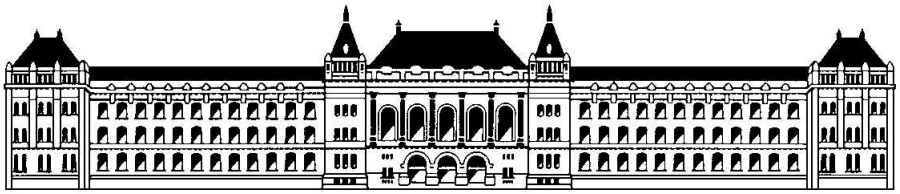


Budapesti Műszaki és Gazdaságtudományi Egyetem
Híradástechnikai Tanszék, CrySyS laboratórium

Járművek követhetőségének vizsgálata VANET környezetben

TDK dolgozat

Szilágyi Éva

V. éves Villamosmérnök hallgató

Konzulens: Dr. Buttyán Levente és Holczer Tamás, CrySyS laboratórium

BME VIK

2008

Tartalomjegyzék

Kivonat

Mindennapi életünktől elválaszthatatlanok a közlekedés és a kommunikáció legkülönbözőbb formái. E kettő összefonódása hívta életre a járműközi kommunikációt, azaz VANET-eket (Vehicular Ad Hoc Network), melyek nagyobb áteresztőképességű, hatékonyabb és biztonságosabb forgalmat ígérnek városokon belül és kívül egyaránt. Az eljárás kínálta lehetőségek kecsegtetőek, így a járművezetők előre értesülhetnek egy esetleges baleset helyszínéről, így kerülve el a továbbiakat, illetve a dugókat.

Azonban más szempontból is elengedhetetlen az ötlet tanulmányozása. Meg kell vizsgálnunk a veszélyeket is: a fokozott kommunikáció lehetőséget nyújt arra, hogy egy autó könnyebben követhetővé váljon, és ezáltal sérüljön a vezető privát szférája. Számos csoportot foglalkoztat ez a terület, így a rendelkezésre álló irodalom tanulmányozása után munkánkhoz a mix zone-ok ötletét vettük alapul. Eszerint az elv szerint, az autók igyekeznek úgy, és olyan helyeken azonosítót váltani, hogy egy esetleges támadó minél nehezebben legyen képes nyomon követni.

Az eddig publikált munkák nem felelnek meg minden igénynek, néhol a támadó modell rossz, vagy a forgalom szimuláció áll távol a valóságtól, de előfordul az is, hogy a fő cél, a balesetmentes közlekedés háttérbe szorul a privát szféra védelméhez képest. Munkánk folyamán az említett gyengeségeket szem előtt tartva implementáltunk különböző azonosító, pontosabban pszeudoním váltási algoritmusokat, és hasonlítottuk össze ezeket annak érdekében, hogy megtaláljuk a leghatékonyabb módszert, mellyel egy adott erősségű támadó ellen védekezhetünk.

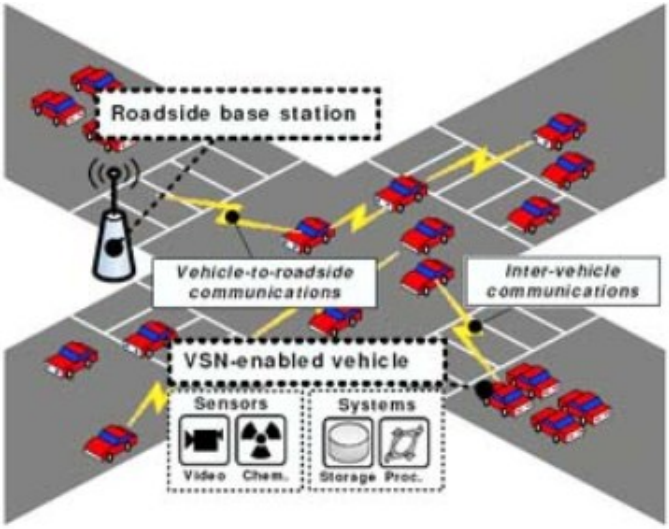
1. Bevezetés

Mindennapi életünktől elválaszthatatlanok a közlekedés és a kommunikáció legkülönbözőbb formái. Így csak idő, és technikai fejlődés kérdése volt, hogy mikor fonódik szorosabban össze e kettő.

A járműközi kommunikáció (VANET, Vehicular Ad Hoc Network) egy lehetséges módja a forgalom biztonságosabbá és hatékonyabbá tételének. Mint minden technikai újdonságnak, így a kommunikációs eszközök, lehetőségek fejlődésének is számtalan előnye van. A VANET-ek segítségével biztonságosabbá tehetjük az utakat. Számtalan emberi élet megmenthető lehetne mindössze azáltal, hogy ha egy-egy balesetről időben értesülnénk. Más szempontból az is nagy előny lehetne, ha hatékonyabbá tennénk vele a közlekedést, azaz jó előre értesülnének a járművezetők a dugók, útlezárások helyzetéről, így el tudnák kerülni ezeket.

Viszont ezen előnyök mellett meg kell vizsgálnunk a kevésbé jó oldalt is. A VANET-ek a támadók számára lehetőséget kínálnak, hogy megsértsék a használói magánszféráját, adott esetben követhetővé váljanak, vagy csak passzív lehallgatással egyszerűen információt gyűjtsenek róluk. Ez a fő ok, ami elrettenti a potenciális használókat, és befektetőket az új rendszertől.

A hálózat technikai megvalósítása sem egyszerű, számtalan próbálkozás zajlik. Szükség van arra, hogy egy adott jármű kommunikálni tudjon egy központi egységgel, és annak esetleges kihelyezett elemeivel (az út mentén található állomásokkal- Roadside Unit - RSU), valamint arra is, hogy a többi járművel (On-Board Unit - OBU) üzenetet tudjon váltani, esetleg megadja a pozícióját.



1. ábra. VANET

Manapság egyre fontosabb téma a személyes szféra védelme, azaz pontosabban például annak az információnak a védelme, hogy egy alkalmazás használója éppen hol található. Beresford és munkatársai által írt cikkben [3] ezt a kérdést elemzik. Azok védelmét szeretnék fokozni, akik olyan szolgáltatásokat vesznek igénybe, amikhez feltétlenül szükséges a helyzetük megadása.

Pillanatnyilag még nem égető a privát szféra védelme, de egyre elterjedtebbek az olyan alkalmazások, ahol szükség van a felhasználó pozíciójának megadására, más részről pedig a követő és megfigyelő készülékek, felszerelések fejlődésével egyre inkább az lesz. (Talán az egyik legfontosabb kérdés a részinformációk összekapcsolása.) Félő, hogy járművünk bárki számára könnyebben követhetővé válik, és ez által esetlegesen támadásoknak lesz kitéve. Több stratégiát dolgoztak már ki a helyhez kötődő információk védelmére. Lehet szabály-alapú elvű, mint például Geographic Location/Privacy vagy pedig a továbbított információkat először egy kontrollált módon degradálják. Ilyen modellben úgy anonimizálják a felhasználó kilétét, hogy korlátozzák azokat a helyeket, ahol a felhasználó megtalálható.

A világon mindenfelé (különösen Európában, az Egyesült Államokban, és Japánban) egyre nagyobb támogatást kapnak a közlekedés biztonságosabbá tételére szolgáló törekvések. Erre egy módszer a járműközi kommunikáció.

A problémát az okozza, hogy azoknak az alkalmazásoknak, melyeknek a biztonság fokozása a célja, folyamatosan szükségük van arra, hogy az egyes járművek broadcast üzenetben közöljék az aktuális helyzetüket, és sebességüket: úgynevezett *szívdobbanás* üzenetekben. Ez segít kiszámíthatóvá tenni a forgalmat a sofőrök számára (meg tudják becsülni a körülöttük lévő autók mozgását), és figyelmeztetheti őket egy-egy kockázatos helyzetre. De egyben ezeknek a szívdobbanás üzeneteknek a lehallgatásával válik követhetővé a jármű.

Erre a problémára egy megoldási javaslat, hogy a járművek bizonyos időközönként azonosítót (logikai és fizikai címeket egyaránt) cserélnek. Nem pontos azonosítót használnak, elégséges egy pszeudoním, mert sem az alkalmazásoknak, sem a többi járműnek nem arra van szüksége, hogy tudja, éppen ki vezet egy adott autót, hanem arra, hogy értesüljön arról, hogy egy adott helyen és időben milyen sebességgel közlekedik az ott található jármű.

Ezek a pszeudonímek a nyilvános kulcsú infrastruktúra elvén alapszanak. Tehát minden résztvevő jármű rendelkezik egy privát- és egy publikus kulccsal. A privát kulcsot csak a tulajdonosa ismeri, ellenben a publikus kulcs nyilvános, bárki megszerezheti. Ha egy üzenetet a privát kulccsal aláírják, akkor a hozzá tartozó publikus kulccsal bárki ellenőrizheti, hogy az üzenet ténylegesen attól származik-e, aki ezt állítja. Ez az aláírás a hitelesítés. (Ha az üzenetet publikus kulccsal kódolják, akkor azt csak a privát kulccsal lehet visszafejteni, ez ad lehetőséget a titkosításra.) Azonban csak akkor támaszkodhatunk egy publikus kulcsra, ha tudjuk, hogy ki birtokolja a hozzá tartozó privát kulcsot. Ezért van szükség a hitelesítés szolgáltatókra, akik aláírt igazolásokat állítanak ki arról, hogy egy adott publikus kulcs (és a hozzá tartozó privát kulcs) kihez tartozik. Ezeket az igazolásokat nevezzük tanúsítványoknak. [12]

A tanúsítványok az alábbi adatokat tartalmazzák:

- Sorozatszám
- Kiállító DN (azaz a CA)
- Érvényesség kezdete, vége
- Tulajdonos DN (a tanúsítvány alanya)
- Tanúsítvány-irányelv (Certificate Policies)

- QCStatement (csak minősített tanúsítványban)
- Tranzakciós limit (csak minősített tanúsítványban)
- Visszavonási információk elérhetősége
- Kulcshasználat (Key Usage)
- Nyilvános kulcs, a CA aláírása, aláíró és hash algoritmusok megnevezése

A tanúsítványokat általában más tanúsítványok alapján ellenőrizhetjük, az ellenőrzést gyökér hitelesítés szolgáltatók publikus kulcsára vezethetjük vissza. Ezekre a kulcsokra jellemző, hogy sokan ismerik és elfogadják.

A hitelesítő központok feladatai közé tartozik:

- a felhasználók azonosítása, és regisztrálása
- számukra a tanúsítvány kibocsájtása
- az, hogy adott esetben nyilvánosságra hozzák a tanúsítványokat
- az, hogy nyilvánosságra hozza, ha a felhasználó visszavonja a tanúsítványt
- a garanciavállalás (saját működésére)

Abban az esetben, amikor a kommunikáló felek anonimitást szeretnének, de ugyanakkor erős végpont hitelesítést, lehetőség van álnevek, pszeudonímek használatára. Viszont ebben az esetben is a valódi adataik ugyanúgy eltárolásra kerülnek a hitelesítő központban, tehát kérdéses esetben meg lehet állapítani, hogy kitől származott egy adott üzenet, még ha pszeudonímet használt is az illető. Ezek a pszeudonímek, (melyek a publikus kulcsból és egy véletlen számsorozatból állnak) folyamatos cserélgetés esetén olyan rövid életűek, hogy nem kerül sor a visszavonásukra. (Előbb lejár az érvényességük, mint lezajlik egy visszavonási procedúra.)

Tehát elmondható, hogy a céloknak teljesen megfelel a pszeudoním is, így ezeket az egyén magán szférájának biztonsága érdekében úgy kell megválasztani, hogy ne lehessen közvetlenül a váltás előtti pszeudonímet összekötni a váltás utánival. Viszont tudnunk kell, honnan származik egy üzenet - így küszöbölve ki azt, hogy esetlegesen egy támadó hamis üzenetekkel árassza el a rendszert, szolgáltatásbénítást okozva, vagy csak egyszerű megtévesztés céljából.

Sajnos egy globális lehallgatás (amikor a támadó a hálózatbeli összes üzenetváltást hallja) esetében ez a módszer sem nyújt elegendő védelmet. A szívdobbanás üzenetek, valamint egyszerű számítások segítségével (ha adott sebességgel közlekedik, akkor hol lehet adott idő múlva) képes lehet járműhöz kötni az egyes pszeudonímeket.

Azonban ez nem egy reális támadó modell, valóságban inkább feloszthatjuk a területet olyan részekre, ahol történhet lehallgatás, és ahol nem, tehát a nem megfigyelt területeken kell azonosítót váltani. Ezen az elven alapszik az úgy nevezett *keverő zóna (mix zones) modell.*

Mindenképpen szükség van arra, hogy pszeudonímeket használjunk, ha csökkenteni akarjuk annak esélyét, hogy egy esetleges támadó azonosítani tudja a jármű tulajdonosát. A [14] cikkben Schoch és munkatársai hívják fel a figyelmet arra, hogy nem szabad megfelejtkezni arról sem, hogy ha egy autó adott sebességgel elágazás nélküli szakaszon közlekedik, akkor a támadó jó eséllyel tudni fogja, hogy bizonyos eltelt idő után hol járhat a követés áldozata, hiába haladt át egy keverő zónán. A kérdés összetettségét mutatja, hogy a támadó esélyeit növeli az a tény, hogy nagyobb valószínűséggel halad egyenesen egy jármű, mint kanyarodik.

E dolgozat témája a követhetőség problémakörének elemzése. A kiküszöbölésre törekszik azáltal, hogy minden jármű - habár rendelkezik egy azonosítóval, mely elengedhetetlen a kommunikációhoz, és jónéhány alkalmazás használatához - bizonyos időközönként lecseréli ezt az egyéni jellemzőt.

A cserére többféle elképzelés létezik.

Az egyik ilyen az említett keverő zóna elmélet. A [3] cikkben Beresford olyan közlekedési területeket javasol, ahol sok autó találkozik, és döntési helyzetben vannak útjuk folytatását illetően (pl. kereszteződések), hogy ott váltsanak azonosítót a járművek. (Ezt a kérdést elemzi Freudiger a [8] cikkében.) Sajnos a járművek nem tudhatják, hogy mikor haladnak megfigyelt, illetve mikor haladnak biztonságos övezetben, így folyamatos azonosító váltás szükséges.

De milyen cserealgoritmus védheti meg hatékonyan járművünket a támadótól úgy, hogy közben a központ, és a többi érintett jármű számára egyértelműen azonosított maradjon? Milyen támadó modellt használjunk a védelem kidolgozásánál?

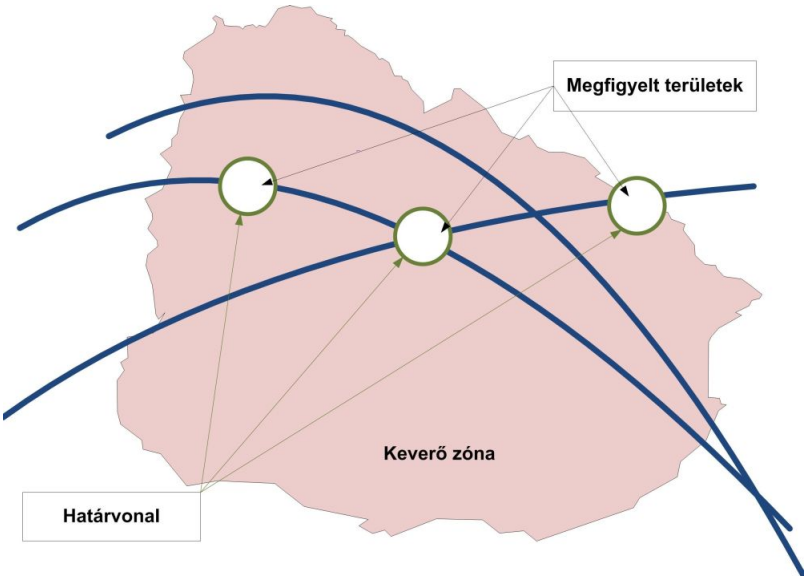
2. Keverő zóna

Keverő zóna modell: A támadó bizonyos helyeken rádióvevő készülékeket helyez el, hogy lehallgathassa az ott elhaladó autók közötti kommunikációt, beleértve bizonyos szintig a szívdobbanás üzeneteiket. De ezen az adott szakaszon kívül a támadó nem hallhatja a kommunikációt.

A modell feltételez egy olyan megbízható middleware rendszert, amely a helymeghatározás alapjául szolgáló rendszer, valamint a megbízhatatlan egyéb alkalmazások között található.

Az ötlet szerint a 2. ábrán is látható két (három) részre oszthatjuk a teret:

- a *megfigyelt terület*, ahol az alkalmazások használata, illetve ezzel egyidejűleg a megfigyelés történhet
- a *keverő zóna*, ahol a felhasználó azonosítót vált a többi ott tartózkodóval együtt, de nem történhet lehallgatás
- a két zóna között található a *határvonal*.



2. ábra. A keverő zóna részei

A többi jármű nem egy követhető felhasználó azonosítót kap, társítva azzal, hogy hol található az adott felhasználó, hanem egy pszeudonímet. Ez a pszeudoním teszi lehetővé a kommunikációt az autók között, és ezt a pszeudonímet cserélik le a felhasználók, amikor beérnek a keverő zónába, illetve csere történik a megfigyelt zónában is (hiszen nem tudják, hogy hol találhatóak a megfigyelő pontok), de ott hatástalan a védekezési kísérlet.

A keverő zóna modell célja a hosszú-távú követés elleni védekezés, de rövid távon is jobban használható, mint sok más eljárás. A nem megbízható alkalmazásokra tekinthetünk úgy, mint egy globális

megfigyelő hálózatra. Az is aggályos, hogy a támadó előzetes megfigyelések alapján, vagy egyszerű számításokkal (attól függően, hogy egy adott jármű mikor lépett be a keverő zónába, mikor valószínű, milyen intervallumban, hogy elhagyja azt) a keverő zónában is követni tudja az áldozatát. Az is neki segít, hogy adott szakaszokon nagyobb a valószínűsége, hogy egy jármű egyenesen halad, mint annak, hogy elkanyarodik. Előzetesen gyűjtött adatok segítségével a támadónak lehet elképzelése arról, hogy áldozata milyen forgalmi minta alapján közlekedik, vagy arról, hogy az egyes napok egyes óráiban mennyi idő szükséges egy bizonyos útszakasz megtételéhez a keverő zónán belül. Tehát az anonimitási szint függ az adott övezet geometriájától, a térbeli és időbeli bontástól, valamint a sofőrök viselkedési mintáitól is.

A keverő zónák elmélete reményt ad arra, hogy a járművek csökkentsék követhetőségüket azáltal, hogy bizonyos időközönként pszeudonímet cserélnek. A [3] cikkben Beresford és kollégái az elmélet matematikai modelljét elemzik. Céljuk a modell alaposabb vizsgálata, finomítása és törekszenek a számítási igények minimalizálására. Hangsúlyt fektetnek a felhasználói visszacsatolásra.

Egyik ötletük a határvonal szerepének erősítése. Azaz ha a támadó nem csak azt figyeli meg, hogy az áldozat melyik zónából melyik másik zónába lépett át, hanem azt is, hogy pontosan hol, akkor jobban, könnyebben tudja követni a keverő zónán kívül. Viszont íme három ok, ami miatt nem érdemes túl kis részekre bontani a határsávot:

- helymeghatározás pontossága
- mintavételezés pontossága (kicsi minták torzítják az eredményeket)
- számítási igények

Azonban nem szabad elfelejteni, hogy a járművezetők nem tudják, hogy hol találhatóak a lehallgatási pontok, ezért ajánlott a folyamatos azonosító váltás.

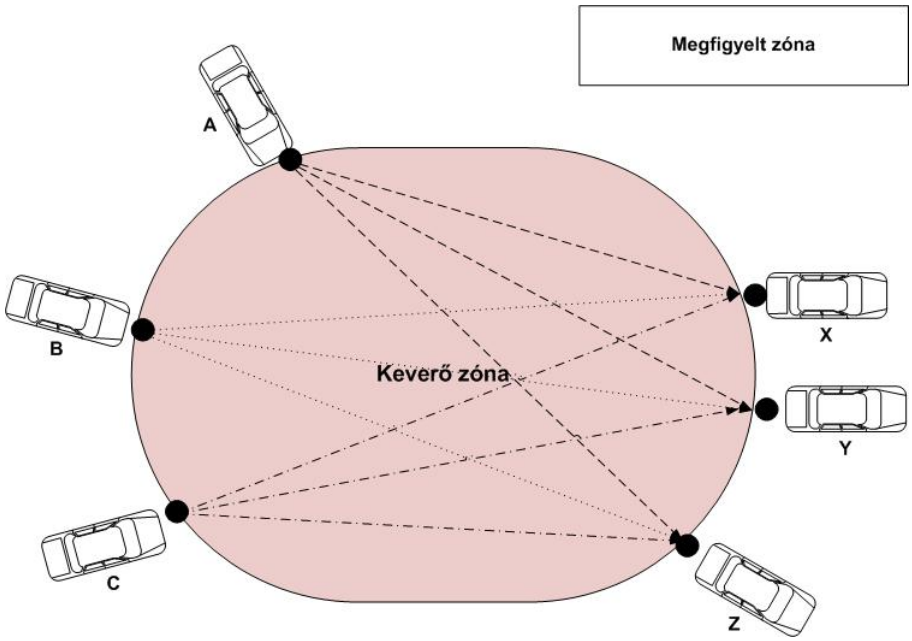
3. Támadó modell

3.1. Támadó tudása

A felhasználók beérkeznek a keverő zónába, ott egy addig nem használt pszeudonímre cserélik a sajátjukat, majd bizonyos idő után az új azonosítóval elhagyják az övezetet.

A támadó a belépési és kilépési pontoknál megfigyelheti az időt, a koordinátákat, és a pszeudonímeket. Célja, hogy ezek alapján az adatok alapján a régi, és az új pszeudonímeket össze tudja kötni, és járműhöz tudja rendelni őket. Lehetősége van arra, hogy térbeli és időbeli megszorításokkal egy forgalmi mátrixot hozzon létre. Ennek a forgalmi mátrixnak a soraiban a bemeneti pontok, oszlopaiban pedig a kimeneti pont találhatók, de természetesen nem minden összekötés lehetséges a megszorító tényezők miatt. Ilyen megszorítások:

- a felhasználó nem hagyhatja el előbb a keverő zónát, mint hogy belépett volna
- nem mozoghat a keverő zóna két olyan pontja között, amik nincsenek összekötve
- bizonyos sebességnél gyorsabban nem közlekedhet



3. ábra. Keverő zóna

A 3. ábrán látható, hogy a keverő zónába belép az A, B illetve C jelű jármű, majd bizonyos idő elteltével a védett területet elhagyja az X, Y, illetve Z azonosítójú autó. A támadó célja, hogy a megszorításokat, valamint a rendelkezésére álló többlet információkat figyelembe véve megállapítsa, hogy melyik váltás előtti, illetve utáni pszeudoním melyik járműhöz tartozik.

Munkánk során olyan támadó modellt használunk, melynek hatékonyságához elengedhetetlen egy tudásbázis. Ez a tudásbázis előzetes tapasztalatok alapján jön létre. Azaz adott területen zajló, adott mennyiségű jármű esetén a támadó megvizsgálja, hogy mely útszakaszok megtételéhez, mennyi idő szükséges. Mivel ez esetenként változhat, ezért tárolja az egyes időtartamokhoz tartozó valószínűségeket.

1. táblázat. Tudásbázis részlete

Tudásbázis			
Kvantált idő	edge26 - edge25	Kvantált idő	edge26 - edge2
'4'	0,123	'21'	0,188
'3'	0,413	'20'	0,269
'2'	0,375	'19'	0,384
'1'	0,089	'18'	0,159

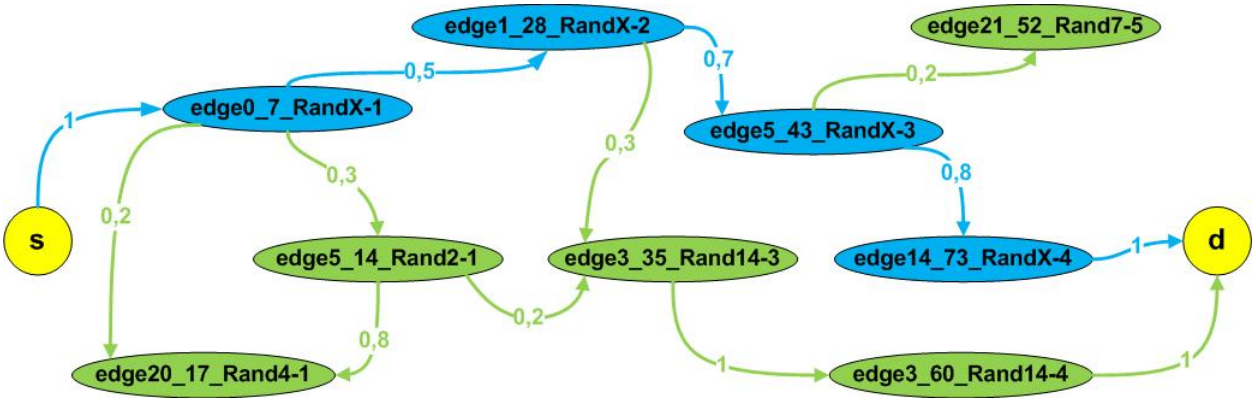
Az 1. táblázatban a tudásbázis kis részletét mutatjuk be. Tartalmazza, hogy melyik két-két élt vizsgáljuk, illetve megmutatja, hogy mekkora a valószínűsége annak, hogy egy adott útszakaszt bizonyos időintervallumon belül tesznek meg a járművek. A dolgozatban a kvantálási érték: 10. A példa táblázatból kiderül, hogy annak, hogy egy jármű 10-19 másodperc közötti idő alatt jusson el az edge26-ról az edge25-re, 0,089 a valószínűsége, még annak, hogy 30-39 másodperc között, annak 0,413, tehát ennek a legnagyobb a valószínűsége.

3.2. Támadó erőssége

A támadók tekintetében megkülönböztetünk gyenge, és erős támadói modellt. Ez azt jelenti, hogy ha egy támadó csak a forgalmi mátrix-ot ismeri, akkor *gyenge támadónak* nevezzük, de ha más információkkal is rendelkezik az áldozatáról, ami megkönnyíti a forgalmi mátrix pontosítását, személyre szabását, akkor már *erős támadóról* van szó. Így például meghatározza a támadó erősségét, hogy rendelkezésére áll-e áldozatának potenciális célpont listája (munkahely, bevásárlóközpont, sporttelep), vagy bárhová tarthat az illető.

Azonban nemcsak két csoportba (erős illetve gyenge támadó) sorolhatjuk be őket. A támadó erősségét nagyban befolyásolja, hány ponton képes lehallgatni a hálózatot, illetve, hogy a lehallgató készülékei milyen hatósugarúak. Azonban nemcsak ezeknek a megfigyelési pontoknak a darabszáma játszik fontos szerepet, hanem az is, hogy hol tudja elhelyezni az eszközöket. A támadó sikerességét befolyásolja a megfigyelt terület összetettsége, tehát hogy hány elkanyarodási lehetősége van áldozatának, illetve hogy mekkora lehet a sebességingadozása. Nagyban könnyíti a helyzetét, ha csak néhány autó van az adott területen.
A támadón kívülálló tényező, de befolyásolja az erősségét, hogy mennyire hatékony a védekezési mechanizmus. Ha csak nagyon ritkán cserél a jármű azonosítót, és azt is olyan területen, ahol szinte egyedül van jelen, akkor a támadónak nagyobb esélye nyílik a követésre, mint egy rendszeresen, kiszámíthatatlanul, megfelelő helyen végrehajtott cseresorozat esetén.

Munkánk folyamán használt támadó modell rendelkezik a potenciális célállomások listájával. A megfigyelt csomópontok számát az egyes változatokban különböző értékűre állítjuk. A tudásbázis és a megszorító feltételek segítségével egy forgalmi gráfot hozunk létre, melyben összekötjük azon eseményeket, amikor a támadó számára megjelenik, illetve amikor eltűnik egy jármű, és fordítva. A gráf éleit a tudásbázisból származó valószínűségek alapján súlyozza. Beépítünk a gráfba egy kezdőpontot, és összekötjük azzal a ponttal, ahol először eltűnik az áldozat járműve. A végponthoz hozzákötjük a potenciális végállomásokat. Ezt követően a legkisebb szorzatú út megtalálása a cél, amit a súlyok logaritmizálásával visszavezetünk a legkisebb összsúlyú út keresésére (ezt másnéven a legrövidebb útnak is hívják) a gráfban a kezdő- és a végpont között. A támadó sikerességét mutatja, ha sikerült végig követnie az áldozatát ezen az útvonalon, és nem tévesztette össze egy másik arra közlekedő autóval.



4. ábra. Gráf

A 4. ábrán egy példa gráf látható. Az érthetőség érdekében az áldozat járműjének azonosítóját RandX-el jelöltük. A forgalom többi résztvevője esetében a Rand szó után szám áll. A gráf élein az előzetesen megszerzett tudásból származó valószínűségek vannak normálva. A kezdőpont az 's' csomópont, ehhez csatlakozik a legelőször eltűnő RandX azonosítójú autó. Utolsó pontja 'd' csomópont, ehhez kapcsolódik az összes olyan jármű, melynek végállomása megegyezik RandX végállomásával. (A modellünk szerint a támadó ismeri áldozata lehetséges célállomásait.)
A gráf csomópontjai három információt tartalmaznak: azt az élt (edge), amin éppen a jármű megtalálható, az időpillanatot (amikor feltűnik, vagy eltűnik az autó), valamint az autó azonosítóját. Azaz az edge0_7_RandX-1 csomópont azt jelenti, hogy a RandX-1 azonosítójú autó éppen az edge0 élen halad, amikor a 7-es időpillanatban feltűnik, vagy éppen eltűnik a támadó számára.
A támadó akkor sikeres, ha olyan utat képes találni a gráfban, melynek csomópontjai csak a követett autóhoz, azaz jelen esetben RandX-hez tartoznak. Azonban előfordulhat, hogy elveszíti szem elől áldozatát, és egy döntési helyzetben egy másik járművet választ, és azt követi tovább a cél felé, vagy pedig újabb hibát követ el, és egy egészen más járműről gondolja azt, hogy az áldozata.

Ebben a dolgozatban a támadó eredményességét vizsgáljuk, azt hogy mennyire tudja nyomon követni, illetve azt, hogy mennyi idő eltelte után téveszti szem elől az áldozatát.

4. Cserealgoritmusok összehasonlítása

4.1. Munkakörnyezet

4.1.1. SUMO

A SUMO (Simulation of Urban MObility) városi közlekedést szimuláló, nyílt forráskódú program. 2000-ben a Centre for Applied Informatics (ZAIK) és az Institute of Transport Research at the German Aerospace Centre munkatársai kezdték fejleszteni ezt a mikroszkopikus forgalmat szimuláló alkalmazást. [2]

Nemcsak azért esett a választás erre a szimulációs programra, mert hordozható és gyors, hanem azért is, mert a járművek ütközésmentesen mozognak, többféle járművet lehet definiálni benne, és minden egyes járműnek meg lehet külön adni az útvonalát.

Munkánk során egy egyszerűsített térképet használtunk, melynél mi határoztuk meg a csomópontokat, és a köztük lévő kapcsolatokat, majd a SUMO programmal generáltuk a különböző forgalmi és támadó modellhez szükséges bemeneti file-okat.

A SUMO által kínált legfontosabb lehetőségek:

- szimuláció parancssorból (SUMO)
- grafikus felület (GUI)
- hálózat építése/konvertálása (NETCONVERT)
- hálózat építése/generálása (NETGEN)
- kezdet-végpont mátrixból útvonalak konvertálása (OD2TRIPS)
- térképek importálása más alkalmazásokból (Visum, Vissim, ArcView, XML-Leírások)
- dinamikus útgenerálás (DUAROUTER)
- olyan részletes file-ok generálása, mely megadja, hogy egy szimuláció folyamán melyik jármű, melyik időpillanatban éppen hol tartózkodott

4.1.2. PERL

Az algoritmusok implementálásához programnyelvet kellett választani. A PERL [1] szó jelentése: Practical Extraction and Report Language (kivonatok és jelentések készítésére szolgáló nyelv). A Perl nyelv interpretált nyelv. Rendszeradminisztrációs feladatok megkönnyítésére jött létre. Elsősorban azért erre a programnyelvre esett a választás, mert rendkívül jól kezeli a reguláris kifejezéseket, valamint ismert nyelvi elemeket tartalmaz, már kis rész ismerete is elég a használatához, és gyors. A nyelv meglévő eszközökre lett alapozva: C, sed, awk és sh programokra.

Az [11] alapján gyűjtöttem össze a PERL legfontosabb tulajdonságait. Perl-ben írt scriptek esetén csak a számítógép hardware korlátai érvényesülnek: képes egy teljes file-t beolvasni egy string változóba, tetszőleges mélységű rekurzió futtatható benne. A hatékonyságot elősegítendő az asszociatív tömbök elérését hash táblákkal gyorsítja. Nagyon gyors és rugalmas mintaillesztő algoritmusa van szövegek keresésére és cseréjére. Képes bináris adatokkal is dolgozni, és ezekből bonyolult adatstruktúrákat felépíteni. Az adminisztrációs feladatok megkönnyítésére az asszociatív tömbökhöz adatbázis file-okat rendelhetünk.

A nyelv erősségei:

- Gyors fejlesztés: A Perl nem igényel fordítást, mert félig interpreteres nyelv, egyből futtatható a forráskód. A hibakeresés a beépített lehetőségeivel egyszerű. A Perl-nél induláskor a futtató környezet félig feldolgozza a forráskódot, ennek eredményeképpen egy gépi kódhoz közeli nyelven lesz elérhető a program, így a Perl interpreteres volta nem válik egyben hátránnyá is.
- Lehetőségek: Telepítése után a reguláris kifejezései segítségével a szövegfeldolgozás rendkívül egyszerű, továbbá objektumorientált, adatbázis-kezelő és hálózati programozási lehetőségek széles tárháza áll a rendelkezésre. Az Internetről letölthető modulok segítségével tovább bővíthetőek a lehetőségek.
- Tanulhatóság: A Perl alapjai viszonylag egyszerűek, fokozatosan tanulható nyelv. Egy adott feladatra általában többféle megoldási lehetőséget ad, lehetőséget nyújtva a kezdőnek hogy egyszerűen, a haladónak, hogy elegánsan oldja meg a problémát.
- Hordozhatóság: Minden elterjedt platformon futtatható. Könnyen kialakítható egy olyan fejlesztői környezet, melyben a fejlesztés Windows környezetben zajlik, a program azonban végül Unix környezetben fog fut - s ehhez akár semmilyen módosítás nem szükséges a forrásban (persze megfelelő körültekintéssel kell eljárni, és nem tartalmazhat operációs rendszertől függő részeket a program).
- Sebesség: Általánosságban elmondható, hogy gyors, persze ez a tulajdonsága természetesen attól függ, hogy mivel hasonlítjuk össze. Igaz, hogy egy hosszabb programot is pillanatok alatt fel tud dolgozni a futtató környezet, gyorsan képes elindulni a programunk, majd pedig gyors feldolgozásra képes. Adott esetben speciális beépített és nagyon jól optimalizált nyelvi elemeinek köszönhetően (reguláris kifejezések, beépített gyorsrendezés) akár egy bináris programnál is gyorsabb működésre képes.

4.2. Megoldási lépések

A dolgozat célja különböző pszeudoním váltási algoritmusok összehasonlítása. Azonban nem csak az algoritmusok különböznek, hanem az egyes algoritmusok paraméterei is.
Az eredmények felé számos előkészítő lépés vezet. Az alábbiakban ezekről következik egy rövid áttekintő:

4.2.1. A SUMO által használt file-ok

Forgalom szimulálásához elengedhetetlen egy térkép, melyen a közlekedési eszközök haladnak. Így első lépésként szükség van a SUMO által feldolgozható bemenetekre, azaz nod.xml, valamint edg.xml file-okra, melyek a térkép csomópontjait, illetve az őket összekötő éleket (utakat) tartalmazzák. Munkánk során saját készítésű térképet használtunk, így mi adtuk meg a nod.xml file-ban található pontokat, és mi definiáltuk az ezeket összekötő éleket.

Részlet a nod.xml file-ból:

```
<nodes>
   <node id="node0" x="0.0" y="0.0" type="priority"/>
   <node id="node1" x="1100" y="8600" type="priority"/>
</nodes>
```

Részlet az edg.xml file-ból:

```
<edges>
   <edge id="edge0" fromnode="node0" tonode="node1" priority="30" speed="35" />
   <edge id="edge1" fromnode="node1" tonode="node2" priority="30" speed="35" />
</edges>
```

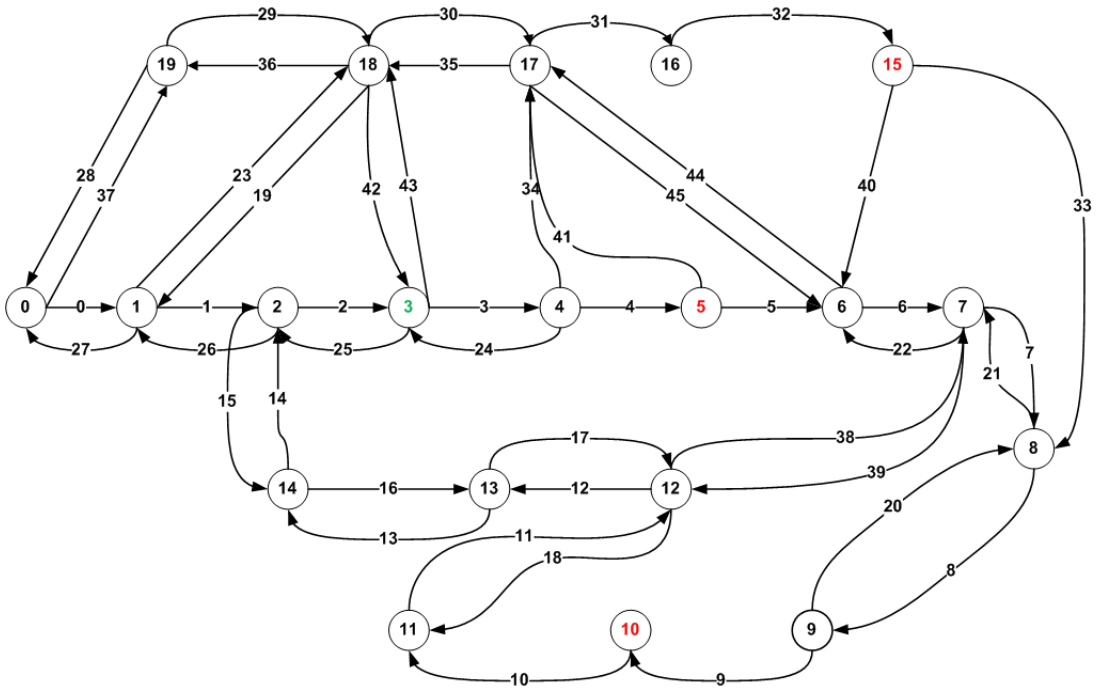
A "netconvert" parancs segítségével létrehozza a térképet, a net.xml file-t.

Munkánk során az 5. ábrán látható térképet használtuk. A hálózatban húsz csomópont és negyvenhat él található. A megfigyelt autó a 3-as csomópontból indul, lehetséges célállomásai az 5, 10 és a 15 csomópont.

A forgalmi információk a térképet alapul véve, a SUMO által generált rou.xml file-ban találhatóak.

Egy .rou.xml file felépítése:

```
<routes>
   <vehicle id="Rand1" depart="0">
      <route>edge1 edge2 edge3 edge34</route>
   </vehicle>
```



5. ábra. Térkép

A SUMO program képes random helyekről indítani az autókat a térképen, de nem képes ezeket időben is elosztani, így ezeket az indulási időpontokat véletlenszerűvé kell tenni annak érdekében, hogy valósághűbb forgalommal dolgozhassunk.
A vizsgálat folyamán azonban nem ezeket a rou.xml file-okat használjuk, hanem az ezek felhasználásával, a SUMO segítségével generált részletes xml file-okat. A SUMO program egy térkép leírás (net.xml) és az egyes autók által bejárt utakat tartalmazó leírás (rou.xml) megadásával a "netstate-dump" parancs segítségével képes egy olyan részletes xml file-t generálni, mely tartalmazza azt, hogy melyik autó, melyik időpillanatban hol volt éppen.

Ezek az alábbi felépítésűek:

```
<sumo-netstate>
   <timestep time="28">
      <edge id="edge6">
         <lane id="edge6_0">
            <vehicle id="Rand1" pos="5.00" speed="0.00"/>
         </lane>
      </edge>
   </timestep>
```

4.2.2. Paraméterek

Annak érdekében, hogy munkánk áttekinthető, és követhető legyen, konfigurációs file-ok írása célszerű. Ezek a file-ok tartalmazzák azokat a paramétereket, amelyeket változtatunk annak érdekében, hogy minél jobb pszeudoním-váltó algoritmust hozzunk létre.

Általunk változtatott paraméterek:

- Támadó erőssége

  Hány lehallgatott csomópont van.
  A dolgozatban összehasonlításra kerül 5, 8, 11, 15 ill. 20 megfigyelt csomópontot feltételező eset.

- Forgalom mérete

  Mennyi autó tartozik a forgalomhoz
  A dolgozatban kis térképrészleten közlekedő, kevés járművet vizsgálunk.

- Célállomások száma

  Feltételezzük, hogy a támadó tisztában van 'áldozata' célállomásaival.
  A vizsgált esetben három célállomás közül választhat a célpont jármű.

- Csereidő

  Milyen időközönként vált pszeudonímet a jármű.
  A dolgozatban a fix idejű váltások 20 illetve 40 másodpercenként történnek, a véletlen idejű váltások pedig a 20 és 40 másodperc közötti intervallumból kerülnek ki.

- Váltás valószínűsége

  Milyen valószínűséggel cserél azonosítót a jármű egy adott időpillanatban.
  A dolgozatban az adott pillanatban 0,05 illetve a 0,025 esélyű váltásokat vizsgáljuk.

4.2.3. Tudásbázis

Mint azt már korábbi fejezetben említettük, szükség van egy támadói tudásbázis létrehozására. A tudásbázist előzetes tapasztalatok alapján hozza létre, tehát korábbi eredmények szerint súlyozza azt, hogy egy adott szakaszt mennyi idő alatt tehet meg az áldozata. Olyan dokumentumot készít magának, mely tartalmazza, hogy melyik pontból melyik pontba lehet eljutni, és ezeket a távolságokat milyen időintervallumon belül képes teljesíteni. A tudásbázist befolyásolja, hogy a támadó hány csomópontot képes lehallgatni, illetve, hogy mennyi jármű közlekedik az adott területen.

4.2.4. Algoritmusok

A munka folyamán három cserealgoritmust implementáltam. A vizsgálathoz ennek a három módszernek az összehasonlítása szükséges. Azonban az egyes algoritmusoknak is több bemeneti paramétere

van, ezért nem elégséges csupán a három közül kiválasztani a leghatékonyabbat, a legmegfelelőbb beállításokra van szükség. A létrehozott három cserealgoritmus közös alapja az, hogy bizonyos időközönként lecserélik az azonosítójukat.

Érdemes még megjegyezni, hogy ezekben a scriptekben a könnyebb szimuláció érdekében nem cseréljük le teljesen az autók azonosítóit, hanem egy további szám kerül hozzáfűzésre, amely azt is megmutatja, hogy hányadszor cseréli az azonosítóját. Azonban az általunk elkészített támadó nem tudja összekötni ezeket az azonosítókat. Pl. Rand33 nevű autó azonosítója az első csere után Rand33-1, a második csere után Rand33-2 lesz. De ez a számon tartható rendszer könnyen megváltoztatható, ha a követés elkerülése a cél.

1. cserealgoritmus. Az első változat egy egyszerű váltást tartalmaz, azaz egy megadott idő után minden egyszerre induló autó egyszerre cseréli le az azonosítóját, és egész útjuk folyamán rendszeresen, az adott csereidőnként változtatják. Ezt a csereidőt a konfigurációs file-ban állítani lehet.

Algoritmus 1

```
1: while timestep do
2:     for all auto do
3:         if auto.szamlalo = csereido then
4:             auto.id ← ujid
5:             auto.szamlalo ← 0
6:         end if
7:     end for
8:     auto.szamlalo++
9: end while
```

2. cserealgoritmus. A második megoldás azt teszi lehetővé, hogy egy megadott tartományból minden autó sorsoljon magának egy cserélési időt, majd miután ez letelt, sorsoljanak egy újabbat. Ebben az esetben befolyásolhatjuk azt a tartományt, ahonnan a járművek azonosítói származhatnak.

Algoritmus 2

```
1: while timestep do
2:     for all auto do
3:         if auto.szamlalo = csereido then
4:             auto.id ← ujid
5:             csereido ← rand(param)
6:             auto.szamlalo ← 0
7:         end if
8:     end for
9:     auto.szamlalo++
10: end while
```

3. cserealgoritmus. A harmadik változat más ötleten alapul. Minden autó, minden időpontban dönt, hogy váltson-e azonosítót, vagy sem. Paraméterként állíthatunk egy értéket, annak valószínűségét, hogy egy adott pillanatban a váltás mellett dönt a jármű.

Algoritmus 3

```
1: while timestep do
2:     for all auto do
3:         if randnum ≤ prob then
4:             auto.id ← ujid
5:         end if
6:     end for
7: end while
```

4.2.5. A támadó eredményessége

A támadó tudása három fő részből tevődik össze:

- Látja, hogy hol jelenik meg az áldozata.
- Érzékeli, amikor belép a keverő zónába, és nem tudja tovább követni.
- Tisztában van áldozata potenciális célállomásaival.

Célja a megjelenési, eltűnési és megérkezési események helyes összekapcsolása. Munkánk során implementálásra kerül egy olyan gráf, melynek csomópontjait az egyes események adják, az éleket pedig a támadó a legjobb tudása szerint helyezi el ezen események között. Az élek súlyait a tudásbázis alapján határozza meg. Akkor sikeres a támadó, ha össze tudja kötni áldozata keverő zóna előtti azonosítóját a keverő zóna utáni új azonosítóval minden egyes alkalommal, amikor az azonosítót cserél. Az így összekötött eseményekből hozza létre a forgalmi mátrixot.
A támadó eredményességét az jelenti, ha végig tudja követni az áldozatát útja folyamán, a felbukkanástól a megérkezésig, tehát mutatni tud egy olyan legrövidebb utat a gráfban, amelynek csomópontjaiban csak az áldozat járművéhez tartozó események találhatóak meg.
Ilyen legrövidebb út keresési algoritmus:

- a Dijkstra algoritmus:

  mohó algoritmus, irányított gráfokban lehet megkeresni a legrövidebb utakat egy adott csúcspontból kiindulva.

- a Bellman-Ford algoritmus:

  egy pontból induló legrövidebb utak (hosszának) meghatározására, ha bizonyos élsúlyok negatívak. De feltesszük, hogy a gráf nem tartalmaz negatív összhosszúságú irányított kört.

- Floyd algoritmus

  segítségével az összes pontpár közötti távolságot meg lehet határozni.

Mivel nem dolgozunk negatív élekkel, és csak meghatározott két pont közötti útra vagyunk kíváncsiak, ezért a Dijsktra algoritmust használjuk.
A gráfba a tudásbázisból kiolvasott élsúlyok kerülnek. Az egy csomópontból kiinduló éleket normalizáljuk, majd logaritmizáljuk, és vesszük az ellentettjüket. Így a legvalószínűbb élek súlya lesz a legkisebb. Ezek után a Dijsktra algoritmussal megkeressük a legrövidebb utat 's' és 'd' csomópont között. Az eredményesség megállapításához megvizsgáljuk, hogy az áldozat autójának mennyi időre van szüksége, hogy elérje a célállomását, illetve, hogy a támadó ezen úton mennyi ideig tudta követni. Ezek aránya mutatja a támadó sikerességét. Két tulajdonságát vizsgáljuk meg az így kapott értékeknek: az átlagukat és a tapasztalati szórásukat.
Ha az átlag egy, vagy kicsivel elmarad egytől, a támadó jó eséllyel sikeres, ha nullához közelebb álló értékeket kapunk, akkor nagyobb az esély arra, hogy ne tudja végig követni az áldozatot.

4.3. Eredmények

Szimulációnkban az áldozat több potenciális célállomással (3 db) rendelkezik. Annak érdekében, hogy vizsgálható eredményekhez jussunk, több forgalmi minta (60 db) esik támadás áldozatául ugyanazzal a támadói tudásbázissal. A különböző forgalmi mintákban az áldozat célállomásai egy adott eloszlás szerint változnak.

A vizsgálat első lépéseként ellenőriztük, hogy ha nem történik pszeudoním váltás, akkor mennyire tudja a támadó követni az áldozatát. Megállapítottuk, hogy a minimális lehallgatási pont (a kiinduló pont és a potenciális megérkezési pontok) elég a támadónak célja eléréséhez, ha nem történik azonosító váltás.

A támadó eredményességének vizsgálatához két értéket hasonlítunk össze az egyes cserealgoritmusok, illetve azok különböző beállításai esetén. Ezekről az eredményekről elmondható, hogy mindegyiknél azonos a forgalom sűrűsége, és az áldozat kiinduló pontja, valamint célállomásai.

Az anonimitás szintjének megállapításához használhatjuk a Levine-Schields-féle taxonómiát [9], mely szerint az anonimitás egy 0 és 1 közé eső érték (azaz egy valószínűség), ahol 0 az anonimitás teljes hiányát jelöli, az 1 pedig a teljes anonimitást. (*A dolgozatban éppen fordítva van, az 1 a teljes követhetőséget jelenti (anonimitás hiánya), míg a 0 a követhetetlenség (tökéletes anonimitás)*)

Legyen $Pr_e(x)$ annak a valószínűsége, hogy x a kezdeményezője egy kommunikációnak az e megfigyelő szerint. Egy anonim csoport számára $\sum_{x \in S} Pr_e(x) = 1$ (ahol S az anonim csoportban részvevők száma). Az x résztvevő számára biztosított anonimitás az e megfigyelővel szemben legyen $d_{x,e}(A)$, ahol A legyen a használt anonimizáló protokoll. Ekkor legyen $d_{x,e}(A) = \sum_{y \in S \neq x} Pr_e(y)$ – illetve ezzel ekvivalensen: $d_{x,e}(A) = 1 - Pr_e(x)$ – ami maga az anonimitás szintje, melynek különböző értékeire például a következő szintek adhatók meg:

1. *Abszolút anonimitás*: Egy támadó nem képes megkülönböztetni a kommunikációkat. Ekkor $d_{x,e}(A) = 1$.
2. *Gyanú felett*: A kommunikáció néhány tényezője ismert a támadó számára, de a kommunikáció kezdeményezője nem megkülönböztethető a többi résztvevőtől: $d_{x,e}(A) \geq (1 - 1/|S|)$ és $d_{y,e}(A) \leq d_{x,e}(A)$, minden $y \neq x \in S$-re (itt $|S| > 1$).
3. *Lehetséges ártatlanság*: annak a valószínűsége, hogy egy x entitás kezdeményezője egy kommunikációnak nem nagyobb, mint annak a valószínűsége, hogy nem kezdeményezője, de a többi entitásnál nagyobb valószínűségű a támadó szemében: $1/2 \leq d_{x,e}(A) \leq d_{y,e}(A)$ minden $y \neq x \in S$ esetén. Ekkor nyilván $d_{x,e}(A) < (1 - 1/|S|)$.
4. *Leleplezve*: Fennáll még a valószínűsége annak, hogy a támadó nem tudja azonosítani a kezdeményezőt, bár ez meglehetősen kicsi. $0 \leq d_{x,e}(A) \leq 1/2$.
5. *Bizonyítottan leleplezve*: A támadó képes bizonyítani a kezdeményező kilétét: $d_{x,e}(A) = 0$.

(*A táblázatokban a fenti számozás alapján hivatkozunk az anonimitási szintekre.*)

Az anonimizáló rendszerek végső célja a lehetséges ártatlanság, vagy annál magasabb szint elérése. Az abszolút anonimitás elvileg sem megvalósítató, mert feltételezhető olyan megfigyelő, aki kellő erőforrással rendelkezik ahhoz, hogy közel minden hálózati forgalmat monitorozzon.

2. táblázat. Lehallgatott csomópontok száma: 5

Lehallgatott csomópont száma: 5				
Cserealgoritmusok	Paraméterek	Követhetőség átlaga	szórása	Anonimitás
Fix időnkénti váltás	20	0,0942	0,5603	3
	40	0,1942	0,5603	3
Véletlen időnkénti váltás	20-40	0,1233	0,5594	3
Fix valószínűségű váltás	0,05	0,1076	0,5740	3
	0,025	0,2008	0,4846	3

A 2. táblázatban található értékek mindössze öt megfigyelt csomópont által nyújtott információkból származnak. Ezért itt van a legnehezebb dolga a támadónak, az anonimitási szint konstans 3. Az eredmények szerint főleg sikertelenek a követési próbálkozások.

3. táblázat. Lehallgatott csomópontok száma: 8

Lehallgatott csomópont száma: 8				
Cserealgoritmusok	Paraméterek	Követhetőség átlaga	szórása	Anonimitás
Fix időnkénti váltás	20	0,1825	0,0495	3
	40	0,3959	0,0679	3
Véletlen időnkénti váltás	20-40	0,2124	0,1885	3
Fix valószínűségű váltás	0,05	0,1510	0,1156	3
	0,025	0,2964	0,1986	3

A 3. táblázatban azokat az eseteket tanulmányoztuk, amikor az előbbiekhez képest még újabb három jól megválasztott lehallgatási pont került a rendszerbe. Az eredmények megfelelnek a várakozásainknak, a támadó itt már sikeresebb, viszont az anonimitási szint még itt is állandó mindegyik esetre nézve, 3.

A 4. táblázatban azokat az eredményeket foglaltuk össze, amiket tizenegy, a támadó által lehallgatott csomópont esetén kaptunk. A táblázatból kitűnik, hogy ha egy jármű adott körülmények között csak negyven másodpercenként vált pszeudonímet, akkor teljes mértékben követhető marad a támadó számára. Azonban ha kétszer ilyen sűrűséggel teszi, sokkal sikertelenebbek a követési próbálkozások. Abban az esetben, ha egy jármű azt az algoritmust használja, miszerint sorsol magának egy adott intervallumból (*itt: [20-40]*) egy véletlen váltási értéket, majd ennek lejárta után egy újabbat, akkor az elemzett forgalom szerint átlagosan 40% a támadó sikeressége. A harmadik váltási algoritmus szerint minden időpillanatban eldönti az autó, hogy cserél-e pszeudonímet, vagy sem. Ha nagyobb eséllyel vált, akkor kevésbé eredményes a támadó, de ebben az esetben még a kisebb valószínűségű váltás esetén is csak átlagosan az utazási idő 56%-áig követhető. Ekkora mennyiségű lehallgatott csomópontnál még érdemes a fix idejű váltásokat használni.

Megvizsgáltuk azt az esetet is, amikor tizenöt csomópont áll a támadó rendelkezésére. Ennek értékeit

4. táblázat. Lehallgatott csomópontok száma: 11

Lehallgatott csomópont száma: 11				
Cserealgoritmusok	Paraméterek	Követhetőség átlaga	szórása	Anonimitás
Fix időnkénti váltás	20	0,1825	0,04957	3
	40	1,0000	0,0000	5
Véletlen időnkénti váltás	20-40	0,3918	0,4582	3
Fix valószínűségű váltás	0,05	0,3504	0,3266	3
	0,025	0,5612	0,4879	4

5. táblázat. Lehallgatott csomópontok száma: 15

Lehallgatott csomópont száma: 15				
Cserealgoritmusok	Paraméterek	Követhetőség átlaga	szórása	Anonimitás
Fix időnkénti váltás	20	1,0000	0,0000	5
	40	1,0000	0,0000	5
Véletlen időnkénti váltás	20-40	0,9992	0,0007	4
Fix valószínűségű váltás	0,05	0,9322	0,8500	4
	0,025	0,9832	0,0168	4

az 5. táblázat tartalmazza. Változás a 4. táblázathoz képest, hogy ennél a lehallgatott csomópont mennyiségnél már érdemesebb a harmadik típusú cserealgoritmust használni. Arra az eredményre jutottunk, hogy szinte 100% annak az esélye, hogy a támadó sikeres lesz. A fix idejű pszeudoním váltás ezen a szinten már nem megfelelő. Az eredmények nagy hasonlóságot mutatnak azzal a változattal, mint amikor az összes csomópontot képes lehallgatni a támadó.

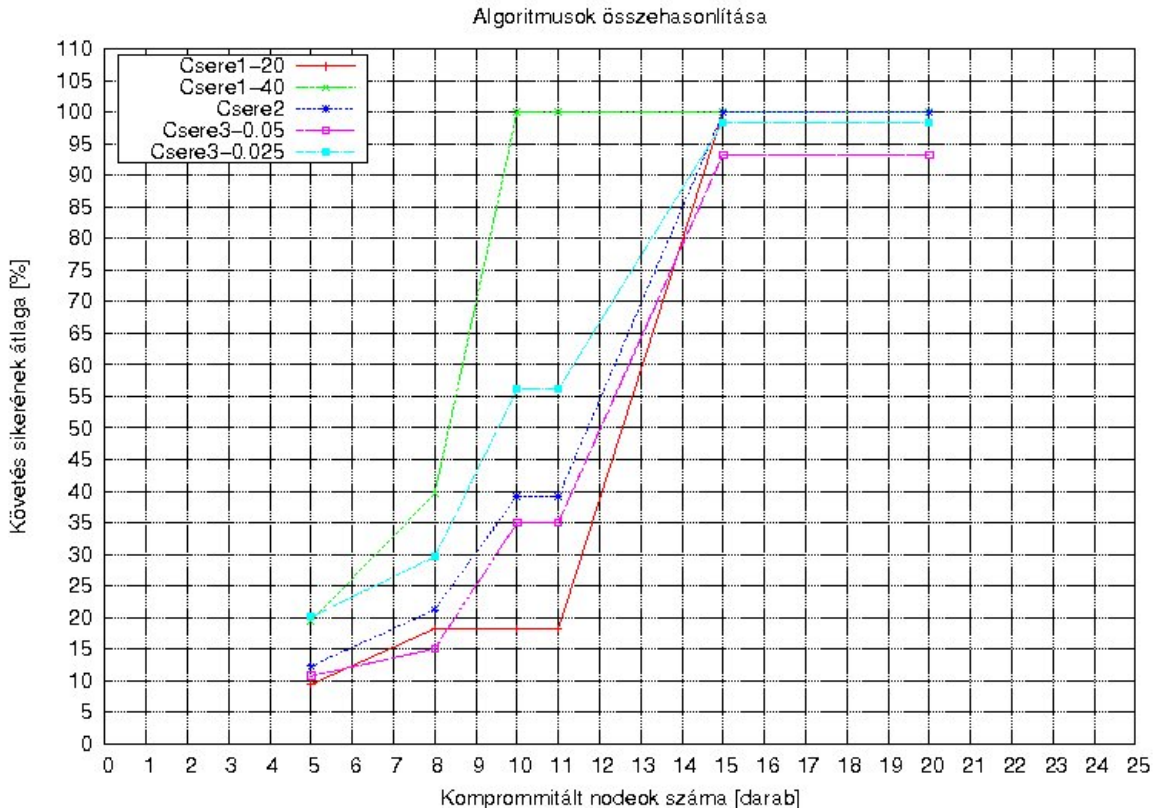
6. táblázat. Lehallgatott csomópontok száma: 20

Lehallgatott csomópont száma: 20				
Cserealgoritmusok	Paraméterek	Követhetőség átlaga	szórása	Anonimitás
Fix időnkénti váltás	20	1,0000	0,0000	5
	40	1,0000	0,0000	5
Véletlen időnkénti váltás	20	0,9990	0,0009	4
Fix valószínűségű váltás	0,05	0,9323	0,8501	4
	0,025	0,9829	0,0171	4

Természetesen megvizsgáltuk, hogy milyen eredményt adnak az algoritmusok, ha minden csomópontot lehallgat a támadó. Ezek összefoglalását a 6. táblázat tartalmazza. A várt eredményre jutottunk, tehát szinte teljes mértékben követhetővé válnak az áldozatok bármely pszeudoním váltási algoritmus

ellenére.
Az eredmények azt mutatják, hogy ha a csomópontok 75%-t megfigyeli a támadó, akkor a járművek védtelenné válnak a követéssel szemben. Ezt azonban azért nem jelenthetjük ki ilyen biztosan, mert a forgalom ugyan életszerű (mivel SUMO-val készül), viszont kis térképrészleten kevés autó közlekedik, így valós környezetben más eredményeket kaphatunk.

A 6. ábra a különböző algoritmusok, és az azokhoz tartozó paraméterek összehasonlítását tartalmazza. Azaz bemutatjuk a támadás sikerességének arányait a lehallgatott csomópontok, és a választott algoritmusok függvényében. A tapasztalatok alapján a lehallgatott csomópontok számának növekedésével, növekszik a támadó sikerességének esélye bármely pszeudoním váltási algoritmus esetén. Érdekes tapasztalat az óriási különbség a két fix váltási idejű algoritmus között. Az a változat, ahol ritkábban kerül sor a cserére, egyértelműen a legkevésbé megfelelő a céljainkhoz, még a másik sok esetben a legalkalmasabb.
Összefoglalásként elmondható, hogy a Csere1-20 és a Csere3-0.05 a két legjobb algoritmus. Azt, hogy melyiket jobb használni, azt a lehallgató csomópontok száma dönti el egy adott forgalomban.



6. ábra. Algoritmusok összehasonlítása

5. Kapcsolódó kutatások

A technika fejlődése, az idő és a tér korlátainak jelentőségének csökkenése magával hozza a biztonság iránti vágy fokozódását. Ennek következménye, hogy egyre több kutatócsoport foglalkozik nem csupán az autók közötti kommunikáció fejlesztésével, hanem a biztonságos adatátvitellel, valamint a járulékos gyengülések kiküszöbölésével, ilyen például az egyes járművek követhetősége.

Egy potenciális megoldást kínál Krishna Sampigethaya munkatársaival a [13] cikkben, melynek alapötlete, hogy véletlenszerű időközönként "eltűnik" az autó, így gátolva meg egy esetleges követést. A CARAVAN-nak nevezett eljárás azon alapszik, hogy a kommunikáció folyamatába random csöndes időszakokat illesztenek be, azt remélve, hogy ennek köszönhetően nem lehet az adott járműhöz kapcsolni az általa bejárt helyszíneket. Ezt a módszert azonban csak autópálya, és véletlenszerűen generált Manhattan forgalmi modellre értékelték ki. A mi szempontunkból azért kiemelkedő munka, mert egy passzív globális támadó (a lehallgató minden jármű minden broadcast üzenetét veszi) elleni védekezést ecsetel.

Másik védekezési mód a dolgozat alapjául is szolgáló keverő zónák (mix zone) elve. Alastair R. Beresford és munkatársai által kidolgozott leírásban [3] azt az ötletet finomítja, hogy az autók keverő zónákon haladjanak át, így a folyamatos pszeudoním cserélgetés esetén rontják a támadó követési esélyeit. Ebben a munkájukban [3] kiemelt figyelmet fordítanak a matematikai modell fejlesztésére, és a számítási igények vizsgálatára, és minimalizálására, de elemzik a kívánt anonimitás szintjét, és megemlítik a felhasználói oldali visszacsatolást is.

Felismerve a VANET veszélyeit, a keverő zónákban használatos pszeudonim váltásokról értekezik Emanuel Fonseca kollégáival az [7] cikkben. Problémaként felvetik, hogy számos bizalmas adat kerül továbbításra a VANET hálózatában, mint például a jármű azonosítója, pozíciója, sebessége, haladási iránya, illetve az ezekhez kötődő időpontok, és ezeket egy támadó könnyen a járművezetőhöz kapcsolhatja, ezzel megsértve a vezető privát szféráját. A cikk írói fő újításukként kiemelik a pszeudonímek integrálását egy valós VANET kommunikációs rendszerbe.

Rongxing Lu és társai cikkükben [10] egy hatékony magánszféra védelmi protokol (efficient conditional privacy preservation - ECPP) leírást kínálnak megoldásként a VANET biztonsági problémáira. Megvalósításként a járművek, és az útmenti egységek közötti rövid lejáratú anoním kulcsváltásokat ajánlják.

Mindenképpen említésre méltó Schoch és kollégái cikke [14], melyben a pszeudoním váltás hátrányos hatásaira (különösen a hatékony útvonalválasztás, vagy a csomagvesztési arány kérdése) próbálnak megoldást keresni. Céljuk egy olyan rendszer, melyben megfelelő szintű a személyes szféra védelme, de a teljesítményre vonatkozó mérések is kielégítő eredményt adnak.

Florian Dötzer írása [5] az autógyártók szemszögéből közelíti meg a kérdést. Jó áttekintést kínál a VANET-ekről, és az ezeket érintő biztonsági kérdésekről, nem csak a külső támadót említi meg, mint veszélyforrás, hanem felveti annak problémáját is, ha a hatóságok helytelenül használják személyes adatainkat. Hiszen a rendszer biztonságos működéséhez elengedhetetlen a felhasználók anonimizálása, de egyben fontos, hogy a hivatalos oldal felé egyértelmű maradjon, hogy kit takar az álnév. A cikkben kompromisszumot keresnek a teljes anonimitás, és a között az állapot között, amikor egyáltalán nem védik a járművek a kilétüket.

Stephan Eichler munkája [6], ötletei nagy hasonlóságot mutat ezzel a dolgozattal. A VANET környezetbeli pszeudoním váltási lehetőségeket elemzi, és szimulációs eredményeket mutat. A probléma az a megközelítésével, hogy a kevéssé realisztikus Manhattan hálós modellt használja szimulációs térképként, azzal a szándékkal, hogy valós forgalmat mutasson. Eredményeivel egy fix pszeudoním váltási értéket támaszt alá. Ezzel az a probléma, hogy ebben az esetben a fő cél sérül, tehát a járművek nem fogják időben megkapni az esetlegesen életmentő információkat. Megközelítése eltér a többi műtől, hiszen csak egy adott jármű környezetében előforduló többi járművet említi meg veszélyforrásként, potenciális támadóként.

A dolgozat a Buttyán Levente, Holczer Tamás és Vajda István munkájára [4] épül. A tovább lépést az mutatja, hogy nem egy keverő zónába belépésről van szó, hanem többön való áthaladáson, és mindezt folyamatos pszeudoním váltásokkal.

6. Összefoglalás

Napjainkban egyre fontosabbá válik mind a járműközi kommunikáció, mind pedig a személyes szféra védelme. E kettő összefonódásának kérdését elemzi ez a dolgozat, azaz azt, hogyan lehet a magánszférát megvédeni, de mégis élvezni az autók közötti kommunikáció előnyeit, kiemelt figyelmet szentelve a követhetőség elkerülésére.

A dolgozatban az egyes pszeudoním váltási algoritmusok összehasonlítása volt a cél, ezért a szimulációkban egy kis térképrészleten vizsgáltuk a forgalmat, illetve azt, hogy ilyen körülmények között mennyire lehet hatékonyan védekezni a járművek követése ellen. A védekezés alapja a keverő zónák elvére épül. Időalapú pszeudoním váltási módszereket alkalmaztunk, azaz a járműveknek lehetőségük van bizonyos időközönként úgy pszeudonímet váltani, hogy a támadó ne tudja követni őket.

Több algoritmust is összehasonlítottunk, annak érdekében, hogy megtudjuk, melyik a leghatékonyabb a támadóval szemben. Láthattuk, hogy védelem nélkül a támadó a forgalom megfigyelésével mindig képes nyomon követni a kiszemelt járművet, azonban pszeudonímek használatával a sikeres követés valószínűsége csökkenthető. Még egy ekkora méretű térképen, kis forgalom mellett is tisztán kitűnik az eredményekből a lehallgatott csomópontok számának (a támadó erejének), illetve az algoritmusoknál alkalmazott paraméterek szerepe.

A pszeudoním váltási algoritmusok közül a Csere1-20, bizonyult a legsikeresebbnek, alacsonyabb számú lehallgatott csomópont esetén. Abban az esetben viszont, amikor sok a lehallgató készülék a forgalomban, a Csere3-0.05 jobb eredményeket mutat. Ezekhez képest a Csere1-40 látványosan gyengébb védelmet tud nyújtani. A Csere2, illetve a Csere3-0.025 nem mutat sem pozitív, sem negatív irányba kiemelkedő értékeket.

A támadó a védelemmel szemben tizenöt csomópont irányítása mellett már hatékonyan tudja követni a kiszemelt járművet, ekkor a csomópontok 75 százalékát uralja, melyről felételezhetjük, hogy a valóságban igen ritkán fordulhat elő, hiszen ekkor voltaképpen szinte az egész hálózat a birtokában van, globális lehallgatás ellen kellene küzdeni. Amennyiben a csomópontok fele, vagy annál kevesebb van a támadó irányítása alatt, úgy a támadás sikeressége 40-50 százalék körüli volt, így elmondhatjuk, hogy a védekezés többé-kevésbé sikerrel zárult.

Kutatásunk következő lépéseként meg szeretnénk vizsgálni valóságos térképek, és nagyobb forgalom esetén az alkalmazott cserealgoritmusokat. Középtávú cél nem csak idő-, hanem téralapú cserealgoritmusok vizsgálata, azaz hogy nem csak azt vizsgáljuk, hogyan lehet védekezni a követhetőség ellen, ha bizonyos időnként pszeudonímet vált a jármű, hanem azt, hogy a váltásokat bizonyos megtett útszakasz után hajtja végre.

7. Irodalomjegyzék

Hivatkozások

[1] A PERL weboldala. http://www.perl.org/.

[2] A SUMO weboldala. http://sumo.sourceforge.net/overview.shtml.

[3] Alastair R. Beresford – Frank Stajano: Mix zones: User privacy in location-aware services. In *IEEE International Workshop on Pervasive Computing and Communication Security (PerSec)* (konferenciaanyag). 2004. Március.

[4] Levente Buttyán – Tamás Holczer – István Vajda: On the effectiveness of changing pseudonyms to provide location privacy in vanets. In *ESAS* (konferenciaanyag). 2007, 129–141. p.

[5] Florian Dotzer: Privacy issues in vehicular ad hoc networks. In *Workshop on Privacy Enhancing Technologies* (konferenciaanyag). 2005. Május.

[6] Stephan Eichler: Strategies for pseudonym changes in vehicular ad hoc networks depending on node mobility. In *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)* (konferenciaanyag). 2007. június, 541–546. p.

[7] Emaunel Fonseca – Andreas Festag – Roberto Baldessari – Rui Aguiar: Support of anonymity in vanets - putting pseudonymity into practice. In *IEEE Wireless Communications and Networking Conference (WCNC)* (konferenciaanyag). 2007. Március.

[8] Julien Freudiger – Maxim Raya – Márk Félegyházi – Panos Papadimitratos – Jean-Pierre Hubaux: Mix-zones for location privacy in vehicular networks. In *ACM Workshop on Wireless Networking for Intelligent Transportation Systems (WiN-ITS)* (konferenciaanyag). 2007. Augusztus.

[9] B. Levine – C. Shields: Hordes: A multicast-based protocol for anonymity. In *B. N. Levine and C. Shields. Hordes: A multicast-based protocol for anonymity. Journal of Computer Security, 10(3):213- 240, 2002.* (konferenciaanyag). 2002.

[10] Rongxing Lu – Xiaodong Lin – Haojin Zhu – Pin-Han Ho – Xuemin Shen: Ecpp: Efficient conditional privacy preservation protocol for secure vehicular communications. In *The 27th IEEE International Conference on Computer Communications* (konferenciaanyag). 2008.

[11] PERL leírás. http://weblabor.hu/cikkek/perlalapjai1.

[12] PKI leírás. http://www.berta.hu/files/berta–pki-gyakorlati-problemak–2008.09.18.ppt.pdf.

[13] Krishna Sampigethaya – Leping Huang – Mingyan Li – Radha Poovendran – K. Matsuura – K. Sezaki: Caravan: Providing location privacy for vanet. In *Proceedings of Embedded Security in Cars (ESCAR)* (konferenciaanyag). 2005. November.

[14] Elmar Schoch – Frank Kargl – Tim Leinmuller – Stefan Schlott – Panos Papadimitratos: Impact of pseudonym changes on geographic routing in vanets. In *Third European Workshop on Security and Privacy in Ad hoc and Sensor Networks (ESAS 2006)* (konferenciaanyag). 2006.

A. Függelék

Az X509 v3 szabvány szerinti Digitális Tanúsítvány felépítése:

- Tanúsítvány (Certificate)
  - Verzió (Version - V3)
  - Sorozatszám (Serial Number)
  - Aláírás algoritmus (Algorithm ID)
  - Kibocsájtó azonosító (Issuer)
  - Érvényesség (Validity)
    - Kezdete (Not Before)
    - Vége (Not After)
  - Tulajdonos azonosító (Subject)
  - Publikus kulcs (Subject Public Key Info)
    - Public Key Algorithm
    - RSA Public Key
  - Toldalékok (Extensions)
- Aláírás algoritmus (Certificate Signature Algorithm)
- Aláírás (Certificate Signature)