemeneti tárak között, majd ebben a sorrendben feldolgozza ezek tartalmát. Egy out_i $(1 \leq i \leq n-k)$ bemeneti tár feldolgozása out_i tartalmának beolvasásából és ennek másolásából áll in_j-be, ahol minden j-re $\mathcal{L}(v_j) \subseteq \mathcal{N}(v_i)$. Hasonlóan out_{A_i} feldolgozása esetén out_{A_i} tartalma átkerül in_j-be, ahol minden j-re $\mathcal{L}(v_j) \subseteq \mathcal{N}(v_i^*)$ $(1 \leq i \leq k, v_i^* \in V^*)$. Látható, hogy C működéséhez szükség van valamilyen véletlen értékre, jelölje ezt r_C.

- H **gép.** Ez a gép modellezi az útvonalválasztó protokoll feletti rétegben elhelyezkedő protokollokat, illetve végsősoron a megbízható csomópontok felhasználóit. H bármikor képes útvonalkeresést kezdeményezni azzal, hogy elhelyez valamelyik M_i req_i tárában egy (c_i, l_{tar}) üzenetet, ahol c_i az kérések megkülönböztetésére szolgáló azonosító, és $l_{tar} \in L$ a cél azonosítója. c_i H gép egy belső változója (állapotának egy része), amely kezdetben 0, és minden egyes alkalommal inkrementálódik, amikor egy req_i-be történik üzenet elhelyezés. Egy ilyen kérésre a válasz a res_i táron kerül visszaadásra. A válasz $(c_i, routes)$ formátumú, ahol c_i a megfelelő kérés azonosítója, míg $routes$ a visszaadott útvonalak halmaza. Néhány protokoll esetén $routes$ mindig szingleton, míg más esetben ez nem feltétlen igaz. Ha nem talált a protokoll futása során útvonalat, akkor $routes = 0$.

  req_i és res_i tárak mellett H hozzáfér ext_j tárakhoz is. Ez modellezi a támadó azon képességét, hogy egy külön csatornán keresztül bármikor képes egy megbízható protokollrésztvevő nevében egy útvonalfelderítés kezdeményezésére. Az ext_j tárból kiolvasott üzenetek formátuma (l_{ini}, l_{tar}), ahol $l_{ini}, l_{tar} \in L$ a kezdeményező és a cél azonosítói abban a kérésben, amit valamely támadó csomópont kezdeményezett. Amikor H értelmezi (l_{ini}, l_{tar}) üzenetet ext_j-ből, ellenőrzi először, hogy $l_{ini} \in \mathcal{L}(v_1, \ldots, v_{n-k})$. Ha az ellenőrzés sikertelen, akkor H figyelmen kívül hagyja az üzenetet, egyébként pedig elhelyez egy (c_i, l_{tar}) kérést req_i-ben, ahol i annak az M_i gépnek az indexe, amelynek azonosítója l_{ini}. Így H-nak szüksége van arra az információra, hogy melyik azonosító melyik M_i $(1 \leq i \leq n-k)$ géphez tartozik. Ezt bemenetként kapja az inicializációs fázis során.

- M_i **gép.** M_i működését elsősorban a modellezendő útvonalválasztó algoritmus határozza meg. M_i req_i bemeneti és res_i kimeneti tárakon keresztül kommunikál H-val. Ezeken a tárakon keresztül kap H-tól kéréseket útvonalkeresésre, és küldi el res_i-n a kérés eredményét a fentiek szerint.

  M_i a kimenő out_i és bemenő in_j tárakon kommunikál a protokoll más résztvevőjével. Mindkét tár tartalmazhat $(sndr, rcvr, msg)$ formátumú üzenetet, ahol $sndr \in L$ a küldő azonosítója, $rcvr \in L \cup \{*\}$ a címzett azonosítója ($*$ jelenti a többcímzettes üzenetet), és $msg \in \mathcal{M}$ pedig az aktuális protokollüzenet. Itt $\mathcal{M}$ az összes lehetséges protokollüzenet halmazát jelenti, amit a vizsgálandó útvonalválasztó protokoll határoz meg.

  Minden protokoll esetében szükséges annak meghatározása, hogy a protokollüzenet kérés vagy válasz. Ezért legyen $type : \mathcal{M} \to \{\mathsf{rreq}, \mathsf{rrep}\}$ egy olyan függvény, amely vissza-

adja bármely protokollüzenetre annak a típusát. Ezeken kívül szükséges még egy protokollüzenetből az útvonalkeresést kezdeményező és a cél azonosítójának a meghatározása is. Így legyen $tar : \mathcal{M} \to L$ és $ini : \mathcal{M} \to L$ olyan függvények, amelyek visszaadják egy üzenetben szereplő kezdeményező és cél azonosítóját.

Elsőként M_i aktiválódása után beolvassa a req_i-ben szereplő kéréseket. Minden egyes H-tól kapott (c_i, l_{tar}) kérésre generál egy msg útvonalkérés üzenetet, majd frissíti belső állapotát az útvonalválasztó protokoll szerint. Végül elhelyezi $(\mathcal{L}(v_i), *, msg)$ üzenetet out_i tárban.

Miután minden kérést sikeresen feldolgozott req_i tárból, M_i beolvassa in_i tartalmát. in_i-ben fellelt üzenetek kivételi sorrendjükben kerülnek feldolgozásra. Minden $(sndr, rcvr, msg)$ üzenetre M_i ellenőrzi, hogy $sndr \in \mathcal{N}(v_i)$ és $rcvr \in \{\mathcal{L}(v_i), *\}$. Ha ezen ellenőrzések sikertelenek, akkor M_i figyelmen kívül hagyja msg-t, egyébként feldolgozza azt, és frissíti belső állapotát az útvonalválasztó protokoll szerint. Ezekután az alábbi eseteket tudjuk megkülönböztetni:

$type(msg) =$ rreq : Ha $tar(msg) \notin \mathcal{L}(v_i)$ (vagyis M_i nem az aktuális útvonalkeresés célja), akkor msg tartalmától valamint M_i belső állapotától függően az eldobásra kerülhet, és ekkor M_i nem generál kimeneti üzenetet. Egyébként M_i többesszórással továbbítja msg-t. Ekkor M_i kimenetként egy msg' üzenetet generál (pl. hozzáfűzi saját azonosítóját az útvonalhoz), és elhelyezi $(\mathcal{L}(v_i), *, msg')$-t out_i-ben.

Ha $tar(msg) \in \mathcal{L}(v_i)$ (vagyis M_i az útvonalkeresés célja), akkor M_i elfogadhatja a kérést, és ekkor generál egy msg' válaszüzenetet a protokollnak megfelelően, majd elhelyezi $(\mathcal{L}(v_i), l, msg')$ üzenetet out_i-be, ahol $l \in \mathcal{N}(v_i)$ az útvonalban szereplő első olyan csomópont azonosítója, ahova a válaszüzenetnek elsőként el kell jutnia. A legtöbb protokollban a válaszüzenetek csak egyetlen útvonalat tartalmazhatnak, és a válasznak ennek a fordítottján kell eljutnia a kezdeményezőhöz. Ekkor l egyértelműen meghatározható abból az útvonalból, amit a válaszüzenet szállít. Máskülönben ha a válaszüzenet több útvonalat tartalmaz, vagy pedig nem a fordított útvonalon kell a válasznak visszajutnia, akkor az útvonalválasztó protokoll specifikálja, hogy a cél miként határozza meg l-t egyértelműen a beérkezett összes kérésüzenet alapján.

$type(msg) =$ rrep : Ha $ini(msg) \notin \mathcal{L}(v_i)$ (vagyis M_i nem az msg-hez tartozó útvonalkeresés kezdeményezője), akkor msg tartalmától valamint M_i aktuális belső állapotától függően msg eldobásra kerülhet, és ekkor nem generál M_i válaszüzenetet. Máskülönben M_i továbbíthatja msg-t, amely esetben M_i a megfelelő msg' protokollüzenetet generálja, és elhelyezi $(\mathcal{L}(v_i), l, msg')$ üzenetet out_i-ben, ahol $l \in \mathcal{N}(v_i)$ az útvonalban szereplő első olyan csomópont azonosítója, ahova a válaszüzenetnek elsőként el kell jutnia. Amint azt fentebb tárgyaltuk, az útvonalválasztó protokollnak specifikálnia kell M_i számára l meghatározásának módját. Ha $ini(msg) \in \mathcal{L}(v_i)$ (vagyis M_i az útvonalkeresés kezdeményezője), akkor M_i generálhat H számára egy választ, amit res_i-n keresztül juttat el neki.

- A_j **gép**. Ezek a gépek reprezentálják a támadót, illetve G_{L_3} gráf szerint az összevont

támadó csomópontokat. Feltételezzük, hogy az egyes csomópontok nem képesek egymással rejtett csatornán keresztül kommunikálni. A kommunikációs képességeket figyelembe véve ezek a gépek nem tudnak többet mint az M_i gépek, tehát in_{A_j} tárból olvasnak és out_{A_j} tárba írnak, ahogyan ezt M_i gépek teszik in_{M_i} és out_{M_i} tárak esetén. Viszont A_j gépek nem a protokoll szerint működnek. Semmilyen megszorítást nem teszünk A_j működésére, csak azt, hogy egy t biztonsági paraméter és a hálózat n méretének függvényében polinomiális idejű legyen. Így tetszőleges támadásokat figyelembe vehetünk az analízis során. Vagyis A_j akár késleltetheti vagy törölheti azon üzeneteket, amiket egyébként a protokoll működése alapján továbbküldene. Ezeken kívül még módosíthat és beszúrhat üzeneteket.

Ezeken felül A_j-k egy külön csatornán küldhetnek útvonalkeresést kezdeményező üzeneteket ext_j táron keresztül, így megadhatják azt, hogy ki kezdeményezi az útvonalkérést és melyik cél felé. ext_j-n a támadó egyszerre több független üzenetet is küldhet, de mivel a támadó nem adaptív, ezért egy kérés eredményét nem használhatja fel egy másik kéréshez.

Amint a leírásból látható minden M_i-nek tudnia kell a saját és szomszédai azonosítóját (G_{L_3} szerint). Így M_i megkapja $\mathcal{L}(v_i)$-t, $\mathcal{N}(v_i)$-t, valamint A_j $\mathcal{L}(a_j)$-t és $\mathcal{N}(a_j)$-t az inicializációs fázisban.

Ezeken felül az egyes gépeknek szükségük lehet valamilyen kriptográfiai „kellékre” (pl. publikus és privát kulcs) az útvonalválasztó protokolltól függően. Ezen „kellékek” eloszlását a következőképpen modellezzük. Legyen adott egy I függvény, amely bemenetként vár egy r_I véletlent, és ebből előállít egy $I(r_I) = (\kappa_{pub}, \kappa_1, \ldots, \kappa_{n-k}, \nu_1, \ldots, \nu_\ell)$ $(1 \leq \ell)$ vektort. κ_{pub} komponens olyan publikus információ, amit minden A_j és M_i ismer. Minden $1 \leq i \leq n-k$-ra κ_i csak M_i számára lesz ismert, míg $1 \leq j \leq \ell$-re ν_j minden A_m számára ismert, ahol $1 \leq m \leq k$. Fontos, hogy az inicializációs fázisban modellezhető a kriptográfiai „kellékek” privát cseréje mind szimmetrikus mind asszimmetrikus rejtjelezés esetén. Asszimmetrikus esetben κ_{pub} tartalmazza minden eszköz publikus kulcsát, míg κ_i $(1 \leq i \leq n-k)$ tartalmazza a megbízható eszközök és ν_j $(1 \leq j \leq \ell)$ a támadó eszközök privát kulcsait. Szimmetrikus esetben κ_{pub} üres, és κ_i valamint ν_j tartalmazzák a privát kulcsokat, amelyek csak v_i és a_m $(1 \leq m \leq k)$ eszközök által ismertek.

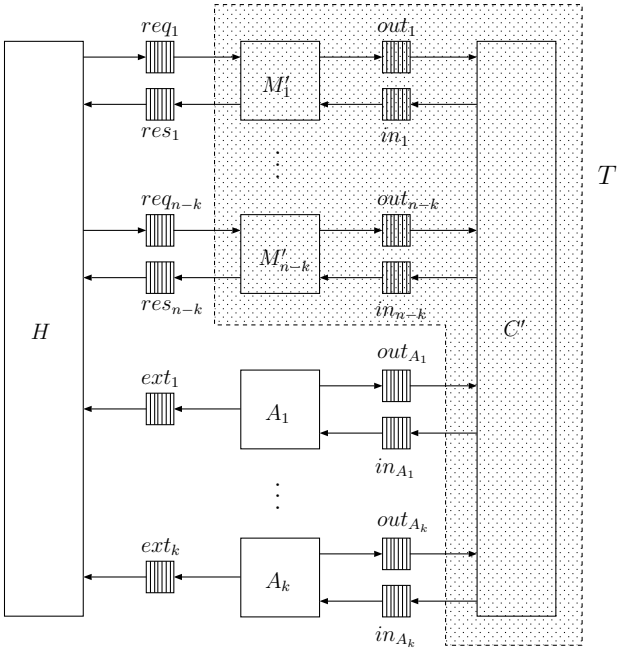
Végül minden M_i és A_j bemenetként kap valamilyen véletlent az inicializációs fázisban. M_i véletlen bemenete r_i, míg A_j véletlen bemenete r_{A_j}.

A számításnak akkor van vége, amikor H a leállási állapotába kerül. A mi esetünkben (amikor A_j-k több egymástól független kérésüzenetet küldhetnek H számára) ez akkor történik meg, amikor H beolvasta mindegyik res_i bemeneti tárból a választ, ami megfelel a req_i-ben elhelyezett kérésnek.

$sys^{\mathsf{real}}_{conf,\mathcal{A}}$ kimenete a válaszüzenetben visszaadott útvonalak halmaza. Jelöljük ezt a kimenetet $view^{\mathsf{real}}_{conf,\mathcal{A}}(r)$-el, ahol $r = (r_I, r_1, \ldots, r_{n-k}, r_{A_1}, \ldots, r_{A_k}, r_C)$. Ezenkívül $view^{\mathsf{real}}_{conf,\mathcal{A}}$ jelölje a $view^{\mathsf{real}}_{conf,\mathcal{A}}(r)$ véletlen valószínűségi változót, amikor r-t egyenletesen választjuk.

4.2.2. Az ideális világ modell

Az ideális világ modell megfelel egy redukált $conf = (G_{L_3}(E, V, \mathcal{L}), V^*)$ konfigurációnak és egy $\mathcal{A}$ támadónak, ahol $\mathcal{A}$ egy Aktív-y-x támadó. Jelöljük ezt a rendszert $sys^{\mathsf{ideal}}_{conf,\mathcal{A}}$-el. Továbbá legyen $k = |V^*|$ és $n = |V|$. $sys^{\mathsf{ideal}}_{conf,\mathcal{A}}$ a $\{H, T, A_1, \ldots, A_k\}$ halmazból áll, ahol az egyes halmazelemek egy-egy Turing-gépnek felelnek meg, ahol H ugyanazt jelenti mint valós világ modell esetben, míg T modellezi a protokoll ideális működését. A_i $(1 \leq i \leq k)$ jelenti az ideális világbeli támadónak megfelelő csomópontokat. T és A_i-k probabilisztikus működésűek. Az ideális világbeli kapcsolatokat ábrázolja az egyes gépek között a 4.3. ábra.



4.3. ábra. Kapcsolatok az ideális világ modellben ($sys^{\mathsf{ideal}}_{conf,\mathcal{A}}$) forrás alapú útvonalválasztásnál.

Egy ideális útvonalválasztó szolgáltatásnak soha nem szabad nem plauzibilis útvonalat visszaadni. Így T szerepe a valós hálózat viselkedésének emulálása, és annak biztosítása, hogy a válaszüzenetek, amelyek nem plauzibilis útvonalat tartalmaznak, kiszűrésre kerülnek. Ezt a következő módon érjük el: T belső struktúrája megegyezik a valós világ modellel (vagyis T futtatja M'_i és C' Turing-gépeket, amelyek ugyanúgy működnek mint M_i és C a valós világ modellben). Így biztosítjuk azt, hogy T emulálni tudja egy valós hálózat működését. Másrészről, mivel C'-t $conf$-al inicializáljuk, ezért az képes megjelölni azokat a válaszüzeneteket, amelyek nem plauzibilis útvonalat tartalmaznak $conf$-ban. M'_i gépek minden üzenetet ugyanolyan módon dolgoznak fel, függetlenül attól, hogy a válasz plauzibilis vagy nem plauzibilis útvonalat tartalmaz (vagyis figyelmen kívül hagyják a korrupciós jelzőt). A kezdeményező először elvégezi a protokoll által specifikált összes ellenőrzést, majd ha ezek sikeresek, akkor ellenőrzi a korrupciós jelzőt. Ha ez be van állítva, akkor eldobja az üzenetet, ha nincs, akkor feldolgozza az üzenetet (visszaadja az útvonalat H számára). Ez azt jelenti, hogy az ideális világ modellben minden olyan üzenet kiszűrésre kerül a kezdeményező által, amely nem plauzibilis útvonalat tartalmaz.

Mivel T emulálja a valós hálózat működését, ezért azokat a támadásokat, amelyeket meg-

engedünk T-ben, megengedünk a valós világ esetén is. Így a mi megközelítésünkben egy ideális világbeli támadó képességei és lehetőségei megegyeznek egy valós világbeli támadóéval. Így mindkettőt $\mathcal{A}$-val jelöljük, az általuk kontrollált csomópontokat pedig A_j-vel $(1 \leq j \leq k)$

Úgy mint a valós világ esetében, az egyes gépek itt is reaktív módon működnek. Mindegyik gépet egy feltételezett, hipotetikus ütemező aktivál menetenként a következő sorrendben: H, T, A, T. Látható, hogy T kétszer aktiválódik egy menetben. A tárak úgy működnek mint valós esetben.

H és A_j ugyanúgy működnek mint a valós világ modellben. A következőkben részletesebben leírjuk T működését:

- T **gép**: T futtatja $\{M'_1, M'_2, \ldots, M'_{n-k}, C'\}$ gépek halmazát, ahol M'_i és C' nagyjából ugyanaz mint M_i és C a valós esetben, annyi különbséggel, hogy M'_i képes a korrupciós jelzővel ellátott üzeneteket feldolgozni, míg C' pedig képes a válaszokat korrupciós jelzővel ellátni.

  Minden egyes menetben, amikor T gépet először aktiválják, az aktiválja $M'_1, M'_2, \ldots, M'_{n-k}$ gépeket ebben a sorrendben, majd visszatér inaktív állapotba várva a második aktiválásra. Második aktiválása során futtatja C'-t. Amikor C' befejezte a feladatát, T visszatér inaktív állapotba (ez jelzi a menet végét).

  Az in'_i $(1 \leq i \leq n-k)$-be C' által elhelyezett üzenetek $(sndr, rcvr, (msg, cf))$ formátumúak, ahol $sndr, rcvr$ és msg ugyanúgy definiálhatóak mint a valós világ modellben, és $cf \in \{\top, \bot\}$ a *korrupciós jelző*, ami jelzi, hogy msg korrupt $(\top)$ vagy sem $(\bot)$. Az üzenetek, amelyek out_i $(1 \leq i \leq n-k)$, out_{A_j} és in_{A_j} $(1 \leq j \leq k)$ tárakba kerülnek azonos formátumúak mint a valós esetben (nincsenek korrupciós jelzővel megjelölve). Fontos megjegyezni, hogy A_j kimeneti és bemeneti tárai szintén a valós esettel egyeznek meg, így egy valós világbeli támadó könnyen „áthelyezhető" az ideális világba.

  Amikor M'_i gép beolvassa $(sndr, rvcr, (msg, cf))$ üzenetet in'_i-ből, ellenőrzi, hogy $sndr \in \mathcal{N}(v_i)$ és $rcvr \in \{\mathcal{L}(v_i), *\}$. Ha ezek sikeresek, akkor elvégzi a protokoll által megkövetelt ellenőrzéseket (MAC, aláírás ellenőrés, stb.). Ha $type(msg) = \mathsf{rrep}$ és $ini(msg) \in \mathcal{L}(v_i)$, akkor M'_i ellenőrzi, hogy $cf = \top$. Ha igaz, akkor M'_i eldobja msg üzenetet, egyébként folytatja a feldolgozását. Ha $type(msg) \neq \mathsf{rrep}$ vagy $ini(msg) \notin \mathcal{L}(v_i)$, akkor cf nem kerül ellenőrzésre. M'_i által generált üzenetekre nem kerülnek korrupciós jelzők, ezek az out_i tárba kerülnek.

  C' gép C géphez hasonlóan minden egyes M'_i-nek és A-nak kimeneti tárát átmásolja a megfelelő szomszédok bemeneti tárába. Viszont minden egyes $(sndr, rcvr, msg)$ üzenet másolása előtt bármely in'_i tárba C'-nek kötelező egy korrupciós jelzőt elhelyezni msg-n. Ezt a következőképpen teszi.

  - ha $type(msg) = \mathsf{rreq}$, akkor C' beállítja cf jelzőt $\bot$-ra.
  - ha $type(msg) = \mathsf{rrep}$ és az összes msg-ben szereplő útvonal plauzibilis G'_{L_3}-ban, akkor C' beállítja cf jelzőt $\bot$-ra.
  - minden más esetben C' cf-et $\top$-ra állítja.

  C' nem tesz az in_{Aj}-be helyezendő üzenetekre korrupciós jelzőt.

A számítás kezdete előtt minden gép bemenetként kap véletlen értékeket. H és A ugyanazokat a kezdeti értékeket kapja bemenetként mint a valós világ modellben. T inicializációja M_i' és C' inicializációjából áll. Minden M_i' és C' ugyanazokat a kezdeti bemeneti értékeket kapja mint M_i' és C valós esetben.

A számításnak akkor van vége, amikor H a leállási állapotába kerül. A mi esetünkben (amikor A_j-k több egymástól független kérésüzenetet küldhetnek H számára) ez akkor történik meg, amikor H beolvasta mindegyik res_i bemeneti tárból a választ, ami megfelel a req_i-ben elhelyezett kérésnek.

$sys^{\mathsf{ideal}}_{conf,\mathcal{A}}$ kimenete a válaszüzenetben visszaadott útvonalak halmaza. Jelöljük ezt a kimenetet $view^{\mathsf{ideal}}_{conf,\mathcal{A}}(r')$-val, ahol $r' = (r'_I, r'_1, \ldots, r'_{n-k}, r'_{A_1}, \ldots, r'_{A_k}, r'_C)$. Ezenkívül $view^{\mathsf{real}}_{conf,\mathcal{A}}$ jelölje a $view^{\mathsf{ideal}}_{conf,\mathcal{A}}(r')$ véletlen valószínűségi változót, amikor r'-t egyenletesen választjuk.

4.3. A biztonságos útvonalválasztás definíciói

A következőkben megadom a biztonságos útvonalválasztás definícióit, de előbb röviden bemutatom a megkülönböztethetetlenség számításelméleti definícióit [9]. *Egy biztonságos protokoll valós világ modelljében elvárjuk, hogy az ne adjon vissza nem plauzibilis útvonalat a szimuláció során.* Ez viszont nem mindig tartható, hiszen elenyésző de pozitív valószínűséggel a támadó általában mindig képes a felhasznált kriptográfiai primitívet megtörni (pl. megtippel egy digitális aláírást), vagyis a biztonságosság szemléletesen $view^{\mathsf{real}}_{conf,\mathcal{A}}$ és $view^{\mathsf{ideal}}_{conf,\mathcal{A}}$ eloszlások megkülönböztethetetlenségére vezethető vissza.

$\mathcal{D}$ és $\mathcal{D}'$ diszkrét valószínűségi eloszlások eloszlásfüggvényét jelölje $P_{\mathcal{D}}(x)$ valamint $P_{\mathcal{D}'}(x)$. $\mathcal{D}$ és $\mathcal{D}'$ eloszlások

- megegyeznek, ha $P_{\mathcal{D}}(x) = P_{\mathcal{D}'}(x)$ minden x-re.
- statisztikailag megkülönböztethetetlenek, ha $\sum_x |P_{\mathcal{D}}(x) - P_{\mathcal{D}'}(x)|$ (vagyis a két eloszlás L_1 távolsága) elhanyagolható.

Jelölje $x \leftarrow \mathcal{D}$ azt az eseményt, hogy x minta $\mathcal{D}$ eloszlásból származik, $Z(x) = 1$ pedig azt az eseményt, hogy $Z : \{0,1\}^\infty \to \{0,1\}$ algoritmus t véges erőforrással (számítási lépések száma, orákulum kérések száma, stb.) $\mathcal{D}$ eloszlásra dönt x minta esetén, míg ha $Z(x) = 0$, akkor Z algoritmus $\mathcal{D}'$ eloszlásra dönt. $\mathcal{D}$ és $\mathcal{D}'$ eloszlás számításelméleti szempontból (t, ε) *megkülönböztethetetlen*, ha

$$\max_Z | \mathbf{P}\{Z(x) = 1 \mid x \leftarrow \mathcal{D}\} - \mathbf{P}\{Z(x') = 1 \mid x' \leftarrow \mathcal{D}'\} | \leq \varepsilon$$

ahol a maximumot a megkülönböztető algoritmusok felett képezzük. Legyen $d_t(\mathcal{D}, \mathcal{D}')$ a fenti képletben definiált maximum. Bizonyítható, hogy $d_t(\mathcal{D}, \mathcal{D}')$ távolság, valamint ha $t = \infty$, akkor $d_t(\mathcal{D}, \mathcal{D}') = V(\mathcal{D}, \mathcal{D}')$, ahol $V(\mathcal{D}, \mathcal{D}')$ $\mathcal{D}$ és $\mathcal{D}'$ eloszlások statisztikai távolsága (vagyis a két eloszlás L_1 távolságának a fele).

Legyen $\mathcal{D}_n$ és $\mathcal{D}'_n$ két eloszlásegyüttes, ahol n jelentheti például valamilyen biztonsági paraméter méretét. $\mathcal{D}_n$ és $\mathcal{D}'_n$ valószínűségi eloszlásegyüttesek *hatékonyan (polinomiálisan)*

megkülönböztethetetlenek, ha bármely $p(n)$ polinom és elegendően nagy n esetén

$$\max_Z \mid \mathbf{P}\{Z(x) = 1 \mid x \leftarrow \mathcal{D}_n\} - \mathbf{P}\{Z(x') = 1 \mid x' \leftarrow \mathcal{D}'_n\} \mid < \frac{1}{p(n)}$$

vagyis a bal oldalon szereplő megkülönböztetés mértéke elhanyagolható n függvényében. [1]

A megkülönböztetés szempontjából három biztonságfogalmat tudunk megfogalmazni.

3. definíció. *Egy útvonalválasztó protokoll tökéletesen biztonságos, ha bármely $conf$ konfigurációhoz és bármely valós világbeli $\mathcal{A}$ támadóhoz létezik egy olyan ideális világbeli $\mathcal{A}'$ támadó, hogy $view^{\mathsf{ideal}}_{conf,\mathcal{A}}$ és $view^{\mathsf{real}}_{conf,\mathcal{A}}$ eloszlások megegyeznek.* ♣

4. definíció. *Egy útvonalválasztó protokoll statisztikailag biztonságos, ha bármely $conf$ konfigurációhoz és bármely valós világbeli $\mathcal{A}$ támadóhoz létezik egy olyan ideális világbeli $\mathcal{A}'$ támadó, hogy $view^{\mathsf{ideal}}_{conf,\mathcal{A}}$ és $view^{\mathsf{real}}_{conf,\mathcal{A}}$ eloszlások statisztikailag megkülönböztethetetlenek.*♣

5. definíció. *Egy útvonalválasztó protokoll (számításelméleti szempontból) biztonságos, ha bármely $conf$ konfigurációhoz és bármely valós világbeli $\mathcal{A}$ támadóhoz létezik egy olyan ideális világbeli $\mathcal{A}'$ támadó, hogy $view^{\mathsf{ideal}}_{conf,\mathcal{A}}$ és $view^{\mathsf{real}}_{conf,\mathcal{A}}$ eloszlások számításelméleti szempontból megkülönböztethetetlenek.* ♣

A tökéletes biztonság implikálja a statisztikai biztonságot, ami pedig magába foglalja a számításelméleti szempontból biztonságosság fogalmát.

Szemléletesen egy protokoll tökéletesen biztonságos, ha igaz, hogy amit egy valós világbeli támadó képes elérni a protokoll futása során, azt ugyanúgy képes egy ideális világbeli támadó is. Ebből az is következik, hogy egy valós világbeli támadó nem képes azt elérni, hogy H egy nem plauzibilis útvonalat kapjon vissza, hiszen egy ideális világbeli támadó definíciója alapján nem képes erre. A statisztikailag biztonságos protokollokra ugyanez érvényes, csak 1-hez közeli valószínűséggel. Biztonságos protokollok esetén a valós és ideális világbeli modellek kimeneteit leíró valószínűségi eloszlások nem különböztethetőek meg hatékonyan, tehát egy gyakorlati alkalmazás számára a valós és ideális modellek ekvivalenseknek tekinthetőek.

Ahhoz, hogy bebizonyíthassuk egy adott útvonalválasztó protokoll biztonságát, találnunk kell minden valós világbeli $\mathcal{A}$ támadóhoz egy olyan $\mathcal{A}'$ ideális világbeli támadót, hogy az 5. definíció teljesüljön. A mi modellünkben egy lehetséges eset amikor $\mathcal{A} = \mathcal{A}'$. Ennek oka, hogy bármely $conf$ konfiguráció esetén $sys^{\mathsf{real}}_{conf,\mathcal{A}}$ működése könnyen *szimulálható* $sys^{\mathsf{ideal}}_{conf,\mathcal{A}}$ működésével, feltéve, hogy a két rendszer ugyanazon r véletlennel lett inicializálva. Ha $sys^{\mathsf{ideal}}_{conf,\mathcal{A}}$ egy üzenetet sem dob el a beállított korrupciós jelző miatt, akkor $sys^{\mathsf{ideal}}_{conf,\mathcal{A}}$ és $sys^{\mathsf{real}}_{conf,\mathcal{A}}$ ugyanolyan működésű rendszerek, vagyis belső állapotuk minden menetben azonos. Ezért $view^{\mathsf{ideal}}_{conf,\mathcal{A}}(r) = view^{\mathsf{real}}_{conf,\mathcal{A}}(r)$ minden r esetén, és így $view^{\mathsf{ideal}}_{conf,\mathcal{A}}$ és $view^{\mathsf{real}}_{conf,\mathcal{A}}$ azonos eloszlású, vagyis a protokoll tökéletesen biztonságos. Ha egy üzenet $sys^{\mathsf{ideal}}_{conf,\mathcal{A}}$ rendszerben eldobásra kerül a korrupciós jelzője miatt, míg $sys^{\mathsf{real}}_{conf,\mathcal{A}}$-ben a protokoll minden verifikációjának eleget téve nem kerül eldobásra, akkor $sys^{\mathsf{ideal}}_{conf,\mathcal{A}}$ és $sys^{\mathsf{real}}_{conf,\mathcal{A}}$ belső állapotai nem biztos, hogy

[1] A képlet a következőkből adódik. Szemléletesen egy ritka esemény ritkán fordul elő még akkor is, ha vizsgált számszor megismételjük a véletlen kísérletet. Egy hatékony algoritmus lépéseinek a száma gyakorlati szempontból $p(n)$. Definíció szerint egy elhanyagolható $f : \mathbb{N} \to \mathbb{N}$ függvényre igaz, hogy $(1 - f(n))^{p(n)} > 0.99$ bármely $p(n)$ polinom esetén.

minden menetben meg fognak egyezni, és a leállási állapot is különböző lehet. Ezt nevezzük *szimulációs hibának*. Ekkor előfordulhat, hogy $view^{\mathsf{ideal}}_{conf,\mathcal{A}}(r) \neq view^{\mathsf{real}}_{conf,\mathcal{A}}(r)$, ugyanakkor a rendszer még statisztikailag biztonságos lehet, ha a szimulációs hiba csak elhanyagolható valószínűséggel következik be, vagyis elhanyagolható valószínűséggel kerül eldobásra egy üzenet a korrupciós jelzője miatt $sys^{\mathsf{ideal}}_{conf,\mathcal{A}}$-ben. A (számításelméleti szempontból) biztonságosságot indirekt módon tudjuk bizonyítani. Ehhez először feltesszük, hogy létezik olyan hatékony Z algoritmus, amely képes megkülönböztetni $view^{\mathsf{ideal}}_{conf,\mathcal{A}}$ és $view^{\mathsf{real}}_{conf,\mathcal{A}}$ eloszlásokat. Ekkor viszont Z felhasználható egy olyan támadó megkonstruálásához, aki képes a felhasznált kriptográfiai primitívet (pl. digitális aláírás) feltörni. A kriptográfiai primitívek biztonsága az erősen egyirányú függvény létezésén alapul.

5. fejezet

Az Ariadne biztonsága

Az Ariadne [4] egy DSR-en alapuló „biztonságos” útvonalválasztó protokoll. Először a 5.1. részben bemutatom az Ariadne egy változatának működését, majd a 5.2. részben kis lépésekből felépített támadást konstruálok ellene, igazolva így, hogy az Ariadne nem biztonságos az előző fejezet modellje szerint.

Az Ariadne protokollt eredetileg a következő kriptográfiai primitívekkel javasolták a szerzői:

- Páronkénti titkos kulcsok (Pairwise shared keys) esetén szükéges egy inicializáló mechanizmus, amely szétosztja n csomópont esetén az $n(n-1)/2$ kulcsot.
- TESLA használatánál szintén szükséges az osztott titkos kulcsok szétosztása a kommunikáló csomópontok között, valamint a hitelesítő nyilvános TESLA kulcs szétosztása minden egyes résztvevőnek.
- digitális aláírásnál feltételezzük, hogy szétosztásra kerül egy hitelesítő kulcs minden egyes csomópont részére.

Az Ariadne-t eredetileg Aktív-1-x valamint Aktív-y-x támadómodellek esetén is biztonságosnak terveztét, ugyanakkor – mint később látni fogjuk – az általunk definiált modellben az Ariadne nem biztonságos már Aktív-1-x támadó esetén sem.

A kártékony üzenetbeszúrás és változtatás ellen MAC-el védekeznek a tervezők, amire a HMAC függvényt javasolják. Itt fontos megemlíteni, hogy a MAC-nek gyorsnak és egyszerűen számíthatónak kell lennie, egyébként kihasználható lenne a protokoll DoS támadással szemben. Ideális esetben a kezdeményező fél (initiator) végzi el ezen hitelesítő kódok ellenőrzését az rrep üzenet megérkezése során. A MAC-hez szükséges szimmetrikus kulcs közös a kezdeményező és a célcsomópont között, más ezt nem ismerheti. Az útvonalon elhelyezkedő csomópontok hitelesítéséhez szerényebb kapacitású eszközök esetén a TESLA vagy a páronkénti osztott kulcsokat használó hitelesítési eljárást javasolják. Nagyobb számítási kapacitás esetén a digitális aláírás lehet másik lehetséges megoldás. A útvonalválasztást kezdeményező megbízik a célcsomópontban, hiszen az képes kontrollálni a kettőjük között menő forgalmat. Az anonimitás és titkosság teljesítése útvonalválasztás esetén nem szükséges.

Az Ariadne működése három fő részre osztható.

1. *Célállomás hitelesíti a hozzá érkező* rreq *üzenetet.* Ennek céljából a kezdeményező elhelyez egy MAC hitelesítő kódot az rreq üzenetben, ami nem más mint a csak kettőjük által ismert titkos kulcson (K_{SD}) valamint egy tetszőleges üzeneten (pl. időbélyeg) számolt ellenőrzőösszeg.

2. *Útvonalinformáció hitelesítése.* A útvonal felderítése során a kezdeményező hitelesíteni szeretné az rrep üzenetben szereplő összes csomópontot, valamint a cél szintén meg akar győződni az rreq üzenetben szereplő csomópontok hitelességéről, hogy az rrep üzenetet csak a hitelesített csomópontok alkotta útvonalon küldje vissza a kezdeményezőhöz. Ezen cél elérése történhet TESLA, digitális aláírás vagy a páronkénti szétosztott osztott kulcsok (egyszerű MAC) felhasználásával, amelyeknél az utolsó a leghatékonyabb számítási igény szempontjából.

3. *Hoponkénti hash-érték számítása.* Az útvonalinformáció hitelesítése nem elegendő, mivel egy támadó képes lehet csomópontokat eltávolítani az rreq üzenetből. Ezért ennek detektálása céljából hoponkénti hash-érték számítást alkalmaznak. Tehát egy előző hop módosítása vagy törlése csak úgy lehetséges, ha a támadó azt az rreq üzenetet kapja meg, amiben még nem szerepel az előző hop, vagy pedig invertálja az egyirányú hashfüggvényt.

A fentiekből látható, hogy az útvonalinformáció hitelesítése szempontjából az Ariadne protokollnak három fajtája létezik. Ezek közül én a *hitelesítő kód*ot használó változatot mutatjuk be, a másik két változat is hasonlóképpen működik.

5.1. Az Ariadne hitelesítő kód használatával

S	:	$h_{\mathsf{S}} = MAC_{\mathsf{SD}}(\mathsf{rreq}, \mathsf{S}, \mathsf{D}, id)$
S $\rightarrow *$	:	(rreq, S, D, id, h_{S}, (), ())
B	:	$h_{\mathsf{B}} = H(\mathsf{B}, h_{\mathsf{S}})$
B $\rightarrow *$	:	(rreq, S, D, id, h_{B}, (B), (mac_{BD}))
C	:	$h_{\mathsf{C}} = H(\mathsf{C}, h_{\mathsf{B}})$
C $\rightarrow *$	:	(rreq, S, D, id, h_{C}, (B, C), (mac_{BD}, mac_{CD}))
D $\rightarrow$ C	:	(rrep, D, S, (B, C), mac_{SD})
C $\rightarrow$ B	:	(rrep, D, S, (B, C), mac_{SD})
B $\rightarrow$ S	:	(rrep, D, S, (B, C), mac_{SD})

5.1. táblázat. Protokollüzenetek az Ariadne működése során. S útvonalkeresést kezdeményez D felé. rreq a kérés-, míg rrep a válaszüzeneteket jelöli. id a kérés véletlen értékű azonosítója, mac_{XY} az X és Y titkos kulcsán és a protokollüzeneten számolt ellenőrzőösszeg (MAC). H a publikus hashfüggvény, h pedig az iterált hash-érték.

Az Ariadne működését mutatja az 5.1. táblázat. Látható, hogy az útvonalat alkotó minden csomópont hitelesíti az rreq üzenetet és így magát az útvonalat. A mac_{SD} jelöli az üzeneten és az S és D azonosítójú csomópontok közös privát kulcsán számolt ellenőrzőösszeget, id pedig a kérés véletlen értékű azonosítóját.

Kezdetben a kezdeményező (initiator) generál egy rreq üzenetet, majd ezt többesszórással eljuttatja minden szomszédjának. Az rreq üzenet tartalmazza a kezdeményező és a célcsomó-

pont egyedi azonosítóját, egy generált véletlent, ami azonosítja az rreq üzenetet és így magát az útvonalkeresést, illetve egy mindezeken és egy csak a cél és kezdeményező csomópont által ismert kulcson generált ellenőrzőösszeget (MAC). Az első csomópont ezen és a saját azonosítóján számol egy hash-értéket egy publikus hashfüggvény alapján, amit elhelyez a listában. A köztes csomópontok az előző hop által elhelyezett hash-érték és saját azonosítójuk alapján számolják tovább a hash-értéket ugyanazzal a publikus hashfüggvénnyel. Minden egyes köztes csomópont, amely megkapja az rreq üzenetet, először kiszámolja ezt a hash-értéket, majd saját azonosítóját hozzáfűzi a listában szereplő azonosítókhoz, végül ezt az új kérést hitelesíti a csak általa és a célcsomópont által ismert kulccsal. Végezetül ezt a hitelesítő kódot csatolja az rreq üzenethez, majd ezt továbbítja minden szomszédjának.

Miután a kérés eljutott a célcsomóponthoz, az ellenőrzi a minden hop által számolt hash-értéket, mivel a kezdeményező által számolt MAC-et ki tudja számolni. Ezekután minden az útvonalban szereplő csomópont hitelesítő kódját ellenőrzi. Ha mindezen ellenőrzés sikeres, akkor a cél generál egy rrep üzenetet, amit visszaküld a forrás felé az rreq üzenetben érkezett útvonal fordítottja szerint. Ez a válaszüzenet tartalmazza a célcsomópont azonosítóját, magát a keresett útvonalat valamint a cél hitelesítő kódját ezeken az elemeken (az ehhez tartozó kulcsot csak a kezdeményező és a cél ismeri). Minden egyes köztes csomópont, amely megkapja ezt az rrep üzenetet, továbbítja a kezdeményező felé módosítás nélkül. Miután az rrep üzenet sikeresen eljutott a forráshoz, az ellenőrzi a cél hitelesítő kódját. Ha a verifikáció sikeres, akkor a kezdeményező elfogadja a protokoll által visszaadott útvonalat. A továbbiakban feltételezzük, hogy minden csomópont, amely egy rreq vagy rrep üzenetet kap, elvégzi az alábbi egyszerű ellenőrzéseket.

- Ha egy q csomóponthoz először érkezik egy rreq üzenet, akkor ellenőrzi, hogy az útvonalban szereplő utolsó csomópont azonosító megegyezik-e q valamely szomszédjának az azonosítójával. Ha ez nem teljesül, akkor ellenőrzi, hogy a kezdeményező azonosítója szerepel-e q szomszédjainak az azonosítói között. Ha ez is sikertelen, akkor eldobja q csomópont a kérést.

- Ha egy q csomóponthoz először érkezik egy rrep üzenet, akkor először ellenőrzi, hogy a saját azonosítója szerepel-e az útvonalban. Ha nem, akkor az rrep üzenetet eldobja. Ha igen, akkor ellenőrzi, hogy az útvonalban q azonosítója előtt illetve után szereplő azonosítók valóban q szomszédjainak az azonosítói. Ha nem, akkor ellenőrzi, hogy van-e még további azonosító az útvonalban. Ha nincs, akkor ellenőrzi, hogy q szomszédjainak az azonosítói között szerepel-e a kezdeményező azonosítója. Ha nem szerepel, akkor eldobja rrep üzenetet. Ha van még további azonosító útvonalban, de nem szerepel előtte azonosító, akkor ellenőrzi, hogy a cél azonosítója megegyezik-e valamelyik szomszédjának az azonosítójával. Ha nem egyezik, akkor eldobja az rrep üzenetet.

5.2. Egy támadás az Ariadne ellen

Ebben a részben bemutatok egy támadást az Ariadne MAC hitelesítést használó változata ellen (a többihez hasonló módon konstruálható támadás) Aktív-1-2 támadót feltételezve. Ebből következik a tény, hogy az Ariadne nem biztonságos Aktív-1-x támadó esetén sem.

Tegyük fel, hogy létezik olyan redukált $conf = (G_{L_3}(E, V, \mathcal{L}), V^*)$ konfiguráció és egy olyan $\mathcal{A}$ támadó, hogy a következő üzenet, amely $\ell_{ini} \in L \setminus L^*$ azonosítójú csomóponthoz érkezik $sys^{\mathsf{ideal}}_{conf,\mathcal{A}}$ rendszerben, egy nem plauzibilis útvonalat tartalmaz:

$$msg = (\mathsf{rrep}, \ell_{ini}, \ell_{tar}, (\ell_1, \ldots, \ell_p), mac_{\ell_{ini},\ell_{tar}})$$

Továbbá tételezzük fel, hogy msg-t egy $\top$ korrupciós jelzővel kaptuk, valamint msg sikeresen teljesítette az Ariadne összes verifikációját ℓ_{ini} csomópontban. Ez azt jelenti, hogy az msg-ben levő MAC helyes, ℓ_1 ℓ_{ini} egy szomszédja, és $h = (\ell_{ini}, \ell_1, \ldots, \ell_p, \ell_{tar})$ egy nem plauzibilis útvonal $conf$-ban. Jelöljük msg-ben id-vel a kérés azonosítóját.

1. lemma. *Ha h nem plauzibilis útvonal $conf$-ban, akkor msg keresztül haladt egy $a \in V^*$ csúcsnak megfeleltetett csomóponton.* ♢

Bizonyítás Tegyük fel az állítás ellenkezőjét, hogy msg nem haladt keresztül egy támadó csomóponton sem, vagyis csak $v \in V \setminus V^*$ csomópontokat érintett az útja során. Ekkor v_1 csomópont ($\ell_{ini} \in \mathcal{L}(v_1)$) csak egy olyan v_2 csomóponttól kaphatta msg-t, amelyre $v_2 \in V \setminus V^*$. Ez viszont mindegyik v_i csomópontra igaz, ahol $\ell_i \in \mathcal{L}(v_i)$ és $2 \leq i \leq p$. Mivel minden i-re $v_i \in V \setminus V^*$, így v_i csomópont csak akkor továbbítja msg üzenetet, ha $\mathcal{L}(v_i)$ a h útvonal része, valamint létezik $r, q \in V$, hogy $\ell_{i-1} \in \mathcal{L}(r)$ és $\ell_{i+1} \in \mathcal{L}(q)$, valamint $(p, v_i) \in E$ és $(q, v_i) \in E$, tehát $v_{i-1} = r$ és $v_{i+1} = q$. Ebből arra a következtetésre jutunk, hogy h útvonal plauzibilis, hiszen $\ell_{ini}|\ell_1|\ldots|\ell_{tar}$ egy létező particionálása h-nak, ami ellentmond a feltételezésnek. ∎

2. lemma. *Ha h nem plauzibilis útvonal $conf$-ban, akkor $v_{tar} \in V \setminus V^*$ ($\ell_{tar} \in \mathcal{L}(v_{tar})$) célcsomópont nem elhanyagolható valószínűséggel hitelesítette msg üzenetet.* ♢

Bizonyítás Az 1. lemma alapján látható, hogy $\mathcal{A}$ támadó egy csomópontján keresztül haladt msg. Legyen ez a legkorábbi időpont (menet) ρ. Mivel $mac_{\ell_{tar},\ell_{ini}}$ hitelesítő kód helyes, és $\ell_{tar} \notin L^*$, ezért $\mathcal{A}$ támadó csak elhanyagolható valószínűséggel volt képes msg üzenetet magától generálni, amiből következik, hogy $\mathcal{A}$ támadó egy $\rho' \leq \rho$ menetben kaphatta meg msg-t, ami már tartalmazta $mac_{\ell_{tar}}$-t. Mivel a megbízható csomópontok (beleértve a célt) csak akkor továbbítják msg üzenetet, ha azt korábban már megkapták, ezért létezik $\rho'' < \rho'$ menet, amiben a cél már hitelesítette msg üzenetet. ∎

3. lemma. *Ha h egy nem plauzibilis útvonal, akkor létezik j ($0 \leq j \leq p-1$, $\ell_0 = \ell_{ini}$, $\ell_{p+1} = \ell_{tar}$) és q ($1 \leq q \leq p - j$), hogy $\ell_j \in \mathcal{L}(v)$, $\{\ell_{q+1}, \ldots, \ell_{j+q}\} \subseteq L^*$, $\ell_{j+q+1} \in \mathcal{L}(v')$, ahol $v, v' \in V \setminus V^*$, és nem létezik $v^* \in V^*$, hogy $(v, v^*) \in E$ és $(v^*, v') \in E$. Ekkor létezik nem elhanyagolható valószínűséggel $a, b \in V^*$, hogy $(v, a) \in E$ és $(b, v') \in E$.* ♢

Bizonyítás Jelen esetben ha h nem plauzibilis, akkor a 2. állítás 2. pontja teljesülhet csak, hiszen az 1. pontbeli állítás az Ariadne esetén csak elhanyagolható valószínűséggel teljesülhet, mivel az ottani feltétel szerint a két azonosítóhoz tartozó megbízható csomópont nem szomszédos.

Tegyük fel a 2. állítás első pontját feltételezve, hogy nem létezik $a \in V^*$, hogy $(v, a) \in E$. Mivel $\mathcal{A}$ támadó csak elhanyagolható valószínűséggel képes $mac_{\ell_t,D}$ ellenőrzőösszeg megkonst-

ruálásához, ezért nem elhanyagolható valószínűséggel v hitelesítette és továbbította

$$(\mathsf{rreq}, \ell_{ini}, \ell_{tar}, id, h_{\ell_i}, (\ell_1, \ldots, \ell_i), (mac_{\ell_1,\ell_{ini}}, \ldots, mac_{\ell_i,\ell_{ini}}))$$

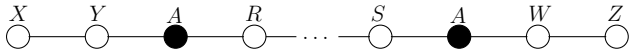
üzenetet. Viszont $v \in V \setminus V^*$ a feltétel miatt, így v nem bocsáthatta ki a fenti üzenetet, hiszen $\ell_{t+q+1} \notin \mathcal{N}(v)$.

Hasonló indirekt gondolatmenettel igazolható a $(b, v') \in E$ eset is. ∎

1. tétel. *Az Ariadne nem biztonságos Aktív-1-2 támadót feltételezve.* ♢

Bizonyítás A protokoll nem biztonságos, ha létezik egy olyan $conf$ konfiguráció és egy olyan $\mathcal{A}$ támadó, hogy $sys^{\mathsf{ideal}}_{conf,\mathcal{A}}$ korrupciós jelzővel adja vissza az útvonalat, míg $sys^{\mathsf{real}}_{conf,\mathcal{A}}$ nem, tehát a visszaadott útvonal nem plauzibilis.

A 1. 2. és 3. lemmák alapján bemutatunk egy Aktív-1-2 támadó általi sikeres támadást az Ariadne protokoll ellen.



5.1. ábra. Egy lehetséges konfiguráció, amely alapján az Ariadne protokoll nem biztonságos

A 5.1. ábrán látható egy lehetséges támadás konfigurációja. A 3. lemma alapján a támadó csomópontjai a 5.1. ábra szerint helyezkednek el, ahol A-val jelöltük ezen csomópontok azonosítóit. Az 1. és 2. lemmák alapján azt is tudhatjuk, hogy nagy valószínűséggel a válaszüzenetet a célcsomópont aláírta, és nagy valószínűséggel az üzenet áthaladt legalább egy támadó csomóponton. Fontos megemlíteni, hogy *egy útvonalban résztvevő csomópontok nem képesek más csomópontok hitelesítő kódját ellenőrizni, hiszen nem ismerik a megfelelő privát kulcsokat.*

X egy útvonalkeresést kezdeményez Z felé. Az első támadó csomópont a következő kérésüzenetet kapja:

$$msg_1 = (\mathsf{rreq}, X, Z, id, h_Y, (Y), (mac_{YZ}))$$

A ezután nem fűzi hozzá a saját MAC-ét a kéréshez, hanem inkább h_Y értéket beilleszti az eredeti MAC helyére, és ezt az üzenetet továbbítja:

$$msg_2 = (\mathsf{rreq}, X, Z, id, h_Y, (Y), (mac_{YZ}, h_Y))$$

Mivel egy MAC formailag úgy néz ki mint egy hash érték (ha a MAC számítást kriptográfiai MAC alapján végzik (pl. HMAC)), és ezt a MAC-et célcsomóponton kívül senki sem tudja ellenőrizni, ezért R csomópont nem fogja detektálni a támadást. Így a második támadó csomóponthoz a következő kérésüzenet érkezik:

$$msg_3 = (\mathsf{rreq}, X, Z, id, h_S, (Y, A, R, \ldots, S), (mac_{YZ}, h_Y, mac_{RZ}, \ldots, mac_{SZ}))$$

Ezekután A eltávolítja a két támadó csomópont között levő csomópontok $(R, \ldots, S)$ azonosítóit és a hozzájuk tartozó MAC összegeket a listából. Ezt A megtudja tenni, mivel az

azonosítólistában hátra fele keresve megtalálja az első támadó csomópont pozícióját, amiből megállapítja, hogy hol helyezkedik el annak a MAC összege a többi aláíráshoz képest. Így A megszerzi h_Y hash-értéket, amiből kiszámítja $h_A = H(A, h_Y)$ hash-értéket, és most már érvényes MAC összeget tud generálni $(\mathsf{rreq}, X, Z, id, h_A, (Y, A), (mac_{YZ}))$ üzeneten, majd továbbítja a következő üzenetet:

$$msg_4 = (\mathsf{rreq}, X, Z, id, h_A, (Y, A), (mac_{YZ}, mac_{AZ}))$$

Mivel a hoponként számolt hash-érték helyes, csakúgy mint az összes MAC msg_4-ben, ezért Z-hez egy érvényes kérés fog érkezni, amire aztán Z a következő választ adja:

$$msg_5 = (\mathsf{rrep}, Z, X, id, (Y, A, W), mac_{ZX})$$

Amikor msg_5 eléri a második támadó csomópontot, az visszailleszti a két támadó csomópont között levő csomópontok azonosítóinak listáját az útvonalba, majd a kapott üzenetet továbbítja:

$$msg_6 = (\mathsf{rrep}, Z, X, id, (Y, A, R, \ldots, S, A, W), mac_{ZX})$$

Mivel a köztes csomópontok nem tudják ellenőrizni a MAC összeget msg_6-ban, üzenet az eljut az első támadó csomóponthoz, amely aztán ismét eltávolítja a köztes csomópontok azonosítóit az útvonalból:

$$msg_7 = (\mathsf{rrep}, Z, X, id, (Y, A, W), mac_{ZX})$$

Ezt az üzenetet továbbítva az sikeresen eljut X csomópontba, amely aztán sikeresen ellenőrzi a cél aláírását, és elfogadja $h = (Y, A, W)$ útvonalat, ahol h egy nem plauzibilis útvonal. ∎

6. fejezet

endairA: Egy bizonyított biztonságú forrás alapú protokoll

Az Ariadne alapján bemutatok egy útvonalválasztó eljárást, ami bizonyíthatóan statisztikailag biztonságos az általam definiált modellben. Az új protokollt endairA-nak fogom hívni (az Ariadne megfordítása). A neve elárulja, hogy egy útvonalválasztás során az útvonalban résztvevő csomópontok a kérésüzenetek helyett a válaszüzeneteket fogják aláírni, és így hitelesíteni azt. Először a 6.1. részben bemutatom az endairA működését, majd a 6.2. részben formálisan belátjuk, hogy biztonságos a modellben. A 6.3. részben leírom az endairA néhány variánsát, amelyek gyakorlati szempontból jelentősek.

6.1. Az endairA specifikációja

$\mathsf{S} \rightarrow *$	:	(rreq, S, D, id, ())
$\mathsf{B} \rightarrow *$	:	(rreq, S, D, id, (B))
$\mathsf{C} \rightarrow *$	:	(rreq, S, D, id, (B, C))
$\mathsf{D} \rightarrow \mathsf{C}$	:	(rrep, D, S, (B, C), (sig_{D}))
$\mathsf{C} \rightarrow \mathsf{B}$	:	(rrep, D, S, (B, C), $(sig_{\mathsf{C}}, sig_{\mathsf{D}})$)
$\mathsf{B} \rightarrow \mathsf{S}$	:	(rrep, D, S, (B, C), $(sig_{\mathsf{B}}, sig_{\mathsf{C}}, sig_{\mathsf{D}})$)

6.1. táblázat. Protokollüzenetek az endairA működése során. S útvonalkeresést kezdeményez D felé. rreq a kérés-, míg rrep a válaszüzeneteket jelöli. id a kérés véletlen értékű azonosítója, míg sig_{X} az X csomópont aláírása a protokollüzeneten.

A 6.1. táblázat mutatja az endairA működését. sig_{D} jelöli a D azonosítójú csomópont digitális aláírását az üzeneten, id pedig a kérés véletlen értékű azonosítóját. Kezdetben az útvonalkeresést kezdeményező csomópont küld egy rreq kérésüzenetet, amiben megnevezi célcsomópont azonosítóját valamint a kérés véletlen értékű id azonosítóját. Minden egyes közbenső csomópont, amely megkapja ezt a kérést, hozzáfűzi saját azonosítóját az eddig felhalmozódott azonosítók után, majd ezt a kérésüzenetet továbbítja. Amikor a kérés eléri a célt, az generál egy rrep válaszüzenetet. A válasz tartalmazza a kezdeményező és a cél azonosítóját, magát az rreq üzenetben felhalmozódott azonosítókból álló útvonalat és a cél aláírását ezeken az elemeken. A válaszüzenetet a cél az rreq üzenetben kapott útvonal fordított sorrendjében

továbbítja a megfelelő szomszédos csomópontnak. Minden közbenső csomópont, amely megkapja a válaszüzenetet, az alábbi három ellenőrzést végzi el.

1. Szerepel-e a saját azonosítója az útvonalban elhelyezkedő azonosítók között.
2. A saját azonosítóját követő és megelőző azonosítók valóban szomszédos csomópontokhoz tartoznak-e.
3. Ellenőrzi az összes aláírást az üzenetben, valamint azt, hogy minden aláíráshoz kölcsönösen egyértelműen hozzárendelhető-e pontosan egy azonosító az útvonalból.

Ha minden ellenőrzés sikeres, akkor aláírja a válaszüzenetet, majd továbbítja a válaszban szereplő útvonal alapján. Ellenkező esetben eldobja a válaszüzenetet továbbítás nélkül. Amikor a kezdeményező megkapja a válaszüzenetet, ellenőrzi, hogy az útvonalban szereplő első azonosító az valójában egy szomszédos csomópont azonosítója-e. Ha igen, akkor ellenőrzi az összes útvonalban szereplő azonosítók szerint a megfelelő aláírásokat, beleértve a célt is. Ha mindez sikeres, akkor elfogadja a válaszüzenetben kapott útvonalat.

Fontos kiemelni az aláírások ellenőrzését az útvonalat alkotó csomópontok által, mivel enélkül az endairA még Aktív-1-x támadó esetén sem lenne biztonságos a modellünkben.

6.2. Az endairA biztonsága

2. tétel. *Az endairA statisztikailag biztonságos, ha a felhasznált $\mathcal{S}$ aláíróséma is statisztikailag biztonságos választott nyíltszövegű támadás esetén.* ♢

Bizonyítás Látható, hogy minden megbízható csomópont azt az msg üzenetet írja alá és továbbítja az üzenetben szereplő útvonal alapján, amit a célcsomópont aláírt. Így biztosítható, hogy minden megbízható csomópont valóban a céltól küldött msg üzenetet továbbítja, ha a kérdéses csomópont valóban szerepel az útvonalban, és az összes aláírás megléte és helyessége garantálja, hogy az útvonal valóban plauzibilis. Annak a valószínűsége, hogy a célcsomópont aláírása, illetve a köztes csomópontok aláírása hamis, elhanyagolható. A következőkben formálisan is igazoljuk a protokoll helyességét.

Egy útvonalválasztó protokoll statisztikailag biztonságos, ha nem plauzibilis útvonalat csak elhanyagolható valószínűséggel ad vissza bármely $conf$ konfigurációra és bármely $\mathcal{A}$ támadó esetén. Pontosabban egy rrep üzenet $sys^{\mathsf{ideal}}_{conf,\mathcal{A}}$ rendszerben elhanyagolható valószínűséggel kerül eldobásra $\top$ korrupciós jelző miatt.

Tegyük fel, hogy létezik olyan $conf = (G_{L_3}(E, V, \mathcal{L}), V^*)$ konfiguráció és $\mathcal{A}$ támadó, hogy a következő üzenet sikeresen eljut egy ℓ_{ini} azonosítójú megbízható csomóponthoz $sys^{\mathsf{ideal}}_{conf,\mathcal{A}}$ rendszerben:

$$msg = (\mathsf{rrep}, \ell_{ini}, \ell_{tar}, (\ell_1, \ldots, \ell_p), (sig_{\ell_{tar}}, sig_{\ell_p}, \ldots, sig_{\ell_1}))$$

Továbbá tételezzük fel, hogy msg-t egy $\top$ korrupciós jelzővel kaptuk, valamint msg sikeresen teljesítette endairA összes verifikációját ℓ_{ini} csomópontban és a köztes csomópontokban. Ez azt jelenti, hogy az msg-ben levő összes aláírás helyes, ℓ_1 ℓ_{ini} egy szomszédja, és $h = (\ell_{ini}, \ell_1, \ldots, \ell_p, \ell_{tar})$ egy nem plauzibilis útvonal $conf$-ban. Ha h nem plauzibilis útvonal, akkor teljesül a 2. állítás.

A következőkben belátjuk, hogy a 2. állítás mindkét pontja csak elhanyagolható valószínűséggel teljesül az endairA futása során, amiből következik, hogy nem plauzibilis útvonal csak elhanyagolható valószínűséggel kerül eldobásra $sys^{\mathsf{ideal}}_{conf,\mathcal{A}}$ rendszerben.

Az 1. eset triviálisan csak akkor lehetséges, ha $\mathcal{A}$ támadó hamisította $sig_{\ell_{t+1}}$ aláírást, hiszen ekkor $(v, v') \notin E$, vagyis $\ell_t \notin \mathcal{N}(v')$, és így v' nem fogja aláírni msg üzenetet.

A 2. esetben tegyük fel, hogy $\mathcal{A}$ nem konstruálta meg egyik megbízható csomópont aláírását sem. v csomópont a következő

$$msg' = (\mathsf{rrep}, \ell_{ini}, \ell_{tar}, (\ell_1, \ldots, \ell_p), (sig_{\ell_{tar}}, sig_{\ell_p}, \ldots, sig_{\ell_{j+1}}))$$

üzenetet csak egy $v^* \in V^*$ csúcsnak megfeleltetett csomóponttól kaphatta. Mivel $\ell_{j+1} \in L^*$, ezért egy megbízható csomópont nem küldhetett egy üzenetet $sig_{\ell_{j+1}}$ aláírással. msg' küldéséhez v^* a következő üzenetet kaphatta:

$$msg'' = (\mathsf{rrep}, \ell_{ini}, \ell_{tar}, (\ell_1, \ldots, \ell_p), (sig_{\ell_{tar}}, sig_{\ell_p}, \ldots, sig_{\ell_{j+q+1}}))$$

hiszen a feltétel szerint $\mathcal{A}$ nem konstruálhatta meg $sig_{\ell_{j+q+1}}$ aláírást, hiszen $\ell_{j+q+1} \notin L^*$. Mivel v^*-nak a modell szerint a gráfban nincs támadó csomópontnak megfeleltetett szomszédja, ezért msg''-t csak megbízható csomóponttól kaphatta. Viszont az egyetlen csomópont, amely msg''-t küldhette, az v'. Ez azt jelentené, hogy $(v, v^*) \in E$ és $(v^*, v') \in E$, ami ellentmond 2. feltétel 2. pontjának. Így az eredeti feltevésünk sem lehetett igaz, vagyis $\mathcal{A}$ megkonstruálta legalább egy megbízható csomópont aláírását.

A fentieket összegezve egy tetszőleges támadó csak akkor járhat sikerrel, ha képes megkonstruálni legalább egy megbízható csomópont aláírását. Tegyük fel, hogy $\mathcal{A}$ képes erre, vagyis létezik olyan rrep üzenet, amely eldobásra kerül nem elhanyagolható valószínűséggel $sys^{\mathsf{ideal}}_{conf,\mathcal{A}}$ rendszerben $\top$ korrupciós jelzője miatt. Ezt felhasználva konstruálunk egy olyan $\mathcal{A}'$ támadót, amely képes a biztonságosnak feltételezett $\mathcal{S}$ aláírósémát megtörni, amellyel viszont ellentmondásba ütközünk.

Alkalmazzuk a redukció módszerét $\mathcal{A}'$ létrehozásához. $\mathcal{A}'$ támadó $\mathcal{S}$ aláírósémát akarja megtörni úgy, hogy ismeri ennek egy tetszőleges puk publikus kulcsát, és nem ismeri az ehhez tartozó prk privát kulcsot. Viszont $\mathcal{A}'$ támadó hozzáfér egy $\mathcal{O}$ orákulumhoz, amely prk privát kulcsot használja az aláíráshoz. $\mathcal{O}$ orákulum minden neki küldött üzenetet aláír prk kulccsal, majd az aláírást visszaküldi. Sikeres támadásnak az minősül, ha $\mathcal{A}'$ képes olyan érvényes aláírást konstruálni egy tetszőleges üzenethez, amely üzenetet előzőleg $\mathcal{O}$ orákulum nem írt alá. Ehhez $\mathcal{A}'$ támadó szimulálja a protokoll futását az alábbi módon. Kiválaszt egy ℓ_k tetszőleges csomópontot. ℓ_k kulccsal rendelkező csomópontnak megfelelteti puk publikus kulcsot. A protokollban résztvevő minden más csomópontnak elküldi puk kulcsot ℓ_k nevében. Ennek eredményeképpen minden csomópont puk publikus kulccsal ellenőrzi ℓ_k csomópont aláírását. Amikor ℓ_k aláírna egy üzenetet, akkor $\mathcal{A}'$ $\mathcal{O}$ orákulumhoz fordul aláírásért ezzel az üzenettel, majd ezt az aláírást és üzenetet küldi tovább ℓ_k nevében. A többi csomópont sikeresen ellenőrzi ezt az aláírást, hiszen ismerik puk kulcsot. Feltételezésünk szerint $sys^{\mathsf{ideal}}_{conf,\mathcal{A}}$ rendszer szimulációja során nem elhanyagolható ϵ valószínűséggel előfordul egy olyan rrep (msg) üzenet, amelyben az összes csomópont aláírása helyes, viszont az útvonal nem plauzibilis. Vagyis

létezik egy olyan ℓ_j csomópont, amelynek aláírását tartalmazza msg, annak ellenére, hogy ezt ℓ_j nem írta alá. Tegyük fel, hogy $k = j$. Ebben az esetben sig_{ℓ_j} egy olyan aláírás, ami puk kulccsal ellenőrizve helyes. Mivel ℓ_j nem írhatta alá msg megfelelő részét, ezért $\mathcal{A}'$ nem hívhatta $\mathcal{O}$ orákulumot sig_{ℓ_j} aláírás generálásához. Így $\mathcal{A}'$-nek sikerült egy olyan aláírást előállítania egy üzeneten, ami puk nyilvános kulccsal ellenőrizve helyes. Mivel $\mathcal{A}'$ ℓ_j-t véletlenül választotta, ezért a $k = j$ esetnek a valószínűsége $\frac{1}{n}$, és így $\mathcal{A}'$ sikerének a valószínűsége $\frac{\epsilon}{n}$, ahol ha ϵ nem elhanyagolható, akkor $\frac{\epsilon}{n}$ sem elhanyagolható. ∎

Összességében a támadó felügyelete alatt levő csomópontok csak elhanyagolható valószínűséggel képesek nem az rrep üzenetben szereplő útvonal szerint továbbítani rrep válaszüzenetet, ami viszont biztosítja azt, hogy az rrep üzenetben a forrás által kapott útvonal valóban plauzibilis útvonal.

6.3. endairA kiegészítések és variánsok

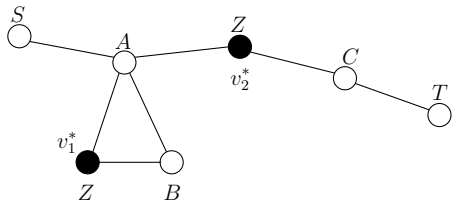
Az endairA amellett, hogy bizonyíthatóan biztonságos a modellünkben, rendelkezik egy másik jelentős tulajdonsággal. Nevezetesen azzal, hogy kevesebb kriptográfiai műveletet igényel az összes csomópont részéről, mivel csak az rrep üzenetek során kell kriptográfiai ellenőrzéseket végezni, és ezt is csak az útvonalban résztvevő csomópontok részéről, hiszen az rrep pont-pont címzéssel halad vissza a forráscsomópont fele. Ezzel szemben az Ariadne esetén minden rreq üzenet aláírásra kerül a hálózatban minden csomópont által, hiszen az rreq üzeneteket többesszórással továbbítják a csomópontok.

Fontos megemlíteni, hogy a modellünkben az egyes csomópontokat statikusnak tételezzük fel (legalábbis az analízis ideje alatt) abban az értelemben, hogy azok nem mozognak, és így magának a hálózatnak a konfigurációja nem változik. Ebből következik, hogy ebben a modellben visszajátszásos támadás (replay attack) nem lehetséges, hiszen ha egy útvonal érvényes, akkor az a hálózat statikus jellege miatt mindig is érvényes lesz az analízis ideje alatt. A másik ok, amiért nem lehetséges visszajátszásos támadás az a támadó nem adaptív jellege, hiszen ha jobban belegondolunk, akkor észrevehetjük, hogy ez a fajta támadás adaptívnak minősül. Ennek ellenére a gyakorlatban jelentős lehet ezen támadás megakadályozása. Ez ellen egy lehetséges védekezés ha megköveteljük a célcsomóponttól, hogy az lásson el minden rrep üzenetet azzal a véletlen azonosító értékkel (id), amit az rrep-hez tartozó rreq üzenetben kapott, és a többi mezővel együtt azt is hitelesítse aláírásával. Így a forrás ellenőrizni tudja, hogy a kapott rrep üzenet valóban friss-e. Természetesen ha megengedjük a csomópontok mozgását az analízis ideje alatt, akkor előfordulhat az az eset, hogy konkrét támadás nélkül a visszaadott útvonal már nem létezik a csomópontok mozgása miatt. Ez ellen csak egy időkorlát bevezetésével védekezhetünk, vagyis a forráscsomópont csak bizonyos ideig fogadhat el válaszokat egy kérésre.

Egy másik probléma lehet, hogy a támadó csomópontok képesek rreq kérésekkel elárasztani a hálózatot bármely megbízható csomópont nevében, hiszen az rreq üzenetek nincsenek hitelesítve. Ez ugyan nem befolyásolja a visszaadott út plauzibilitását, vagyis a modell szerinti biztonságot, de a gyakorlatban ez is problémaként jelentkezhet. Ez elkerülhető, ha például megköveteljük, hogy a forráscsomópont is írja alá az rreq üzenetet. Ez persze többletterhelést

jelent a többi csomópontnak, mivel azoknak ellenőrizni kell a forrás aláírását.

Végezetül egy utolsó gyakorlati probléma lehet az, hogy az rrep üzenetben minden aláírást minden csomópontnak ellenőrizni kell, ami szintén túl költséges néhány alkalmazás esetén. Egy jó megoldásnak kínálkozna, ha megköveteljük, hogy minden köztes csomópont csak a cél aláírását ellenőrizze az rrep üzenetben. Így viszont a protokoll már nem lesz biztonságos a modellünkben, ahogy azt a 6.1. ábra mutatja.



6.1. ábra. Egy támadás konfigurációja az endairA protokoll ellen, ha csak a célcsomópont aláírásának ellenőrzését követeljük meg a köztes csomópontoktól.

Tegyük fel, hogy S csomópont útvonalkeresést kezdeményez T csomópont felé. Megmutatom, hogy az (A, B, Z, C) útvonal elfogadásra kerül S csomópont által, annak ellenére, hogy nem plauzibilis útvonal. v_2^* csomópont A csomópontnak küldi a következő üzenetet B nevében, miután megkapta a válaszüzenetet C csomóponttól.

$$(\mathsf{rrep}, S, T, id, (A, B, Z, C), (sig_T, sig_C, sig_Z))$$

A elfogadja az üzenetet, mert T aláírása érvényes, A szerepel az útvonalban, valamint S és B csomópontok A szomszédai. Így A aláírja az üzenetet, és elküldi azt S csomópontnak:

$$(\mathsf{rrep}, S, T, id, (A, B, Z, C), (sig_T, sig_C, sig_Z, sig_A))$$

S nyilvánvalóan eldobja ezt az üzenetet, mivel B aláírása hiányzik az üzenetből. Viszont v_1^* képes venni A adását, és így értelmezni képes az A által küldött üzeneteket. Így először eltávolítja A aláírását, majd elküldi B csomópontnak a következő üzenetet:

$$(\mathsf{rrep}, S, T, id, (A, B, Z, C), (sig_T, sig_C, sig_Z))$$

B elfogadja ezt az rrep üzenetet, aláírja, és a következő üzenetet elküldi A csomópontnak:

$$(\mathsf{rrep}, S, T, id, (A, B, Z, C), (sig_T, sig_C, sig_Z, sig_B))$$

Végül A ismét elfogadja a válaszüzenetet, és elküldi a következő üzenetet S számára:

$$(\mathsf{rrep}, S, T, id, (A, B, Z, C), (sig_T, sig_C, sig_Z, sig_B, sig_A))$$

Ezt már elfogadja S csomópont, annak ellenére, hogy nem plauzibilis útvonalat tartalmaz. Ez a támadás az endairA eredeti változata ellen nem lehetséges, mivel A nem továbbítaná az első rrep üzenetet, amikor B aláírása hiányzik az üzenetből. Viszont ha minden köztes csomópont csak a cél aláírását ellenőrzi, akkor biztosítani kell, hogy minden köztes csomópont

egy kérést csak egyszer dolgozzon fel. Ez történhet úgy, hogy például megjegyzi az *id* értéket minden válaszüzenetnél. Ez viszont egy állandóan növekedő *id* listát (log) eredményezne. Egy lehetséges megoldás a növekedő listával szemben, ha a forrás választ egy t időkorlátot, és ezt szintén elküldi a kéréssel együtt, majd elindítja a számlálóját. Minden választ erre a kérésre csak az időzítő lejárta előtt fogad el. Minden csomópont az első érkező válasszal együtt elindít egy időzítőt t lejárati idővel, amíg nem jár le az időzítő, addig az adott csomópont eldobja az *id*-vel érkező válaszokat, ha az adott *id* már szerepel a listájában. Az időzítő lejártával viszont eltávolítja az *id* értéket a listából (ekkorra már a forrás időzítője lejárt, és nem fogad el ezzel az *id* értékkel érkező válaszokat), és így ismét elfogad ezzel az *id*-vel érkező válaszokat. Ezen megoldáshoz nem szükséges időszinkronizáció a csomópontok között.

7. fejezet

A tábla alapú ad hoc útvonalválasztás modellje

Az útvonalválasztó tábla – röviden: tábla – alapú útvonalválasztó protokollok nem explicit módon adják vissza az útvonalválasztás eredményét, vagyis magát az útvonalat, hanem a rendszer állapota reprezentálja a protokoll lefutása végén magát az eredményt. Mivel minden csomópont lokálisan dönt egy csomag továbbítása felől, ezért a hálózatban levő összes megbízható csomópont tábláinak az összessége írja le a rendszernek az állapotát. Precízebben fogalmazva ha a rendszer a protokoll lefutása során „korrekt" rendszerállapotokon keresztül éri el a futás eredményét reprezentáló végső állapotot, akkor a protokoll biztonságos. Mivel a támadó csomópontok a saját tábláikat szabadon, minden kötöttség nélkül módosíthatják, akár nem a protokoll szabályai szerint, ezért számunkra ezek a táblák nem tartoznak bele a rendszer állapotába.

A dolgozat hátralevő részében ezen tábla alapú útvonalválasztó protokollokkal szeretnék részletesebben foglalkozni. Először a 7.1. részben megadom a hálózat statikus modelljét, majd a 7.2. részben formálisan is megfogalmazom a rendszerállapot és korrekt rendszerállapot fogalmakat, végül a forrás alapú protokollokhoz hasonlóan a 7.3. részben itt is megadom a rendszer szimulációs modelljét, amely a dinamikus viselkedést jellemzi.

7.1. A hálózat statikus modellezése

Hálózati modell: A 3.1. részben ismertetett modellt fogjuk tovább bővíteni a célból, hogy képesek legyünk modellezni a tábla alapú protokollokat. A 3.1. részben bemutatott $G_{L_3}(E, V, \mathcal{L})$ gráfot tovább bővítjük, és bevezetjük egy kapcsolat költségének és a csomóponti költségek fogalmát. Egy kapcsolat költségén a minimális adattovábbítási költséget értjük egy kapcsolat (link) esetén (pl. adattovábbítási idő), míg a csomóponti költségen a minimális adatfeldolgozási költséget értjük egy csomópont esetén (pl. adatfeldolgozási idő). A költségek hozzárendelését az egyes csomópontokhoz és kapcsolatokhoz két függvény végzi. $\mathcal{C}_{node} : V \to \mathbb{R}$ egy csomóponthoz rendeli a csomóponti költséget, míg $\mathcal{C}_{link} : E \to \mathbb{R}$ egy kapcsolathoz annak a költségét. A továbbiakban $\mathcal{C}$ függvény *node* és *link* indexét figyelmen kívül hagyjuk, mivel az argumentum típusa egyértelműen meghatározza magát a függvényt egy alkalmas környezetben. Egy tipikus függvénydefiníció

a következő: $\mathcal{C}(v) = 1$ minden $e \in E$ csúcsra, és $\mathcal{C}(e) = 0$ minden $e \in E$ élre. Ez a gyakorlatra leginkább jellemző eset, amikor egy út költségét az úton elhelyezkedő csúcsok száma adja (hop szám). Így egy olyan L_3 gráfot kapunk tábla alapú protokollok esetén, mely kiegészül ezen költségfüggvényekkel: $G_{L_3}(E, V, \mathcal{L}, \mathcal{C}_{link}, \mathcal{C}_{node})$

Támadó modell: Teljesen egyenértékű a 3.2. részben ismertetett modellel.

Konfiguráció: Jelen esetben a konfiguráció szintén egy $conf = (G_{L_3}(E, V, \mathcal{L}, \mathcal{C}_{link}, \mathcal{C}_{node}), V^*)$ páros, ahol V^* a támadó csomópontoknak megfeleltetett csúcsok halmaza. A konfiguráció itt hasonlóképpen értelmezhető mint a 3.3. részben, vagyis a szomszédos támadó csomópontokhoz tartozó csúcsok összevonása hasonlóképpen történik. A továbbiakban itt is – ha külön nem említjük – redukált konfigurációkról lesz szó, melyek L_3 gráfjaiban a megbízható és a támadó csomópontok által használt csomópontazonosítók halmaza diszjunkt.

7.2. Rendszerállapotok és korrekt rendszerállapotok

Ahogy azt már a bevezetőben említettem tábla alapú útvonalválasztás során az egyes csomópontok lokálisan döntenek egy csomag továbbítása felől. Így egy konfiguráció által képviselt rendszer jellemzéséhez szükségünk van annak állapotának jellemzésére, vagyis hogy az egyes megbízható csomópontok tábláinak a bejegyzéseiben mi áll, hiszen ezen bejegyzések összessége a konfigurációval együtt már egyértelműen leírja az egész rendszer állapotát. Más szavakkal fogalmazva itt most a protokoll által visszaadott érték magának a rendszernek az állapota, vagyis a megbízható csomópontok táblabejegyzéseinek az összessége. Egy ilyen bejegyzés három értéket tartalmaz a mi modellünkben:

1. A célcsomópont azonosítóját.
2. A célcsomóponthoz vezető következő csomópont (hop) azonosítója.
3. A célcsomóponthoz vezető út feltételezett költségét, amely út a 2. bejegyzésben szereplő csomóponton keresztül halad.

Általában a 3. érték a feltételezett legkisebb költséget jelenti.

Így formálisan egy rendszer egy állapota egy $Q \subset (V \setminus V^*) \times L \times L \times \mathbb{R}$ halmazzal reprezentálható úgy, hogy bármely $(v, \ell_{tar}, \ell_{nxt}, c)$ és $(v', \ell'_{tar}, \ell'_{nxt}, c')$ négyesre Q-ban ha $v = v'$ és $\ell_{tar} = \ell'_{tar}$ és $\ell_{nxt} = \ell'_{nxt}$, akkor $c = c'$. Egy $(v, \ell_{tar}, \ell_{nxt}, c) \in Q$ egy táblabejegyzésnek feleltethető meg v táblájában, ahol ℓ_{tar} a cél, ℓ_{nxt} pedig a célhoz vezető úton levő első szomszédos csomópont azonosítója, amely útnak a feltételezett költsége c. Szemléletesen azon négyesek összessége, amelyeknek első helyén v szerepel, alkotják v tábláját. Az összes Q-beli négyes pedig az összes megbízható csomópont táblabejegyzéseinek az összessége. Feltételezzük, hogy egy csomópont táblája egy célcsomópontra több bejegyzést is tartalmazhat, de mindegyik útra a következő csomópont azonosítójának különbözőnek kell lennie.

6. definíció (Korrekt rendszerállapot). *Egy Q rendszerállapot korrekt, ha minden $(v, \ell_{tar}, \ell_{nxt}, c) \in Q$-ra létezik egy olyan $v_1, v_2, \ldots, v_p$ csúcsok sorozata V-ben, hogy $(v_i, v_{i+1}) \in E$ minden $1 \leq i < p$-re, és*

- $v_1 = v$,
- $\ell_{tar} \in \mathcal{L}(v_p)$,
- $\ell_{nxt} \in \mathcal{L}(v_2)$ *és*
- $\sum_{i=2}^{p-1} \mathcal{C}_{node}(v_i) + \sum_{i=1}^{p-1} \mathcal{C}_{link}(v_i, v_{i+1}) \leq c$. ♣

Intuitíve egy rendszer állapota korrekt, ha a megbízható csomópontok összes táblabejegyzései korrektek abban az értelemben, hogy ha v táblája rendelkezik egy bejegyzéssel ℓ_{tar} azonosítójú cél felé ℓ_{nxt} mint következő csomópontazonosítóval és c költséggel, akkor valóban létezik a hálózatban egy olyan út, amely

- v csomópontból indul;
- egy olyan csomópontban végződik, amely használja az ℓ_{tar} azonosítót;
- v egy olyan szomszédján megy keresztül, amely használja az ℓ_{nxt} azonosítót, és
- ezen út költsége kisebb vagy egyenlő mint c.

A útköltségre vonatkozó feltétel a következőkkel magyarázható. Legtöbb esetben egy támadónak az lehet a célja, hogy megpróbálja csökkenteni a forrás által látott úthoz tartozó költség értékét, vagyis az út hosszát – amely út egy támadó csomóponton halad keresztül – kevesebbnek tüntesse fel a forrás számára, mint ami az valójában. Így nagy valószínűséggel a forrás a kisebb költségű útra fog dönteni, amin jelen esetben egy támadó csomópont helyezkedik el. Így a támadó csomópont elérheti indirekt módon, hogy saját magán haladjon át a cél és forrás közötti adatforgalom. Ha ezt egy támadó képes elérni egy protokoll esetében, akkor a rendszer egy nem korrekt rendszerállapotba kerül.

7.3. A rendszer szimulációs modellje

A 4.2. részhez hasonlóan itt is leírom a szimulációs paradigma adaptációját tábla alapú útvonalválasztó protokollokra. Definiálom a valós és ideális világ modelleket, amelyek szintén tartalmaznak támadót, amely támadóról itt is csak annyit követelünk meg, hogy a biztonsági paraméter (pl. kriptokulcs hossza) függvényében polinomiális számú számítási lépést tehet. Így *minden* lehetséges kivitelezhető támadást megengedünk, ami egy elég általános megközelítés. Az ideális világ modell csakúgy mint eddig most is „biztonságos” a konstrukciójából adódóan, vagyis az összes nem tolerálható kategóriába tartozó támadás sikertelen az ideális világbeli rendszerrel szemben.

7.3.1. A valós világ modell

A valós világ modell megfelel egy redukált $conf = (G_{L_3}(E, V, \mathcal{L}, \mathcal{C}_{link}, \mathcal{C}_{node}), V^*)$ konfigurációnak és egy $\mathcal{A}$ támadónak, ahol $\mathcal{A}$ egy Aktív-y-x támadó. Jelöljük ezt a rendszert szintén $sys^{\mathsf{real}}_{conf,\mathcal{A}}$-val. Továbbá legyen $k = |V^*|$ és $n = |V|$. $sys^{\mathsf{real}}_{conf,\mathcal{A}}$ a $\{M_1, M_2, \ldots, M_{n-k}, H, A, \ldots, A_k, C\}$ halmazból áll, ahol az egyes halmazelemek egy-egy

Turing-gépet jelentenek, amelyek átmeneti tárakon (buffer) keresztül kommunikálnak egymással akárcsak a 4.2.1. részben. Minden egyes M_j megfelel egy megbízható csomópontnak, ami megfelel $G_{L_3}(E, V, \mathcal{L}, \mathcal{C}_{link}, \mathcal{C}_{node})$ gráf egy csúcsának. H a megbízható felhasználó által futtatott protokoll egy absztrakciója, A_i $(1 \leq i \leq k)$ jelenti a támadó csomópontokat, amelyek szintén megfelelnek G_{L_3} gráf csúcsainak. C Turing-gép reprezentálja a rádiókapcsolatokat, vagyis G_{L_3} éleit. Feladata a hozzá kapcsolódó átmeneti tárak közötti üzenetcsere megvalósítása. Minden gép probabilisztikus működésű. Mivel a valós világ modell felépítése és működése szinte teljesen azonos a 4.2.1. részben tárgyalt modellel, ezért a részletes ismertetéstől itt most eltekintek.

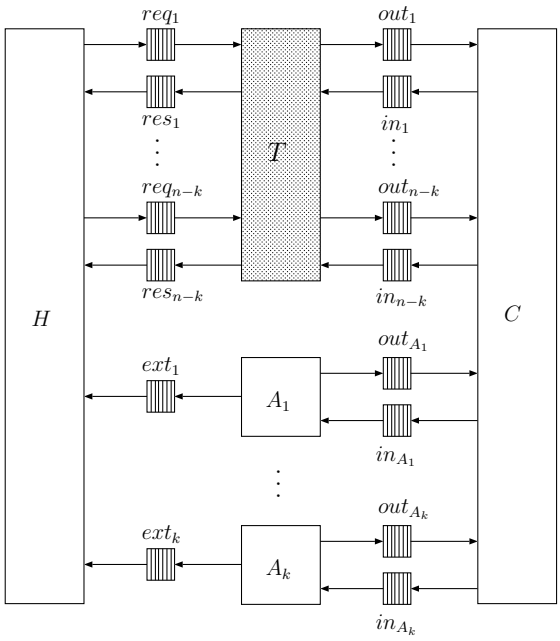
A számításnak akkor van vége, amikor H a leállási állapotába kerül. A mi esetünkben (amikor A_j-k $(1 \leq j \leq k)$ több egymástól független kérésüzenetet küldhetnek H számára) ez akkor történik meg, amikor H beolvasta mindegyik res_i $(1 \leq i \leq n-k)$ bemeneti tárból a választ, ami megfelel a req_i-ben elhelyezett kérésnek. Ez a válasz lehet egy time-out esemény is. $sys^{\mathsf{real}}_{conf,\mathcal{A}}$ kimenete a megbízható csomópontokhoz tartozó útvonalválasztó táblák összessége, vagyis a 7.2. részben definiált négyesek egy halmaza. Jelöljük ezt a kimenetet $view^{\mathsf{real}}_{conf,\mathcal{A}}(r)$-val, ahol $r = (r_I, r_1, \ldots, r_{n-k}, r_{A_1}, \ldots, r_{A_k}, r_C)$. Ezenkívül $view^{\mathsf{real}}_{conf,\mathcal{A}}$ jelölje a $view^{\mathsf{real}}_{conf,\mathcal{A}}(r)$ véletlen valószínűségi változót, amikor r-t egyenletesen választjuk.

7.3.2. Az ideális világ modell

Az ideális világ modell megfelel egy redukált $conf = (G_{L_3}(E, V, \mathcal{L}, \mathcal{C}_{link}, \mathcal{C}_{node}), V^*)$ konfigurációnak és egy $\mathcal{A}$ támadónak, ahol $\mathcal{A}$ egy Aktív-y-x támadó. Jelöljük ezt a rendszert $sys^{\mathsf{ideal}}_{conf,\mathcal{A}}$-val. Továbbá legyen $k = |V^*|$ és $n = |V|$. $sys^{\mathsf{ideal}}_{conf,\mathcal{A}}$ a $\{H, T, A_1, \ldots, A_k\}$ halmazból áll, ahol az egyes halmazelemek egy-egy Turing-gépnek felelnek meg, ahol H ugyanazt jelenti mint valós világ modell esetben, míg T modellezi a protokoll ideális működését. A_i $(1 \leq i \leq k)$ jelenti az ideális világbeli támadónak megfelelő csomópontokat. T és A_i-k probabilisztikus működésűek. Mivel a modell T gépet leszámítva úgy működik mint a valós világ modell esetében, ezért itt csak T működésének ismertetésére szorítkozom. A 7.1. ábra szemlélteti az ideális modellben az egyes Turing-gépek kapcsolatait.

T gép az M_i $(1 \leq i \leq n-k)$ gépek működését emulálja azzal a különbséggel, hogy T-t $conf$ konfigurációval inicializáljuk, így az detektálni képes ha a rendszer inkorrekt állapotba kerül. Ha ez bekövetkezik, akkor T megjegyzi, hogy a rendszer inkorrekt állapotba került, de maga a számítás folytatódik mintha semmi sem történt volna.

A számításnak akkor van vége, amikor H a leállási állapotába kerül. A mi esetünkben (amikor A_j-k több egymástól független kérésüzenetet küldhetnek H számára) ez akkor történik meg, amikor H beolvasta mindegyik res_i bemeneti tárból a választ, ami megfelel a req_i-ben elhelyezett kérésnek. Ez a válasz lehet egy time-out esemény is. $sys^{\mathsf{ideal}}_{conf,\mathcal{A}}$ kimenete vagy a megbízható csomópontokhoz tartozó útvonalválasztó táblák összessége – ha T nem detektált a futás közben inkorrekt állapotot – vagy egy speciális szimbólum, ami azt jelzi, hogy a futás közben valamikor a rendszer inkorrekt állapotba került. Jelöljük ezt a kimenetet $view^{\mathsf{ideal}}_{conf,\mathcal{A}}(r')$-val, ahol $r' = (r'_I, r'_1, \ldots, r'_{n-k}, r'_{A_1}, \ldots, r'_{A_k}, r'_C)$. Ezenkívül $view^{\mathsf{ideal}}_{conf,\mathcal{A}}$ jelölje a $view^{\mathsf{ideal}}_{conf,\mathcal{A}}(r')$ véletlen valószínűségi változót, amikor r'-t egyenletesen választjuk.



7.1. ábra. Kapcsolatok az ideális világ modellben ($sys^{\mathsf{ideal}}_{conf,\mathcal{A}}$) tábla alapú útvonalválasztás esetében.

7.4. A biztonságos útvonalválasztás definíciói

A biztonságosság a 4.3. részhez hasonlóan itt is szemléletesen a $view^{\mathsf{real}}_{conf,\mathcal{A}}$ és $view^{\mathsf{ideal}}_{conf,\mathcal{A}}$ eloszlások megkülönböztethetetlenségére vezethető vissza, tehát *egy biztonságos protokoll ideális világ modelljében elvárjuk, hogy az ne adjon vissza speciális szimbólumot, más szavakkal a rendszer ne kerüljön inkorrekt állapotba a szimuláció során.* Ez viszont nem mindig tartható, hiszen elenyésző de pozitív valószínűséggel a támadó általában mindig képes a felhasznált kriptográfiai primitívet megtörni (pl. megtippel egy digitális aláírást), vagyis a biztonságosság szemléletesen $view^{\mathsf{real}}_{conf,\mathcal{A}}$ és $view^{\mathsf{ideal}}_{conf,\mathcal{A}}$ eloszlások megkülönböztethetetlenségére vezethető vissza. A továbbiakban csak a felhasznált statisztikai biztonság definícióját mondjuk ki, a tökéletes és számításelméleti biztonság fogalmai valamint a bizonyítások technikája a 4.3. részbeli definíciókhoz és ismertetett technikához hasonlatosak.

7. definíció. *Egy tábla alapú útvonalválasztó protokoll statisztikailag biztonságos, ha bármely $conf$ konfigurációhoz és bármely valós világbeli $\mathcal{A}$ támadóhoz létezik egy olyan ideális világbeli $\mathcal{A}'$ támadó, hogy $view^{\mathsf{ideal}}_{conf,\mathcal{A}}$ és $view^{\mathsf{real}}_{conf,\mathcal{A}}$ eloszlások statisztikailag megkülönböztethetetlenek.* ♣

8. fejezet

Az SAODV biztonsága

Az SAODV (Secure AODV) [15] az Ad hoc On-demand Distance Vector (AODV) [16] útvonalválasztó protokoll egy „biztonságos” változata. A 7. részben ismertetett modellt ezen a példán szeretném szemléltetni. A 8.1. részben bemutatom röviden az SAODV működési mechanizmusát, majd a 8.2. részben megmutatom, hogy nem biztonságos a modellemben, nevezetesen két egyszerű támadást konstruálok ellene. Az A függelékben leírok egy további adaptív Aktív-2-x támadást az SAODV ellen.

8.1. Az SAODV működése

Az SAODV működési mechanizmusa nagyon hasonló az AODV működéséhez azzal a kivétellel, hogy az SAODV olyan kriptográfiai eljárásokat használ, amivel garantálni tudja az útvonalválasztó üzenetek integritását, és egyúttal megakadályozza a hop szám (hop count) információ illetéktelen manipulálását. Minden SAODV útvonalválasztó üzenet (rreq és rrep) rendelkezik egy változó és egy nem változó résszel. A nem változó részbe tartozik a csomóponti sorozatszám, a forrás- és célcsomópont címe valamint a kérés azonosítója. A változó rész tartalmazza a hop szám információt. Minden részt különböző kriptográfiai eljárással védenek a nem kívánt módosítástól.

A nem változó részt a forrás (rreq esetén) és a cél (rrep esetén) digitális aláírásával védik. Ez biztosítja, hogy a nem változó részek illetéktelen módosítása detektálhatóvá válik.

Ahhoz, hogy a hop szám információ illetéktelen módosítását megakadályozzák, a szerzők hash-láncok használatát javasolják. Amikor egy csomópont egy rreq vagy rrep üzenetet küld, először beállítja a HopCount mező értékét 0-ra, majd a MaxHopCount mező értékét a TimeToLive értékre (ez származhat a IP fejlécből is). Ezután generál egy random *seed* számot, és ezt elhelyezi a Hash mezőbe. Mindezek után kiszámolja a TopHash értéket úgy, hogy a *seed* értéket iteratívan hasheli MaxHopCount számszor egy publikus h hash-függvénnyel ($\mathsf{TopHash} = h^{\mathsf{MaxHopCount}}(seed)$). A MaxHopCount és TopHash mezők az üzenet nem változó részéhez tartoznak, míg a HopCount és a Hash mezők a változó részhez. Minden közbenső csomópont, amely megkapja az rreq vagy rrep üzeneteket, hasheli a Hash mező értékét ($\mathsf{MaxHopCount} - \mathsf{HopCount}$) számszor ($h^{\mathsf{MaxHopCount}-\mathsf{HopCount}}(\mathsf{Hash})$), és ellenőrzi, hogy az így kapott érték egyezik-e a TopHash értékkel. Ha igen, akkor az üzenet továbbítása előtt a csomópont a HopCount mezőt eggyel növeli, majd frissíti a Hash értékét úgy, hogy eggyel tovább

hasheli annak értékét (Hash = h(Hash)). A MaxHopCount, HopCount, Hash, TopHash mezők valamint h részletes specifikációja megtalálható [15]-ban.

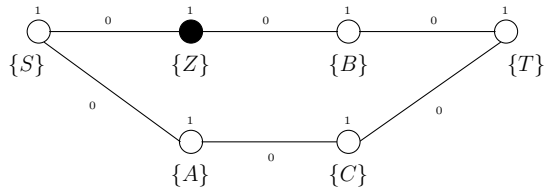
A hash-lánc használatának ötlete abban rejlik, hogy ha adottak a Hash, MaxHopCount és TopHash mezők értékei, akkor bárki ellenőrizheti a HopCount mező értékét. Másrészről a hash-láncban az előző érték csak elhanyagolható valószínűséggel számolható ki a Hash mező értékéből a felhasznált hash függvény egyirányú tulajdonsága miatt. Ez a mechanizmus biztosítja, hogy a támadó nem képes a hop számot csökkenteni, így nem képes azt sem elérni, hogy egy útvonal rövidebbnek látszódjon a forrás vagy cél számára, mint amilyen az valójában. Ugyanakkor ez – mint később látni fogjuk – nem teljesül általában a valóságban, hiszen egy támadó csomópont a HopCount mező érték inkrementálása és a Hash mező frissítése nélkül szabadon továbbíthat egy rreq vagy rrep üzenetet, amellyel így támadható lesz a protokoll.

8.2. Egyszerű támadások az SAODV ellen

A 7.4. részbeni definíciónak megfelelően egy tábla alapú útvonalválasztó protokoll akkor biztonságos, ha garantálja, hogy a megbízható csomópontok tábláiba inkorrekt állapotot okozó bejegyzések csak elhanyagolható valószínűséggel kerülnek a támadó csomópontok manipulációja által. SAODV esetében egy v csomópont akkor hoz létre egy bejegyzést a táblájában egy ℓ_{tar} azonosítójú csomópont számára, ha az egy elég friss üzenetet kap, amihez ℓ_{tar} egy érvényes digitális aláírása tartozik. Valójában a tény, hogy ez az üzenet megérkezett v-hez azt jelenti, hogy szükségszerűen léteznie kell egy útvonalnak v és egy ℓ_{tar} azonosítójú csomópont között, máskülönben az üzenet nem érhette volna el v-t. Viszont az SAODV azt nem képes biztosítani, hogy a következő hop és a hop szám információk az újonnan generált táblabejegyzésben korrektek legyenek. Ezt illusztrálja a következő két egyszerű példa.

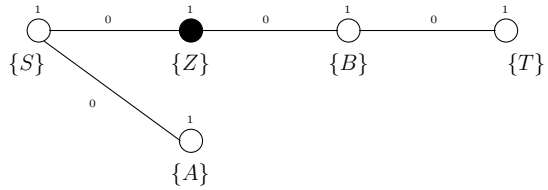
1. támadás. Vegyük alapul a 8.1. ábrán látható konfigurációt. Mivel az SAODV a hop számot használja a útvonal hosszának megállapításához, ezért a csomóponti költségeket konstans 1-nek, míg az adatkapcsolati költségeket konstans 0-nak vesszük minden csomópont és adatkapcsolat esetében. Tegyük fel, hogy az S azonosítójú csomópont egy útvonalkeresést kezdeményez a T azonosítójú csomópont felé. Amikor az rreq üzenet a Z azonosítójú támadó csomópontot eléri, az nem növeli a hop számot, és ennek megfelelően nem is frissíti a hash-értéket. Így amikor ezt az üzenetet a T csomópont megkapja, akkor az létrehoz egy $(S, B, 1)$ bejegyzést a táblájában. Ezt a bejegyzést T felül sem fogja írni amikor a másik rreq üzenetet megkapja a C azonosítójú csomópont felől, mivel az utóbbi egy 2 értékű hop számmal fog rendelkezni. Ez azt jelenti, hogy az egész rendszer egy inkorrekt állapotban fog megállni, mivel nincs olyan útvonal a hálózatban, amely egy T azonosítójú csomópontból indul, egy B azonosítójú csomóponton halad át és egy S azonosítójú csomópontban végződik, és amelynek a költsége kisebb vagy egyenlő mint 1.

Fontosnak tartom megjegyezni, hogy az SAODV ezen gyengeségét már annak szerzői is felfedezték és meg is említették (lásd 5.3.5. részt a [15]-ban). Az én célom csak annyi volt, hogy bemutassam a modellem használhatóságát egy egyszerű valós példán.



8.1. ábra. Egy konfiguráció, ahol SAODV protokollt futtatva egy támadó elérheti, hogy a T azonosítójú csomópont egy inkorrekt költségértékű bejegyzést hozzon létre az útvonalválasztó táblájában.

2. támadás. Vegyük alapul a 8.2. ábrát. Ismét tegyük fel, hogy az S azonosítójú csomópont egy útvonalkeresést kezdeményez a T azonosítójú csomópont felé. Továbbá tegyük fel, hogy egy rreq üzenet elérte a célt, és hogy az válaszolt egy megfelelő rrep üzenettel. Amikor ez az rrep üzenet eléri a Z azonosítójú támadó csomópontot, akkor az továbbítja a kapott rrep üzenetet az S azonosítójú csomópontnak A nevében. Ennek következtében az S azonosítójú csomópont létrehoz egy $(T, A, 2)$ táblabejegyzést. Látható viszont, hogy nem létezik olyan út S azonosítójú csomópontból T azonosítójú csomópont felé, amely egy A azonosítójú csomóponton halad át. Más szavakkal a rendszer ismét inkorrekt állapotban állt meg. Ismereteink szerint az SAODV ezen gyengeségét még nem publikálták.



8.2. ábra. Egy konfiguráció, ahol SAODV protokollt futtatva egy támadó elérheti, hogy S azonosítójú csomópont egy inkorrekt bejegyzést hozzon létre az útvonalválasztó táblájában T azonosítójú célcsomópontra helytelen A – mint következő hop – értékkel.

9. fejezet

Az ARAN biztonsága

Az ARAN (Authenticated Routing for Ad hoc Networks) szintén egy másik biztonságos tábla alapú útvonalválasztó protokoll ad hoc hálózatok számára, amelynek részletes leírása [17]-ben olvasható. Először röviden bemutatom az ARAN működését a 9.1. részben, majd formálisan bebizonyítom a 9.2. részben, hogy ez biztonságos a 7. részben definiált modell szerint.

9.1. Az ARAN működése

Az SAODV-hez hasonlóan az ARAN is nyilvános kulcsú kriptográfiát használ az útvonalválasztó üzenetek (rreq és rrep) integritásának a védelmére. Kezdetben egy S forráscsomópont útvonalkeresést kezdeményez azzal, hogy többesszórással továbbítja a következő üzenetet minden szomszédja számára.

$$(\mathsf{RREQ}, T, cert_S, N_S, t, Sig_S)$$

ahol RREQ jelzi, hogy ez egy rreq üzenet, S és T rendre a forrás- és célcsomópont azonosítói, N_S egy S által generált nonce (egyszer használatos véletlen érték), t az aktuális időbélyeg, $cert_S$ a forráscsomópont nyilvános kulcsának tanúsítványa és Sig_S pedig a forrás aláírása mindezeken az elemeken. N_S egy monoton módon növő érték, amely a t és S értékekkel együtt egyértelműen azonosítja az üzenetet, így detektálhatóvá és eltávolíthatóvá válnak a duplikált rreq és rrep üzenetek.

Később a közbenső csomópontok is aláírják a többesszórás miatt a hálózatban terjedő kérést. Így az rreq üzenetek a következő formával rendelkeznek általános esetben:

$$(\mathsf{RREQ}, T, cert_S, N_S, t, Sig_S, Sig_A, cert_A)$$

ahol A annak a közbenső csomópont azonosítója, amely aktuálisan a kérést küldte. Amikor A egy szomszédját, legyen ez most B, eléri ez a kérés, akkor az ellenőrzi mindkét aláírást és a nonce érték frissességét. Ha ezek az ellenőrzések sikeresek, akkor B bejegyzi magának az útvonalválasztó táblájába, hogy létezik egy út S csomópontba (mint célcsomópontba) A csomóponton (mint következő hopon) keresztül. Ezekután B eltávolítja A tanúsítványát és aláírását, aláírja a kérést, hozzáfűzi a saját tanúsítványát és aláírását az üzenethez, majd a

kapott következő üzenetet többesküldéssel továbbítja a szomszédainak:

$$(\mathsf{RREQ}, T, cert_S, N_S, t, Sig_S, Sig_B, cert_B)$$

Amikor a T célcsomópontot eléri az első rreq üzenet, amely ehhez az útvonalkereséshez tartozik, akkor T elvégzi azokat az ellenőrzéseket és módosításokat, amelyeket hasonló módon a közbenső csomópontok is elvégeztek. Ezután egy rrep üzenettel válaszol a kérésre. Ez az rrep üzenet az rreq üzenetek által felfedezett úton terjed visszafele a szomszédok közötti pont-pont címzéssel. A T által küldött rrep üzenet a következő formával rendelkezik:

$$(\mathsf{RREP}, S, cert_T, N_S, t, Sig_T)$$

ahol RREP jelzi, hogy ez egy rrep üzenet, N_S és t pedig rendre a nonce és időbélyeg értékek, amelyek az rreq üzenetből származnak, S a forráscsomópont azonosítója, $cert_T$ a T célcsomópont nyilvános kulcsának tanúsítványa, míg Sig_T a T aláírása mindezeken az elemeken.

Az rreq üzenetekhez hasonlóan az rrep üzenetek is aláírásra kerülnek a közbenső csomópontok által. Így az rrep üzenetek általános formája a következő:

$$(\mathsf{RREP}, S, cert_T, N_S, t, Sig_T, Sig_B, cert_B)$$

ahol B annak a csomópontnak az azonosítója, amely éppen továbbította az rrep üzenetet.

Egy A csomópont, amelyet elér az rrep üzenet, ellenőrzi a válaszban található mindkét aláírást, és ha mindkettő érvényes, akkor A továbbítja a választ annak a szomszédjának, amelytől előzőleg a válaszhoz tartozó kérést kapta. De mielőtt ezt megtenné A, eltávolítja a B csomópont tanúsítványát és aláírását az üzenetről, és elhelyezi a válaszban a saját tanúsítványát és aláírását:

$$(\mathsf{RREP}, S, cert_T, N_S, t, Sig_T, Sig_A, cert_A)$$

Ezeken felül A még létrehoz egy új bejegyzést az útvonalválasztó táblájában egy útvonalról, amely T csomópontba (célcsomópont) B csomóponton (mint következő hopon) keresztül halad.

Amint az a leírásból is látszik az ARAN nem használ hop számokat az útvonal hosszának a méréséhez. Ehelyett a csomópontok az elsőként beérkezett útvonalválasztó üzenet alapján frissítik a táblájukat. Minden később jövő üzenetet, amely ugyanahhoz az útvonalkereséshez tartozik, figyelmen kívül hagynak. Ez más szavakkal azt jelenti, hogy az ARAN nem a legrövidebb, hanem a leggyorsabb útvonalat választja ki, vagyis azt, amelyiken az útvonalkereséshez tartozó üzenetek a legrövidebb időn belül megérkeztek. Így itt az úthossz-metrika a üzenetek terjedésének (fizikai) ideje.

9.2. Az ARAN biztonsága

3. tétel. *Az ARAN statisztikailag biztonságos, ha a felhasznált aláíróséma is statisztikailag biztonságos választott nyíltszövegű támadás esetén.* ♢

Bizonyítás Mivel az ARAN az üzenetek terjedési idejét használja úthossz-metrikának, ezért feltételezzük, hogy a csomóponti költségek a mi modellünk szerint jelen esetben az üzenetek minimális feldolgozási ideje (a csomópontoknál), és a kapcsolati költségek pedig a minimális adattovábbítási időt reprezentálják (a csomópontok közötti kapcsolatokon). Ezeken felül még azzal a pesszimista feltevéssel is élünk, hogy a támadó csomópontoknál figyelembe vett feldolgozási idő 0, ami formálisan azt jelenti, hogy $\mathcal{C}_{node}(v) = 0$ minden $v \in V^*$-re.

A modellünkkel összhangban azt is feltesszük, hogy minden egyes útvonalválasztó táblabejegyzés explicit módon tartalmazza az útvonal hosszának az értékét is. A mi esetünkben ennek az értéknek a metrikája az az idő, ami ahhoz szükséges, hogy egy útvonalkereséshez tartozó üzenet (rreq vagy rrep) a létrehozásától számítva elérje azt a csomópontot, amely a bejegyzést létrehozza. Habár ezek az időértékek nincsenek explicite reprezentálva az ARAN útvonalválasztó tábláiban, ez még nem gyengíti a protokoll biztonságának bizonyításának értékét. Másként leírva mi ugyanazokat a táblabejegyzéseket használjuk a modellben mint az ARAN, figyelembe véve a cél és a következő hop azonosítókat.

Ahhoz, hogy bebizonyíthassuk az ARAN biztonságosságát, találnunk kell egy megfelelő $\mathcal{A}'$ ideális világbeli támadót bármely valós világbeli $\mathcal{A}$ támadóhoz, hogy a 7. definíció teljesüljön. A modellünk konstrukciója alapján erre egy alkalmas jelölt az $\mathcal{A}' = \mathcal{A}$ támadó, mivel ebben az esetben a valós és ideális világ modellbeli támadók lépései pontosan ugyanazok (ha persze ugyanazzal a véletlen bemenettel lettek inicializálva). Ha nem került az ideális világbeli rendszer inkorrekt állapotba a számítás során, akkor nem csak az egyes lépések, hanem a két modell kimenete is ugyanaz lesz. Másrészről, ha egy inkorrekt állapotba jut a rendszer az ideális világ modellben, akkor a két világ modell kimenete különbözni fog, mivel ekkor az ideális világ modell kimenete egy speciális szimbólum lesz. Így a 7. definíció szerint a protokoll akkor lesz biztonságos, ha inkorrekt állapot csak elhanyagolható valószínűséggel fordul elő. A következőkben megmutatjuk, hogy ez teljesül az ARAN esetében.

Az inkorrekt állapot elérése azt jelenti, hogy a megbízható csomópontok egyike, pl. v, egy inkorrekt bejegyzést hoz létre az útvonalválasztó táblájában. Legyen ez az inkorrekt bejegyzés $(\ell_{tar}, \ell_{nxt}, c)$. Mivel $v \in V \setminus V^*$, ezért az csak akkor módosít (vagy hoz létre új) bejegyzést a táblájában, ha egy olyan üzenetet kapott, amelyet ℓ_{tar} is – mint a célcsomópont – és ℓ_{nxt} is – mint a következő hop ℓ_{tar} fele – aláírt, $\ell_{nxt} \in \mathcal{N}(v)$, valamint c idő volt szükséges ahhoz, hogy az üzenet a küldőtől eljusson v-hez. Ekkor $(\ell_{tar}, \ell_{nxt}, c)$ bejegyzés az alábbi három esetek egyikében lehet inkorrekt:

1. Nem létezik útvonal v csomópontból egy olyan másik csomópontba, amely az ℓ_{tar} azonosítót használja.

2. Léteznek útvonalak v csomópontból egy olyan másik csomópontba, amely az ℓ_{tar} azonosítót használja, de egyik sem megy keresztül v olyan szomszédján, amely az ℓ_{nxt} azonosítót használja.

3. Léteznek útvonalak v csomópontból egy olyan másik csomópontba, amely az ℓ_{tar} azonosítót használja, és keresztül megy v olyan szomszédján, amely az ℓ_{nxt} azonosítót használja, de mindegyik költsége nagyobb mint c.

Az első esetben ha ℓ_{tar} aláírása az útvonalkereséshez tartozó üzenetben nem lett hamisítva,

akkor a tény, hogy v-t elérte az üzenet azt bizonyítja, hogy létezik egy útvonal v és egy másik ℓ_{tar} azonosítót használó csomópont között (mivel máskülönben az üzenet nem érhette volna el v-t). Így az 1. eset csak akkor lehetséges, ha ℓ_{tar} aláírását sikerült valakinek hamisítania. Ez viszont csak elhanyagolható valószínűséggel történhetett meg a felhasznált aláíróséma feltételezett biztonsága miatt.

A 2. esetben ha ℓ_{nxt} aláírása nem lett megkonstruálva az üzenetben, akkor v', ahol $(v, v') \in E$, valamint $\ell_{nxt} \in \mathcal{L}(v')$, valóban látta és aláírta az üzenetet. Jelen esetben ugyanaz az érvelés alkalmazható v' esetében mint amit bemutattunk az 1. esetben v csomópontra: a tény, hogy v'-t elérte az üzenet azt bizonyítja, hogy létezik egy útvonal v' és egy másik ℓ_{tar} azonosítót használó csomópont között (mivel máskülönben az üzenet nem érhette volna el v-t), és így létezik útvonal v és egy ℓ_{tar} azonosítót használó csomópont között, amely keresztül megy v' csomóponton (mivel $(v', v) \in E$). Ez azt jelenti, hogy a 2. eset csak akkor lehetséges, ha ℓ_{tar} vagy ℓ_{nxt} vagy mindkettő aláírását hamisították, aminek a valószínűsége ugyancsak elhanyagolható.

Végezetül a 3. esetben legyen R azon létező útvonalaknak a halmaza, amelyek v-ből indulnak, ℓ_{tar} azonosítót használó csomópontban végződnek, valamint keresztül mennek v egy olyan szomszédján, ami ℓ_{nxt} azonosítót használja. Legyen c' az R-ben levő útvonalak költségei közül a legkisebb. Feltevésünk szerint $c' > c$. Ha ℓ_{tar} és ℓ_{nxt} aláírások a v-által fogadott üzenetben nem hamisítottak, akkor az üzenetnek valamelyik R-ben levő útvonal szerint kellett haladnia. Viszont tudjuk, hogy ez az üzenet nem érhette el v-t olyan c időn belül, amelyre $c < c'$, hiszen a csomóponti és kapcsolati költségek a minimális üzenetfeldolgozási és adattovábbítási időt reprezentálják a csomópontoknál illetve a köztük levő kapcsolatoknál. Más szavakkal a támadó nem képes felgyorsítani az üzenetek átviteli és feldolgozási idejét a megbízható csomópontoknál. Így a 3. eset csak akkor lehetséges, ha vagy ℓ_{tar} vagy ℓ_{nxt} vagy mindkettő aláírását hamisították, aminek a valószínűsége szintén elhanyagolható. ∎

10. fejezet

ASAR: Egy bizonyított biztonságú hibrid protokoll

Az ASAR (A Secure Ad Hoc Routing) egy olyan általam tervezett hibrid protokoll, amely a forrás és tábla alapú útvonalválasztó protokollok tulajdonságait ötvözi. Tulajdonképpen az eddigi eredmények szintéziseként tekinthető. Az észrevételem az volt, hogy nem létezik olyan tábla alapú protokoll, amely biztonságos a 7. részben definiált modellben, és hop számot használ az útvonal hosszának a megállapítására. Ahogy azt a 9.2. részben megmutattam az ARAN ugyan egy biztonságos protokoll ebben a modellben, de ez az üzenetek terjedési idejét használja fel mint úthossz-metrikát. Ez viszont természetesen korlátot is szab a protokoll alkalmazhatóságának. Az fizikai idő jellegű metrikával az ARAN ugyan hatékonyabban védekezik a csatornás támadás ellen, de egyúttal sokkal érzékenyebb a csomóponti késleltetésekre is. Ebben rejlik az ASAR előnye és egyben hátránya is.

Az ASAR nem használ fizikai idő jellegű metrikát, ennek ellenére, ahogy azt a 10.2. részben látni fogjuk, mégis bizonyíthatóan biztonságos a modellünkben. Természetesen a hop szám használata maga után vonja annak minden hátrányát is. Ezek közül a legfontosabb, hogy véleményem szerint nem lehet állandó hosszúságú útvonalválasztó üzeneteket garantálni az útvonalfelderítés során, ha modellünkkel összhangban Aktív-y-x támadót tételezünk fel, hiszen intuitíve látható, hogy mindegyik közbenső csomópontnak hitelesítenie kell magát az egyes útvonalválasztó üzenetekben. Így viszont már nem tartható az üzenetek konstans hossza. Innen már egyszerűen az is következik, hogy az útvonalfelderítési szakaszban alkalmazható az endairA (6. rész) protokoll megfelelő része a megfelelő kiegészítésekkel, míg a másik, útvonalkarbantartó szakasz, már teljesen a tábla alapú protokollok mintájára működik, vagyis minden csomópont egy lokális útvonalválasztó tábla alapján továbbítja az adatcsomagokat. Másként fogalmazva míg az útvonalválasztó üzenetek (rreq és rrep) explicite az egész útvonalat tartalmazzák, addig az adatcsomagok mindig csak a célcsomópont címét. A 10.1. részben először bemutatom az ASAR útvonalfelderítési szakaszának működését, majd végül a 10.2. részben formálisan is igazolom ennek biztonságát.

10.1. Az ASAR specifikációja

Mivel, ahogy azt a bevezetőben említettem, az ASAR útvonalfelderítési szakasz az endairA-hoz hasonlatos, ezért itt most csak az alapvető eltéréseket írom le a 6.1. résztől. Itt is feltételezzük, hogy a útvonalkeresést kezdeményező csomópont és a célcsomópont megbízhatóak. A működés a 6.1. táblázat szerinti, a válaszüzenetek (rrep) terjedésével minden az útvonalban résztvevő megbízható csomópont a következő műveleteket végzi el. Legyen a válaszban szereplő útvonal $(\ell_1, \ldots, \ell_p)$ most a forrás- és célcsomópont azonosítókkal (ℓ_1, ℓ_p) együtt. Az ellenőrzéseket végző megbízható csomópont azonosítója legyen ℓ_i $(1 \leq i \leq p)$.

1. ℓ_i szerepel-e az útvonalban elhelyezkedő azonosítók között.

2. Ha $2 \leq i \leq p-1$, akkor ℓ_{i-1} és ℓ_{i+1} azonosítók valóban szomszédos csomópontokhoz tartoznak-e, illetve ha $i = 1$, akkor ℓ_2 szomszédos-e, vagy ha $i = p$, akkor ℓ_{p-1} szomszédos-e.

3. Ellenőrzi az összes aláírást az üzenetben, valamint azt, hogy minden aláíráshoz kölcsönösen egyértelműen hozzárendelhető-e pontosan egy azonosító az útvonalból.

4. Ha $i \leq p-1$, akkor frissíti útvonalkereső tábláját minden olyan táblabejegyzésre, ahol a bejegyzéshez tartozó célcsomópont azonosítója ℓ_j $(i+1 \leq j \leq p)$. A bejegyzést a $(\ell_j, \ell_{i+1}, j-i-1)$ hármas alkotja, ahol ℓ_j a célcsomópont azonosítója, ℓ_{i+1} a következő hop azonosítója ℓ_j felé, $j-i-1$ pedig ℓ_i és ℓ_j között levő csomópontok (hopok) feltételezett száma.

Ha az ℓ_1 azonosítójú forráscsomópontot is elérte a válaszüzenet, akkor az a sikeres verifikációkat és frissítéseket követően megkezdheti az adatcsomagok küldését ℓ_2 szomszédja felé olyan üzenetek formájában, amelyek csak a célcsomópont ℓ_p címét tartalmazzák. Minden további köztes csomópont a táblájának megfelelően továbbítja ezeket az üzeneteket.

10.2. Az ASAR biztonsága

4. tétel. *Az ASAR statisztikailag biztonságos, ha a felhasznált aláíróséma is statisztikailag biztonságos választott nyíltszövegű támadás esetén. Továbbá minden korrekt Q rendszerállapotban igaz $sys^{\mathsf{real}}_{conf,\mathcal{A}}$ rendszerben, hogy minden $(v, \ell_{tar}, \ell_{nxt}, c) \in Q$ elemre*

$$0 \leq c - \sum_{i=2}^{p-1} \mathcal{C}_{node}(v_i) - \sum_{i=1}^{p-1} \mathcal{C}_{link}(v_i, v_{i+1}) \leq y$$

ahol $(v_i, v_{i+1}) \in E$ minden $1 \leq i < p$-re, $v_1 = v$, $\ell_{tar} \in \mathcal{L}(v_p)$, $\ell_{nxt} \in \mathcal{L}(v_2)$, míg y a támadó által használt csomópontazonosítók száma. ♢

Bizonyítás (Vázlat) Mivel jelen esetben az útvonalhossz-metrikának a csomópontok (hopok) száma felel meg, ezért a csomóponti költség értelemszerűen konstans 1 minden csomópontnál, míg a csomópontok közötti adatkapcsolati költség konstans 0 minden kapcsolat esetén.

Ahhoz, hogy bebizonyíthassuk az ASAR biztonságosságát, találnunk kell egy megfelelő $\mathcal{A}'$ ideális világbeli támadót bármely valós világbeli $\mathcal{A}$ támadóhoz, hogy a 7. definíció teljesüljön. A modellünk konstrukciója alapján erre egy alkalmas jelölt az $\mathcal{A}' = \mathcal{A}$ támadó. A 7. definíció szerint a protokoll akkor lesz biztonságos, ha inkorrekt állapot csak elhanyagolható valószínűséggel fordul elő. A következőkben megmutatjuk, hogy ez teljesül az ASAR esetében is.

Az inkorrekt állapot elérése azt jelenti, hogy a megbízható csomópontok egyike, pl. v, egy inkorrekt bejegyzést hoz létre az útvonalválasztó táblájában. Legyen ez az inkorrekt bejegyzés (ℓ_j, ℓ_{i+1}, c), és legyen az ezt okozó rrep üzenet

$$msg = (\mathsf{rrep}, \ell_1, \ell_p, (\ell_2, \ldots, \ell_{p-1}), (sig_{\ell_p}, \ldots, sig_{\ell_{i+1}}))$$

ahol $\ell_i \in \mathcal{L}(v)$ $(1 \leq i \leq p-1)$ és $i+1 \leq j \leq p$. A 2. tételből következik, hogy ha v elvégezte a szükséges verifikációkat, akkor nagy valószínűséggel[1] $h = (\ell_i, \ldots, \ell_j)$ plauzibilis útvonal, hiszen a tételből tudjuk, hogy a $(\ell_1, \ldots, \ell_p)$ útvonal plauzibilis, és h ennek egy része. Vagyis ha h nem lenne plauzibilis, akkor $(\ell_1, \ldots, \ell_p)$ sem lenne az, amivel viszont ellentmondásra jutnánk. Így ha v egy olyan (ℓ_j, ℓ_{i+1}, c) bejegyzést hoz létre a táblájában, amire

1. nem létezik útvonal v csomópontból egy olyan másik csomópontba, amely az ℓ_j azonosítót használja, vagy

2. léteznek útvonalak v csomópontból egy olyan másik csomópontba, amely az ℓ_j azonosítót használja, de egyik sem megy keresztül v olyan szomszédján, amely az ℓ_{i+1} azonosítót használja,

akkor h útvonal nem lehet plauzibilis, ami ellentmondáshoz vezet. A továbbiakban legyen a 2. definíció szerint a h útvonalnak megfeleltetett csúcsok sorozata $(v_1, \ldots, v_k)$ $(2 \leq k \leq j)$, ahol $\ell_i \in \mathcal{L}(v_1)$ és $\ell_j \in \mathcal{L}(v_k)$.

Legyen az

$$x = \sum_{t=2}^{k-1} \mathcal{C}_{node}(v_t) + \sum_{t=1}^{k-1} \mathcal{C}_{link}(v_t, v_{t+1})$$

a $(v_1, \ldots, v_k)$ útvonal költsége, ahol $\sum_{t=1}^{k-1} \mathcal{C}_{link}(v_t, v_{t+1}) = 0$, míg $\mathcal{C}_{node}(v) = 1$ minden v csúcs esetén. A következőkben belátjuk, hogy nem elhanyagolható valószínűséggel

$$x \leq c \leq x + y$$

amiből átrendezéssel következik a tétel második állítása.

A plauzibilis útvonal definíciójából tudjuk, hogy h útvonal egyértelműen particionálható olyan részsorozatokra, amely részsorozatok egyértelműen hozzárendelhetőek $(v_1, \ldots, v_k)$ csúcsokhoz.

Tegyük fel, hogy $x > c$, ahol nyilván $x = k - 2$. Tudjuk, hogy $c = j - i - 1$, így kapjuk, hogy $j - i + 1 < k$. Ez azt jelenti, hogy legalább egy csomóponthoz nem tartozik azonosító az útvonalban, hiszen $j - i + 1$ a h útvonalat alkotó csomópontazonosítók száma. Így h útvonal nem lehet plauzibilis, vagyis ellentmondásra jutottunk.

[1] nem elhanyagolható, 1-hez közeli valószínűséggel

Tegyük fel, hogy $x + y < c$ $(0 \leq y)$ vagyis $c - x > y$. Innen következik, hogy

$$(j - i + 1) - k > y \tag{1}$$

Tudjuk, hogy $j-i+1 \geq k$. Ekkor (1) csak úgy teljesülhet, ha h útvonal tartalmaz legalább $y+1$ darab támadó azonosítót, mivel egy megbízható csomópont pontosan egy azonosítót használ, valamint egy támadó csomópont a modell szerint nem használhat megbízható csomóponthoz tartozó azonosítót. Ez viszont csak úgy lehetséges, ha a támadó csomópont legalább egy azonosítót kétszer szúr be h útvonalba, ami viszont így már nem lenne plauzibilis, tehát ellentmondásra jutnánk. ∎

11. fejezet

Kapcsolódó munkák

A bizonyított biztonságelmélet kialakulása a XX. század végére tehető [9], viszont először ezt a számításelméleti megközelítést csak kulcscsere és hitelesítés problémákra alkalmazták [10]. Az összetettebb kriptográfiai primitívek biztonsága az erősen egyirányú függvény létezésére épít, jobban mondva arra a feltételezésre, hogy létezik számításelméleti szempontból nehéz probléma (NP-teljes probléma). Az összes kriptográfiai építőelem (álvéletlen generátor, álvéletlen függvény és permutáció, digitális aláírás, stb.) létezése indirekt módon bizonyítható egy redukciós módszerrel. Például ha létezik erősen egyirányú függvény, akkor létezik álvéletlen generátor is. A redukciós bizonyítás során azt látjuk be, hogy ha nem létezik álvéletlen függvény, akkor erősen egyirányú függvény sem létezhet. Hasonló redukcióval látható be az is, hogy ha nem létezik álvéletlen permutáció, akkor álvéletlen generátor sem létezhet. Így haladunk felfele egészen addig, amíg eljutunk az összetettebb primitívek (pl. digitális aláírás, MAC) biztonságának a bizonyításáig. Ebből rögtön kiderül az is, hogy ha valaki esetleg bizonyítaná, hogy nem léteznek számításelméleti szempontból nehéz problémák (pl. javasol egy hatékony algoritmust egy NP-teljes problémára), akkor az egész „modern" kriptográfia alapjaiban összeomlana.

Az eddigiek során több biztonságos ad hoc útvonalválasztó protokollt is javasoltak [6], viszont ezen protokollok biztonságát csak informálisan igazolták, ami maga után vonja ennek minden hátrányát és veszélyét is. A legtöbb esetben ez nem kielégítő „bizonyítás". Több más olyan munka is létezik [7], ahol bizonyított biztonságelmélettel foglalkoznak, viszont ezek az elemzések nem terjednek ki ad hoc útvonalválasztó protokollokra. Ezen protokollok tervezése során a szerzők először felsorolták a lehetséges támadásokat modellek segítségével vagy anélkül ([4], [15], [17]), majd figyelembe véve a hálózati sajátságokat és lehetőségeket terveztek egy olyan „biztonságos" protokollt, amely ellen az általuk felsorolt támadások nem kivitelezhetőek. Persze ezt is csak informális érveléssel bizonyították, ahelyett, hogy azt visszavezették volna a fent bemutatott redukciós módszer segítségével a felhasznált kriptoelem már bizonyított biztonságára. A probléma az volt, hogy nem volt megfogalmazva az, hogy konkrétan mit értünk egy ad hoc útvonalválasztó protokoll biztonságán. Ennek következtében aztán mindenki tervezett egy biztonságos protokollt a saját informális modelljében. Az általam használt támadómodell alapjai megtalálhatóak [4]-ben, de itt sem formális keretek között leírva.

A szimulációs paradigmát először [3]-ben alkalmazták ad hoc útvonalválasztásra (részletes általános leírása megtalálható [19]-ben), ahol már bebizonyították formálisan az Ariadne

gyengeségét Aktív-1-1 támadó esetén. Itt már szintén rámutattak a hash-láncok gyengeségére ad hoc környezetben, valamint hasonló támadást konstruáltak ebben a modellben az SRP ellen is. Ebben a modellben a támadó viszont csak egy csomóponttal és egy azonosítóval rendelkezett. Az itt javasolt modell így nem volt teljesen alkalmas Aktív-y-x támadómodell esetén, ami egy erősebb támadót tételez fel. Így módosítottam ezt a modellt, hogy legyen lehetőség általánosabb támadómodell esetén is formális bizonyításra. Az endairA protokollt szintén [3]-ban javasolták először, ahol viszont nem volt specifikálva az útvonalban résztvevő csomópontok által történő aláírások ellenőrzése. Bizonyítható, hogy enélkül az endairA sem biztonságos még Aktív-1-x támadó esetén sem, ahogyan azt a dolgozatban láttuk. Az endairA protokoll kiegészítéseit és biztonságának bizonyítását a modellben először [1]-ben dolgoztam ki a témavezetőm irányítása alatt.

A tábla alapú (distance vector routing) protokollok biztonságával [2]-ben foglalkoztam először, ahol szintén a témavezetőmmel együttműködve kidolgoztam az itt is bemutatott formális modellt.

A szimulációs paradigmán kívül más formális megközelítést is javasoltak [8] [11], de ezek vagy túl specifikusak és nem annyira használhatóak általánosabb esetben [11], vagy pedig olyan környezetben használják őket, ahol nem teljesülnek az alkalmazott módszer követelményei [8]. A másik de hasonló megközelítés az enyémhez a [20]-ban található, ahol szintén egy belső támadó modellezésére alkalmas formális modellt javasolnak. Ez a strand spaces modellhez [21] hasonló, amelyet kulcscsere protokollok formális verifikációjához javasoltak. Itt két tulajdonsággal definiálták a biztonságos útvonalválasztást. A létezés tulajdonsága (liveness property) szerint lehetséges útvonalat találni a hálózatban, míg a biztonságosság tulajdonsága (safety property) szerint ezen útvonalban nem szerepel támadó csomópont. Véleményem szerint ez irreális követelmény, hiszen a gyakorlatban ezt nem lehet garantálni, valamint a támadó csomópontok is becsületesen követhetik a protokoll működési szabályát. A forrás alapú protokollok hasonló biztonsági követelményét már megfogalmazták [22]-ban, de itt is csak informálisan. A [11] szintén egy másik formális módszerrel (CPAL-ES) végez elemzést, viszont ez csak az SRP protokollra hegyeződik ki, és nem elég általános. Például a biztonsági cél is túlságosan SRP specifikus, amivel így a módszer nem alkalmazható általánosabb esetben. Ráadásul egy olyan támadást hoz fel példának, ami inkább féreglyukakkal operál, ami az én véleményem szerint nem az útvonalválasztás elleni támadásnak minősül, hanem a szomszédok felderítése elleninek. Másrészről viszont a CPAL-ES jól automatizálható az én módszeremmel szemben. A [8]-ben BAN logikával [23] már elemezték szintén az SRP protokollt, de azt nem a megfelelő kontextusban tették, mivel a BAN logikában nincs arra lehetőség, hogy a protokollból származtatva definiáljuk a biztonságos útvonalválasztás célját. A másik probléma, hogy a BAN logikában feltételezzük, hogy minden protokollrésztvevő megbízható, és nem kompromitálnak titkos adatokat (pl. titkos kulcsokat). A BAN logika helytelen alkalmazására példa, hogy az SRP szerzői az elemzés ellenére nem fedezték fel a [3]-ban leírt támadást.

Az utóbbi években illetve évtizedekben a kriptográfiának két jól elkülöníthető megközelítése alakult ki, amely a témával foglalkozó közösséget is megosztja. Az egyik az egyszerű ugyanakkor hatékony *formális* nézőpont, a másik pedig a részletes *számításelméleti* megközelítés, melynek alapja a valószínűségszámítás és a komplexitáselmélet. Mindkét megközelítés

helyes és hasznos, hiszen mindkettő célja a biztonság bizonyítása, csak más eszközökkel és más absztrakciós szinten. A kriptográfia formális elemzése hasznos lehet olyan esetekben, ahol különböző rendszerek tervezése, analízise és implementációja során szükséges a kriptográfia használata. Ilyenkor kényelmes lehet például ha az adott rejtjelezési algoritmust nem vesszük figyelembe, hanem magasabb szintről közelítjük meg a követelményeket az adott kriptográfiai eljárással szemben. Több megközelítés lehetséges, melyek mind különböző technikákat [12][13] használnak a modellezésre (processz algebra, temporális logika, stb.). Ezek a módszerek a biztonságot garantálják egy felsőbb absztrakciós szinten. A számításelméleti megközelítés – ahol minden valószínűségszámításról és Turing-gépekről szól – nézőpont képviselői néha naívnak és túlságosan absztraktnak, a valóságot nem hűen tükröző módszernek tekintik a formális elemzést, ugyanakkor komplex rendszerek tervezésénél és a verifikációs műveletek automatizálásában vitathatatlan az előnye. Látható, hogy a két nézőpont között egy furcsa és nem túl előnyös „válaszfal" figyelhető meg. [5]-ben ezt a falat próbálják meg áthidalni úgy, hogy egy számításelméleti igazolást adnak a rejtjelezés formális modelljére. Ez viszont csak egy lépcsőfok a probléma megoldásában. Számomra mindez azért is fontos, mert a dolgozatban én is egy számításelméleti modellt javasoltam és használtam, viszont minden olvasó számára feltűnhet a bizonyítások hasonlósága, ami talán az automatizálás lehetőségét rejti magában egy alkalmas formális nyelv használatával [12].

12. fejezet

Összegzés

Ebben a dolgozatban leírtam egy olyan formális modellt, amiben precízen definiálható, hogy mit értünk biztonságos útvonalválasztás alatt mobil ad hoc hálózatokban, és így lehetőség nyílik egy adott útvonalválasztó protokoll biztonságosságát (vagy éppen nem biztonságosságát) igazolni ebben a modellben. A modell alapja a szimulációs paradigma, melyet kriptográfiai protokollok biztonságának a bizonyítására használnak fel. A dolgozat újdonsága, hogy ezt a modellt adoptálta mind forrás alapú mind pedig útvonalválasztó tábla (vagy távolsági vektor) alapú ad hoc útvonalválasztó protokollokra, és így sikerült egy olyan formális keretrendszert alkotni, amellyel a gyakorlatban is sikerrel lehet biztonsági elemzéseket végezni.

Az ad hoc hálózat statikus modellezésének bemutatása után formálisan definiáltam a hálózat dinamikus reprezentációját a szimulációs paradigma segítségével. Rámutattam arra, hogy az ad hoc hálózatok az alkalmi és önszerveződő jellegük miatt nem modellezhetőek az eddig megismert technikákkal és módszerekkel, hiszen azok nem veszik figyelembe ezen hálózatok alapvető tulajdonságait. Így nemcsak egy új hálózati modellt, de egy új támadómodellt is fel kellett építeni. Mind a forrás alapú mind pedig a tábla alapú protokolloknál részletesen leírtam a dinamikus működést reprezentáló valós és ideális világ modelleket, amelyek általánosan alkalmasak vezetéknélküli ad hoc hálózatok jellemzésére, és specifikusan ad hoc útvonalválasztó protokollok elemzésére is. A valós világ modell formalizálja egy útvonalválasztó protokoll valóságos működését, míg az ideális világ modell formalizálja egy biztonságos útvonalválasztó eljárással szembeni követelményeket. Ezen követelmény forrás alapú protokollok esetén az, hogy sohasem szabadna egy biztonságos útvonalválasztó eljárásnak nem plauzibilis útvonalat visszaadnia, míg tábla alapú protokollok esetén az, hogy sohasem juthat a protokoll futása során a rendszer inkorrekt állapotba. Természetesen léteznek olyan támadások, amelyek ellen vagy nem tudunk védekezni, vagy pedig túl költséges lenne ellenük védekezni, ezért ezeket mind megengedjük, toleráljuk, és beépítjük a modellbe. A biztonságos útvonalválasztás formális definícióját a valós és ideális világ modellek kimeneteinek a számításelméleti megkülönböztethetetlensége alapján adtam meg. A megkülönböztetés mértéke három különböző definíciót eredményezett, melyek közül gyakorlati szempontból a statisztikai megkülönböztetésnek van nagyobb jelentősége.

A modellek használhatóságát valóságos példákon szemléltettem. A dolgozat során látható volt, hogy rendkívül szövevényes támadások konstruálhatóak olyan protokollokkal szemben,

melyek biztonsága csak informális érveléssel igazolt. A forrás alapú protokollok elemzése során bemutattam az Ariadne protokoll működését MAC típusú üzenethitelesítés esetén, majd beláttam, hogy ez a protokoll nem biztonságos a modellemben. Ezekután leírtam egy olyan új protokoll (endairA) specifikációját, amelyről formálisan bebizonyítottam, hogy biztonságos a modellben. Rámutattam arra, hogy kisebb módosításokkal az endairA a gyakorlatban is használható protokoll lehetne. A tábla alapú protokollok esetén először az SAODV útvonalválasztó protokollt elemeztem, melynek során beláttam, hogy az nem biztonságos a modellemben. Az ARAN ezzel szemben egy biztonságos protokoll ugyanebben a modellben, amit formálisan is igazoltam. Az ARAN viszont nem hop számot használ úthossz-metrikaként, ezért terveztem egy új hibrid protokollt (ASAR), amely az endairA-hoz teljesen hasonlóan minden közbenső csomóponttal hitelesíti az útvonalválasztó üzenetet, viszont az adatcsomagok továbbítása már az útvonalfelderítés során feltöltött táblák segítségével történik. Az ASAR biztonsága szintén formálisan igazolt a modellben.

A modell egyértelmű hátránya, hogy nem kellően automatizálható, amire már korábban utaltam. Jelenleg a bizonyítások túl sok konstruktív ötletet tartalmaznak, amelyek egy gép számára jelenleg még kivitelezhetetlenné teszik ezeket a bizonyításokat. Egy lehetséges jövőbeli kutatási irányvonalat jelenthet ezen bizonyítások automatizálása például processz kalkulus segítségével (SPI [12], PI [13]), amely viszont a kriptográfiai primitívekre kriptográfiai szempontból eleve biztonságos fekete dobozokként tekint egy felsőbb absztrakciós szintről. Ezzel szemben a szimulációs paradigma nem él ezzel a feltételezéssel amint azt láttuk, hanem valószínűségi eszközökkel fejezi ki a biztonságosság fogalmát. Egy lehetséges megoldás lehet a kettő közötti szakadék áthidalása [5] ad hoc hálózatok esetén is.

Köszönetnyilvánítás

Ezúton szeretném megköszönni konzulensemnek dr. Buttyán Leventének és dr. Vajda Istvánnak a szakmai tanácsait, megjegyzéseit és segítségét. Ezenkívül köszönöm Bacsárdi Lászlónak a hasznos megjegyzéseit és észrevételeit, valamint Paráda Lászlónak, Veiland Tamásnak, Patakfalvi Tamásnak, Petri Csanádnak és Gál Csabának az egyéb hasznos segítséget.

Rövidítések, jelölések

Jelölés	*Magyarázat*
$\mathcal{A}$	Aktív-y-x támadó
$\mathcal{O}$	orákulum
L	csomópontazonosítók halmaza
L^*	támadó által használt csomópontazonosítók halmaza
$G_{L_2}(E,V)$	L_2 gráf
$G_{L_3}(E,V,\mathcal{L},\mathcal{C}_{link},\mathcal{C}_{node})$	L_3 gráf forrás alapú protokollok esetén
$G_{L_3}(E,V,\mathcal{L})$	L_3 gráf tábla alapú protokollok esetén
$\mathbb{G}$	L_3 gráfok halmaza
$\mathcal{N}$	$V \to 2^L$; csúcshoz szomszédos csúcsok azonosítóit rendeli
$\mathcal{L}$	$V \to 2^L$; csúcshoz saját azonosítóit rendeli
$\mathcal{S}$	aláíróséma
$conf$	konfiguráció
$\mathbb{K}$	konfigurációk halmaza
$\mathbb{K}^{dj}$	$\mathbb{K}^{dj} \subset \mathbb{K}$; ahol a támadó nem használhat már kirendelt azonosítót
$\mathcal{K}$	$V^* \to 2^{V^*}$; egy támadó csúcshoz tartozó blokk csúcsainak halmaza
$\triangleright$	$\triangleright \subseteq \mathbb{K}^{dj} \times \mathbb{K}^{dj}$; konfiguráció redukció
$\lvert V \rvert$	V halmaz elemszáma (kardinalitása)
V^*	támadó csomópontoknak megfeleltett csúcsok halmaza
$\varnothing$	üres halmaz
$\mathcal{G}$	alkalmazott gráfalgoritmus a konfiguráció redukció során
$\mathcal{C}_{node}$	$V \to \mathbb{R}$; csomóponti költségfüggvény
$\mathcal{C}_{link}$	$E \to \mathbb{R}$; adatkapcsolati költségfüggvény
$sys^{\mathsf{ideal}}_{conf,\mathcal{A}}$	ideális világ modell
$sys^{\mathsf{real}}_{conf,\mathcal{A}}$	valós világ modell
$view^{\mathsf{ideal}}_{conf,\mathcal{A}}$	ideális világ modell kimenetének eloszlása
$view^{\mathsf{real}}_{conf,\mathcal{A}}$	valós világ modell kimenetének eloszlása
rrep	route reply üzenet
rreq	route request üzenet
rerr	route error üzenet

Rövidítés	*Jelentés*
nonce	egyszer használatos véletlen érték
AODV	Ad hoc On-Demand Vector Routing
DSR	Dynamic Source Routing
SRP	Secure Routing Protocol
ZRP	Zone Routing Protocol
LAR	Location-Aided Routing
GPS	Global Positioning System
MAC	Message Authentication Code
HMAC	Keyed-Hashing for Message Authentication
TESLA	broadcast authentication protocol for authenticating routing messages
DoS	Denial of Service
SAODV	Secure AODV
ARAN	Authenticated Routing for Ad hoc Networks
ASAR	A Secure Ad hoc Routing
IP	Internet Protocol
MANET	Mobile Ad hoc Networks
IETF	Internet Engineering Task Force
OSI	Open System Interconnection
ISO	International Standards Organization

Irodalomjegyzék

[1] G. Ács, L. Buttyán, and I. Vajda. Provably Secure On-demand Source Routing in Mobile Ad Hoc Networks. Technical Report, Budapest University of Technology and Economics, March 2005. Available on-line at `http://eprint.iacr.org/` under report number 2004/159.

[2] G. Ács, L. Buttyán, and I. Vajda. Provable Security Of On-demand Distance Vector Routing in Wireless Ad Hoc Networks. submitted for publication, April 2005.

[3] L. Buttyán and I. Vajda. Towards provable security for ad hoc routing protocols. In *Proceedings of the ACM Workshop on Security in Ad Hoc and Sensor Networks (SASN)*, October 2004.

[4] Y.-C. Hu, A. Perrig, and D. Johnson. Ariadne: A secure on-demonad routing protocol for ad hoc networks. In *Proceedings of the ACM Conference on Mobile Computing and Networking (Mobicom)*, 2002.

[5] M. Abadi, P. Rogaway. Reconciling two views of cryptography (the computational soundness of formal encryption), In *Proceedings of the IFIP Conference on Theoretical Computer Science*, Springer LNCS 1872, 2000.

[6] Y.-C. Hu and A. Perrig. A survey of secure wireless ad hoc routing. *IEEE Security and Privacy Magazine*, 2(3):28–39, May/June 2004.

[7] W. Mao. *Modern Cryptography: Theory and Practice*. Prentice Hall PTR, 2004.

[8] P. Papadimitratos and Z. Haas. Secure routing for mobile ad hoc networks. In *Proceedings of SCS Communication Networks and Distributed Systems Modelling Simulation Conference (CNDS)*, 2002.

[9] O. Goldreich. *Foundations of Cryptography*. Cambridge University Press, 2004.

[10] M. Bellare, R. Canetti, H. Krawczyk. A Modular Approach to the Design and Analysis of Authentication and Key Exchange Protocols. In *Proceedings of the ACM Symposium on the Theory of Computing*, 1998.

[11] J. Marshall. An Analysis of the Secure Routing Protocol for mobile ad hoc network route discovery: using intuitive reasoning and formal verification to identify flaws. MSc thesis, Department of Computer Science, Florida State University, April 2003.

[12] M. Abadi and A. D. Gordon. A Calculus for Cryptographic Protocols: The Spi Calculus. In *Proceedings of the Fourth ACM Conference and Communications Security*, 1997.

[13] R. Milner, J. Parrow, D. Walker; A Calculus of Mobile Processes, Part I-II. *Information and Computation*, September 1992.

[14] D. B. Johnson, D. A. Maltz, Y.-C. Hu, and J. G. Jetcheva. The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks, Internet Draft, February 2002.

[15] M. G. Zapata and N. Asokan. Securing ad hoc routing protocols. *Proceedings of the ACM Workshop on Wireless Security (WiSe)*, 2002.

[16] C. Perkins and E. Royer. Ad hoc on-demand distance vector routing. In *Proceedings of the IEEE Workshop on Mobile Computing Systems and Applications*, pp. 90-100, February 1999.

[17] K. Sanzgiri, B. Dahill, B. Levine, C. Shields, and E. Belding-Royer. A secure routing protocol for ad hoc networks. In *Proceedings of the International Conference on Network Protocols (ICNP)*, 2002.

[18] Danny Dolev and Andrew C. Yao. On the security of public-key protocols. *IEEE Transactions on Information Theory*, 2(29):198 208, March 1983.

[19] B. Pfitzmann and M. Waidner. A model for asynchronous reactive systems and its application to secure message transmission. In *Proceedings of the IEEE Symposium on Security and Privacy*, May 2001.

[20] S. Yang and J. Baras. Modeling vulnerabilites of ad hoc routing protocols. In *Proceedings of the ACM Workshop on Security of Ad Hoc and Sensor Networks*, May 2003.

[21] J. Guttman. Security goals: packet trajectories and strand spaces. In *Foundations of Security Analysis and Design*, edited by R. Focardi and R. Gorrieri, Springer LNCS 2171, 2000.

[22] P. Papadimitratos and Z. Haas. Secure routing for mobile ad hoc networks. In *Proceedings of SCS Communication Networks and Distributed Systems Modelling Simulation Conference (CNDS)*, 2002.

[23] M. Burrows, M. Abadi, and R. Needham. A logic of authentication. *ACM Transactions on Computer Systems*, 8(1):18-36, February 1990.

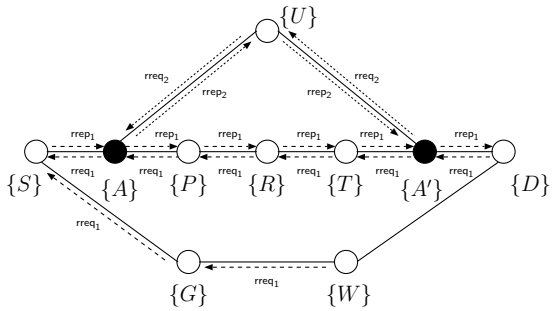
[24] Y.-B. Ko and N. H. Vaidya. Location-Aided Routing (LAR) in mobile ad hoc networks. *Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking*, October 25-30, 1998.

[25] Z. J. Haas and M. R. Pearlman. The Zone Routing Protocol (ZRP) for Ad Hoc Networks. *Internet-draft*, July, 2002.

A. Függelék

Egy adaptív támadás az SAODV protokoll ellen

Eddig csak nem adaptív támadásokat mutattam be az SAODV protokoll ellen. A következőkben azt szeretném demonstrálni, hogy létezik ellene adaptív Aktív-2-x támadás is, amely a felhasznált hash láncok gyengeségét mutatja ad hoc környezetben. A támadás menetét szemlélteti a A.1. ábra.



A.1. ábra. Egy adaptív Aktív-2-x támadás az SAODV protokoll ellen. A nyilak nem az üzenetek haladási irányát mutatják, hanem arra a csomópontra mutatnak, amely mint célbejegyzés szerepel a csomópontok tábláiban.

Ebben a konfigurációban a támadó két csomópontot irányít és két azonosítót használhat, melyeket A és A' jelöl, vagyis $L^* = \{A, A'\}$. Tegyük fel, hogy az S forráscsomópont útvonalkeresést kezdeményez D célcsomópont felé. Az egyszerűség kedvéért szintén feltételezzük, hogy kezdetben minden útvonalválasztó tábla üres a hálózatban, vagyis teljesül, hogy S kérésére egyik közbenső csomópont sem képes válaszolni. A támadás lépései a következők:

1. S csomópont többesszórással elküldi minden szomszédjának az rreq_1 kérésüzenetet, nevezetesen A és G csomópontoknak. Az SAODV szabályoknak megfelelően S elhelyez egy generált véletlen értéket ($seed_1$) az rreq_1 üzenetben. Ezekután aláírja rreq_1 üzenet nem változó részeit garantálva ezzel a kérés integritását.

2. Amikor A csomópontot eléri rreq_1 üzenet S csomóponttól, az egy új útvonalkeresést kezdeményez A' csomópont fele, vagyis elküld egy rreq_2 üzenetet U csomópontnak.

Viszont mielőtt aláírná és elküldené rreq_2 üzenetet, A elhelyezi benne $seed_1$ véletlen értéket, mintha az egy újonnan generált véletlen érték lenne.

3. Előbb vagy utóbb mind rreq_1 mind pedig rreq_2 eléri A' csomópontot, mivel U sikeresen ellenőrzi rreq_2 üzenetet, hiszen az A csomóponttól származó aláírás korrekt, valamint A módosítás nélkül továbbította rreq_1 üzenetet, tehát nem alkalmazta $seed_1$ véletlen értéken az előírt hash függvényt.

4. Így A' csomóponthoz kettő kérésüzenet érkezik korrekt aláírásokkal: rreq_1 üzenet $\mathsf{Hash}_1 = h^3(seed_1)$ értékkel és rreq_2 üzenet $\mathsf{Hash}_2 = h(seed_1)$ értékkel, ahol Hash_1 és Hash_2 a megfelelő Hash mezők értékeit jelentik rendre rreq_1 és rreq_2 üzenetekben. A' Hash_1 mező értékét $h(seed_1)$-re állítja, mivel ismeri $h(seed_1)$ értéket rreq_2 üzenetből. Anélkül, hogy válaszolna rreq_2 kérésre A' továbbküldi rreq_1 üzenetet D csomópont felé, és vár a visszaérkező válaszra.

5. Ekkor két eset történhet:

   - D csomópont előbb kapja meg rreq_1 üzenetet W csomóponttól mint A' csomóponttól. Ekkor D mindkét kérésre válaszolni fog, mivel az A' csomóponttól származó rreq_1 kisebb hop számmal rendelkezik mint a W csomóponttól származó.
   - D csomópont előbb kapja meg rreq_1 üzenetet A' csomóponttól mint W csomóponttól. Ekkor D csak az A' csomóponttól származó kérésre fog válaszolni, mivel az kisebb hop számmal rendelkezik mint a W csomóponttól származó kérés.

   Így D csomópont mindenképpen válaszol rreq_1 kérésre rrep_1 válasszal, és elküldi ezt a választ A' csomópontnak pont-pont címzéssel, valamint frissíti a saját útvonalválasztó táblájáta egy $(S, A', 2)$ „inkorrekt”[1] bejegyzéssel.

6. Amikor A' csomópontot eléri rrep_2 üzenet, akkor A' ugyanazokat a lépéseket végzi el a válaszüzenettel mint amit A tett a kérésüzenetekkel a 2. pontban. Vagyis A és A' kooperatíven ugyanazokat a műveleteket végzik rrep_2 üzenet esetében mint amiket az rreq_1 üzenet esetében végeztek csak fordított szereposztásban.

A támadás eredményeképpen S forráscsomópont megkapja rrep_1 üzenetet A csomóponttól egy érvénytelen hop szám értékkel, és így egy $(D, A, 2)$ „inkorrekt” bejegyzés kerül az útvonalválasztó táblájába. Az útvonalfelderítés végeztével S forráscsomópont az adatcsomagokat A csomópont felé fogja küldeni G csomópont helyett.

[1] persze ez nem a 7. rész szerinti modell szerint inkorrekt, mivel ott nem is engedtünk meg ilyen jellegű adaptív támadást. Inkorrekt abban az értelemben, hogy nem létezik 2 hosszú út D csomópontból S csomópontba egy A' azonosítót használó csomóponton keresztül.