híradástechnika

1945 VOLUME LXIII. 2008

hírközlés • informatika

Szoftverbiztonság

Kvantumkriptográfia

DRM rendszerek

Botnetek

2008/11

A Hírközlési és Informatikai Tudományos Egyesület folyóirata
a Nemzeti Hírközlési és Informatikai Tanács együttműködésével
és a Nemzeti Kulturális Alap támogatásával

nka

Tartalom

Címlapfotó: Dankó András

A Híradástechnika folyóirat megújulása elé

Kedves Olvasóink!

Az eddigiekben megpróbáltuk összehangolni azt a kettős cékitűzésünket, hogy lapszámainkban egyaránt helyet kapjanak az új kutatási eredményeket bemutató közlemények és a színvonalas szakmai ismeretterjesztést szolgáló áttekintő cikkek. A jövőben – figyelembe véve olvasóink érdeklődését és elvárásait – lényeges előrelépést szeretnénk elérni az áttekintő cikkek számát és minőségét illetően. Egyben megfontoltuk azt a tényt is, hogy amennyiben a kutatási jellegű cikkek csak magyarul látnak napvilágot, akkor óhatatlanul szűk olvasóközönség számára érdekesek csupán, miközben az angol számokban való publikálásuk egyrészt az adott témát művelő, jóval tágabb nemzetközi közösség számára teszik lehetővé a közzétételt, másrészt idézhetővé, referálhatóvá válnak ezek a munkák.

A fentiek alapján a szerkesztőbizottság – a HTE vezetőségének jóváhagyása és támogatása mellett – a jövőben szeretné megvalósítani azt, hogy a magyar számok döntő részben szélesebb közönségnek szóló áttekintő cikkekből álljanak, melyek mellett rendszeresen közölnénk könyvismertetőket, projektbeszámolókat, szakmai híreket, érdekességeket, interjúkat is.

A magyar folyam így jobban betölthetné azt a szerepét, hogy a szakma egyetlen magyar nyelvű, színvonalas ismeretterjesztő folyóirataként közvetítse az egyes részterületeket helyzetét, fejlődésének irányait és legújabb eredményeit a jelenleginél szélesebb olvasótábor számára és formálja, befolyásolja a magyar szaknyelvet.

Terveink szerint új rovatokkal fogjuk bővíteni lapszámainkat, azt tervezzük, hogy rendszeresen jelentkezünk könyvismertetésekkel, konferenciákról, fontos szakmai eseményekről szóló beszámolókkal, hazai és nemzetközi projektek ismertetéseivel, a HTE szakosztályok tevékenységét bemutató cikkekkel, valamint egyetemi és kutatóintézeti egységek bemutatkozásaival.

Publikációs fórumként, bírált kutatási cikkek megjelentetésére az angol nyelvű számok fognak szolgálni. Ezekben a számokban lehetnek az eredmények hozzáférhetők, idézhetők, hivatkozhatók az alapvetően nemzetközi kutató közösség számára. Fokozatosan szeretnénk megteremteni a lehetőségét annak, hogy az angol folyamot a későbbiekben bírált folyóiratként ismerje el a nemzetközi szakmai közösség. Ehhez első fontos lépésként az eddigi évi 2-ről 4-re növeljük az angol kiadások számát.

Bár az eddigiekben is törekedtünk a kutatási cikkek független bíráltatására, a fenti elképzelés sikeréhez a nemzetközi folyóiratokban szokásos bírálati procedúra általános és következetes alkalmazására lesz szükség. Kialakítunk egy állandó bírálói kört, lehetőleg minél több külföldi szakember bevonásával. A jelenlegi szerkesztőbizottságunk mellé létrehozunk egy International Advisory Committee-t, amelynek tagjai ösztönöznék saját környezetükben a lapunkban történő publikálást és közreműködnének a bírálati folyamat lebonyolításában.

Szerkesztőbizottságunk tagjai a jövőben is egy-egy fontos részterület „gazdái” maradnak és a továbbiakban is tervezzük célszámok, célszám-részek megjelentetését, többek között az alábbi területeken:

– vezetéknélküli és mobil kommunikáció,
– optikai hírközlés,
– digitális műsorszórás,
– infokommunikációs szolgáltatások,
– internet-technológiák és alkalmazások,
– médiainformatika,
– multimédia rendszerek,
– kábeltelevíziós rendszerek
– távközlési szoftverek,
– adat- és hálózatbiztonság,
– űrtávközlés,
– infokommunikáció a közlekedésben,
– gazdasági és szabályozási kérdések,
– az infokommunikáció társadalmi vonatkozásai.

Az új szerkesztési elveknek megfelelően a 2009-es évben a következőképpen alakul majd a magyar és angol számok megjelenése:

Január, április, július és október, tehát negyedévente: angol számok – „Infocommunications Journal” címmel. Február, április, június, augusztus, október és december, tehát kéthavonta: a „Híradástechnika” magyar számai.

Bízunk benne, hogy a tervezett változtatások megnyerik olvasóink tetszését és a korábbiaknál többen fogják haszonnal forgatni számainkat. Természetesen várjuk cikkeiket is, mind a magyar, mind az angol számokba!

Zombory László, a szerkesztőbizottság elnöke
és Szabó Csaba Attila főszerkesztő

Informatikai biztonság

buttyan@hit.bme.hu

A Híradástechnika jelen száma négy áttekintő jellegű ismeretterjesztő cikket tartalmaz az IT biztonság különböző területeiről.

Az első cikk – *Szekeres László, Tóth Gergely: Szoftverbiztonság* – a szoftverhibákból származó biztonsági problémákkal foglalkozik. A mai szoftverek komplexitása elképesztően nagy, ezért szinte biztosan tartalmaznak programhibákat. Ezek a hibák egy rosszindulatú támadó számára gyakran olyan lehetőségeket rejtenek, amelyek segítségével a támadó könnyen visszaéléseket tud elkövetni, és akár a szoftvert futtató hoszt felett teljes uralmat át tudja venni. Éppen ezért a szoftverhibák kihasználása gyakran más támadások kezdőlépése. A cikk áttekintést ad a szoftverbiztonság mai helyzetéről: A szerzők ismertetik a probléma jelentőségét, kiterjedtségét és a benne rejlő kockázatokat, majd bemutatnak néhányat a tipikus szoftverhibákból. Ezt követően a hagyományos védekezési módszereket veszik sorra, rávilágítva e módszerek hiányosságaira, majd a szoftverfejlesztés folyamata közben alkalmazható védelmi lehetőségekről írnak. A cikk végül egy kitekintéssel és néhány kutatási irány felvázolásával zárul.

Szentgyörgyi Attila és Szabó Géza: Bevezetés a botnetek világába címmel a botnetek témakörébe vezeti be az olvasót. A botnet (robot network) egy támadó által vezérelt, fertőzött számítógépekből álló hálózat, melyet a támadó összehangolt támadások kivitelezésére tud felhasználni. A botnetek napjaink egyik legelterjedtebb és legveszélyesebb károkozói. Az átlagfelhasználó keveset tud a működésükről és a védekezési módszerekről, pedig az első botnet megjelenése óta sok év eltelt már. Ezért a szerzők összefoglalják a botnetek működési elvét, áttekintik a botnetek életciklusát a keletkezéstől, a támadások végrehajtásán keresztül a megszűnésükig, valamint a botnetek felfedezését szolgáló módszereket is és egy rövid jövőképpel zárják a cikket.

Harmadik cikkünk – *Fehér Gábor, Polyák Tamás, Oláh István: DRM technológiák* – szintén kurrens témával, a digitális tartalmakhoz kapcsolódó szerzői jogok védelmével foglalkozik. Az analóg világban egy rögzített mű másolása minőségromlással járt, így aki igazi minőségre vágyott, az kénytelen volt fizetni a tartalomért. Napjainkban azonban más a helyzet, hiszen a digitális másolat minősége megegyezik az eredeti mű minőségével. Szükségszerű tehát a digitális tartalom védelme az illegális másolás és terjesztés ellen, valamint a legális felhasználás lehetővé tétele a tartalomhoz kapcsolt felhasználási jogok definiálásával. Ezzel a védelemmel és jogkezeléssel foglalkozik a DRM (Digital Rights Management). A cikk célja, hogy az olvasót megismertesse a DRM technológia alapjaival és a javasolt DRM rendszerek működésével. A szerzők röviden tárgyalják a DRM alternatíváit is.

Végül negyedikként, *Gyöngyösi László és Imre Sándor: A kvantumkriptográfia infokommunikációs alkalmazásai* című írása egy jövőbe mutató témával, a kvantum-alapú kriptográfiával ismerteti meg az olvasót. Mint ismeretes, a napjainkban alkalmazott nyílvános kulcsú titkosító algoritmusok biztonsága nehéznek vélt matematikai problémákra, például a faktorizáció nehézségére épül. Egy kvantumszámítógép azonban ezeket a problémákat is hatékonyan (polinomiális lépésszámmal) oldaná meg. A kvantumszámítógépek megjelenésével tehát a jelenlegi titkosítási módszerek nagy része veszélybe kerül, így a jövőben olyan titkosítási módszereket kell találnunk, amelyek megvédenek bennünket egy kvantumszámítógéppel rendelkező támadótól is. Valójában régóta ismeretes a tökéletes rejtjelezés fogalma és algoritmusa, amit még egy kvantum-támadó sem tud feltörni, ám az is ismert, hogy a tökéletes rejtjelezéshez nagy mennyiségű véletlen kulcsbitet kell a kommunikáló feleknek megosztaniuk, ami a gyakorlatban szinte használhatatlanná teszi a tökéletes rejtjelezőt. Szerencsére éppen a kvantummechanika az, ami erre a problémára megoldást kínál a kvantum-alapú kulcscsere formájában, melynek segítségével a felek képesek megegyezni egy a tökéletes rejtjelezésre alkalmas véletlen kulcsban. A cikk szerzői a kvantum-kulcscsere elméleti alapjainak ismertetésén túl, annak gyakorlati megvalósításáról is beszámolnak és az olvasó képet formálhat arról, hogy hol tart ma ez a technológia.

E számunkban helyet kapott egy beküldött kutatási cikk is: *Bodrog Levente, Horváth Gábor, Vulkán Csaba: A TCP/HSDPA rendszer átvitelének analitikus modellje*. Ebben a szerzők a TCP-átvitelt modellezik mobil, adatforgalmat nyújtó, HSDPA környezetben, a TCP csomagvesztési valószínűsége és a körbefordulási ideje segítségével, megalkotják a HSDPA-t leíró sorbanállási hálózatot, amely tartalmazza a torlódási pontokat és protokollrétegeket, amelyek hatással vannak a vesztésre és a késleltetésre.

Buttyán Levente
vendégszerkesztő

Szabó Csaba Attila
főszerkesztő

Szoftverbiztonság

SZEKERES LÁSZLÓ, TÓTH GERGELY

SEARCH-LAB Kft.
{laszlo.szekeres, gergely.toth}@search-lab.hu

Kulcsszavak: szoftverbiztonság, puffer-túlcsordulás, szoftvertesztelés, fuzzing, statikus elemzés, verifikáció

A programfejlesztés mai technikája mellett a jelenlegi rendszerekben rendkívül sok olyan programozási hiba marad, amelyek a rendeltetésszerű használat során nagyon ritkán jelentkeznek. Azonban ezen ártalmatlannak tűnő, a hétköznapi működést legtöbbször nem is befolyásoló hibák egy rosszindulatú támadó számára gyakran olyan lehetőségeket rejtenek, amelyek segítségével könnyen visszaéléseket tud elkövetni. A probléma fontosságát és a veszély nagyságát csak növeli, hogy adott esetben a támadó számára egyetlen hiba megtalálása és kihasználása elegendő a védelmi eszközök megkerüléséhez és a rendszer feletti teljes irányítás átvételéhez. Mivel ezek a hibák rendkívül komoly veszélyt jelentenek, a megelőzésük és az ellenük való védekezés óriási fontosságú.

1. Bevezetés

A mai társadalmunk nagymértékben függ az informatikai rendszerektől, és ez a függőség rohamos mértékben növekszik. Ez újabb és újabb biztonsági követelmények elé is állítja rendszereinket, melyeknek a mai technológia sokszor nem tud megfelelni. A szoftverfejlesztés ma egyike a legnehezebb mérnöki feladatoknak. Elsősorban ennek köszönhető az, hogy a működésben lévő szoftverekben rengeteg a hiba, sokszor nem megfelelően működnek, valamint megbízhatatlanok, nem robusztusak. Sajnos ezek a hibák gyakran nem csupán a funkcionalitásban okoznak hiányosságokat, hanem biztonsági szempontból is: biztonsági lyukakat, sebezhetőségeket eredményeznek.

A cikkben áttekintést szeretnénk adni az olvasónak a mai szoftverbiztonság helyzetéről. A második szakaszban tisztázzuk a szoftverbiztonság szó jelentését. Ezután ismertetjük a probléma jelentőségét, kiterjedtségét és a benne rejlő kockázatokat. A negyedik szakaszban bemutatunk néhányat azokból a tipikus szoftverhibákból, melyek a problémák gyökereit képezik. Ezekután bemutatjuk a hagyományos védekezési módszereket a szoftver sebezhetőségei ellen, miközben rávilágítunk e módszerek hiányosságaira. Az ezt követő szakasz azokat a lehetőségeket veszi számba, melyek a fejlesztők rendelkezésére állnak, hogy elkerüljék vagy időben megtalálják a veszélyt okozó hibákat. Végül kitekintést teszünk a jövő felé és bemutatunk néhány, a területet érintő reményteli kutatási irányzatot.

2. Mi a szoftverbiztonság?

A szoftverbiztonság szó alatt az irodalom két jól elkülöníthető területet is ért. A két területet egymástól elválasztó kérdés az, hogy kit védünk, illetve kit tekintünk támadónak. Az egyik lehetőség, amikor magát a szoftvert, pontosabban annak fejlesztőjét védjük a szoftvere illegális másolásától, visszafejtésétől (reverse engineering) vagy rosszindulatú módosításától. Ebben a modellben magát a szoftvert tekintjük „ártatlannak”, amely nem bízik az őt futtató, potenciálisan rosszindulatú hosztban. Ez a terület elsősorban a szellemi tulajdonjogok védelméről szól. A továbbiakban ezzel a területtel nem is foglakozunk. Azoknak, akik érdeklődnek a téma iránt Collberg áttekintő cikkét [1] ajánljuk.

A másik terület, amivel ez a cikk is foglalkozik, nem a szoftver védelmére, hanem az azt futtató hoszt illetve a felhasználó védelmére összpontosít. Vagyis a modell itt a szoftvert, a programkódot tarja megbízhatatlannak a hoszt vagy a felhasználó szempontjából. Milyen veszélyeknek van kitéve ebben a modellben a felhasználó?

Az egyik veszélytípust az úgy nevezett rosszindulatú kódok (malicious code) alkotják, melyek szándékosan valamilyen nem megengedett műveletet szeretnének a hoszton végrehajtani. Ilyenek a mindenki által jól ismert vírusok, férgek, trójai programok, kémszoftverek, logikai bombák és így tovább. A másik veszélyforrás, amely az imént említett rosszindulatú kódok túlnyomó többségének létezését is lehetővé teszi, az a számítógépen futtatott operációs rendszerben és alkalmazásokban lévő, általában nem szándékosan elkövetett hibákból adódó sebezhetőségek jelenléte. A továbbiakban szoftverbiztonság alatt ezt az utólag felvázolt területet értjük, vagyis a nem megbízható kód modellt tekintjük relevánsnak.

Továbbá nem összekeverendő a szoftverbiztonság fogalma azokkal a biztonsági szoftverekkel, amelyek éppen valamilyen biztonsági funkciót valósítanak meg (például rejtjelezés, naplózás, hozzáférés-védelem). Ide sorolhatók még például a tűzfal vagy vírusirtó programok. Hogy érzékeltessük a két terület közötti különbséget, jogosan tehetnénk fel a kérdést, hogy egyáltalán egy behatolásdetektáló rendszer, vagy egy ví-

rusirtó telepítése növeli-e, vagy – a benne potenciálisan megbújó biztonsági lyukak révén – éppen csökkenti egy rendszer biztonságát [2].

Tulajdonképpen mit is jelent az, hogy biztonságos egy szoftver? Ha egy mondatban szeretnénk megfogalmazni, akkor azt mondhatnánk, hogy az a szoftver biztonságos, ami azt teszi, amit elvárunk tőle, hogy tegyen, és – ami ugyanolyan fontos – semmi egyebet. A programfejlesztés mai technikái mellett ez sajnos nem tűnik megvalósíthatónak. A szoftverekben olyan hibák maradnak, amelyek sérülékennyé, sebezhetővé és támadhatóvá teszik az azt futtató rendszert.

3. Mekkora a probléma?

Nap mint nap olvashatunk híreket számítógépes betörésekről, a levélszemétről (spam), botnetekről, vírusokról és férgekről (worm). Ezen problémák mérhetetlen károkat, dollárbilliókban mérhető veszteségeket okoznak évente. Ha jobban a dolgok mögé nézünk, akkor kiderül, hogy az illetéktelen számítógépes behatolások valamilyen szoftversebezhetőség kihasználásán keresztül történnek meg. Az internetes férgek gyors terjedését szintén programozói hibák, illetve az azok által keltett szoftverbiztonsági sebezhetőségek teszik lehetővé.

A hibát kihasználva a férgek egy rejtett hátsó ajtó program (rootkit) telepítésével zombi gépekké változtatják az internetre csatlakozó számítógépeket, melyek botnet hálózatot alakítanak ki, amit pedig tömeges spamküldésre, valamint összehangolt szolgáltatásmegtagadásos támadásokra használnak. Az asztali operációs rendszerek többsége olyan kémprogramokkal fertőzött, amelyek bizalmas információkat küldenek annak felhasználójáról. Ezek a programok az esetek túlnyomó többségében úgy települnek fel, hogy a felhasználó meglátogat egy rosszindulatú weboldalt, amely a böngészőben vagy annak valamely pluginjében lévő szoftverhibát kihasználva tetszőleges kódot tud lefuttatni a számítógépen. Látható tehát, hogy az infokommunikációs rendszerek legnagyobb problémáit és kockázatait alapvetően a rossz minőségű szoftver okozza.

A fenyegetettség nagysága ráadásul folyamatosan növekszik. A növekvő összekötöttség, az Internetre csatlakozó eszközök és szolgáltatások egyre növekvő száma (gondoljunk csak a mobiltelefonokra vagy a népszerű webes szolgáltatásokra) a támadási lehetőségek számát is növeli. Ezen kívül a szoftverek komplexitása is növekszik. A Windows XP operációs rendszer 40, míg a Windows Server 2003 már 50 millió sornyi forráskódból állt. Minél több sor kód, annál több programozói hiba, és minél több programozói hiba, annál több potenciális biztonsági sebezhetőség kerül a végtermékekbe. Az *1. ábrán* statisztika látható a 2002 és 2006 között talált és publikált szoftveres sebezhetőségek számáról, a NIST [3] sebezhetőségi adatbázisa alapján.

Érdemes megfigyelni az exponenciálisan növekvő tendenciát a 2003-as évtől kezdődően. Ma, hogyha fény derül egy biztonsági hibára, akkor az mindaddig támadhatóvá teszi az érintett rendszereket, amíg azt a hibát ki nem javítják, be nem foltozzák. Ha figyelembe vesszük ezt az időrést és a fenti statisztikát a talált hibák számát illetően, a mai Internetre csatlakozó rendszereinkről nyugodtan kijelenthetjük, hogy állandó veszélynek vannak kitéve.

4. Néhány tipikus hiba és sebezhetőség

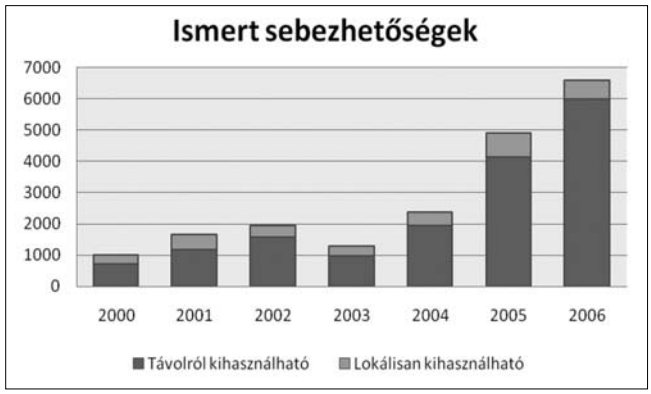
Biztonsági hiba nagyon sok helyen kerülhet a rendszerbe: már rögtön a követelmények meghatározásánál, vagy a rendszer tervezésénél, az implementálás során, de akár még a használat, illetve működtetés közben is okozhat egy nem megfelelő konfiguráció vagy környezet biztonsági hiányosságokat. Ebben a szakaszban bemutatásra kerül néhány, a legnagyobb számban előforduló biztonsági szempontból veszélyes, az implementáció során elkövetett programozási hiba.

Puffertúlcsordulás – az öreg hiba

A biztonsági szempontból veszélyes programozási hibák közül a legrégebb óta jelenlévők, nagyon sűrűn elkövetettek és legveszélyesebbek azok, amelyek puffertúlcsorduláshoz vezethetnek. A puffertúlcsordulás akkor történhet meg, amikor a szoftver egy fix hosszúságú tömböt (puffert) lefoglal a memóriában és a tömb írásakor nem ellenőrzi annak határait. Ilyenkor a támadónak lehetősége nyílik arra, hogy egy lefoglalt tömböt túlírva (tipikusan valamilyen túlzottan hosszú bemenet segítségével) felülírjon a program működése szempontjából fontos adatokat a memóriában. Ezek a hibák elsősorban a C/C++ programozási nyelv sajátosságai miatt fordulnak elő.

A legnagyobb veszély akkor jelentkezik, ha a szóban forgó fix hosszúságú tömböt lokális változóként definiálják, ugyanis ilyenkor a tömb a vermen (stack) tárolódik, amiből következően a tömb határán túlírva lehetőség nyílik a függvény visszatérési címének felülírására és ezáltal az eredeti futási útvonal eltérítésére. Vegyük szemügyre például az *2. ábrán* látható hibásan megírt programot.

1. ábra
Talált sebezhetőségek száma 2000 és 2006 között



```
void bad_func(char *userinput)
{
  int i=1;
  int j=2;
  int k=3;
  char buffer[100];

  strcpy(buffer, userinput);
  printf("%s", buffer);
}

int main(int argc, char *argv[])
{
  bad_func(argv[1]);
  return 0;
};
```

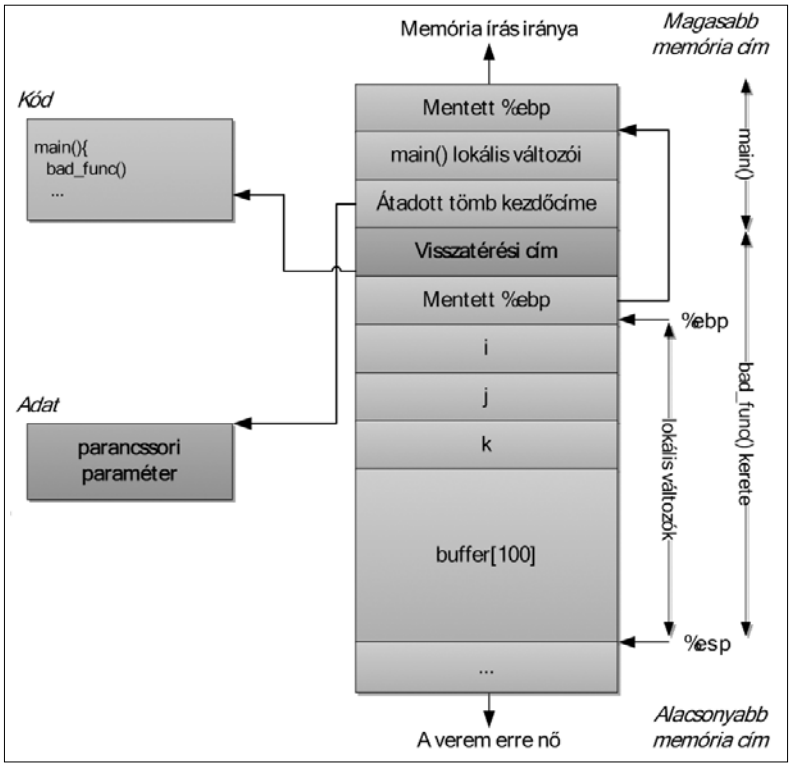
2. ábra Hibás C forráskód

A program az első argumentumaként kapott karaktersorozat kezdőcímét átadja a `bad_func` nevű függvénynek. Ezután a `strcpy()` könyvtári függvény a megadott címen lévő karakterláncot a lokálisan deklarált `buffer` nevű tömbbe másolja. A függvény meghívásakor (az x86 architektúrát feltételezve) a verem állapota a *3. ábrán* látható.

Amikor a megadott karakterláncot a `buffer` változóba másolja a `strcpy()` függvény, a lefoglalt hely kezdőértékétől addig írja a verem tartalmát, amíg a bemeneti karakterlánc végét jelző `0` értékű bájtot el nem éri. Tehát ha a bemenetnek egy 100 hosszú karakterláncnál hosszabbat adunk meg, akkor a másoláskor a vermen feljebb lévő elemek is átíródnak. Láthatjuk, hogy a visszatérési cím is felülírható, tehát a támadó tetszőleges kódsorozatra képes átirányítani a program futását.

A túlcsordulás legegyszerűbb kihasználási módja az, ha egyből magában a bemenetben olyan futtatható kódsorozatot helyez el a támadó, amely például egy hálózati porton keresztül hozzáférhetővé teszi a parancssort, és a visszatérési címet úgy írja felül, hogy az erre a kódsorozatra mutasson. Így a függvényvisszatéréskor a támadó kódja lefut, ezáltal csatlakozni tud a géphez és átveheti az irányítást. Ha nem lokális változóként deklarálunk egy puffert, hanem dinamikusan foglaljuk, akkor nem a stacken, hanem a heapen allokálódik számára hely. Ez hasonlóképpen túlírható és elérhető, hogy tetszőleges memóriacímet tetszőleges értékre módosítsunk, ami a fent említett támadásokra, azaz tetszőleges kódsorozat futtatására biztosít lehetőséget [4].

3. ábra A verem állapota



Egész számokkal kapcsolatos hibák

Az egész számokkal kapcsolatos hibák alapvetően a számítógépek számábrázolásának korlátai miatt jelentkeznek, illetve a nem megfelelő hiba- vagy kivételkezelés miatt. Ezek a hibák általában nem okoznak önmagukban sebezhetőséget, de nagyon gyakran miattuk jönnek létre puffer túlcsordulásra lehetőséget adó sebezhetőségek [5].

a) Aritmetikai túlcsordulás

Az aritmetika túlcsordulás (integer overflow) akkor következik be, amikor egy egész számot nagyobbra növelünk (például egy összeadás vagy szorzás művelettel), mint amekkora maximális értéket tárolni tud a számábrázolás. Ha például felhasználjuk ezt a számot egy memóriafoglalásnál, elképzelhető, hogy a szám túlcsordulása miatt túl kevés memóriát foglalunk, és ezzel egy puffer túlcsordulásos sebezhetőséget hozunk létre a heap-en.

b) Előjelezési hiba

A legtöbb programozási nyelvben, ha a programozó definiál egy egész számot, akkor, ha csak explicite nem definiálja előjel nélkülinek, az egy előjeles szám lesz. Később, ha ezt az értéket átadja egy függvénynek, amely egy előjel nélküli számot vár paraméteréül, akkor a számot implicite előjel nélkülivé konvertálja a fordító (casting), és a továbbiakban úgy is értelmezi. Ez azért jelenthet problémát, mert egy negatív szám, még előjelesként értelmezve például átmegy egy puffertúlcsordulás kivédésére beszúrt maximális hosszt vizsgáló feltételen, majd az ezt követő másolást végrehajtó függvény paramétereként már előjel nélküliként, egy nagy számmá válik, és így ismét egy puffer túlcsordulásos sebezhetőséget idéz elő.

c) Eltérő bitszélesség

Előfordulhat, hogy egy nagyobb méretű változót (például egy 32 bites integert) szeretnénk kisebb területen eltárolni (például egy 16 bites short változó helyén), amely nem képes azt befogadni, ezért az érték csonkolódik. Természetesen egy csonkolódott változóval való memóriafoglalás vagy paraméterellenőrzés is okozhat sebezhetőséget.

A printf() formátumleíró helytelen használata

A szabványos C könyvtár közkedvelt kiíró függvénye a `printf()`, illetve annak változatai. Ezek előnyös, jól használható szolgáltatása, hogy egy formátumleíró sztring megadásával egyszerűen leírható, hogy a megadott, különböző típusú paraméterek, a megjelenített szövegben hol és milyen alakban jelenjenek meg.

Abban az esetben azonban, ha a formátumleíró kívülről módosítható, az támadásra ad lehetőséget [6]. Egy ilyen tipikus hiba látható a *4. ábrán*.

```
int main(int argc, char *argv[])
{
  printf(argv[1]);
  return 0;
};
```

4. ábra printf() hibás használata

A parancssori paraméterben vezérlő karaktereket elhelyezve, a támadó olyan „hibás működést" képes előidézni, amely révén információkat tud kiolvasni a program memóriájából, manipulálni képes memóriacímek tartalmát és akár át is tudja venni a vezérlést a támadott gép felett.

Például a fenti programot a „`%X%X%X`" paraméterrel meghíva, a program kiírja hexadecimális számrendszerben a vermen tárolt értékeket (amelyek között titkosnak minősülő adatok is lehetnek). A „`%s%s%s`" paraméterrel pedig mutatóként értelmezi a stack-en található értékeket, így nem csak a veremből, hanem e pointerek által mutatott memóriatartományból is egyszerűen kiírattathatunk információkat, melyek szintén lehetnek bizalmas jellegűek (egy rejtjelkulcs vagy jelszó).

A vezérlő karakterek között azonban a „`%n`" a legérdekesebb, mert ez nem csupán a megjelenést befolyásolja, hanem memóriapozíciók felülírására is képes. Ennek a vezérlő karakternek a funkciója, hogy a paraméterként megadott pointer által mutatott memóriapozícióra kiírja, hogy az adott `printf()` végrehajtása során eddig hány karakter jelent meg a képernyőn. Tekintve, hogy megfelelő sztring megválasztásával a képernyőn megjelenített karakterek számát könnyen befolyásolni lehet, gyakorlatilag megoldható, hogy a „`%n`" tetszőleges értéket írjon be a megcímzett memória rekeszbe. Tehát e hiba kihasználásával szintén, ahogyan azt már stacken, illetve heap-en történő túlcsordulásoknál is láttuk, a támadónak tetszőleges kód futtatására van lehetősége.

Web-es típushibák

Napjainkban egyre nagyobb hangsúlyt kapnak a webes szolgáltatások, ezért az e rendszerekben előforduló egy-két típushibáról is említést teszünk. Ahogyan az előző, C/C++ programoknál előforduló hibák esetében is, itt is egyrészt a nem megfelelő bemenetellenőrzés, valamint a mögöttes rendszer felépítése, tulajdonságai hibáztathatóak sebezhetőségekért.

a) Parancs befecskendezés

Tegyük fel, hogy egy webszerveren futó CGI alkalmazás egy űrlapban megadható e-mail címet felhasználva, a következő utasítást hajtja végre: „`cat somefile | mail emailaddress`", ahol az `emailaddress` a felhasználó által megadott paraméter. A támadó ilyenkor, ha nincs megfelelő paraméter ellenőrzés, az e-mail címként a „`eve@attacker.com | rm -fr /`" karakterláncot megadva, például képes lehet letörölni a web szerver minden adatát.

b) SQL befecskendezés

Olyan rendszereknél, ahol a háttérben egy SQL adatbázis működik, a felhasználó által megadott adatok sokszor egy SQL parancsba vagy lekérdezésbe ágyazódnak bele. Ilyenkor, ha a támadó SQL parancs elemeket illeszt az általa megadott adatokba, az eredeti parancs értelmét meg tudja változtatni, tipikusan például egy adatbázisból történő jelszóellenőrzést meg tud kerülni.

c) Cross-site scripting (XSS)

A Cross Site Scripting sebezhetőség akkor jöhet elő, ha például egy webes alkalmazás fejlesztője egy űrlapba írható adatokat nem ellenőrzi, majd később a bevitt adatokat megjeleníti. Ilyenkor a támadó általában egy JavaScript kódot helyezve az űrlapba, majd az eredményt megjelenítő URL-t elküldve áldozatának, lefuttathatja saját kódját, amely már az áldozat jogosultságaival rendelkezik.

5. Hagyományos védekezési módszerek

A hagyományos védekezési módszerek a szoftverekben rejlő potenciális sebezhetőségek ellen védekeznek valamilyen módon magán a hoszton, vagy a belső hálózat határvonalán.

Hozzáférés védelem

Az operációs rendszerek (OS) egyik fő feladata a számítógépes biztonság egyik alapelvének betartatása, miszerint minden modul (felhasználó, processz) csak azokhoz az erőforrásokhoz férjen hozzá, amire feltétlenül szüksége van (least privilege principle). Az OS által biztosított hozzáférés védelmi mechanizmusokkal határt szabhatunk az egyes támadások lehetőségeinek, de eliminálni őket nem tudjuk. Szigorúbb hozzáférésvédelmi politikát valósíthatunk meg továbbá olyan megoldásokkal, mint például a SELinux [7].

Memóriavédelem

A puffertúlcsordulásnál láttunk, hogy a támadó általában a vermen vagy a heapen futtatja az általa a memóriába juttatott kódot. Ezeket a memóriaterületeket „nem futtatható" területekként kell megjelölni. Ezt biztosítja például Windows rendszerekben a Data Execution Prevention (DEP), vagy Linuxon a PaX vagy az Exec Shield. Ezek a megoldások természetesen nem védenek minden támadás ellen, például a „Return-to-libc" [8] támadással megkerülhetőek.

Nagy segítség a támadó számára a kihasználáskor, hogy a virtuális memóriában, azonos architektúrán a

memória címek állandóak. Így tudhatja, hogy mit mire kell átírni, és hogy bizonyos „hasznos” függvények hol találhatók.

Egy másik lehetőség, hogy megnehezítsük ilyenkor a támadó dolgát, ha ezt a memória elrendezést véletlenszerűvé tesszük úgy, hogy mindig változtatunk a kódszegmens, a programkönyvtárak, a stack és a heap báziscímén. Ezt a technikát ASLR-nek (Address Space Layout Randomization) hívjuk. Természetesen ezt a védelmet is ki lehet játszani, de a sikeres támadások számát csökkenteni képes.

Védekezés ismert rosszindulatú kódok ellen

A vírusirtók és behatolás detektáló rendszerek olyan rosszindulatú programok és támadások ellen képesek védeni elsősorban, melyek a múltból már ismertek. Ahogyan a 2. szakaszban arra már utaltunk, ezek a kiegészítő szoftverek ugyanúgy növelik a rendszer komplexitását, mint bármilyen más alkalmazás, így a sebezhetőség potenciált is. A veszélyt fokozza, hogy különösen az antivírus és IDS szoftverek mélyen beépülnek az operációs rendszerekbe, ezért egy esetleges biztonsági lyuk teljes körű hozzáférést képes biztosítani egy potenciális támadó számára.

Hálózati határvédelem

A hálózati rétegben is védekezhetünk a támadások ellen. Tűzfalak segítségével kikényszeríthetjük a hálózathozzáférési politikánkat. Ez gyakorlatilag ugyanazt jelenti, mint az operációs rendszer hozzáférésvédelme, csak a hálózati erőforrásokról van szó. Segítségükkel leszűkíthetjük a támadási felületet, de természetesen az elérhetően maradt szolgáltatásokban lévő sebezhetőségek ellen nem véd.

Foltozás

Mivel nem hibamentesen kerülnek ki a szoftverek a fejlesztőktől, egy hiba felbukkanása után azt utólag kell kijavítani. Nagyon fontos, hogy ezek a hibajavítások minél gyorsabban jussanak el a felhasználókhoz, és fel is települjenek. Ahogy azt már említettük, a reakcióidő miatt az idő nagy részében a sebezhetőségek kihasználásra várnak.

Látható, hogy az eddig felsorolt védelmi mechanizmusok mind a számítógépre telepített szoftverekben már meglévő sérülékenységek *kihasználását* nehezítik, vagy a támadási felületet szűkítik. Ezek a technikák képesek a kockázatok bizonyos mértékű enyhítésére, azonban mivel ezek természetüket tekintve reaktív jellegűek, nem előzik meg a sebezhető pontok kialakulását.

6. Megelőzés a fejlesztés során

A sebezhetőségek kialakulásának elkerülése, megakadályozása, vagy még időben való detektálása a szoftverfejlesztők feladata. Effajta preventív technikák alkalmazására a szoftverfejlesztés életciklusának minden állomásán szükség van.

Védekezés a szoftverfejlesztés folyamán

Biztonsági szempontokat is figyelembe vevő szoftverfejlesztésnél már a követelmények meghatározásánál számolni kell a lehetséges fenyegetettségekkel. Praktikus gyakorlat a használati esetek (use case) mellé a potenciális visszaélési eseteket is meggondolni és felsorolni. Az architektúrális és részletes tervezés folyamataiba is kockázatelemzés bevonása szükséges, számba véve a lehetséges hibákat, sebezhetőségeket. Az implementáció folyamán érdemes a manuális kódszemlézést automatikus statikus kódelemző eszközökkel segíteni. A tesztelés fázisában pedig egyrészt a biztonsági funkciókat a hagyományos tesztelési módszerekkel kell ellenőrizni, majd a teljes rendszeren egy veszélyalapú biztonsági tesztelést kell végrehajtani.

Típusbiztos programozási nyelvek használata

A használt programozási nyelv nagymértékben befolyásolja egy szoftver támadhatóságát. A mai operációs rendszerek, eszközmeghajtók és rengeteg felhasználói szoftver is C illetve C++ nyelven íródik. Ezek a nyelvek túl sok szabadságot adnak a programozónak, hogy elég biztonságosak lehessenek. Más programozási nyelvek (pl. Java, C#, Python) szigorú típusossággal (típus-biztonság), mutatók kiküszöbölésével és egyéb biztonsági megfontolásból bevezetett szabályokkal és megkötésekkel eleve kiküszöbölnek olyan tipikus hibákat, mint például a C programokban sokszor előforduló puffer túlcsordulás. Természetesen vannak területek, ahol a C/C++ használata elkerülhetetlen, például teljesítménykritikus szoftvereknél. Megjegyezzük, hogy léteznek, még ha csak kutatási célból is, olyan C nyelvre alapuló módosított nyelvek illetve fordítók is, amelyeket úgy terveztek, hogy ne lehessen elkövetni használatuk során a legtipikusabb hibákat (pl. Cyclone [9] vagy Ccured [10]).

Statikus kódelemzők

Az elmúlt években a statikus kódelemzők használata nagymértékben elterjedtté vált a manuális kódszemlézés kiegészítőjeként, illetve automatizálásaként. Ezen kereskedelmi forgalomban kapható programanalizáló eszközök használata ma már sok szoftverfejlesztő cég fejlesztési folyamatának része. Több ilyen automatikus statikus analízist végrehajtó szoftver is létezik, amely képes kimutatni bizonyos tipikus biztonsági szempontból veszélyes programozói hibát, több száz hibadefiníció illetve szabály alapján, a forrást elemezve [11]. Ilyenek például a FindBugs, a Coverity vagy a Fortify Source Code Analyze. E szoftvereknek az előnye az, hogy elvben teljes kód-lefedettséget tudnak biztosítani (a gyakorlatban ez nem mindig kivitelezhető). Hátrányuk pedig, hogy általában túl nagy a hibás riasztások (false positive) aránya. Ez nagyban meg tudja nehezíteni használójuk munkáját.

Dinamikus feketedoboz-alapú biztonsági tesztelés

A Washingtoni Egyetem kutatói 1990-ben úgy teszteltek szabványos UNIX alkalmazásokat, hogy hosszú, véletlenül generált inputot küldtek a programok beme-

netére [12]. A tesztelt programoknak körülbelül 30%-a elszállt, vagy kiakadt. A módszert „fuzzing”-nak nevezték el. Hogy a technika a biztonsági tesztelésre még alkalmasabbá váljon, párhuzamosan többen is, két szempontból fejlesztették azt tovább.

Ez egyik oldala a fejlődésnek, hogy figyelembe vesszük a szoftver által elvárt bemeneti struktúráját is (pl. egy Word dokumentum) és egy struktúra leírás alapján generálunk véletlen módon helytelen, de strukturálisan helyes bemenet. Ezzel nagyban növelhető a kódlefedettség.

A másik, hogy a tipikus hibákra általában meghatározhatók olyan bemeneti minták, melyek az adott típushibát relatíve nagy valószínűséggel felszínre képesek hozni. Ilyen minták például a puffer túlcsordulásnál a nagyon hosszú, vagy éppen üres bemenet, vagy a lezáró `0x00` karakter hiánya. Az egészekkel kapcsolatos hibákat, a nulla, a negatív, a nagyon kicsi, illetve nagyon nagy értékek vagy a 2 hatványai idézhetik elő könnyen.

Folytathatnánk a sort a `printf()` hibával („`%s%s%s`” vagy „`%n%n%n`”), SQL injection-nel (`' OR username IS NOT NULL OR username = '`, vagy `1' OR '1'='1`, `...`) és így tovább. Ha a véletlen tesztvektor generálás során olyan heurisztikákat alkalmazunk, melyek ilyen mintákat hoznak létre, még tovább növelhetjük a hibák megtalálásának valószínűségét.

Tehát ennél a biztonsági tesztelési módszernél egy olyan feketedoboz-alapú tesztelést hajtunk végre, amelynél (ellentétben a hagyományos teszteléssel) nem a funkciók specifikációi alapján határozzuk meg a teszteseteket, hanem a potenciális programozói hibák által kialakulható sebezhetőségek alapján. Ilyen fejlett fuzzing eszközök a Peach [13] vagy a Flinder [14], amely olyan összetett protokoll-implementációkat is képes tesztelni, mint például az SSL könyvtárak [15].

Ezen eszközök előnye, hogy minden talált hiba valós hiba (csak true positive), ellentétben a statikus elemzőkkel, viszont az általuk elérhető kódlefedettség jóval kisebb. Említést kell tennünk arról is, hogy természetesen ezeket az eszközöket a támadók is használják a kész szoftvereken, ezért fontos, hogy még a fejlesztői oldalon megtörténjenek ezek a vizsgálatok.

7. Mit tartogat a jövő?

Ebben a szakaszban néhány reményteli kutatási irányzatot mutatunk be, melyek nagyban hozzájárulhatnak a jövőben a biztonságosabb szoftverfejlesztéshez és megbízhatóbb szoftverekhez.

A jövő megoldásai elsősorban a formális módszerek (tételbizonyítók, modellellenőrzők) fejlesztésére, használatuk megkönnyítésére, a szoftverfejlesztési folyamatba való teljes körű beépítésére alapulnak. Pár éve Tony Hoare meg is hirdette az „Informatika nagy kihívása” projektet [16], melynek végcélja egy olyan eszközkészlet megalkotása, mely teljes és automatikus programverifikációra képes.

Mi most két területet szeretnénk kiemelni: az egyik az automatikus kódalapú teszteset-generálás, a másik a programozási nyelv alapú biztonsági megoldások.

Automatikus szoftvertesztelés

A kódszemlézés automatizálása a statikus kódelemzők fejlődésének köszönhetően ma már viszonylag jól használható technológia. Ami viszont a következő évek egyik nagy kutatási feladata az a szoftvertesztelés – minél kiterjedtebb – automatizálásának megoldása.

A fuzzing roppant népszerű módszer lett az elmúlt években, hiszen viszonylag egyszerűen, gyorsan és olcsón lehetett vele akár nagyon komoly biztonsági hibákat is találni bármilyen szoftverben. Annak ellenére, hogy ez valóban hatásos módszer, nagyon komoly korlátai is vannak.

Képzeljük csak el, hogy egy kihasználható programozó hiba például egy `if (x==12)` utasítás igaz ágán helyezkedik el, ahol `x` az egyik bemeneti változó. Annak a valószínűsége, hogy egyáltalán egy teszteset során futni fog a hibás kódrészlet, egy 32 bites `x` változó esetén $1/2^{32}$, hiszen az x bemenetet véletlen módon generáljuk. Éppen ezért a fuzzing általában nagyon kicsiny kódlefedettséget biztosít.

A Microsoft Research kutatói olyan megközelítéssel álltak elő a probléma kezelésére, amit „whitebox fuzzing”-nak neveztek el. A megoldás a dinamikus tesztgenerálás és a szimbolikus futtatás [17] meglehetősen régi ötletére alapszik. Egy véletlenszerűen választott kezdeti bementettel szimbolikusan futtatják a vizsgált programot, miközben bemeneti kényszereket gyűjtenek az érintett feltételes elágazások alapján. Az összegyűjtött kényszereket a bemeneti adatokra vonatkozóan azután szisztematikusan negálják, majd kényszermegoldó eljárásokkal (constraint solver) újabb bemeneteket, teszteseteket generálnak, melyek már a program más részeit hozzák működésbe.

Például az előbbi példát tekintve, a szimbolikus futtatás során az `if (x==12)` elágazásnál, egy `x=0` kezdeti változó az $x \neq 12$ kényszert hozza létre. Ha ezt a kényszert negáljuk és megoldjuk, akkor az `x=12` értéket kapjuk, mint következő tesztesetet, ami már lefedi az elágazás igaz ágát, ahol a feltételezett hibánk el van rejtve [18].

Nyelvalapú biztonság

Ahhoz, hogy a jövőben eleve kizárjunk bizonyos biztonsági szempontból veszélyes programozási hibákat, sokkal mélyebb szinten, alapjaiban kellene megváltoztatni a programozási nyelveket, fordítóprogramokat, valamint a futtatókörnyezeteket. Az ilyen típusú megoldások területe az úgynevezett nyelvalapú biztonság (language-based security), melyet a terület egyik éllovasa Schneider [19] a következőképpen definiált, meglehetősen tág értelmezésben: „azon módszerek halmaza, melyek a programozási nyelvek elméletére és implementációjára alapozva, beleértve ide a szemantikákat, típusokat és az optimalizálást, a biztonság kérdésére próbálnak megoldást nyújtani”.

Az egyik ötlet amit „Proof-Carrying Code”-nak [20] nevez az irodalom az, hogy a fordító generáljon formális bizonyítást arra vonatkozóan, hogy a lefordított program megfelel a különböző biztonsági feltételeknek, majd ezt a bizonyítást mellékelje a lefordított állományhoz. A programot futtató hoszt ezt a bizonyítást ellenőrizheti a futtatás előtt, hogy megbizonyosodjon róla, hogy a szoftver megfelel a követelményeknek.

Ide tartozik még egy másik javaslat is, amely a „Typed Assembly Language” [21] nevet kapta. Ez egy olyan kibővített assembly nyelv, amely a változók típusinformációit is magában képes foglalni. Így futtatás előtt meg lehet bizonyosodni a lefordított állomány típusbiztonságáról anélkül, hogy olyan köztes bájtkód-reprezentációkat használnánk, mint amilyet a Java vagy .NET platformok.

8. Összefoglalás

Megállapítottuk, hogy az informatikai rendszerek biztonságának kérdése a legnagyobb részben valójában szoftverminőségi kérdés. Amíg a szoftverek minőségében nem történik áttörő fejlődés, addig a szoftverbiztonság, így az egész IT biztonság terén sem fogunk lényeges javulást tapasztalni. Az is egyértelművé vált, hogy nem létezik egy minden problémára orvosságot nyújtó megoldás. Egyszerre kell a szoftverfejlesztési metodológiákat, a programozási nyelveket, fordítókat és operációs rendszereket, futtatókörnyezeteket alapjaiban megváltoztatni.

A formális módszerek fejlődése sokat ígér, de ne feledjük, hogy programozói hibák mindig voltak, vannak és lesznek is, így várhatóan az azokból adódó sebezhetőségek sem fognak teljes mértékben megszűnni.

A szerzőkről

SZEKERES LÁSZLÓ 1983-ban született Szegeden. 2007-ben diplomázott mérnök informatikusként a BME Méréstechnika és Információs Rendszerek Tanszékén, Infokommunikációs Rendszerek Biztonsága Szakirányon. Jelenleg biztonsági kutató-fejlesztőként dolgozik a SEARCH-LAB Biztonsági Értékelő Elemző és Kutató Laboratóriumban. Érdeklődési területe a számítógépes biztonságon belül a szoftverbiztonság, biztonsági tesztelés, biztonságos programozási nyelvek és típuselmélet. CISA és CISSP vizsgákkal rendelkezik.

TÓTH GERGELY (CISA) a SEARCH-LAB Kft. értékelő és tesztelő csoportjának vezetője. A Budapesti Műszaki és Gazdasátudományi Egyetemen végzett mérnök-informatikusként és 10 éve foglalkozik IT biztonsággal. Számos magyar és EU K+F projektben vett részt és több mint 20 ipari biztonsági értékelési projektet vezetett. Emellett a SEARCH-LAB által fejlesztet automatikus biztonsági tesztelő keretrendszer, a Flinder vezető fejlesztője.

Irodalom

[1] C. Collberg, C. Thomborson, „Watermarking, tamper-proofing and obfuscation – tools for software protection”, IEEE Transactions on Software Engineering, Vol. 28, 2002, pp.735–746.

[2] R. Giobbi, „Avast! antivirus buffer overflow vulnerability”, US-CERT 2007, http://www.kb.cert.org/vuls/id/125868

[3] „National Vulnerability Database”, http://nvd.nist.gov/

[4] Matt Conover and w00w00 Security Team, „w00w00 on Heap Overflows”, 1999.

[5] Blexim: „Basic Integer Overflows,” Phrack, Vol. 11, 2002.

[6] A. Thuemmel: „Analysis of format string bugs,” Manuscript, 2001.

[7] B. McCarty: SELinux, O&Reilly, 2004.

[8] J. Pincus, B. Baker, „Beyond Stack Smashing: Recent Advances in Exploiting Buffer Overruns,” IEEE Security & Privacy, 2004, pp.20–27.

[9] T. Jim et al., „Cyclone: A safe dialect of C”, USENIX Annual Technical Conference, 2002, pp.275–288.

[10] G.C. Necula et al., „CCured: type-safe retrofitting of legacy software,” ACM Transactions on Programming Languages and Systems (TOPLAS), Vol. 27, 2005, pp.477–526.

[11] „Static code analysis,” Software, IEEE, Vol. 23, 2006, pp.58–61.

[12] B.P. Miller, L. Fredriksen, B. So, „An empirical study of the reliability of UNIX utilities,” Communications of the ACM, Vol. 33, 1990, pp.32–44.

[13] M. Eddington: „Peach fuzzer framework”, http://peachfuzzer.com/

[14] SEARCH-Lab Ltd.: „Flinder”, http://www.flinder.hu/

[15] L. Szekeres, „Biztonsági szempontból veszélyes programozói hibák felderítése automatizált módszerekkel”, Tudományos Diákköri Konferencia, BME, 2006.

[16] C. Hoare, „The Verifying Compiler: A Grand Challenge for Computing Research”, Modular Programming Languages, 2003, pp.25–35. http://www.springerlink.com/content/t96b79tanjm9d4tc

[17] J.C. King, „Symbolic execution and program testing”, Commun. ACM, Vol. 19, 1976, pp.385–394.

[18] P. Godefroid et al., „Automating software testing using program analysis” Software, IEEE, Vol. 25, 2008, pp.30–37.

[19] F.B. Schneider, G. Morrisett, R. Harper, „A Language-Based Approach to Security,” Lecture Notes in Computer Science, Vol. 2000, 2001, pp.86–101.

[20] G.C. Necula, „Proof-carrying code,” Proc. of the 24th ACM SIGPLAN-SIGACT symposium on Principles of programming languages, Paris, France, ACM, 1997, pp.106–119. http://portal.acm.org/citation.cfm?doid=263699.263712

[21] G. Morrisett et al., „TALx86: A realistic typed assembly language,” In Second Workshop on Compiler Support for System Software, 1999, pp.25–35.

Bevezetés a botnetek világába

SZENTGYÖRGYI ATTILA, SZABÓ GÉZA

Budapesti Műszaki és Gazdaságtudományi Egyetem, Távközlési és Médiainformatikai Tanszék
{szgyi, szabog}@tmit.bme.hu

BENCSÁTH BOLDIZSÁR

Budapesti Műszaki és Gazdaságtudományi Egyetem, Híradástechnikai Tanszék
boldi@crysys.hit.bme.hu

Kulcsszavak: robothálózat, botnet, detektálás, darknet, honeypot

A botnet (robot network) nem más, mint számítógépek támadó által vezérelt serege. A gépek nem a támadó tulajdonát képezik, hanem többnyire rosszindulatú kóddal fertőzött otthoni számítógépek. A botnetek napjaink egyik legelterjedtebb és legveszélyesebb károkozói. Sok évvel megjelenésük után is az átlagfelhasználó keveset tud róluk, a működésükről és a védekezési módszereikről, pedig pont az ő gépeiket használja fel leggyakrabban a támadó a saját céljaira. Cikkünk célja, hogy közelebb hozzuk az olvasót ehhez a technikához: összefoglaljuk a botnetek működési elvét, kommunikációjuk és működésük lehetséges módozatait.

1. Bevezetés

Az utóbbi időkben az egyik legismertebb hálózati alkalmazásként emlegetik a robotok hálózatait, a botneteket. Sajnos ez a népszerűség nem a nagyszámú boldog felhasználó visszajelzésének köszönhető, hanem a botnetek által okozott károknak. Egyre-másra jelennek meg híradások arról, hogy egy-egy botnet-támadás milyen károkat okozott különböző szolgáltatóknak és cégeknek.

Az ilyen támadó hálózatok nemcsak a cégeket, megtámadott rendszereket, hanem az átlagembereket is megkárosítják, hiszen számítógépeik erőforrásait károkozására használják fel, miközben akár megszerezhetik a megtámadott gép gazdájának személyes adatait is, vagy akár kéretlen reklámleveleket is küldhetnek nekik. Mint látható, sok egymással összefüggő internetes támadást lehet kapcsolatba hozni a botnetek létezésével. A cikk célja, hogy összefoglalja a botnetekkel kapcsolatos információkat, hogy hatékonyan lehessen fellépni a károkozók ellen.

2. Mi a botnet?

A botnet szó a „roBOT NETwork” angol kifejezésből származik. Az első robotnak nevezett programok az Internet úgynevezett IRC (Internet Relay Chat), internetes beszélgetési szolgáltatásához kapcsolódóan jöttek létre, és olyan feladatokat láttak el, mint üzenetek átadása, üdvözlés, bizonyos jogosultságok biztosítása stb. Ezeket a távirányítható, illetve programozott robotokat kezdték el röviden „bot” néven említeni. Később jelentek meg a rosszindulatú céllal létrehozott „botok”, sokszor továbbra is az IRC hálózaton át koordinált szoftverek, amelyek összehangolt támadásokat tudtak indítani.

A „botok” előre programozott feladatot hajtanak végre, például kéretlen reklámleveleket (spam) generálnak és továbbítanak, vírusokat terjesztenek vagy DoS (Denial-of-Service – szolgáltatásmegtagadásos) támadásokat visznek véghez [8]. A botnet kifejezésben a „network” fogalom azért jelent meg, mert ezek a robotok hálózatba vannak szervezve: az egyes robotok vagy egymással, vagy kevés számú (tipikusan 1-2) vezérlővel (controller) állnak kapcsolatban. Magukat az értelem nélküli robotokat zombiknak, a hálózatot pedig zombi hadseregnek is hívják. A botnetek fontosságát számos kutató felismerte már, és több publikáció is született már a tárgykörben, így hasznos lehet a további irodalmak áttanulmányozása is [2,3,6,7].

3. Botnetek keletkezése

A botnetek az életük során hasonló funkcionális lépéseken mennek keresztül, ezeket életciklusoknak nevezhetjük. Ha értjük a botnetek életciklusát, akkor nagyobb eséllyel vesszük azokat észre, és jobban lehet reagálni a veszélyre.

Egy botnet létrejöttéhez szükség van olyan számítógépekre (áldozatokra), amelyek az adott robot kódját futtatják, ez az első lépés a botnet létrehozásában. A támadók általában úgy jutnak ilyen gépekhez, hogy valamilyen módszert kihasználva terjeszteni próbálják a zombi kódját, hasonlóan más rosszindulatú kódokhoz. Amint megfelelő mennyiségű számítógépen fut a rosszindulatú kód, a támadó elkezdheti az így kialakult hálózat koordinációját, vezérlését.

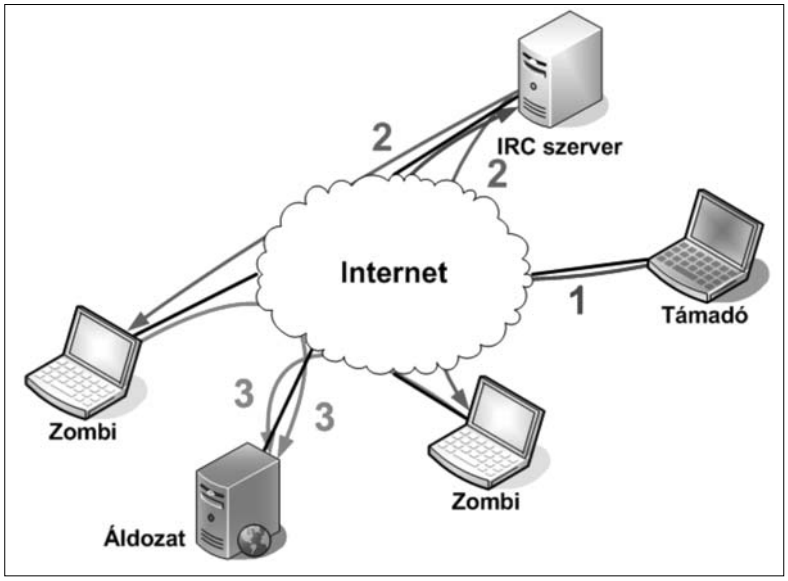
A vezérléshez speciális módszereket használhatnak fel, hogy a vezérlő kiléte titokban maradhasson, ám a botnet mégis koordinált módon működjön, megfelelően végezze a támadásokat. A botgazda parancsára a hálózat támadni kezd, hasonló módon a támadás – legyen az spam küldése vagy egy DoS támadás – rövid időn belül le is állítható.

A működő botnet egy dinamikus közeg: egyes elemeit, a felhasználók gépeit „megjavítják”, így letörlésre kerül a rosszindulatú kód és a botnetből ezek a gépek

kiesnek. Új gépek is beléphetnek ugyanakkor a hálózatba. A hálózat, annak mérete és elemei így folyamatos változásban vannak.

A botnetek megszűnésére kevés példát láttunk, kevés tény ismert. Gyakori, hogy a botnet gazdája mégis lelepleződik, jogi eljárás indul ellene, ilyenkor a botnet elérkezhet megszűnéséhez. Több dolog is történhet egy botnet megszűnése közelében. A tulajdonos átadhatja vezérlését egy másik gazdának, aki sajátjaként kezelheti a hálózatot, vagy akár beolvaszthatja saját hálózatába. Az is elképzelhető azonban, hogy a botnet gazdátlanná válik, senki sem irányítja és gondozza, így egy ideig működik, majd a vezérlési rendszer szétesik. Kevés tapasztalat van azonban ezekről a folyamatokról.

A botnethez szükséges rosszindulatú kód terjesztésére az egyik leggyakoribb módszer az e-maileken keresztül terjedés. A közismert „vírusos e-mail" jellegű terjedés mellett azonban néhány egzotikusabb módszert is felhasználnak a botnetek létrehozására, mint például a gépeken más rosszindulatú kód által hagyott kiskapuk megkeresését, vagy a nyers erő támadást jelszavak kitalálására.



1. ábra
Egy zombihálózat felépítése és a támadás folyamata

3.1. A trójaiak által hagyott kiskapuk

A botnetek egy része még speciálisabb módszert is felhasznál: más kártékony kódok után hagyott kiskapukat keres az Interneten. A botnet kliensek maguk is ebbe a kategóriába sorolhatók, hisz lehetővé teszik a távoli gép vezérlését, azonban ezen túlmenően is számos ilyen kártékony kód létezik, és egy részüknél a távvezérlés könnyen átvehető, mert az adott rosszindulatú kód például nem tartalmaz hitelesítést, bárki vezérelheti a kódot és így átveheti a gép irányítását.

3.2. Jelszó kitalálása és „nyers erő"-támadások

A botnet létrehozásához olyan egyszerű támadásokat is fel lehet használni, mint a jelszavak kitalálása, próbálgatása.

Példaként az RBot (és más bot-családok is) a jelszótalálgatás több fajtáját is használják. Az RBot terjedését manuálisan indították távirányítással. Az RBot a 139-es és 445-ös portokra próbál csatlakozni véletlenül kiválasztott célpontokról, egy kiválasztott név és jelszó segítségével. Ezeket a portokat a Windows rendszer használja a fájlmegosztás és más hálózati szolgáltatások során. Ha sikerül a csatlakozás, akkor megpróbálja kitalálni a megfelelő felhasználói nevet és jelszót. Ha sikertelen a csatlakozási kísérlet, vagy semelyik tesztelendő név és jelszó párosra nem reagál a célgép, akkor a bot feladja a célgép támadását és másik potenciális áldozatot keres. A botnak egy beépített listája van a tipikus felhasználói nevekről, amelyekkel csatlakozni próbál.

Több behatolás detektáló rendszer is képes a nagyszámú bejelentkezési kísérlet alapján a fertőzött gépeket azonosítani és kiszűrni. A belépési kísérleteket a megtámadott gép eseménynaplója is tartalmazhatja.

3.3. Botnet-kliensek gyülekezése

Gyülekezéskor a botnet-kliens a botnetet irányító központtal veszi fel valamilyen módon a kapcsolatot.

A legegyszerűbb botnetek az Internet egy speciális szolgáltatását, a szöveges beszélgetést biztosító IRC rendszert használják. Amikor egy újabb botnet-elem, zombi csatlakozik a rendszerhez, az hozzákapcsolódik az előre beprogramozott IRC szerverek egyikére, és ott fellép a megadott csatornára. Ez a csatorna egy elszigetelt beszélgetési terület, amely kiválóan alkalmas arra, hogy elszigetelt módon, a botokat vezérlő személy parancsokat küldjön mindazon botoknak, amelyek csatlakoztak a rendszerhez. A támadó ezt követően az IRC csatornán kiadott parancsokkal tudja utasítani a zombi hadseregét. Ez a folyamat látható az *1. ábrán*.

Először a támadó csatlakozik az IRC szerverhez és kiadja a támadás parancsot (1). A parancs így eljut az IRC szerveren levő parancsokat figyelő zombi gépekhez (2). A parancsot értelmezve a zombik koordináltan támadást indítanak (3).

A botnetek belső rendszere kifinomult biztonsági elemeket is tartalmazhat: az irányításhoz jelszó alapú azonosítást és rejtjelezett kommunikációt is használhatnak. Az új botnet-kliens frissítéseket is letölthet. Ezek a frissítések sebezhetőségi információkat, új irányítóközpont címeket, vagy akár újabb funkciókat is tartalmazhatnak.

A botnetkliens-program, illetve a támadó azt is figyelemmel kísérheti, hogy esetleg a fertőzött gépen más támadó is elhelyezett-e rosszindulatú kódot. Amennyiben igen, úgy speciális eljárásokkal eltávolíthatja, vagy deaktiválhatja mások programjait. Hasonlóan járhat el az antivírus-programok és más védelmi szoftverek esetében.

4. Botnet-típusok

A botneteket hálózati technikájuk szerint két fő csoportra oszthatjuk:

	Tervezés	Botnet detektálás	Késleltetés	Túlélés	Vezérlő detektálás
Centralizált	Könnyű (1)	Könnyű (1)	Kicsi (3)	Rossz (1)	Könnyű (1)
P2P	Nehéz (3)	Nehéz (3)	Közepes (2)	Kiváló (3)	Nehéz (3)

1. táblázat Botnet-típusok tulajdonságai

• **Centralizált botnetek:** Ezek (közel) csillagtopológiájú hálózatok. Kevés számú vezérlő (tipikusan 1-2) van a hálózatban és a botnet minden tagja ezzel a vezérlővel áll kapcsolatban. Ezt a típusú botnetet a legkönnyebb felismerni.

• **Peer-to-Peer (P2P) botnetek:** Ilyen esetben nincs központi vezérlő, a botnet elemei egymással kommunikálnak. A támadó a parancsot egy botnak adja ki, mely továbbítja ezt a többi bot felé. A P2P botnet megvalósítása nehezebb, hisz egy modernebb technológiát képvisel, amelyben még vannak nyitott kérdések. A hálózatban részt vevő gépek gyorsan változhatnak, az alkalmazott P2P eljárásnak tehát igen hatékonynak kell lennie. A másik oldalról nézve, ilyen esetben a támadás koordináló vezérlő azonosítása nehezebb, így a támadó védettebb helyzetben lehet.

Az *1. táblázat* foglalja össze, hogy az egyes botnetek milyen tulajdonságokkal rendelkeznek.

Számos különböző botnet-kliens és így számos különböző botnet létezik. Tevékenységükben és felépítésükben vannak különbségek, de ezek jelenleg tartalmi lényegükben csak kisebb mértékben térnek el. Az egyes botnetekről, kliensekről internetes adatbázisokból, levelezési listákból és speciális publikációkból lehet több információt szerezni (lásd pl. [7])

5. A botnetek tevékenysége

A botnetek számos különböző tevékenységet látnak és láthatnak el, ezek közül a főbb tevékenységek a következők:

• **Új botok beszervezése.**
A botnet méretének növelése érdekében újabb célpontokat szervezhet be a hálózatba.

• **Szolgáltatás-megtagadásos támadás.**
Hatalmas erőforrásait felhasználva felemésztheti a célpontok erőforrásait, megbénítva azokat.

• **Levélszemét terjesztése.**
Gépek tízezrei segítségével kéretlen reklámlevelek millióinak, sőt, százmillióinak kiküldésére van lehetőség, ami jelentős bevételi forrást jelenthet. A botnetek többek között a kéretlen reklámleveleknek köszönhetik térnyerésüket, mert ezen keresztül váltak igazán pénztermelő lehetőséggé.

• **Illegális tartalom tárolása.**
A botnet, mint egy zombihadsereg egy gyakorlatilag végtelen tárolókapacitással rendelkező háttértárat jelent a támadók számára, hogy illegális tartalmakat (lopott mozifilmeket, lemásolt játékokat, drága szoftvereket) tároljanak. A fájlokat a gép felhasználójától elrejtett helyeken is tárolhatják.

• **Adatgyűjtés.**
Mivel a botkliensek gépek ezrein futnak, könnyűszerrel megszerezhetik a gépeken futó szenzitív adatokat, neveket, jelszavakat, e-mail címeket stb., ahogy azt más spyware programok is megtehetik.

6. Botnetek felismerése

A botnetek működését megértve lehetőségünk nyílik azok detektálására is. A botnetek az elosztottság előnyét használják ki, hogy detektálásuk nehézkes legyen. Egy túlságosan elosztott rendszer azonban nem elég hatékony, így a botnet tulajdonosoknak is kompromisszumot kell kötni a detektálhatóság és a használhatóság terén. Ez a tulajdonság adja meg a lehetőséget a botnetek detektálására.

Botnetek detektálása két fő módszerrel történhet:

– *Felhasználók szintjén*, amikor a felhasználók gépére telepített kódot próbáljuk meg vírusirtó programok vagy behatolásfelismerő rendszerek (IDS, Intrusion Detection System) segítségével megtalálni.

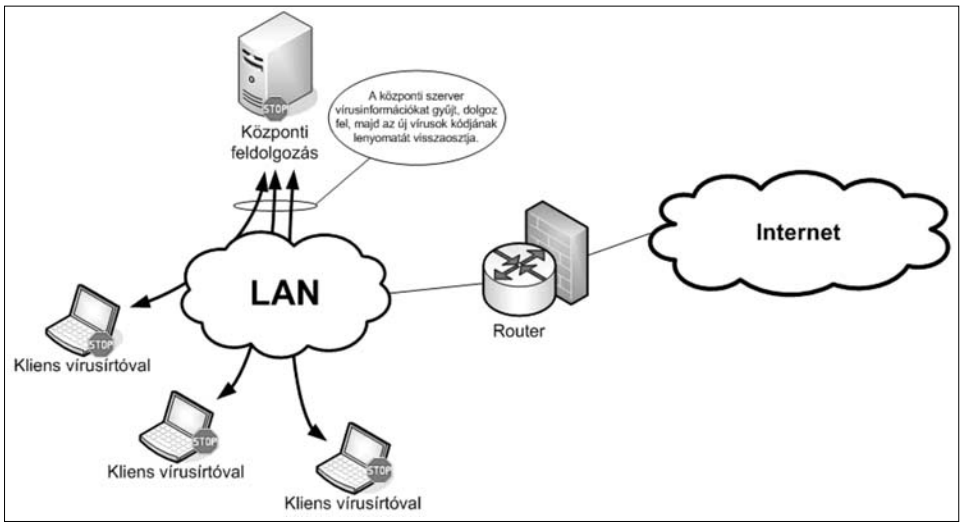
– *Hálózat szintjén*, amikor a teljes (al)hálózat forgalmát vizsgálva próbáljuk a botnetek forgalmát és tevékenységeit detektálni.

6.1. Detektálás felhasználó szinten

Alapvetően a botnetek két szinten detektálhatók. A legkézenfekvőbb megoldás a *felhasználói szintű felismerés.* Ekkor a végfelhasználónál telepített vírusirtó (illetve komplex védelmi) szoftverek segítségével észlelhetők a számítógépre telepített botnetek. A megoldás akkor lenne igazán sikeres, ha minden felhasználó rendszeresen használna vírusirtót, hiszen így garantálni lehetne a védelmet az ismert botnetek terjedése, fenntartása ellen. A felhasználói szintű védekezés azonban nem mindig lehet sikeres: sok esetben a felhasználók jelentős részénél a rosszindulatú kód hosszú időn át futásképes marad és a botnetek mérete csak csökken, de csökkent méretben is igen nagy kapacitással rendelkeznek.

Az egyéni felhasználók mellett a vállalatoknak is nagy figyelmet kell fordítaniuk számítógépeik karbantartására. Szinte mindegyik védelmi szoftver rendelkezik központosított menedzsmenttel *(2. ábra)*, jelentés és naplózás funkcióval. A központi felismerést segítheti az antivírus szoftver naplófájljainak központilag történő gyűjtése is.

Egy lokális fertőzés esetében így több gépen is sikerülhet azonosítani a veszélyforrást, mielőtt az nagyobb károkat okozhatna akár a saját hálózatunkban, akár mások hálózatában.



2. ábra
Központosított vírusfelismerés és feldolgozás

Az antivírus szoftverek mindazonáltal komplexebb feladatokat is elláthatnak, ha valamilyen egyéb hálózatbiztonsági szoftverrel – például tűzfallal – is képesek együttműködni. A gyanús viselkedési minta származhat abból a következtetésből, hogy egy program portokat nyit a felhasználó számítógépén, vagy gyanús, például nagy mennyiségű forgalmat generál bizonyos portokon. Ideális esetben a felhasználóknak maguknak kellene megszabni, hogy milyen általuk futtatott szoftverek használhatják az internetkapcsolatot, azonban ez a felhasználóktól nem várható el, pedig a védelmi szoftverek már ma is képesek lennének ilyen kifinomult ellenőrzések megvalósítására is.

Az antivírus rendszerek alapvetően két módon próbálják meg felismerni a kártékony programokat. A legegyszerűbb módszer, hogy a már ismert kártevő kódját felhasználva abból egy lenyomatot (úgynevezett szekvenciát) készítenek, ezt tárolják, majd a víruskeresés során a fájlokban ezeket a lenyomatokat keresik. Természetesen a különböző vírusirtó programok más és más algoritmusokat használnak, így ugyanarról a víruskódról más és más lenyomatot tárolnak. A megoldás előnye, hogy gyors és megbízható, hátránya viszont, hogy csak ismert kártevők vagy azok ismert variánsainak felismerésére használható.

A másik módszer a kártékony kódok keresésénél a *heurisztikus eljárás,* amelynek lényege, hogy akkor próbálja meg detektálni a vírusokat és más rosszindulatú kódokat, amikor azok lenyomata még nem létezik az adatbázisban. A kártevők megjelenése, a lenyomat elkészítése és a frissítés között eltelt idő alatt a kártevők szabadon garázdálkodhatnak, ezt illusztrálja a *3. ábra*.

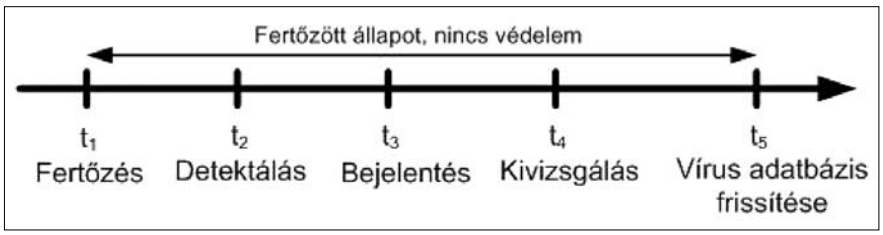
Ennek kivédésére jelentek meg a heurisztikát használó módszerek, amelyek kódokra és eseményekre alkalmazott szabályok pontozása alapján számítanak ki egy értéket, majd ennek függvényében döntik el, hogy az adott program kártevőnek minősül-e vagy sem. A heurisztikus eljárások előnye, hogy olyan kártékony kódokat is képesek felismerni, amelyeknek a lenyomata még nem szerepel az adatbázisban. Hátránya viszont, hogy sokkal nagyobb valószínűséggel hibáznak, így kártevőnek vélhetnek ártalmatlan alkalmazást és fordítva. Ez utóbbi tulajdonság miatt manapság inkább a lenyomat alapú eljárások használata a megszokott.

A felhasználói szintű védekezés hasznos a botok felderítésében, a korai felismerésben és segíthet a bothadseregek visszaszorításában, azonban önmagában nem nyújt megfelelő védelmet a heterogén felhasználói környezet miatt a bothálózatok ellen. Az is látható, hogy ez az alacsony szintű védelem önmagában nem alkalmas a botgazdák felderítésére és felelősségre vonására, továbbá a támadók lépéselőnye miatt a védelmi szoftverek előtt járva mindig használhatnak olyan módszereket, amelyekre a védelmi szoftverek még nem készültek fel.

6.2. Botnetek detektálására ismert módszerek hálózati szinten

A botnetek felderítésének egy másik módszere a *hálózati szintű felismerés*. Ebben az esetben egy egész (al)hálózat forgalmát vizsgálva, például lehallgatással történik a botnetek keresése és blokkolása. A megoldás előnye, hogy a rendszer a felhasználóktól és a botok kódjától is teljesen független, így az ismert forgalmi mintával rendelkező botok könnyen kiszűrhetőek, valamint a forgalomból bizonyos esetekben következtetni lehet a támadó kilétére is. Ennek következtében a felelősségrevonás is nagyobb eséllyel történhet meg, mint a felhasználói szintű védelem esetében.

A hálózati szintű felismerés több részre osztható a módszer függvényében. A leggyakrabban használt eljárások a csapdagépek és csapdahálózatok (honeypotok és honeynetek) és a behatolásdetektálás (IDS, Intrusion Detection System).



3. ábra
Lenyomat alapú felismerés folyamata a fertőzéstől az adatbázis frissítésig

6.2.1. IDS – Behatolásdetektálás

Behatolásdetektálás alatt olyan eljárásokat értünk, amelyek automatikusan képesek jelezni az operátor felé a gyanús tevékenységet hálózatunkban és számítógépeinken, illetve a biztonsági szabályok megszegését. Hálózati szinten a csomagok vizsgálatával foglalkozik az IDS, de a felhasználó szintű detektáláshoz hasonlóan egy IDS is működhet lenyomatok (szekvenciák, illetve szignatúrák) alapján, illetve alkalmazhat valamiféle heurisztikus módszert, ez utóbbit anomália-detekciónak is nevezik.

6.2.2. Darknetek és Honeypotok

A botnetek terjedésük közben véletlenszerű célpontokat választva keresik a támadható gépeket az Interneten. Innen ered az ötlet, hogy csapda elhelyezésével megismerhetőek a fertőzött számítógépek. Az ilyen csapdákat (angolul *honeypot,* bővebben lásd [3,4]) helyezhetjük a szokásos internetes környezetbe, de felhasználhatunk olyan IP tartományokat is, amelyek ugyan le vannak foglalva és az útvonalválasztás is működik hozzájuk, de nincsenek használatban. Az ilyen nem használt internetes címtartományokat nevezik *darknetnek.* A honeypotok gyűjtött adatainak integrálása, közös kezelése is megoldható, ezt általában honeynetnek hívjuk.

A csapdagépeket interakciós szintjük szerint kategorizálhatjuk, többnyire alacsony, közepes és magas interakciójú csapdákról beszélhetünk. Alacsony interakciós szinten a csapda szinte semmit nem enged a támadónak, elfogadja a támadást jelentő adatcsomagokat (legyen az e-mail, vagy valamilyen hiba kihasználása), de a támadónak nem enged további lehetőségeket. A másik véglet esetén, a magas interakciójú honeypot a támadás sikeres lezajlását is végrehajtja, lehetőséget biztosít a támadónak a rosszindulatú kód telepítésére és futtatására. Az alacsony interakciójú csapdák így többnyire listaszerű adatgyűjtésre szolgálhatnak fertőzött gépekről, míg a magas interakciójú csapdák segítségével pontosan megismerhető egy-egy botnet kliens működése, a botnethálózat vezérlése, de akár a botnetben használt rejtjelezés kulcsai is rögzíthetőek. A közepes interakciójú csapdák pedig a kettő között helyezkednek el: emulálják a támadható program működését, üzeneteit és válaszait, de valójában az adott programot nem futtatják. Az emuláció során a káros kód (botnet kliens) letöltődik, de a csapdát felállító fél anélkül vizsgálhatja, ismerheti meg azt, hogy az valójában lefutna. A különböző rendszerek összehasonlítását tartalmazza a *2. táblázat*.

6.2.3. Naplófájlok és hálózati forgalom analízise

A botnetek működésére a hálózati forgalom analízise is információt nyújthat. A folyamatos, céleszközzel történő megfigyelés (behatolásdetektáló eszközök, IDS-ek) mellett elegendő lehet azonban csak egyes naplófájlok vizsgálata. Ilyen lehet a kapcsolók (switchek), vagy az útvonalválasztók (routerek) naplójának vizsgálata. Ezekben a naplófájlokban a modern hálózati berendezések esetén konkrét hálózati forgalominformációk mellett már támadásokról szóló riasztásokról is információkat szerezhetünk.

Külön érdemes megemlíteni a Cisco által kifejlesztett Netflow [5] megoldást. Ez egy tárolási formátum és egyszerre egy vizsgálati módszer is. A hálózati berendezéseinken részletes információk menthetők el a hálózati forgalom egyes kapcsolatairól. Ez történhet teljeskörűen vagy részben, mintavételezéssel. A mentett adatok elemzésére modern eszközök állnak rendelkezésre, ezekkel is felfedezhetőek feltört számítógépek, bot klienseket futtató védett gépek.

6.2.4. Egyéb megoldások

A fent említett megoldásokon kívül számos cikk foglalkozik botnetek felismerésével. Ezek még kísérleti fázisban lévő megoldások, ezért az elérhető szoftverek ezeket a módszereket általában nem alkalmazzák.

Az egyik alapvető ötlet az IRC szervereken működő botok parancs-csatornáinak figyelése. Az IRC azért is

2. táblázat Honeypot-alapú támadásfelismerés típusai, működési elvük, és tulajdonságaik

	Működési elve	Előnye	Hátránya
Alacsony interakciójú honeypot	Teljes alhálózatot emulál, de szemben a darknettel, válaszol is a kérésekre	A teljes alhálózat modellezhető	Kártevők konkrét azonosítása nem megoldott, főként megfigyelésre alkalmas
Közepes interakciójú honeypot	Alkalmazási réteg virtualizációja: alkalmazások szimulálása	Káros kód nem települ, de könnyen elkapható	Károkozó hálózati forgalma nem vizsgálható
Magas interakciójú honeypot	Konkrét rendszerek megvalósítása	0-day támadások detektálhatók, kórokozók működése könnyen megfigyelhető	Minden támadás nem szűrhető, csak a telepített sérülékeny alkalmazásoké, több malware fertőzése nehezen szétválasztható

előnyös a botgazda számára, mert nem közvetlenül kommunikál a botokkal, így tartózkodási helye rejtve marad még akkor is, ha néhány bot kommunikációjára fény derül. Az IRC forgalmat nemcsak a csatornán folyó beszélgetéssel, hanem a kliensekhez közel, a hálózati kommunikáció vizsgálatával is ellenőrizhetjük. Statisztikai módszerek segítségével megkülönbözethető lehet a valós személyek kommunikációja a botokétól.

Az IRC alapú botnet-detektálásra adott módszerek viszonylag széles körűek. Ám sokkal nehezebb detektálni az elosztott rendszerben (például P2P) működő botneteket. A nehézséget az adja, hogy amíg egy IRC alapú botnet detektálása során egy központi elem keresésére van lehetőség (IRC szerver), addig egy elosztott rendszer esetében ilyen host nincs.

A P2P botok detektálásakor kihasználható [1], hogy a botok egymáshoz csatlakoznak, ezért egy portnak vagy port-tartománynak folyamatosan nyitottnak kell lennie, hogy a többi bot forgalmát fogadni tudja. Ezeket a portokat keresve, esetleg a portok forgalmát monitorozva lehetőség nyílhat a botok megfigyelésére. A módszer hátránya, hogy a porthoz nem köthető teljes bizonyossággal botnet-forgalom, így sok téves riasztás is keletkezhet. További megoldás lehet a sikertelen kapcsolódások figyelése, hiszen a botok megpróbálnak kapcsolódni a megadott címlistához, ám ez gyakran sikertelen. Ezen kívül, ha több host is próbál fix IP-címhez csatlakozni, és az nem érhető el, akkor az szintén gyanús viselkedésre utalhat.

A fent említett megoldásokon túlmenően egyre újabb és újabb megoldások látnak napvilágot a botnetek felderítésére, azonban a botok is fejlődnek: kommunikációjuknak elrejtése és új botnet variánsok megjelenése megnehezíti a fertőzött forgalom és a káros kódok detektálását.

7. Összefoglalás és jövőkép

Cikkünkben ismertettük a botnetek fogalmát, működését, hatásait és a felismerés lehetőségeit. Láthattuk, hogy a botnetek a mindennapjaink részévé váltak, hiszen számítógépeinket fertőzve elosztottan visznek végbe támadásokat, illetve küldenek kéretlen reklámleveleket a botgazda utasítására. A botnetek fejlődése mindemellett a mai napig folytatódik. A régebben nagy port kavart, több százezres, sőt a legnagyobbnak becsült hálózatok esetében több milliós zombi hálózatok helyett manapság inkább már a kisebb hálózatokat preferálják a botgazdák, hiszen ezek detektálásának a valószínűsége jóval kisebb. A központosított megoldások helyett pedig előtérbe kerülnek a peer-to-peer alapokon működő botnetek is. Ennek oka a nehezebb felismerésben és a rugalmasságban rejlik.

A jövőben alkalmazott botnetek vezérlésére a botgazdák valószínűleg már titkosított parancs-csatornát használnak a lehallgatás és a lenyomat alapú felismerés elkerülése érdekében. Mindazonáltal a felismerés megnehezítésére egyrészt váltogathatják a portokat és a protokollokat, másrészt pedig a statisztikai keresés ellen paddinget is alkalmazhatnak a parancsokat hordozó üzenetek szofisztikálásához.

Elmondható, hogy a botnetek nemcsak a jelent és a múltat, hanem a jövőt is képviselik. Nemcsak a meglévő eszközeinken fogják kifejteni tevékenységüket, de azokon is, amelyek még meg sem születtek. Biztosak lehetünk benne, hogy okostelefonjaink és más eszközeink célját fogják képezni a jövő botnetjeinek és abban sem kételkedhetünk, hogy sokáig együtt kell még élnünk a botnetek létezésével.

A szerzőkről

SZENTGYÖRGYI ATTILA a BME Villamosmérnöki és Informatikai Karán diplomázott 2006-ban telekommunikációhoz és hálózatbiztonsághoz kötődő szakirányokon. Jelenlegi doktori tanulmányait a Távközlési és Médiainformatikai Tanszéken a HSNLab tagjaként folytatja. Érdeklődési körébe tartoznak a vezetéknélküli hálózatok biztonsági kérdései, az ad hoc és peer-to-peer hálózatok biztonsága, a behatolásdetekció és -megelőzés, különösképp a botnetek vizsgálata és az azonosító alapú kriptográfiai eljárások alkalmazhatósága.

SZABÓ GÉZA Kecskeméten született 1982-ben. Egyetemi diplomáját a Budapesti Műszaki és Gazdaságtudományi Egyetemen szerezte 2006-ban. Munkája során internetes forgalom felismeréssel és modellezéssel foglalkozik. Az Ericsson Magyarországnál dolgozik kutatóként és PhD hallgató a Távközlési és Médiainformatikai Tanszék Nagysebességű Hálózatok Laboratóriumában a BME-n.

Irodalom

[1] Schoof, R., Koning, R.,
„Detecting peer-to-peer botnets”,
University of Amsterdam, 2007.
http://staff.science.uva.nl/~delaat/sne-2006-2007/p17/report.pdf

[2] C. Schiller, J. Binkley, G. Evron, C. Willems, T. Bradley, D. Harley, M. Cross,
Botnets: The Killer Web App. Syngress, 2007.
ISBN 1597491357

[3] N. Provos, T. Holz,
Virtual Honeypots: From Botnet Tracking to Intrusion Detection, Addison-Wesley Prof., 2007.
ISBN 0321336321

[4] Wicherski, G.,
„Medium Interaction Honeypots”, 2006.
http://www.pixel-house.net/midinthp.pdf

[5] Cisco Systems,
Introduction to Cisco IOS NetFlow, 2007.
http://www.cisco.com

[6] Strayer, W. T., Walsh, R., Livadas, C. Lapsley, D.,
„Detecting Botnets with Tight Command and Control”,
31st IEEE Conference on Local Computer Networks (LCN’06), 2006.

[7] Barford, P., Yegneswaran, V.,
„An Inside Look at Botnets”,
Advances In Information Security, Springer, 2007.
ISBN 978-0-387-32720-4

[8] F.C. Freiling, T. Holtz, G. Wicherski,
Botnet Tracking: Exploring a Root-Cause Methodology to Prevent Distributed Denial-of-Service Attacks,
LNCS, Vol. 3679, 2005.

DRM technológiák

FEHÉR GÁBOR, POLYÁK TAMÁS, OLÁH ISTVÁN

Budapesti Műszaki és Gazdaságtudományi Egyetem, Távközlési és Médiainformatikai Tanszék
{feher, polyak, olah}@tmit.bme.hu

Kulcsszavak: szerzői jogok védelme, tartalomszolgáltatók, DRM

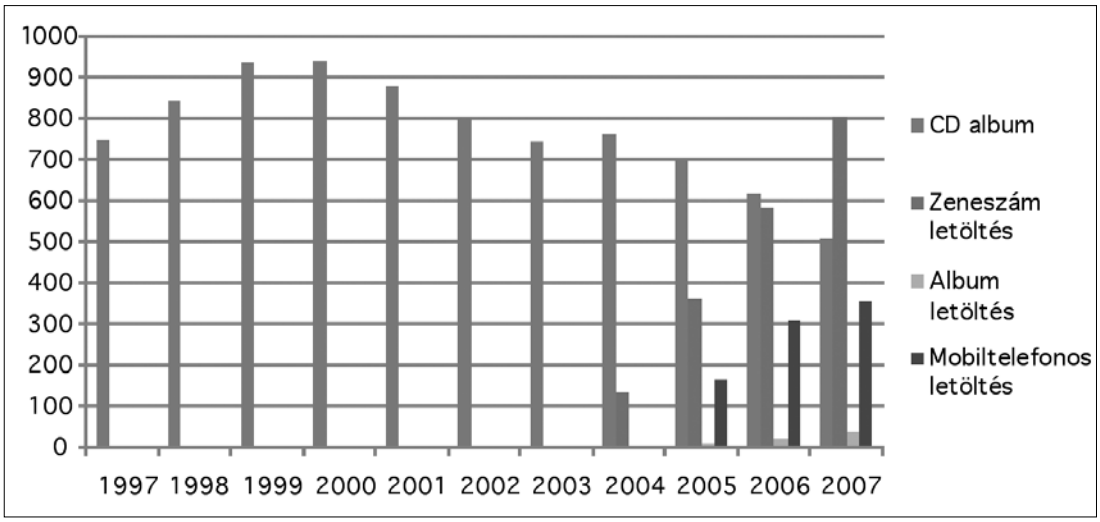
A szerzői jogok védelme örök probléma a társadalomban: biztosítani kell, hogy akik értékes tartalmat állítanak elő, megkaphassák érte a jussukat. Az analóg világban a problémát egyszerűen lehetett kezelni, mivel a másolás közben a mű minősége romlott, így aki igazi minőségre vágyott, annak muszáj volt fizetnie a tartalomért. A digitális világban azonban már más a helyzet, a digitális másolat minősége egy az egyben megegyezik az eredeti mű minőségével. Szükségszerű tehát, hogy a tartalmat védjük, a tartalomhoz köthető jogokat pedig kezeljük. Ez a védelem és jogkezelés a DRM (Digital Rights Management). A cikk célja, hogy az olvasót megismertesse a DRM technológia alapjaival és a felhasználásával.

1. Bevezetés

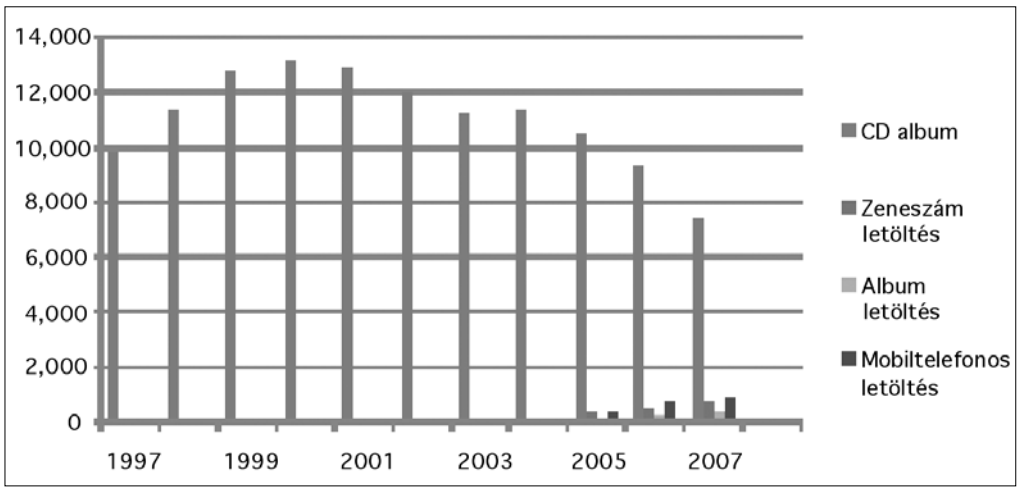
A digitális technika és az internet megjelenése és elterjedése számos jelentős és visszafordíthatatlan változást okozott több piacon is. Az egyik ilyen jelentősen érintett piac a szerzői jog oltalma alá eső tartalmak terjesztésével és kereskedelmével foglalkozó piac volt. A hagyományos zenei kazetták, CD-k, videokazetták és DVD-lemezek kiadói számos új, eddig ismeretlen kihívással kellett (és kell ma is) szembenézzenek. A hagyományos analóg adathordozókhoz képest (hangkazetták és videokazetták) a CD és DVD lemezek sokkal jobb minőségben és digitálisan tárolták a tartalmakat. Mivel ezek az adathordozók lehetővé tették, hogy otthoni körülmények között minőségveszteség nélkül lehessen másolatot készíteni róluk, megjelenésük potenciális veszélyt jelentett a kiadók számára. A felhasználók ugyanis ugyanazt a vizuális- és hangélményt kapták a másolt tartalomtól, mint amilyet a bolti változattól kaphattak.

A digitális tartalomtovábbítás elterjedését egy másik tény is elősegítette: megjelentek olyan tömörítési algoritmusok, amelyek akár tizedére, századrészére, vagy – videók esetében – akár ennél is jobban képesek voltak tömöríteni a tartalmakat. Ilyen úttörő volt a hanganyagok MPEG 1 Layer 3 (MP3) tömörítése és a videótartalmak tömörítésére szolgáló különböző MPEG 1-2-4 videó tömörítési eljárások, melyek a kis méret ellenére jó minőséget nyújtottak.

Az internet megjelenésével lehetővé vált, hogy ezeket a tömörített, kicsi és viszonylag jó minőségű digitális tartalmakat a felhasználók egymás között másolgassák. Ezt a folyamatot csak fokozta az egyre nagyobb sebességet kínáló szélessávú internetelérés. Kialakultak különböző fájlcserélő hálózatok, ahol a felhasználók különösebb bonyodalom nélkül juthattak hozzá illegálisan a tartalmakhoz. A kiadók szövetségeinek véleménye szerint ez a gyakorlat vonta maga után azt, hogy csökkenni kezdtek a lemezeladások, más csoportok szerint azonban máshol kell keresni a népszerűségvesztés okát. Tény azonban. hogy a hagyományos CD és DVD eladási modell mellett új módokat kellett keresni a tartalmak értékesítéséhez. Az *1. és 2. ábra* az amerikai lemezkiadók szövetségének (RIAA, Recording Industry Association of America) eladási adatait mutatja be.



1. ábra
RIAA szövetség által eladott médiapéldányok (millió db)
Forrás: Recording Industry Association of America



2. ábra
RIAA szövetség által eladott médiapéldányok bevétele (nettó millió USD), Forrás: Recording Industry Association of America

Az ábrákon jól látszik, hogy a legálisan forgalmazott médiapéldányok tekintetében a CD egyre kevésbé preferált, míg a digitális letöltések száma rohamosan nő. A 2. ábrán viszont az is látható, hogy a letöltött tartalmak után a kiadóknak nincsen akkora nyereségük, mint a hagyományos médiahordozók esetén. Összességében tehát még a legális digitális letöltések térnyerése is jelentős bevételtől fosztja meg a kiadókat.

2. A kiadók válasza: DRM

A kiadók látva a terjedő illegális fájlcsere-hálózatok népszerűségét, szerették volna elejét venni a tartalmak további illegális cseréjének. Mindezt úgy próbálták elérni, hogy megpróbálták megakadályozni, hogy a szerzői jogi védelem alatt álló tartalmak bekerülhessenek a fájlcserélő hálózatokba, valamint megpróbálták megakadályozni, hogy az ilyen hálózatokból letöltött tartalmak bekerülhessenek a legális tartalmak közé, vagyis lejátssza őket egy általános otthoni lejátszó eszköz. Ennek a problémának a kezelésére jöttek létre a különböző digitális jogkezelő rendszerek (DRM, Digital Rights Management).

A DRM rendszerek lehetővé teszik a kiadók és terjesztők számára, hogy menedzseljék a rendszerükben található tartalmakat. Legtöbbször a tartalmakhoz különböző jogokat és korlátozásokat lehet csatolni, amik segítségével megszabható, hogy a felhasználó hogyan használhatja fel a tartalmat. Meg lehet szabni, hogy ki játszhatja le a rendszerben megvásárolt filmet, milyen lejátszón lehet lejátszani, lehet-e róla másolatot készíteni stb. Az, hogy egy tartalomszolgáltató vagy kiadó milyen jogokat és korlátozásokat enged meg, az adott cég üzleti modelljétől függ. Ezen kívül persze fontos része egy DRM rendszernek, hogy milyen titkosítást használ, hogyan használja azt, valamint, hogy milyen egyéb védelmi elemeket definiál.

Egy DRM rendszer legtöbbször többféle technológiai elemből épül fel:

Titkosítás: Segítségével titkosítani lehet a tartalmakat, hogy azokhoz csak az férjen hozzá, aki jogosult rá. Ezt egy titkos kulccsal oldják meg, úgy hogy a kulcsot csak az (vagy annak a lejátszója) ismeri, aki megvásárolta a tartalmat.

Digitális vízjelezés: Vízjelezés segítségével úgy lehet információt elrejteni a tartalomban, hogy az észrevétlen marad az egyszerű felhasználó számára. Az elrejtett információ lehet a tartalom tulajdonosának valamilyen azonosítója, vagy a tartalmat letöltő felhasználó azonosítója. Jelenleg azonban nincs még olyan vízjelező algoritmus, amit ne lehetne eltávolítani az algoritmus ismeretében. Egyelőre egy lehetséges védelem az eltávolítással szemben az algoritmus titokban tartása, ami persze meggátolja az együttműködést más rendszerekkel, ugyanakkor nem garantálható, hogy az algoritmus nem kiismerhető és feltörhető.

Jogleíró nyelv: A felhasználó számára biztosított jogokat és megkötéseket valamilyen módon le kell írni, hogy az érthető legyen a különböző eszközök számára. Fontos tulajdonsága egy leíró nyelvnek, hogy milyen a kifejezőképessége. Ilyen nyílt jogleíró nyelv szabvány például az ODRL (Open Digital Rights Language).

Eszközök, amik betartatják a szabályokat: A jogleírásban meghatározott szabályokat tartatják be a felhasználóval. Például a lejátszó nem engedi lejátszani a fájlt, ha elfogyott a megvásárolt lejátszások száma.

Kommunikációs protokollok: Azoknak a kommunikációs protokolloknak az összessége, amik segítségével a különböző eszközök kommunikálnak egymással.

2.1. Az érintett iparágak és szereplőik

A DRM rendszereknek több érintettje is van. Sok érintett az itt felsorolt szerepek közül akár többet is betölthet.

Tartalom-előállítók: Ide sorolhatóak a művészek és filmstúdiók, akik a tartalmakat előállítják. Az ő érdekük, hogy minél többet értékesítsenek a jogvédett tartalomból és hogy minél ismertebbek legyenek. Van azonban példa már arra is, hogy az előállítók (általában maguk az előadók) hajlandóak lemondani az értékesítésből származó közvetlen bevételről a növekvő népszerűségből fakadó bevétel javára.

Tartalomszolgáltatók: Ők juttatják el az online tartalmakat a felhasználókhoz. Ezek lehetnek a kiadók tulajdonában, vagy függetlenek, mint például az Ama-

zon. Érdekük, hogy minél olcsóbb legyen a továbbítás, vagyis ne kelljen licenszdíjat fizetniük egy DRM rendszerért, de ugyanakkor biztonságos legyen a rendszerük, hogy a kiadók megbízzanak bennük, így minél több kiadóval tudjanak szerződni.

Hardvergyártók: Ők gyártják a tartalomlejátszó eszközöket, mint amilyenek a DVD lejátszók és a hordozható mp3 lejátszók. Céljuk, hogy minél olcsóbb legyen az eszközük, ezért nem akarnak olyan technológiát a gépeikbe építeni, amit a felhasználó nem fizet ki. Kénytelenek ugyanakkor DRM-et integrálni az eszközökbe, különben a védett tartalmakat nem fogja tudni lejátszani az eszköz.

DRM rendszerek szállítói: DRM rendszereket állítanak elő, és értékesítenek a piacon a tartalomszolgáltatóknak és a hardvergyártóknak. Céljuk, hogy az ő rendszerük legyen a legelterjedtebb.

Tartalomfogyasztók: A felhasználók, akik egyszerűen tartalmat akarnak fogyasztani, zenét hallgatni és videókat nézni, és nem akarnak olyan dolgokkal foglalkozni, mint a DRM. Egyesek fizetni se akarnak a tartalomért, mások nem akarják, hogy a DRM megmondja nekik, hogy melyik eszközökön lehet lejátszani a tartalmakat.

2.2. A DRM rendszerek gyenge pontjai

Sokan kritizálják a DRM rendszereket és nem alaptalanul. A fő kritikusok két táborból kerülnek ki: a felhasználók közül, valamint a tartalomszolgáltatók közül.

A felhasználó szempontjából a legnagyobb problémát az jelenti, hogy sokszor nehézkes ezeknek a rendszereknek a használata és sok olyan korlátozást kényszerítenek rá a felhasználóra, amivel nem nagyon akar együtt élni.

A legfontosabb ezek közül, hogy a tartalomszolgáltatók legtöbbször készüléktípushoz és készülékpéldányokhoz kötik a tartalmak lejátszását, ezért azt nem lehet egy másik lejátszón meghallgatni, vagy például az autónkban lévő lejátszóra átmásolni, még akkor sem, ha esetleg azon is van valamilyen DRM rendszer, csak éppen nem az a fajta, mint amilyenben a tartalmat megvásároltuk.

A kiadók szempontjából a legnagyobb probléma ezekkel a rendszerekkel, hogy még mindig nem teljesen tökéletesek, azaz gyakran feltörik a rendszereket. Egyre inkább elterjedt az a nézet, hogy egy rendszernek nem kell feltörésbiztosnak lennie, azonban ezt olyan nehéz legyen a felhasználónak megtenni, hogy inkább a zenék megvásárlását választja.

Szintén nagy probléma, hogy nincs átjárás a különböző DRM szabványok között, azaz nem lehet lejátszani például a mobiltelefon-rendszerben vásárolt tartalmat egy PCs DRM rendszerben. Léteznek azonban törekvések arra, hogy kialakuljon valamiféle szabvány az átjárásra.

3. Mai DRM technológiák

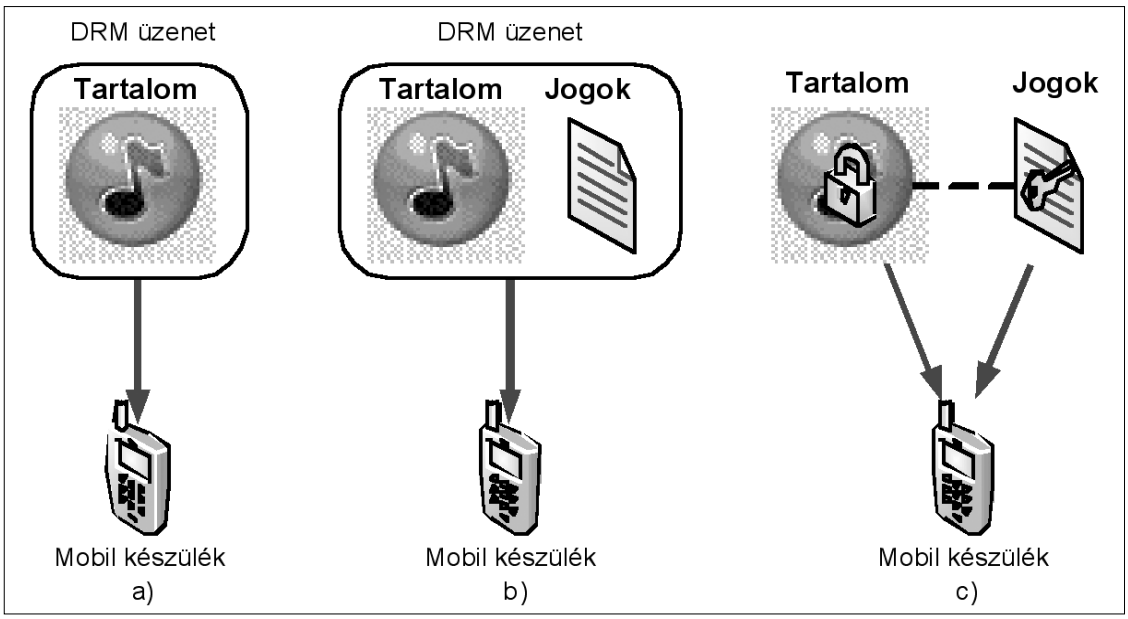
Aki ma DRM-védett tartalmat szeretne kínálni, sokféle DRM technológia közül választhat. A választásban döntő lehet, hogy a védett tartalmat milyen lejátszó platformon kívánja elérhetővé tenni. Mint írtuk, gondot jelent azonban az, hogy az egyes technológiák között az átjárás nehézkes vagy nem megoldható.

A mobiltelefonon elérhető technológiák esetében létezik egy DRM megoldás, amely igen széles körben támogatott. Ez az OMA DRM. Más platformok, mint például a PC-k esetén azonban nem létezik ennyire széles körű támogatása egyetlen technológiának sem.

3.1. Open Mobile Alliance – OMA DRM

Az Open Mobile Alliance 2002 júniusában jött létre, jelenleg több mint 350 nemzetközi cég alkotja. Az OMA tagjai az egész mobil szolgáltatási értékláncot lefedik:

– *Mobileszköz- és rendszergyártók:*
Ericsson, Thomson, Siemens, Nokia, Philips, Motorola, Texas Instruments stb.



3. ábra
OMA DRM 1.0 által támogatott metódusok:
a) Forward lock,
b) Combined delivery ,
c) Separate delivery

– *Mobilszolgáltatók:*
  Vodafone, T-Mobile, Orange, Telefónica stb.
– *Szoftverforgalmazók:*
  Microsoft, Sun, Microsystems, IBM, Oracle, Symbian stb.
– *Tartalomszolgáltatók:*
  Time Warner, Yahoo stb.

A testület a mobil távközlési ipar számára készít nyílt specifikációkat, elősegítve ezzel az eszköztől, hálózattól, szolgáltatótól független mobil szolgáltatások fejlesztését.

Az OMA által kidolgozott szabványok többek között digitális médiaobjektumok jogvédelmét megvalósító rendszerek specifikációit is tartalmazza (OMA DRM).

3.1.1. OMA DRM 1.0

A szervezet 2004 júniusában fogadta el a szabvány 1.0-ás verzióját, amit jelenleg több mint 550 mobiltelefon típus támogat.

A szabvány lényege, hogy a megvásárolt médiaobjektum használata meghatározott jogosultságokhoz kötött. Ugyanahhoz a tartalomhoz többféle jogosultság is tartozhat (pl. egy film esetén a felhasználó választhat korlátozott és korlátlan számú megtekintés között). A jogok leírása egy, a szabványban meghatározott XML formátumú fájl segítségével történik. A jogok érvényesítését a készülékeken található DRM ügynök *(DRM agent)* végzi.

A digitális tartalom egy DRM üzenetben *(DRM Message)* kerül terjesztésre. A tartalom sokféle lehet, kezdetekben azonban ez inkább csak csengőhangokra, háttérképekre, operátor logókra, játékokra korlátozódott. A DRM üzenet tulajdonképpen a bináris tartalom illetve az esetlegesen hozzá kapcsolódó leíró fájlok MIME csomagolása. A DRM üzenetben a tartalom lehet akár titkosítva is.

Az OMA DRM 1.0 a tartalom terjesztésben három fő metódust támogat *(lásd az előző oldali 3. ábrán)*.

Tiltott továbbítás (Forward lock)

A felhasználó letölt a készülékére egy médiaobjektumot a szerverről. A tartalom egy DRM üzenetben érkezik, a letöltéshez szükség van a tartalom URL-jére. Letöltés után a tartalom szabadon megtekinthető, ahányszor csak a felhasználó kívánja, azonban nem továbbítható más készülékekre. A továbbítás tiltását a készülék DRM ügynöke felügyeli, illetve szintén az gondoskodik arról, hogy csak olyan alkalmazás érje el a tartalmat, amely megbízható.

Kombinált letöltés (Combined delivery)

Ebben az esetben a letöltött DRM üzenet a választott médiaobjektumon kívül egy jogosultságobjektumot is fog tartalmazni. Ez az objektum határozza meg, hogy a felhasználó miként használhatja a médiaobjektumot. A jogosultságobjektumon keresztül lehetőség van például arra, hogy a felhasználó beletekinthessen a megvásárolandó tartalomba.

A jogosultság leírása a *Right Expression Language (REL)* segítségével történik. A nyelv kialakításánál az egyszerűségre törekedtek, így csak az eljárások és kényszerek minimális halmazát tartalmazza. Az engedélyhez köthető funkciók a következőek: lejátszás, megjelenítés, végrehajtás és nyomtatás. A kényszerek a funkciókat limitálják, tehetik ezt darabszámra, meghatározott időpontok között vagy meghatározott időintervallumra. A tartalom továbbítására itt sincs lehetőség.

Szétválasztott letöltés (Separate delivery)

Szétválasztott letöltésről beszélünk, ha a médiaobjektum és a jogosultságobjektum külön (akár különböző csatornán, különböző időben, különböző szerverről stb.) érkezik meg a készülékre. A médiaobjektum egészen addig nem használható, amíg a hozzá tartozó jogosultsági objektum nem áll rendelkezésre.

Ez a metódus lehetővé teszi a tartalmak továbbítását más készülékekre. A médiaobjektum továbbítása után a céleszköznek is be kell szereznie a tartalomhoz tartozó jogosultságokat, különben a DRM ügynöke nem fogja engedélyezni a tartalom használatát.

Streaming OMA DRM 1.0 rendszerben

Bár az OMA DRM 1.0 szabványt nem úgy fejlesztették, hogy kimondottan alkalmas legyen stream-elt tartalom lejátszására, mégis kisebb kerülő úton lehetőségünk van erre is. Az ajánlás szerint, ilyenkor a videófolyam specifikációja van a DRM üzenetben. A specifikáció leggyakrabban egy SDP (Session Description Protocol) üzenet és tartalmazza a videófolyam eléréséhez szükséges URL-t. A lejátszás során ügyelni kell arra, hogy amennyiben a tartalom csak egyszer nézhető meg, olyan lejátszót válasszunk, amely nem képes a kapott tartalmat elmenteni. A biztonság fokozására a folyamot titkosítani is szokták, ekkor a dekódoláshoz szükséges kulcs is a védett SDP üzenetben található.

3.1.2. OMA DRM 2.0

Az Open Mobile Alliance 2006 júniusában fogadta el szabvány második verzióját, ami tulajdonképpen az első verzió „Szétválasztott letöltés” metódusának kiterjesztése. Később, 2008 elején a 2.0 szabványt módosították, így lett belőle 2.0.1. Ebben a verzióban csak a szétválasztott letöltés modell használható, azonban az nagyobb funkcionalitást és biztonságot kínál, mint az 1.0 verzióban. A legfőbb újítások:

– A tartalom és jogok mozgatása különböző eszközök között
– Tartalom exportálása offline eszközökre (pl. mp3 lejátszó)
– Szabad tartalommegosztás felhasználó csoportok között (domain)
– PKI alapú kölcsönös azonosítás a felhasználó eszköze és a jogkezelő között
– Bővülő leírás a jogokhoz
– P2P szuperdisztribúció támogatás

Az OMA 2-es szabvány ugyanakkor még koránt sem annyira elterjedt, mint előző verziója, így aki mobiltelefonos DRM platformban gondolkozik, annak még mindig egy megfontolandó lehetőség az OMA DRM 1.0.

Az architektúra résztvevői és elemei

A DRM architektúra részeit a *4. ábra* mutatja.

DRM ügynök: Hasonlóan az előző verzióhoz, egy megbízható entitás a felhasználó készülékén, ő felelős azért, hogy egy tartalmat csak a hozzá tartozó jogosultságobjektumban meghatározott módon lehessen felhasználni.

Tartalomszolgáltató: Elérhetővé teszi a DRM tartalmat. A szabvány pontosan definiálja a DRM tartalom formátumát. A tartalom DRM ügynökhöz juttatása különféle átviteli módszerekkel (HTTP, WAP, MMS stb.) történhet.

Jogosultságszolgáltató: Engedélyeket és megkötéseket rendel a DRM tartalomhoz, majd létrehozza az ezeket tartalmazó jogosultságobjektumot. A DRM tartalom nem használható a jogosultságobjektum nélkül, és csak annak megfelelően használható.

Felhasználó: DRM tartalmat igényel, ezekhez csak a DRM ügynökön keresztül fér hozzá

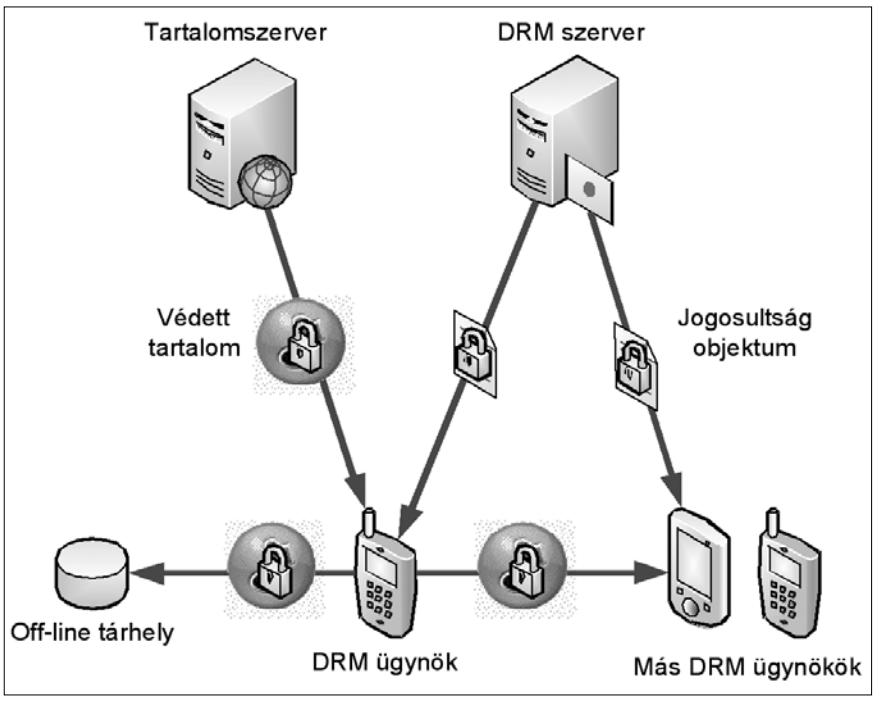
Készüléken kívüli tárhely: Az OMA DRM 2.0 szerinti DRM tartalom biztonsága nem sérül, ha azt a felhasználó a készüléken kívül (is) eltárolja. Ennek oka lehet biztonsági másolat, tárhely felszabadítása a készüléken stb. a jogosultságobjektumok közül csak azok tárolhatók készüléken kívül, amelyek nem tartalmaznak állapotinformációt (például fennmaradó lejátszások száma).

Biztonság

Az OMA DRM 2.0 nagy erőssége az első verzióval szemben a biztonság. A szabvány biztosítja, hogy a tartalomhoz csak az férhessen hozzá, aki arra jogosult, valamint a jogosultságobjektumot csak a megfelelő készülék (vagy csoport) tudja értelmezni.

A biztonságos DRM tartalom létrehozása és küldése a következő lépésekből áll:

4. ábra OMA DRM 2.0 architektúra



1. Tartalom csomagolása:

A tartalomszolgáltató egy biztonságos konténerbe (DCF, DRM Content Format) csomagolja a média objektumot. A DRM tartalmat egy szimmetrikus tartalomtikosító kulcs (CEK, Content Encryption Key) segítségével rejtjelezi.

2. DRM ügynök hitelesítése:

Minden DRM ügynök rendelkezik egy publikus/privát kulcspárral és egy tanúsítvánnyal. A tanúsítvány kiegészítő információkat tartalmaz, mint a gyártó vagy a készülék típusa. Ennek segítségével a tartalom- és a jogosultságszolgáltató képes hitelesíteni az ügynököt.

3. A jogosultságobjektum generálása:

Elkészül a jogosultságobjektum XML fájlja. Ez tartalmazza a CEK-et is. Ez biztosítja, hogy a tartalmat ne lehessen használni a jogosultság objektum nélkül.

4. A jogosultságobjektum védelme:

A jogosultságobjektum küldése előtt annak érzékeny részei (pl. a CEK) titkosításra kerülnek. Ez a titkosítás a DRM ügynök publikus kulcsával történik. Ez biztosítja, hogy csak a megfelelő DRM ügynök férjen hozzá a DRM tartalomhoz. A DRM szerver ezen kívül digitálisan aláírja a jogosultság objektumot.

5. Szállítás:

A DRM- és a jogosultságobjektum készen állnak arra, hogy eljuttassák őket a DRM ügynökhöz. Mivel mindkettő biztonságos, tetszőleges szállítási protokoll (HTTP, Wap Push, MMS stb.) használható.

3.1.3. OMA DRM 2.1

A második verzió kiadása után az újabb piaci igényekre reagálva az OMA elkészítette a DRM szabvány 2.1-es verzióját. Ennek architektúrája megegyezik a 2.0-val, azonban beleépítettek néhány új felhasználási lehetőséget:

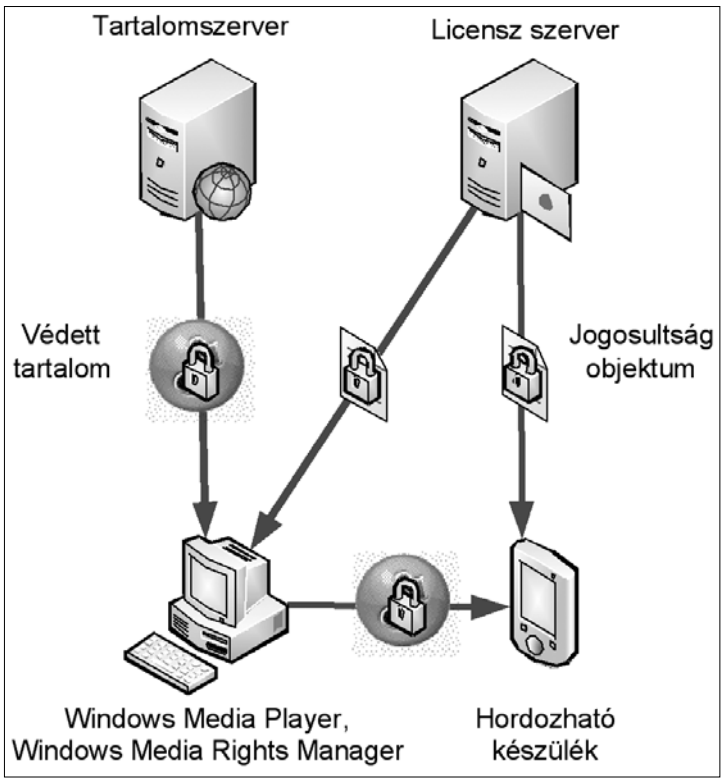
– *Mérések támogatása:* A jogosultság kibocsátójának szüksége lehet információra a különböző tartalmak felhasználásairól.

– *Jogosultság feltöltése:* Lehetőség van a jogosultságobjektumot a DRM szolgáltatóhoz feltölteni. Erre akkor lehet szükség, ha a felhasználó át akarja mozgatni a jogosultságobjektumot egyik készülékről a másikra.

– *Megerősítés a jogosultságobjektum telepítéséről:* A DRM ügynök megerősítő üzenetet küld a DRM szervernek a jogosultságobjektum telepítése után.

3.2. Microsoft Windows Media DRM

A Microsoft Windows Media DRM a Microsoft saját jogvédelmi megoldása, ami végponttól-végpontig tartó biztonságot garantál. A rendszer aktuális verziója a 2004-ben kiadott Microsoft Windows Media DRM 10.



5. ábra
A Windows Media Rights Manager architektúrája

A *5. ábra* mutatja a rendszer architektúráját. Látható, hogy a rendszer felépítése nagyrészt egyezik az OMA DRM 2.0 felépítésével. A felhasználó csak akkor férhet hozzá a becsomagolt tartalomhoz, ha rendelkezik a szükséges jogosultsággal.

A Windows Media Rights Manager által szolgáltatott biztonságos tartalom létrehozása és küldése a következő lépésekből áll:

1. Csomagolás: A Windows Media Rights Manager egy kulcs segítségével titkosítja a médiafájlt. Ezt a kulcsot a rejtjelezett licenszobjektum fogja tartalmazni. A csomagolt médiafájlba egyéb információk is kerülnek, például a cím, ahonnan a licensz megszerezhető. A rendszer a Microsoft saját médiaformátumait használja (*.wma illetve *.wmv fájlok).

2. Elosztás: A csomagolt fájl elérhetővé tételére több lehetőség van: feltölthető egy webszerverre, terjeszthető CD-n, e-mail-en küldhető stb. A rendszer a tartalmak másolását, továbbküldését is engedélyezi.

3. Licensz-szerver: A tartalomszolgáltató választ egy licensz szolgáltatót, aki a tartalmakra vonatkozó specifikus jogokat és szabályokat fogja tárolni. A licensz-szerver feladata a felhasználó licensz igényének elbírálása.

4. Licenszkérelem: Védett tartalom lejátszásához a felhasználónak rendelkeznie kell a titkosítás feloldásához szükséges kulccsal. Az ezt tartalmazó licenszet a licensz-szervertől kérvényezi. A licenszkérelem automatikusan megtörténik, amikor a felhasználó első alkalommal tekinti meg a tartalmat. A rendszer ilyenkor vagy egy regisztráció oldalra irányítja a felhasználót, vagy a háttérben kéri le a licenszet.

5. Médiafájl lejátszása: A tartalom lejátszásához Windows Media Rights Manager-t támogató lejátszóprogramra van szükség. A felhasználó a licenszben meghatározott feltételek szerint tekintheti meg a tartalmat. Ez különböző jogosultságokat tartalmazhat: kezdeti időpontot és időtartamot, lejátszások számát stb. Lehetőség van egy licenszen belül több készülék számára is jogokat biztosítani.

3.3. RealNetworks – Media Commerce Suite

A RealSystem által kifejlesztett DRM rendszer szintén az on-line terjesztett tartalmak védelmét hivatott ellátni. Számos üzleti modellt támogat, mint feliratkozás (subscription), Video on Demand (VOD) stb. Létező RealSystem rendszerek is kiegészíthetők vele.

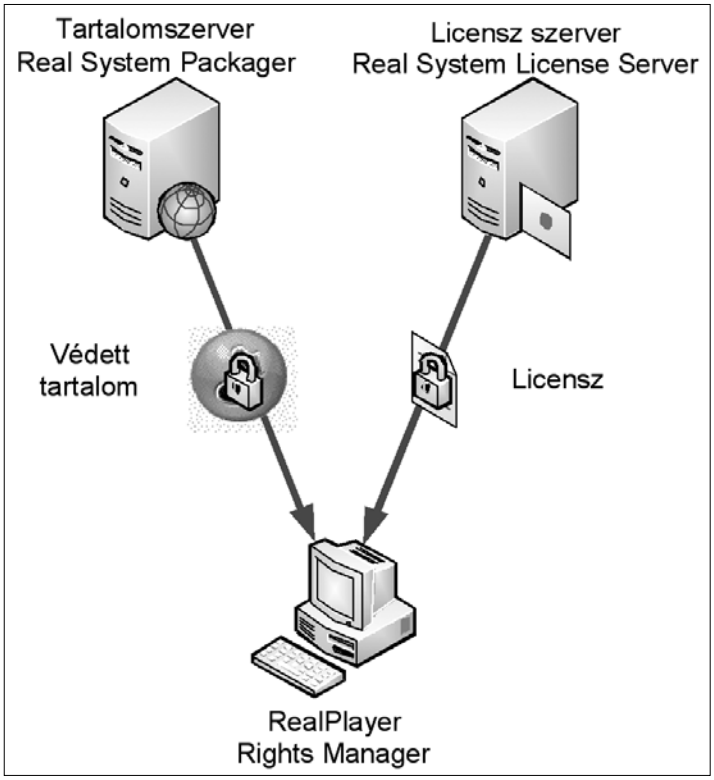
A RealSystem Media Commerce Suite négy komponenst nyújt a médiatartalmak védelmére, terjesztésére és a jogosultságok érvényesítésére:

- **RealSystem Packager**
  Tartalomszolgáltatók részére nyújtott szoftver, segítségével a médiafájlok biztonságos formátumba csomagolhatók.
- **RealSystem License Server**
  HTTP szerver, ami licensz kérelmeket fogad, és licenszeket generál, amik lehetővé teszik a védett médiafájlokhoz való hozzáférést.
- **Media Commerce kiegészítés a RealPlayer-hez**
  Egy megbízható kliens, ami képes biztonságos RealMedia fájlokat (*.rms) értelmezni.
  Ez a fájlformátum speciálisan erre a célra, RealMedia tartalom biztonságos tárolására lett létrehozva.
  A kiegészítés biztosítja, hogy a tartalom megbízható környezetben, a rendelkezésre álló jogosultságoknak megfelelően lesz felhasználva.
- **RealSystem biztonságos fájlformátum beépülő modul**
  RealSystem szerver számára teszi lehetővé a védett tartalom terjesztését (streaming).

3.4. Marlin DRM

A Marlint 2005 januárjában alapította öt vállalat: az Intertrust, a Panasonic, a Philips, a Samsung és a Sony. Céljuk egy DRM-en alapuló tartalom megosztási platform létrehozása volt. 2005 októberében az alapító vállalatok bejelentették a Marlin Developer Community (MDC) megalakítását. A közösség tagjai számára nyilvános a Marlin specifikációja, eszközei, SDK-ja stb. A tagok így részt vehetnek a rendszer fejlesztésében, tesztelésében, a forráskód felülvizsgálatában. Az MDC ezen kívül tréningeket és más eseményeket is szervez.

Az alapítók által létrehozott másik szervezeti egység, a Marlin Trust Management Organization (MTMO) felügyeli az MDC munkáját. Ez a szervezet végzi a Marlin termékek számára a kulcsok és licenszek kezelését, valamint a közösség által fejlesztett Marlin termékeket kötelezi a meghatározott szabályok, feltételek betartására.



6. ábra
RealSystem Media Commerce Suite Architektúrája

A Marlin DRM architektúra is ugyanazokból a szereplőkből áll, mint az OMA DRM: tartalomszolgáltató, jogosultságszolgáltató és felhasználó. A tartalom- és jogosultságobjektumok használata is az OMA DRM-ben leírtakhoz hasonlóan történik: a tartalomszolgáltató egy szimmetrikus kulcs segítségével titkosítja a tartalom objektumot. A jogosultságobjektum pedig tartalmazza a tartalom felhasználási szabályait, valamint a tartalom titkosításának feloldásához szükséges kulcsot. A két rendszer közti fő különbség a jogosultság értelmezésében van.

3.4.1. Csomópont- és kapcsolatobjektumok

A Marlin csomópont (*node*) és kapcsolat (*link*) objektumokat használ a résztvevők (felhasználók, tartalomszolgáltatók és jogosultságszolgáltatók) közötti reláció kifejezéséhez. A tradicionális DRM rendszerekben a jogosultság közvetlenül az azt lekérő készülékhez van kötve. A Marlinban a jogosultság felhasználóhoz van kötve és a felhasználók és készülékek, vagy felhasználók és előfizetések közötti kapcsolatok csomópontok és kapcsolatok rendszereként vannak tárolva. A csomópontok szerinti szétválasztás nagyon rugalmassá teszi a rendszert.

A csomópontok a rendszer logikai entitásait reprezentálják: készülékek, felhasználók, feliratkozások stb. A kapcsolatok a csomópontokat kötik össze, egy irányított gráfot hozva létre.

Erre mutat példát az *7. ábra*.

7. ábra
Csomópontokból és kapcsolatokból álló irányított gráf

Egy csomópont akkor használhatja egy másik tartalmat, ha az irányított gráfban vezet hozzá út. A 7. ábra példájában „Készülék A” használhatja a tartalomszolgáltató által küldött tartalmat, míg „Készülék C” nem.

3.5. Apple FairPlay

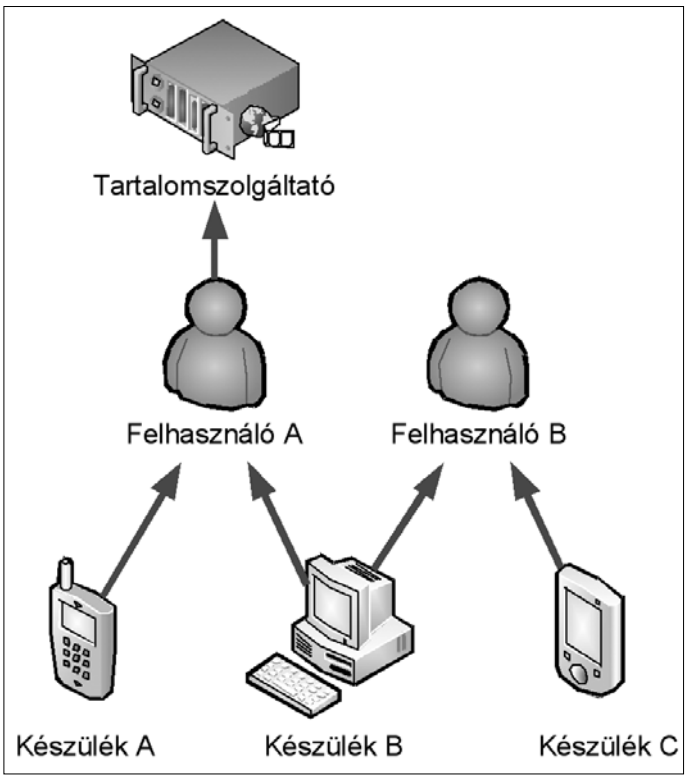
A FairPlay az Apple Inc. által létrehozott és alkalmazott DRM technológia. Ismertségét az iTunes on-line zeneboltnak köszönheti, az innen vásárolt zeneszámokat a FairPlay technológia védi. Az AAC fájlok titkosított formátumban érkeznek, a felhasználó a következő megkötésekkel használhatja őket:

– egy zeneszám tetszőleges iPod készülékre másolható;
– egy zeneszám maximum 5 különböző, az iTunes Store oldalán regisztrált számítógépen játszható le;
– a zeneszám akárhányszor másolható audio CD-re, azonban ugyanaz az összeállítás maximum hétszer írható fel.

A jelenleg támogatott készülékek: Apple iPod, Apple iPhone, Motorola ROKR E1, Motorola SLVR, Motorola RAZR V3i.

3.6. SUN DReaM (DRM/everywhere available)

A Sun Microsystems nem gyárt szórakoztatóelektronikai eszközöket, 2005 szeptemberében mégis beszállt a DRM üzletbe, mivel a DRM piac egyre növekszik és az elérhető DRM megoldások csak a nagy gyártók, vagy konzorciumok saját megoldásai körül csoportosulnak. A Sun ezzel szemben egy licenszdíjmentes változatot ígér nyílt forrással, ahol magának a gyártónak azért van némi kontrollja az elkészült szabványon.



A Sun DRM architektúrájának az alapja az DRM-OPERA. Ez a megoldás egy korábbi EU által támogatott projektből származik, ahol a hangsúly az együttműködésen volt. Előnye, hogy a tartalom csak a hitelesített felhasználóhoz kötődik, és nem függ attól az eszköztől, amit a felhasználó éppen használ.

A SUN DReaM megoldás legnagyobb előnye az OPERA projektből átvett együttműködésben van, valamint abban, hogy a megoldás használata nem lesz licenszköteles.

3.7. A DRM elterjedése

A DRM esetében is nagyon fontos tényező az elterjedtség. Az OMA DRM 1.0-t a legtöbb mobiltelefon támogatja, azonban a biztonság és felhasználhatóság szempontjából kedvezőbb 2.0 (vagy 2.1) szabványt már kevesebben. A többi szabvány, amelyek mögött egy vagy több, de összességében kevés számú cég áll, várhatóan csak a saját platformján tudja elterjeszteni saját megoldását. Ennek a technikai okok mellet szabadalmi okai is vannak. Ilyen esetben például várható, hogy a Windows alapú rendszereken mindig is Windows DRM fut majd, az Apple készülékein a FairPlay, míg a Panasonic, Philips, Samsung, valamint a Sony készülékeken a Marlin DRM.

Az elterjedtséggel szorosan összefügg a szabadalmak kérdése. A legtöbb DRM szabvány mögött álló cég licenszdíjat kér technológiájuk felhasználásáért. Ezért is van az, hogy az egyik DRM szabványban érdekelt cég nem fogja a konkurens technológiát alkalmazni. Vannak azonban olyan megoldások is, ahol nincs licenszdíj. Elterjedtség szempontjából a DRM technológiák közötti átjárhatóság egy nagyon fontos kérdés, és – bár a felhasználók örülnének ilyen megoldásoknak –, úgy tűnik, hogy a gyártók nem látják érdekeltségüket ezen a területen.

4. Élet DRM nélkül

A DRM-et teljesen elutasítók legfőképpen azt hozzák fel a DRM rendszerek ellen, hogy az csak egy olyan torzszülött, ami azért jött létre, hogy a kiadók (jobb híján) fenn tudják tartani a régi rendszert, vagyis hogy fizetünk a tartalmakért. Ezzel szemben sokkal jobb volna, ha olyan új üzleti modelleket dolgoznának ki, ami jobban illeszkedik ahhoz, hogy a tartalmakat teljesen szabadon másolgatják a felhasználók. A kiadók azonban nagyon erősek, így nehéz velük szembeszállni. Azt például, hogy milyen online zeneboltokban árulják a tartalmakat, a kiadók döntik el, legtöbbször az alapján, hogy a bolt milyen DRM rendszerrel van felszerelve.

4.1. DRM-mentes üzleti modellek

Sok üzleti modell azonban már most szakít a DRM-mel és helyette DRM mentesen vagy akár ingyenesen teszi elérhetővé a tartalmat. A DRM-védelem nélküli tartalmak letöltés után akár szabadon másolhatóvá válnak.

Az egyik legfontosabb dolog ezzel az elvvel kapcsolatban, hogy a kiadók döntik el, hogy odaadják-e a számaikat azoknak a kereskedőknek, akik nem használnak semmilyen DRM rendszert. A fő visszatartó erő, hogy eddig még nem sikerült bizonyítani, hogy jobban megéri DRM mentes tartalmat eladni a felhasználóknak, mint DRM-védettet.

A DRM-mentesség értéket jelent a felhasználónak, ezért a jelenleg üzemelő zeneboltok egy része DRM-védett, és DRM-mentes tartalmakat is kínál, utóbbiakat valamivel drágábban, mint a védett tartalmakat, hiszen az árban így kompenzálja a kieső eladása utáni veszteséget. A felhasználók többnyire azért veszik meg a DRM-mentes tartalmakat, mert esetleg több lejátszón is le akarják játszani azt.

Azok a tartalomszolgáltatók, akik teljesen védelemmentes tartalmat is kínálnak, többféle üzleti modellel dolgoznak. Van olyan, akiknél számonként kell fizetni, mint a legnépszerűbb eMusic, Amazon MP3 vagy Aime Street. Léteznek olyan szolgáltatók, akik csak online hallgatásra kínálják a számokat, mint amilyen például a radio.blog.club.

Újfajta kezdeményezések is napvilágot láttak. 2007 szeptemberében a Radiohead együttes saját kiadóját megkerülve, a honlapjáról ingyen letölthetővé tette legújabb albumát. A rajongóknak annyit kellett fizetnie érte, amennyit akartak, de akár ingyen is letölthették. Egy hónappal az indulás után minden harmadik letöltő fizetett valamennyit a számokért, átlagban 6 dollárt. Ennek nyomán a Radiohead körülbelül 2,4 millió dollár bevételt eszközölt ebből az akcióból (amit már nem kell megosztania a kiadóval).

Hasonlóan nagy lépés volt a kiadók átformálásában, hogy a popénekesnő Madonna 2007 októberében szerződést bontott 20 éves kiadójával és az új szerződését a következő 3 albumára már egy koncertszervező céggel kötötte meg. Így Madonna, a tartalom tulajdonosa már egy csatolt terméken keresztül jut a pénzéhez, illetve az is előfordulhatna, hogy most már maga a zenei tartalom is csak csatolt termék.

Az említett zenei példák mellett olyan is akad, ahol a készülék árába tervezik beépíteni a tartalomletöltés előfizetését. Bár az ilyen jellegű üzleti modellek még nem elterjedtek, mégis számos kísérlettel találkozunk.

5. Összefoglalás és jövőkép

Cikkünkben ismertettük a DRM technológia mögött található motivációkat. Amennyiben a régi tartalom és annak elosztásának gyakorlatát tekintjük, elengedhetetlen, hogy a digitális médiával együtt megjelenjen a DRM, amely annak védelmével hivatott foglalkozni. A DRM térnyerésével a kiadók tovább folytathatják bevált üzleti modelljeiket. A jól működő DRM így orvosság a média tartalom kiadói számára a felhasználók illegális másolásai ellen.

Az egyszerűen kivitelezhető, bár illegális másolást megismerő felhasználóknak viszont ez a modell nem

tetszik. A másolást nem tartják nagy bűnnek, a kiadókat viszont, látva az elrettentő pereket, elítélik. Nagyjából itt tart ma a világ.

Ugyanakkor az illegális másolások és a kezdeti DRM rendszerek gyengesége azt is eredményezte, hogy a kiadók lassan kezdik belátni, hogy új értékesítési modellekre is szükség van a hagyományos eladások mellett vagy helyett. Jelenleg az online zeneboltok szaporodásának és a havidíjas tartalomfogyasztás elterjedésének is tanúi lehetünk. A DRM technológia itt is segít, és úgy látszik, az új modellek esetében a felhasználók befogadják már a DRM-et.

A közeljövőben várhatóan a tartalomkínálat még inkább a letöltéses eladások irányába fog mozdulni és egyre kevésbé lesz népszerű a tartalmak fizikai hordozókon, mint például CD lemezeken történő árusítása. Az egyre nagyobb számú fogyasztó már rákényszerítheti a DRM technológiák mögött álló cégeket, hogy dolgozzák ki a technológiák közötti átjárás megvalósítását. Így végül a DRM elhozza mindenki megelégedettségét, a szerzőknek és kiadóknak nem kell tartani a jelentős mértékű illegális másolásoktól, a felhasználók pedig reális áron jutnak a tartalomhoz, amit bármilyen DRM képes készülékkel le is hallgathatnak vagy meg is nézhetnek. Ez azonban sajnos még csak a jövő...

Irodalom

[1] Iannella, R.,
The Open Digital Rights Language:
XML for Digital Rights Management .
Information Security Technical Report,
Volume 9, Issue 3, July-September 2004, pp.47–55.

[2] OMA (2004).
Open Mobile Alliance DRM Specifications,
Version 1.0, Approved Enabler, 2004. június,
http://www.openmobilealliance.org/Technical/
release_program/drm_v1_0.aspx

[3] OMA (2008).
Open Mobile Alliance DRM Specifications,
Version 2.0.1, Approved Enabler, 2008. február,
http://www.openmobilealliance.org/Technical/
release_program/drm_v2_0.aspx

[4] OMA (2007).
Open Mobile Alliance DRM Specifications,
Version 2.1, Candidate Enabler, 2007. július,
http://www.openmobilealliance.org/Technical/
release_program/drm_v2_0.aspx

Hírek

Vegyes vállalatot hozott létre az operatív irányítás terén szerzett tapasztalatáról ismert két cég, a Siemens AG és a Gores Group. Az elsősorban vállalatok bővítésével foglalkozó amerikai magántőkebefektető társaság, mely széles körű tapasztalatokkal rendelkezik a technológiai és a távközlési szektor vállalatainak irányításában, történetének eddigi legnagyobb volumenű felvásárlásával a **Siemens Enterprise Communications** vállalat üzletrészének 51 százalékát vásárolta meg. Az egységes vállalati kommunikáció területén a világ egyik vezető cégeként működő, 13 ezer alkalmazottat foglalkoztató vállalat 2007-ben 3,1 Milliárd eurós forgalmat produkált. Amint azt a Siemens magyarországi leányvállalatának sajtótájékoztatóján Jürgen Liss ügyvezető igazgató elmondta, hogy a tranzakció kapcsán 350 millió eurót fektetnek a vegyes vállalatba – a kutatásra és fejlesztésre amúgy is betervezett összegeken és a normál üzletmenet keretén belül felmerülő kiadásokon felül. A beruházások és a további párhuzamos akvizíciók nyomán létrejött SEN Group célja a Siemens Enterprise Communications értékesítési szervezetének lehető legnagyobb mértékű hasznosítása, a további terjeszkedés elősegítése, valamint a vállalat átalakításának ösztönzése hardverszállítóból szoftvereket és szolgáltatásokat kínáló társasággá.

•

A **Sun Microsystems**, Inc. megjelentette a Solaris operációs rendszer legújabb, 10 10/08 számú verzióját. Az új verzió a Solaris 10 operációs rendszer alapvető erősségeire építve segít az ügyfeleknek maximalizálni az erőforrások kihasználását és a rendszerteljesítményt, kezelni az összetett adatközpontokat, illetve fenntartani az üzletvitel folytonosságát és csökkenteni a költségeket. A Solaris 10 10/08 verzió számos termékfrissítést és fejlesztést tartalmaz, amelyek közül több az OpenSolaris közösség munkájának köszönhető.

•

A HP újgenerációs adattárolói virtualizációs megoldása a **HP StorageWorks** SAN Virtualizációs Szolgáltatási Platform (SVSP) néven kerül forgalomba mely, tovább növeli az adattárolók hatékonyságát és egyszerűsíti a különböző adattároló rendszerek menedzsmentjét. A HP SVSP egy új, hálózat-alapú adattároló platform, amely egyesíti a HP és akár más gyártótól származó tárolórendszerek erőforrásait. Az új platform együttműködik a HP Storage Works Modular Smart Array-jel (MSA) és az Enterprise Virtual Array-jel (EVA), valamint számos más gyártó megoldásával, így központosítja a virtuális SAN környezet menedzsmentjét.

A kvantumkriptográfia infokommunikációs alkalmazásai

GYÖNGYÖSI LÁSZLÓ, IMRE SÁNDOR

Budapesti Műszaki és Gazdaságtudományi Egyetem, Híradástechnikai Tanszék
{gyongyosi, imre}@hit.bme.hu

Kulcsszavak: kvantumkriptográfia, kvantumkommunikáció, kvantuminformatika

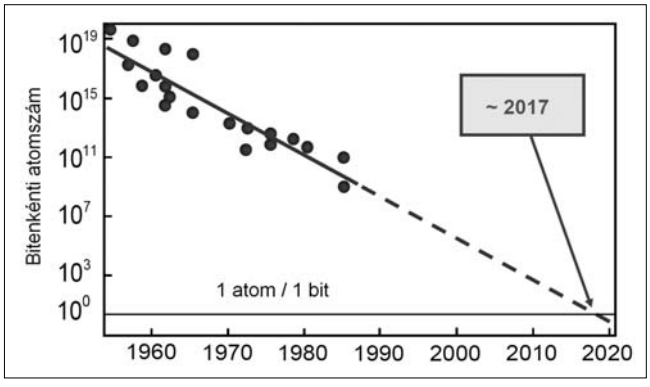
A Moore-törvény alapján, 2017-re várhatóan egy bit információt egy atom tárol majd, így már néhány éven belül elérkezhet a kvantuminformatika világa. A kvantumszámítógépek megjelenésével a jelenlegi titkosítási módszerek nagy része veszélybe kerül. A kvantumszámítógép működése a kvantumelméletre épül, és alkalmas arra, hogy minden mai modern, feltörhetetlennek vélt kódot másodpercek alatt feltörjön. A rejtjelezők ezért már ma olyan módszeren dolgoznak, amely a kvantumszámítógéppel szemben is képes megőrizni a titkokat. Az új, abszolút feltörhetetlen kód: a kvantumkriptográfia. A kvantumkriptográfia alapú titkosítást már a gyakorlatban is megvalósították, laboratóriumi és szabadtéri körülmények között is. A protokoll működőképes, és valóban egy olyan titkosítási módszert nyújt, amely elméletileg sem törhető fel.

1. Bevezető

A Moore-törvényt figyelembe véve – amely szerint a számítógépek bonyolultsága exponenciálisan nő az időben – 2017-re várhatóan egy bit információt egyetlen atomban fogunk kódolni. Ahogyan azt tehát a számítástechnika mai helyzetéből jósolni lehet, a hagyományos technológiák hamarosan elérik a végső fizikai határokat, az elemi műveleteket egyetlen elektron hajtja majd végre. A kvantumeffektusok így már néhány éven belül olyan nagymértékű hatást gyakorolhatnak a számításokra, amely alapvetően befolyásolja, meghatározza a későbbi fejlesztések irányvonalát.

A *kvantumszámítógépek* megjelenésével a jelenlegi titkosítási módszerek nagy része veszélybe kerül. A napjainkban alkalmazott nyílvános kulcsú titkosító algoritmusok biztonsága ugyanis nehéznek vélt matematikai problémákra, például a faktorizáció nehézségére épül, melyek megoldásához szükséges lépésszám *exponenciálisan* növekszik az input méretének növekedésével. A kvantumszámítógép azonban ezeket a nehéz problémákat polinomiális lépésszámmal oldaná meg, és így hatékonyan feltörhetővé tenné a mai rejtjelező algoritmusokat.

1. ábra A számítástechnika fejlődési üteme



A Peter Shor által 1994-ben közzétett *kvantumalgoritmus* például *polinomiális* idő alatt képes megoldani a faktorizáció problémáját. Az algoritmus egyrészt azon alapul, hogy a faktorizációval szemben, a legnagyobb közös osztó megtalálására ismert gyors klasszikus algoritmus is, másrészt pedig a faktorizációs probléma visszavezethető a periódus keresési feladatra. A kvantumalgoritmus *kvantum-regisztereken* végzi el a prímtényezőkre bontást, a faktorizálandó szám maradékosztályainak periodicitási tulajdonságát kihasználva. A kvantumalgoritmussal, egyetlen kvantumszámítógép segítségével *másodpercnyi* időtartam alatt feltörhető azon kód, mely napjaink klasszikus számítógép-hálózatainak együttesen is több hónapig tartana [6].

A jövőben így olyan titkosítási módszereket kell találnunk, amely megvédenek bennünket a kvantumszámítógépek támadásától. A *kvantumkriptográfia* lehet az a titkosítási eljárás, amely ellenáll a kvantumszámítógépek hatalmas számítási teljesítményének is. A kvantumkriptográfia az *egyszeri kulcsos módszert* (OTP, One Time Pad) használja az adatok titkosítására, mely, mint ismeretes, *elméletileg sem törhető fel,* szemben a napjainkban alkalmazott titkosítási eljárások *gyakorlati feltörhetetlenségével.*

2. A kvantumkriptográfia megszületése

Amíg a rejtjelfejtők a kvantumszámítógépre várnak, addig a rejtjelezők olyan módszeren dolgoznak, amely a kvantumszámítógéppel szemben is képes megőrizni a titkokat, azaz *még kvantumszámítógéppel sem törhető fel.* A módszer a kvantumelméletre épül, ugyanúgy, mint a kvantumszámítógép. Az abszolút feltörhetetlen kód a kvantumkriptográfia.

A kvantumkriptográfia története a hatvanas évek végén kezdődött. Stephen Wiesner ekkor vetette fel a kvantumpénz fogalmát. A kvantumpénz elméleti alap-

ja a fotonok fizikája volt. Wiesner ötletét nem valósították meg, azonban egy régi barátja, Charles Bennett figyelmét felkeltette. Wiesner odaadta a kvantumpénzzel kapcsolatos jegyzeteit Bennettnek. Bennettnek azonnal megtetszett az ötlet. Sokat gondolkozott azon, hogyan lehetne a gyakorlatban is megvalósítani. A kvantumpénz ötletét megosztotta Gilles Brassarddal, a Montreali Egyetem számítógéptudósával. Többször megvitatták a dolgot, s arra a következtetésre jutottak, hogy Wiesner ötletét a kriptográfiában lehetne hasznosítani. Wiesner kvantumpénze azért biztonságos, mert a bankjegyekbe zárt fotonok polarizációját lehetetlen megállapítani.

Bennett és Bassard a kódolt üzenetek polarizált fotonok formájába öntésén, s azok ílymódon történő továbbításán kezdett el gondolkodni [1,2]. Az így küldött kódüzeneteket elméletileg a támadó, Eve nem tudná elolvasni, s ezáltal megfejteni sem [3].

2.1. Kvantumrendszerek jellemzése

A fizikai rendszerek időfejlődését a klasszikus fizikában a Hamilton-féle kanonikus egyenletek írják le, míg a kvantumrendszerek időfejlődésének leírására a Schrödinger-egyenlet szolgál. A Schrödinger-egyenletben egy kvantumrendszer kezdeti $|\psi(0)\rangle$ állapotából történő reverzibilis időfejlődését a $|\psi(t)\rangle = U_t\,|\psi(0)\rangle$ transzformáció szabja meg, ahol U_t az időfejlődést leíró evolúció-operátor. Az U_t operátor mindig *unitér*, így minden kvantumtranszformáció unitér leképzést realizál a kvantumrendszeren belül, a végrehajtott transzformáció pedig logikailag reverzibilis. Egy kvantumrendszer állapotterét hullámfüggvények Hilbert-tereként ábrázoljuk. A Hilbert-tér egy $|\psi(t)\rangle$ egységvektora reprezentálja a kvantumrendszer egy adott időpontbeli állapotát.

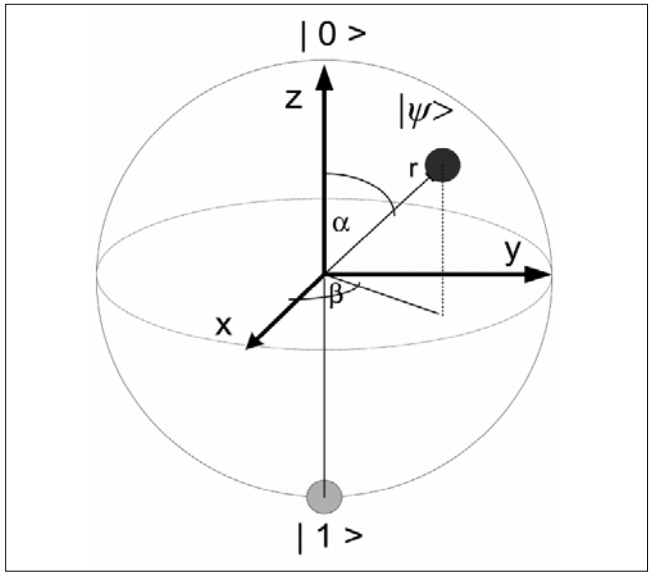
A kvantumállapotokat, valamint a rájuk ható transzformációkat leírhatjuk vektorokkal vagy mátrixokkal, de célszerűbb a Dirac-féle *„bra/ket”* szimbólumok használata. A $|x\rangle$ jelölés egy *„ket”*, ami egy *oszlopvektornak* felel meg, míg a $\langle x|$ jelölés egy *„bra”*-t, azaz egy *sorvektort* jelent, amely éppen a $|x\rangle$ *„ket”* adjungáltja. A *„bra/ket”*-ek leírhatók vektorokkal is:

$$|0\rangle = \begin{pmatrix}1 & 0\end{pmatrix}^T \text{ és } |1\rangle = \begin{pmatrix}0 & 1\end{pmatrix}^T.$$

A mikrorészecskék tulajdonságainak magyarázásakor a részecske állapotváltozásait *komplex számokkal,* valószínűségi amplitúdókkal írjuk le [3].

2.1.1. A kvantumbit

Egy klasszikus rendszeren belüli, „klasszikus értelmezésű” bit, a két logikai állapot között nem vehet fel értékeket. Ezzel ellentétben, a *kvantumbitek* lehetnek a 0 és 1 állapot között is, amelyet az $\alpha|0\rangle + \beta|1\rangle$ állapotvektorral írhatunk le, ahol α, β a $|0\rangle$ és $|1\rangle$ bázisállapotokhoz tartozó *valószínűségi amplitúdók*. A kvantumállapot mérése során, az α, β valószínűségi amplitúdóknak megfelelő valószínűséggel kerül a rendszer a $|0\rangle$ vagy $|1\rangle$ kimeneti állapotok valamelyikébe. A valószínűségi amplitúdókra fennáll $|\alpha|^2 + |\beta|^2 = 1$ normáltsági feltétel, az egyes kimeneti állapotokhoz tartozó mérési valószínűségek pedig ezen valószínűségi amplitúdók négyzetével jellemezhetőek. Így, az $\alpha|0\rangle + \beta|1\rangle$ állapotú kvantumbiten végrehajtott mérés eredménye $|\alpha|^2$ valószínűséggel $|0\rangle$, illetve $|\beta|^2$ valószínűséggel $|1\rangle$ lesz. A kvantumbitek állapotának szemléltetésére a Bloch-gömböt használjuk. A Bloch-gömb egy-egy feléhez a kvantumbit egy-egy bázisállapota tartozik. Általánosan, a gömb északi fele a $|0\rangle$ állapotnak felel meg, a déli fele pedig $|1\rangle$-nek, a többi pont pedig ezen két bázisállapot *szuperpozíciója*.



2. ábra A kvantumbit szemléltetése Bloch-gömbön

A Bloch-gömbi reprezentáció során két fontos szöget különböztetünk meg. Az α szög a $|0\rangle$ és $|1\rangle$ vektor arányát jelenti az összegben, – azaz az adott állapothoz tartozó valószínűségi amplitúdókat – míg a β szög a *relatív kvantum fázist* jelöli. Az állapotvektor a Bloch-gömbön bárhol elhelyezkedhet, így a kvantumbit a felvehető *végtelen sok* állapot közül bármelyikben lehet. Az α szög az $\vec{r}$ vektor és a *z* tengely által bezárt szög, a β szög pedig a vektor irányát határozza meg.

2.2. Foton, mint kvantumbit

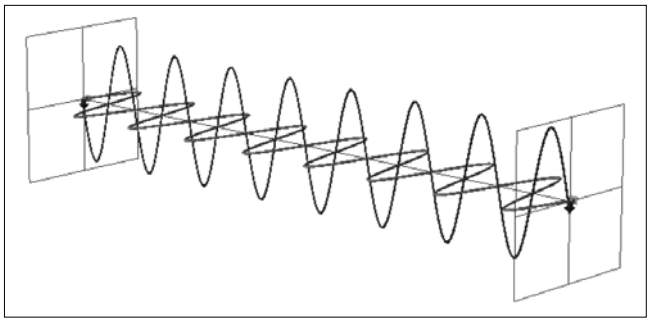
A kvantumbitek megvalósíthatóak fotonokkal is, hiszen a fotonok polarizációs szögei megfeleltethetőek a kvantumbitek $|0\rangle$ és $|1\rangle$ bázisállapotainak. A fotonok horizontális $|h\rangle$ és vertikális $|v\rangle$ polarizációs állapotait így a következőkben a $|0\rangle$ és $|1\rangle$ bázisértékekkel azonosítjuk. A kvantumbitként alkalmazott foton $|\psi\rangle$ állapotát, azaz polarizációját is leírhatjuk a $|\psi\rangle = a\cdot|\rightarrow\rangle + b\cdot|\uparrow\rangle$ állapotvektorral, ahol a $|\rightarrow\rangle, |\uparrow\rangle$ jelölés alatt a *vízszintes,* illetve *függőleges* polarizációt értjük. A klasszikus bitekhez hasonlóan, amelyek a 0 vagy 1 állapotban lehetnek, a fotonok is felvehetik a $|0\rangle$ vagy $|1\rangle$ állapotot, vagy akár e két állapot lineáris kombinációjának megfelelő $|\psi\rangle = a\cdot|\rightarrow\rangle + b\cdot|\uparrow\rangle$ *szuperpozíciós* állapotot. A foton polarizációját a $|\psi\rangle$ irányvektor jelképezi a függőleges és vízszintes polarizációk bázisában.

A kvantummechanika mérési posztulátuma szerint egy méréshez mindig tartozik egy *ortonormált* bázis,

amely mérés a mérendő $|\psi\rangle$ kvantumállapotot az ortonormált bázis egyik bázisvektorába transzformálja át. Így, az $|\psi\rangle = a \cdot |\rightarrow\rangle + b \cdot |\uparrow\rangle$ polarizációjú foton, rektilineáris $\{|\rightarrow\rangle, |\uparrow\rangle\}$ bázisú mérési eredménye $|a|^2$ valószínűséggel $|\rightarrow\rangle$, valamint $|b|^2$ valószínűséggel $|\uparrow\rangle$ lesz.

A fotonok esetében kétféle bázisban kódoljuk, illetve dekódoljuk a kvantumállapotokat, a vízszintes-függőleges bázisállapotokat tartalmazó $\{|\rightarrow\rangle, |\uparrow\rangle\}$ *rektilineáris*, illetve az átlósan polarizált kvantumállapotokat tartalmazó *diagonális* $\{|\nearrow\rangle, |\nwarrow\rangle\}$ bázisban. A természetes fény rengeteg atom, illetve molekula által kibocsátott sugárzásból áll, azonban a síkban poláros fényben az elektromos térerősség vektor egyetlen síkban halad.

A *3. ábrán* láthatjuk, hogy az elektromos térerősség vektor a *z* terjedési iránnyal merőlegesen, az *xy* síkban halad.



3. ábra A fény polarizációja

3. A kvantumkriptográfia működési elve

A kvantumkriptográfia általános modelljében a két kommunikációs fél, Alice és Bob egy kétirányú *klasszikus*, valamint egy egyirányú, Alice-tól Bob felé irányuló *kvantum-csatornán* keresztül állnak kapcsolatban egymással [1]. A kvantumcsatorna felhasználásával a részecskék kvantumállapotukat megőrizve küldhetők át. A csatornák nem megbízhatóak, hiszen a klasszikus csatorna lehallgatható, a kvantumcsatornán pedig a támadó, Eve elfoghat és újraküldhet részecskéket [2].

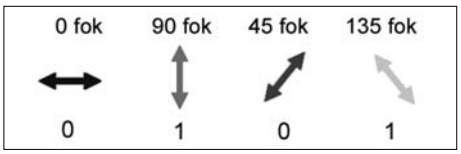
A kvantumkriptográfia segítségével azonban Alice és Bob képesek megegyezni egy olyan kulcsban, amit rajtuk kívül senki más nem ismer. A közös kulcs kialakítását már sikeresen megvalósították, 1997-ben Highes és társai 24 km-es távolságon mutatták be a protokoll működését, egy szabványos üvegszálas optikai kábelen keresztül. 2002-ben pedig 10 km-es távolságon, a légkörben is sikerült megvalósítaniuk a kísérletet.

3.1. A titkos kulcs kialakítása

A kulcskialakítás első szakaszában Alice $\langle\updownarrow,\leftrightarrow\rangle$ rektilineáris (vízszintes-függőleges) és $\langle\nearrow,\nwarrow\rangle$ diagonális (átlós) polarizációs séma véletlenszerű váltogatásával küld egy, egyesekből és nullákból álló véletlenszerű fotonfüzért.

A protokoll általános modelljét a *4. ábrán* láthatjuk.

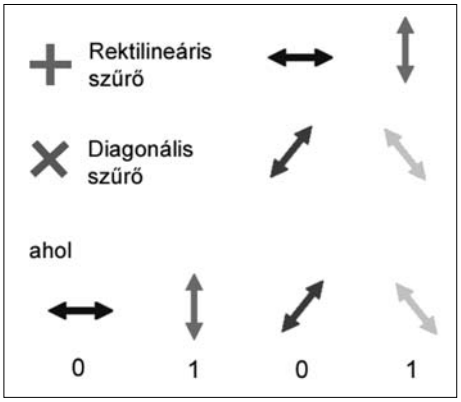
Az 1-eseket és 0-kat bizonyos *polarizáltságú* fotonok helyettesítik. A fotonok polarizációs szögeihez rendelt logikai értékeket az *5. ábra* mutatja.



5. ábra A fotonokhoz tartozó bináris értékek

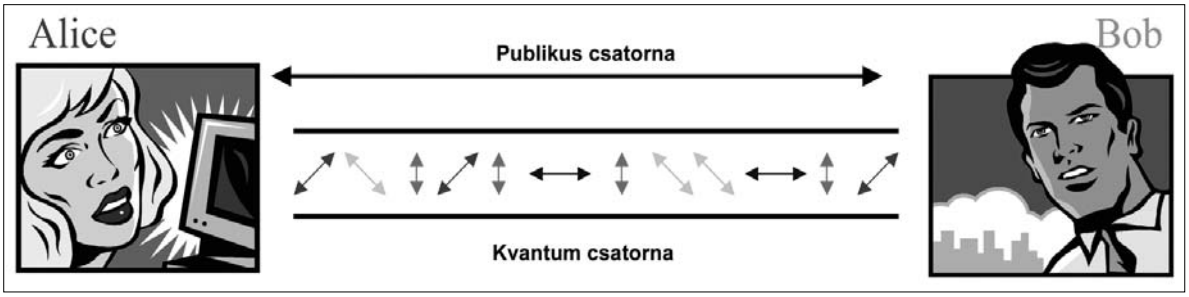
A vízszintesen polarizált foton $\leftrightarrow$ logikai 0-át, míg a függőlegesen polarizált $\updownarrow$ foton a logikai 1-et jelenti. Hasonlóképpen, az átlósan polarizált fotonok közül a 45 fokos szögben polarizált $\nearrow$ foton jelenti a 0-át, míg a 135 fokos $\nwarrow$ a logikai 1-et.

Arra, hogy Alice fotonokkal helyettesítse az egyeseket és nullákat, két módszert alkalmazhat, a $\langle\updownarrow,\leftrightarrow\rangle$ elemű rektilineáris, illetve a $\langle\nearrow,\nwarrow\rangle$ elemű diagonális módszert. A $\langle\updownarrow,\leftrightarrow\rangle$ rektilineáris módszer esetén a logikai 0 értéket a $\leftrightarrow$, a logikai 1-et pedig a $\updownarrow$ polarizációs állapot reprezentálja. A $\langle\nearrow,\nwarrow\rangle$ diagonális, azaz átlós módszer esetében a logikai nullát a $\nearrow$, az 1-et pedig a $\nwarrow$ kvantumállapot jelenti. Az egyes bázisokhoz tartozó polarizációs állapotokat a *6. ábrán* láthatjuk.



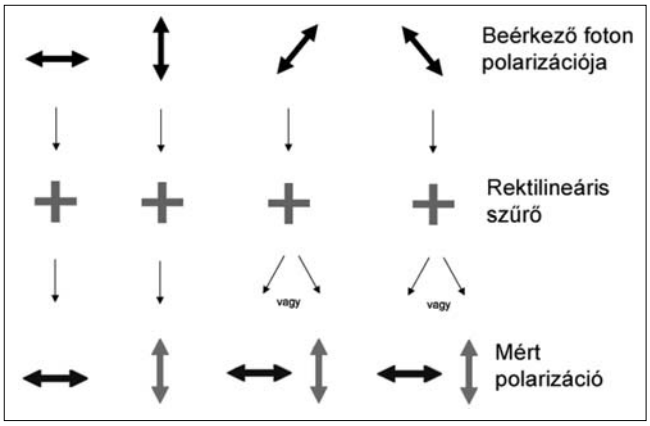
6. ábra A rektilineáris és diagonális szűrővel előállítható fotonok és azok értékei

Bobnak, a dekódoló oldalon minden egyes foton polarizációját meg kell állapítania, tehát minden egyes alkalommal el kell döntenie, hogy hogyan állítsa be polárszűrőjét. Bob azonban nem tudhatja, hogy melyik fotont milyen módszerrel küldte Alice, ezért az *esetek fe-*



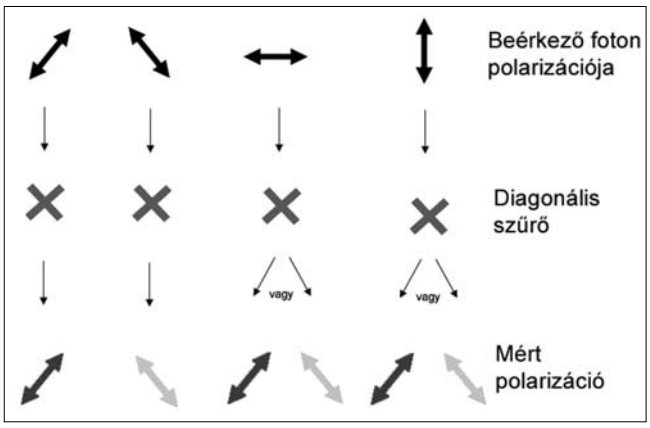
4. ábra A kvantumkriptográfia általános modellje

lében csak tévesen tudja megállapítani a polarizációt. Bob a $\langle \updownarrow, \leftrightarrow \rangle$ bázisban tökéletesen felismeri a függőlegesen és vízszintesen polarizált fotonokat, az átlósakat azonban nem. A $\langle \nearrow, \nwarrow \rangle$ bázisban kódolt kvantumbiteket így *véletlenszerűen* függőlegesnek vagy vízszintesnek azonosítja. A rektilineáris bázisban végrehajtott mérések lehetséges kimeneteleit a *7. ábrán* foglaltuk össze.



7. ábra
Lehetséges mérési eredmények rektilineáris szűrő esetében

Hasonlóképpen, ha $\langle \nearrow, \nwarrow \rangle$ szűrőt alkalmaz, akkor az átlósan polarizált fotonokat tökéletesen felismeri, de a vízszintes és függőleges fotonokat helytelenül átlós polarizáltságúaknak azonosítja, *véletlenszerű logikai értékekkel*. A diagonális bázisú mérések lehetséges kimeneteleit a *8. ábrán* láthatjuk.



8. ábra
Lehetséges mérési eredmények diagonális szűrő esetében

Egy bináris üzenet elküldésekor Alice a $\langle \updownarrow, \leftrightarrow \rangle$ rektilineáris és $\langle \nearrow, \nwarrow \rangle$ diagonális módszert véletlenszerűen váltogatja.

Legyen példánkban az átküldendő üzenet a következő, 12 bites bináris sztring: *„011010111010”*. Az Alice által elküldött kvantumállapotok vételekor Bobnak meg kell állapítania a fotonok polarizációját. Mivel nem tudja, hogy Alice melyik fotonnál milyen polarizációs sémát alkalmazott, így Bob véletlenszerűen váltogatja a rektilineáris és diagonális detektorát. Törvényszerűen időnként eltalálja melyik a helyes, másszor nem, ekkor azonban rosszul értelmezheti Alice fotonját.

Abban az esetben, ha a csatornát nem hallgatta le Eve, akkor Bob a *9. ábrán* látható kulcshoz juthat.

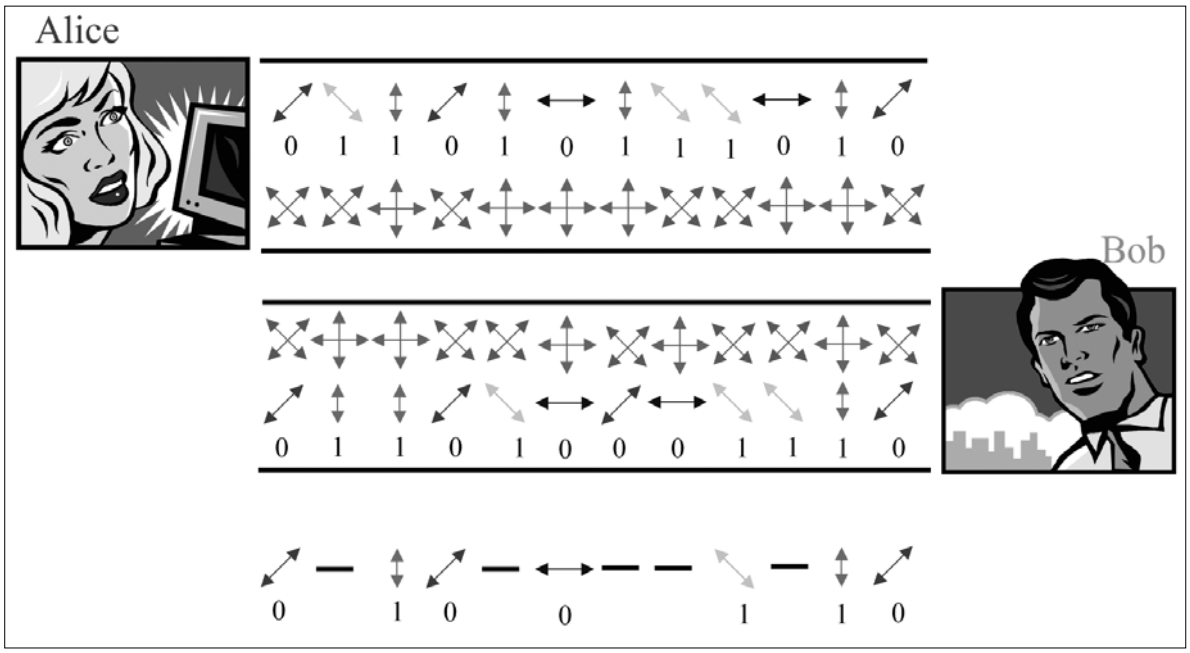
Amennyiben Bob eszerint az ábra szerint választotta meg detektorait, akkor Alice *„011010111010”* üzenetét *„011010001110”*-nak dekódolhatta. Azaz, ha Bob diagonális szűrőt használ az első foton vizsgálatához, akkor helyes eredményt kap, azaz $\nearrow$-t. Viszont, ha Bob rektilineáris szűrőt használ az első foton vizsgálatához, akkor a $\nearrow$ polarizációjú fotont tévesen $\updownarrow$ vagy $\leftrightarrow$ polarizáltságúnak fogja értelmezni. Ha $\leftrightarrow$ polarizáltságúnak értelmezi, akkor az nem okoz problémát Bobnak, hiszen szintén logikai nullát reprezentál. Abban az esetben, ha Eve *nem próbálta meg lehallgatni a csatornát,* akkor biztosak lehetünk abban, hogy ahol Bob azonos polarizációjú szűrőt választott, ott ugyanazon az értéket kapja, mint amit Alice elküldött.

A következő szakaszban, a helyzet tisztázása érdekében Alice a publikus csatornát használva felhívja Bobot, s közli vele, hogy milyen polarizációs sémát használt az egyes fotonokon. A fotonok pontos állapotait azonban nem árulja el. Alice így például elmondhatja Bobnak, hogy az első fotont a $\langle \nearrow, \nwarrow \rangle$ séma szerint kódolta, azonban azt már *nem közli,* hogy amit küldött az $\nearrow$ vagy $\nwarrow$ állapotú foton volt-e. Ezt követően, Bob közli Alice-szel a helyesen dekódolt kvantumbitek *sorszámait*. Ezen pozíciókban Bob helyesen vizsgálta be a fotonokat, így helyesen állapította meg azok logikai értékeit is. Alice és Bob így figyelmen kívül hagyhatja azon fotonokat, amelyeknél Bob rosszul választott bázist, s a továbbiakban csak a helyesen értelmezett fotonokkal foglalkoznak. Az előbbi példában a polárszűrők sorrendje *„X++XX+X+XX+X”* volt, így a megtartott bitfüzérünk *„0100110”* lett.

A kulcsmegosztáshoz, s ezáltal a kvantumkriptográfia megvalósításához három *előkészítő* szakasz szükséges. A három szakasz tehát lehetővé teszi Alice-nek és Bobnak, hogy megállapodjanak egy normál számsorozatban. A kialakított üzenet egyik meghatározó tulajdonsága azonban, hogy az teljesen véletlenszerű, az üzenet ugyanis Alice teljesen véletlenszerű logikai érték illetve detektorválasztásából generálódott. Maga a számsorozat pedig nem hordoz tényleges üzenetet, mindössze egy *véletlenszerű kulcs,* amely teljesen véletlenszerűen kialakított füzért használjuk az *egyszer használatos kód* (OTP) szimmetrikus kulcsaként.

3.2. Eve megjelenése

Az előbbi példánál nem feltételeztük azt, hogy Eve hallgatózna, így nem kaphatott Bob téves eredményt *megfelelő bázisválasztás* esetében sem. Tekintsük most azt az esetet, amikor Eve hallgatózik a kommunikációban. Eve a kvantumcsatornán keresztül próbál hozzájutni a titkos kulcsunkhoz, azonban Eve sem tudhatja azt, hogy Alice milyen polarizáltságú szűrőt alkalmazott fotonjai elküldéséhez. Így például, ha az előző példában küldött üzenet esetén Eve $\langle \nearrow, \nwarrow \rangle$ szűrőt használ az első foton vizsgálatához, akkor helyes eredményt kap, azaz $\nearrow$-t. Viszont, ha rektilineáris szűrővel próbálja meg megállapítani a kvantumbit állapotát, akkor a $\nearrow$ polari-

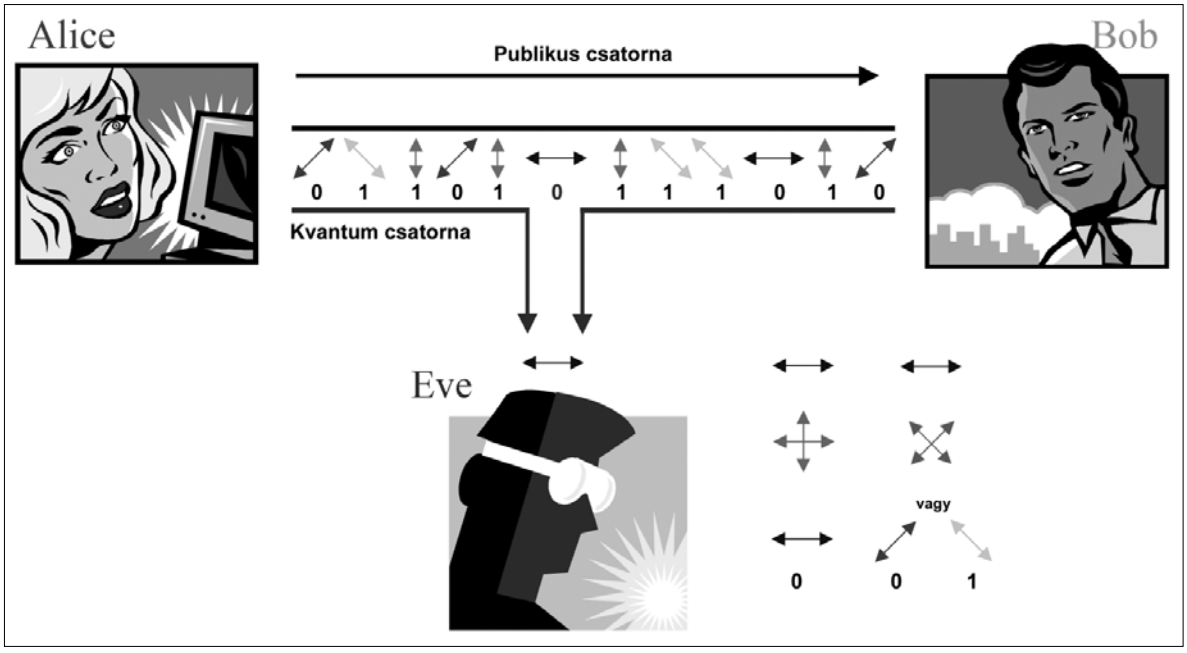


9. ábra Alice és Bob detektor-egyeztetést követően kialakított kulcsa

zációjú fotont tévesen ↕ vagy ↔ polarizáltságúnak fogja értelmezni. Amennyiben ↔ polarizáltságúnak értelmezi, valamint Bob a ⟨↗,↘⟩ bázisú szűrő helyett a téves ⟨↕,↔⟩ bázist választja a kvantumállapot detektálásához, akkor azzal ténylegesen nem okozna problémát, hiszen ezen polarizációs állapot is logikai nullát reprezentál. Azonban, ha ↕-nek értelmezi – *annak eredeti értékét megváltoztatva* – már logikai egyet továbbít a kvantumcsatornán keresztül. A téves detektorválasztásokat azonban a felek kiszűrik, így ezen bit mindenféleképpen kikerül a végleges kulcsból.

Eve helyzetét nagymértékben megnehezíti az, hogy minden egyes fotont csak *egyetlen egyszer* vizsgálhat. A fotont nem oszthatja két fotonra, és nem vizsgálhatja mindkét bázisban egyszerre. Eve így nem lehet biztos abban, hogy az elfogott kódszöveg pontos-e, ennek következtében nincs reménye a megfejtésére sem. Eve megpróbálhatja bemérni az Alice által elküldött fotont, de nem tudja, hogy rektilineáris vagy diagonális bázist használjon-e. Így az esetek *felében* rosszul dönt. Ekkor azonban még mindig pontosan olyan helyzetben van, mint Bob, aki szintén csak az esetek felében találja el a helyes bázist. Ezt követően azonban Alice közli Bobbal, hogy melyik fotonnál melyik lett volna a helyes detektor, így csak azok a fotonok kerülnek a kulcsfüzérbe, amelyeket Bob jól mért be. Eve-en azonban ez nem segít, mivel ezeknek a fotonoknak a felénél nem megfelelő detektort használt, ezért a kulcsot alkotó fotonok felének polarizációját is rosszul méri be.

A kvantumkriptográfia így lehetővé teszi, hogy Alice és Bob megállapodjon egy kulcsban, amely titkos kulcsot Eve csak hibásan lehet képes beazonosítani. Eve jelenléte a kvantum-kommunikációban pedig egyértelműen detektálható, a kvantumcsatornán okozott *irrever-*



10. ábra Eve hallgatózik a kvantum-csatornán

zibilis zavarok következtében. A mérési próbálkozásokkal Eve megváltoztatja a foton polarizációját, s ezen polarizációváltozások nyilvánvalóak Alice és Bob számára [7].

Abban az esetben például, ha Alice ↔-t küld, Eve pedig a nem megfelelő ⟨↗,↖⟩bázisú detektorral vizsgálja, akkor a detektor arra kényszeríti a beérkező ↔ állapotú fotont, hogy ↗ vagy pedig ↖ állapotban lépjen tovább. Ha Bob a maga ⟨↕,↔⟩ bázisú detektorával megvizsgálja az átalakított fotont, akkor lehetséges, hogy az Alice által küldött ↔-t kapja, de az is lehetséges, hogy ↕-ként fogja értelmezni, ami helytelen. Alice tehát egy vízszintesen polarizált fotont küldött, amihez Bob a megfelelő detektort használta, mégis rosszul mérte be az elküldött fotont. Ha tehát Eve rossz detektort választ, akkor *„csavar"* bizonyos fotonokon, amivel a vevőt esetenként hibára késztetheti, még akkor is, ha megfelelő detektort használ. Azonban ha Alice és Bob végez egy rövid ellenőrzést, akkor ezek a hibák kiküszöbölhetőek.

Alice-nek és Bobnak meg kell győződniük arról, hogy a kialakított füzér azonos-e. A füzér azonosságáról megbizonyosodni a legegyszerűbben úgy lehetne, ha Alice felhívná Bobot és közölné vele a sorrendet. Ekkor azonban ha Eve elfogja a hívást, hozzájuthatna a teljes kulcshoz. A teljes füzér egyeztetése azonban felesleges, ugyanis elég, ha Alice véletlenszerűen kiválaszt bizonyos mennyiségű elemet a bitfüzérből. Ha Bob ezeket helyesnek nyilvánítja, akkor elenyészően alacsony a valószínűsége annak, hogy Eve lehallgatta az eredeti adást.

Miután Alice és Bob nyíltan egyeztette a számokat, ezeket elvetik és kettejük közös, bináris számjegyekből álló egyszeri kulcsa az egyeztetésnél felhasznált bitek számával csökken. Amennyiben Alice és Bob eltérésre bukkan a vizsgált bitek között, akkor tudni fogják azt, hogy Eve hallgatózik. Ekkor az egész kulcsot kénytelenek eldobni.

3.2.1. Téves bázisú lehallgatás következményei

A következőkben tekintsük azt az esetet, amikor Eve téves polarizációjú szűrővel próbálja meg bemérni az Alice által küldött ↔ vízszintes polarizáltságú fotont. Eve ⟨↕,↔⟩ bázis helyett ⟨↗,↖⟩ bázist használ, miáltal módosítja a Bob felé továbblépő foton polarizációját. Példánkban legyen a *módosított foton* polarizációja↖. Ebben az esetben, ha Bob megfelelő bázisú detektort választ – tehát azt, amit Alice eredetileg is használt – akkor véletlenszerűen ↔-t vagy ↕-t kap. A *11. ábrán* azon esetet modelleztük, amikor a vevő a módosított polarizációjú fotont ↔-nak méri.

Ez esetben Bob nullát kapott, ami a detektoregyeztetésnél sem derül ki, ugyanis mindketten azonos polarizációjú szűrőt választottak és a kapott logikai érték is megegyezik a küldöttel. A kulcs tehát *0100110* lesz. Most nézzük azt, ha Bob tévesen 1-et kap, azaz a↖ polarizációjú fotont a rektilineáris szűrővel ↕-nek méri. A mérések kimenetele ekkor a *12. ábrán* látható módon alakul.

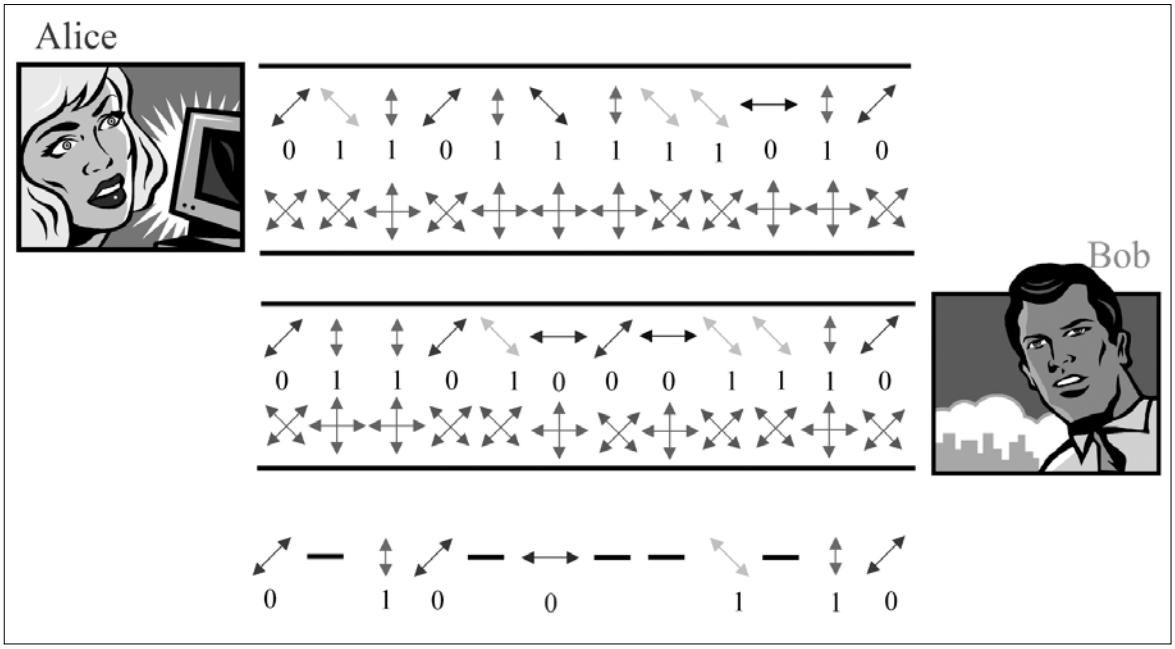
Ebben az esetben az ellenőrző szakaszban egyértelműen fény derül arra, hogy azonos bázisú detektorhasználat esetén eltér a küldött és mért érték.

3.3. A protokoll lépéseinek összefoglalása

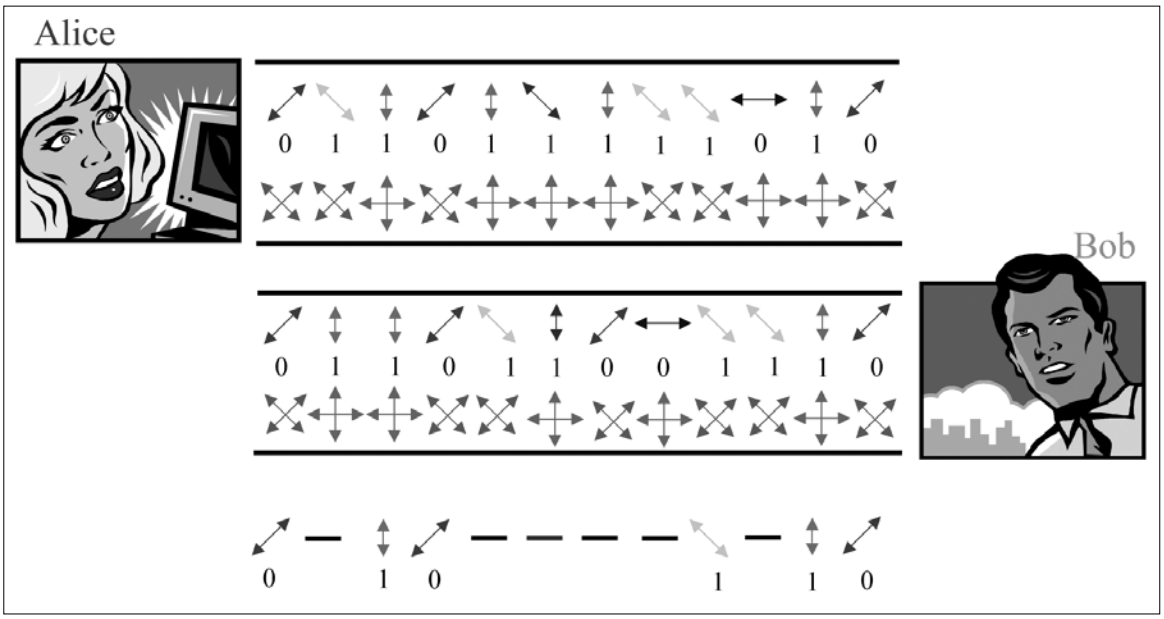
A kvantumkriptográfia tehát egy olyan titkosítási módszer, amely *fizikailag* teszi lehetetlenné az Alice és Bob közötti kommunikáció pontos lehallgatását. A kvantumkriptográfia segítségével Alice és Bob teljesen titokban megállapodhat egy egyszeri kulcsban, s a továbbiakban ezen kulccsal kódolják üzeneteiket [1].

Összegzésként a módszer öt fő lépése:

1. *Alice fotonfüzért küld Bobnak, aki ezt bevizsgálja.*
2. *Alice közli Bobbal, hogy az érkező fotonoknál melyik esetben választotta a megfelelő detektort. A helyes mérés eredményét nem árulja el, ezért azt a beszélgetést Eve hiába hallgatja le.*



11. ábra Bob azonos polarizációjú detektor estén helyes eredményt kapott



12. ábra Bob ebben az esetben rossz értéket kapott

3. *Alice és Bob elveti a nem megfelelő méréseket, csak a többivel foglalkoznak.*
4. *Alice és Bob néhány számjegy egyeztetésével ellenőrzi a kulcs érintetlenségét.*
5. *Ha az ellenőrzés eredménye megfelelő, akkor az egyszeri kulccsal kódolhatják üzeneteiket. Ha nem, akkor viszont tudják, hogy Eve hallgatózott, így kénytelenek újrakezdeni a műveletet.*

3.4. A kvantumkommunikáció sikeres lehallgatásának valószínűsége

A protokollon belüli kvantumkommunikációban Eve csak bizonyos valószínűséggel lehet képes helyesen meghatározni a kvantumcsatornára küldött kvantumállapot bázisát, illetve a helyes polarizációs állapotot. A protokoll által alkalmazott kvantumkommunikáció tulajdonságaiból következően leírhatjuk a sikeres kvantumállapot azonosításához tartozó explicit valószínűségeket is.

Abban az esetben, ha Eve megpróbálja bemérni az Alice által küldött kvantumállapotot, az esetek 50%-ban rossz bázist fog választani, miáltal az adott fotont kicseréli egy másikra. Eve így 50%-os valószínűséggel kap azonos állapotot. Ugyanakkor 50%-os valószínűséggel rossz bázist használ, azaz a fotonokat csak újabb 50%-os valószínűséggel tudja helyesen beazonosítani. Vagyis, ha *rossz bázist* választ, akkor összesen $1/2 \cdot 1/2 = 1/4$, tehát 25% valószínűséggel *helyes bitet* mér, illetve 25% valószínűséggel rosszat. Ugyanakkor, ha Eve jó detektort választ, – aminek a valószínűsége 50% – akkor biztosan jó állapotot mér. Eve-nek így *összesen 75% esélye van* arra, hogy jó állapotot mérjen és 25% annak a valószínűsége, hogy rosszat. Így Eve közbeavatkozása, minden fotonnál 25% valószínűséggel hibát okoz a kommunikációban.

Miután Eve bemérte Alice elküldött kvantumállapotát, azt visszahelyezi a kvantumcsatornára, majd továbbítja Bob felé. A Bobhoz kerülő, bemért kvantumállapot logikai értéke így az esetek 25%-ban eltér az Eve által visszahelyezett bit értékétől, hiszen Bob szintén 25% valószínűséggel mást fog mérni, mint amit Eve küldött. Tehát a 75%-os helyes érték 25%-ban rossz eredményt fog szolgáltatni, amely így $3/4 \cdot 1/4 = 3/16 = 0.1875$, tehát 18,75%-nyi hibát jelent. Továbbá, a 25% hibásan továbbküldött fotont pedig 75% valószínűséggel rossznak fogja mérni Bob, amely ismét $3/16 = 0.1875$, azaz 18,75%-nyi hibát jelent a kvantum-kommunikációban.

Összefoglalva, Bob *18,75% + 18,75% = 37,5%*-ban nem azt fogja fogadni, amit Alice eredetileg küldött, függetlenül attól, hogy éppen azonos bázist használtak-e, hiszen Eve mindkettőjüktől függetlenül tudja csak megválasztania a bázisát. Egy ilyen jelentős hibát Alice és Bob könnyen észrevehet, ha néhány azonos bázissal átküldött bitet egyeztetnek a klasszikus csatornán, amit elvetnek a kulcsból. A kvantumbitek hitelesítése során, a felek kiszűrik a téves bázisú kiolvasásokat, majd a megmaradt, helyes bázisban dekódolt kvantumbitek helyességét ellenőrzik, azok bitértékeinek összehasonlításával. A téves bázisú dekódolások eltávolítása után kialakult bitsorozatban lévő különbségek pedig egyértelműen felfedik Eve közbeavatkozását.

3.5. A kvantumkód megszerzésének valószínűsége

A protokollon belüli kvantumkommunikáció lehallgatásával Eve, az esetek (3/4) részében juthat helyes eredményre, így egy N kvantumbites kvantumkód észrevétlen lehallgatásának valószínűsége $(3/4)^N$, ami elhanyagolható, ha N értéke megfelelően megválasztott.

Így, már egy igen alacsony értéket – mindösszesen 50 kvantumbitet – tartalmazó kulcs esetén, Eve mindösszesen $(3/4)^{50} = 0.0000005663216564269$ valószínűséggel lehet képes helyesen beazonosítani a küldött bitsorozatot. A protokollon belüli kvantumkommunikáció észrevétlen támadása így $(3/4)^N$ valószínűséggel maradhat csak felderítetlenül, ami elhanyagolhatónak tekinthető a gyakorlatban alkalmazott N értékek mellett. Eve támadása így nagyon nagy valószínűséggel kiszűrhető, hiszen a kvantumcsatorna támadása nem marad-

hat észrevétlen, mivel elkerülhetetlen hibákat okoz a kvantumkommunikációban. Amennyiben a felek nem találnak eltérést a helyes bázisban dekódolt kvantumbiteket tartalmazó bitfüzérben, akkor Alice és Bob biztos lehet abban, hogy az elküldött biteket nem szerezte meg senki.

A gyakorlatban a kvantumbitet kibocsátó forrás, az átviteli csatorna és esetlegesen maga az adattároló egység is szolgálhat zajforrásként a kvantumkommunikációban, miáltal romolhat a letisztázott bitsorozat tökéletes állapota. A hibával számolnunk kell, elkerülhetetlen, mindaddig, amíg az egy tolerálható érték alatt marad. Eve esetleg próbálkozhatna azzal is, hogy visszafogja magát és hallgatózását bizonyos küszöb alatt tartja, így abban reménykedve, hogy a terminálnak nem tűnik fel tevékenysége. Azonban a gyakorlatban ezen próbálkozása csak elhanyagolhatóan kis valószínűséggel segítené a kvantumbitek sikeres megszerzésében.

3.6. A kvantumkriptográfia működésének formális modellezése

A kvantumkriptográfia esetében a véletlenszerűség kitüntetett szerepet kap, hiszen az adó által elküldött foton bázisától és polarizációjától kezdve, a lehallgató szintén véletlenszerű mérési eredményein keresztül, egészen a vevő szintén véletlenszerűen bemért fotonjáig, a fő szerepet a véletlenszerű működés játssza. A protokoll során a $k \in \{0,1\}^N$, $N>0$ közös kulcs kialakítása egy dedikált kvantumcsatornán keresztül történik, bármiféle előzetes információcsere nélkül [5]. Miután a közös kulcs kialakult, Alice és Bob szimmetrikus kulcsú titkosítást alkalmazva kódolják üzeneteiket.

3.6.1. A protokoll lépéseinek formális leírása

Kommunikáció a kvantumcsatornán keresztül:

1) Alice generál egy *n* bitből álló, véletlenszerű bitsorozatot. A bitsorozat az átküldeni kívánt értékeket szimbolizálja. $A = \{a_i \mid 0 \le i \le n-1\} = [a_0, a_1, \ldots a_{n-1}]$.

2) Alice, az *A* halmazban lévő véletlenszerű bitekhez, szintén véletlenszerűen választ bázist. A bázis lehet rektilineáris, ekkor a $\beta = \langle \updownarrow, \leftrightarrow \rangle$ jelölést használjuk, illetve lehet diagonális, ekkor a $\beta = \langle \nearrow, \searrow \rangle$ jelölést alkalmazzuk. A bitekhez tartozó detektorsorrendet így a következőképpen jelölhetjük:

$$B = \{\beta_i \mid 0 \le i \le n-1\} = [\beta_0, \beta_1, \ldots \beta_{n-1}].$$

3) Alice, az *A* halmaz bitjeit a *B* halmazban lévő, indexnek megfelelő bázissal kódolja, majd a létrehozott kvantumbitet átküldi a kvantumcsatornán.

4) Bob minden egyes fotont egyenként detektál.

5) A fotonok detektálásához véletlenszerűen választ bázist, majd dekódolja a kvantumbitet. Bob minden egyes fotont β' bázisban dekódol, azaz vagy $\langle \updownarrow, \leftrightarrow \rangle$ bázist választ, vagy $\langle \nearrow, \searrow \rangle$ bázist.

A kapott dekódolt bitsorozat Bob oldalán tehát a következő:

$$A' = \{a'_i \mid 0 \le i \le n-1\} = [a'_0, a'_1, \ldots a'_{n-1}].$$

Kommunikáció a publikus csatornán keresztül:

1) *Bázisegyeztetési szakasz*

Ebben a szakaszban Bob közli Alice-el, hogy az *A'* dekódolt bitsorozatban, az adott a'_i bit detektálásához milyen β_i bázist választott.

2) *Hibás detektorválasztások kiszűrése*

Miután Bob közölte Alice-szel a választott detektorokat, Alice elárulja az adott a'_i bithez tartozó bázist. Alice ezek után az *A* sorozatból eldobja azokat a biteket, ahol különböző detektorokat választottak. Ugyanígy tesz Bob is a másik oldalon, így a kulcsban csak azon a_i, a'_i bitek maradnak, amelyekre teljesül a $\beta_i = \beta'_i$ összefüggés.

A kialakított kulcs az *elsődleges* kulcs. Az Alice és Bob oldalán kialakult kulcs jelölése legyen $k_{ELSŐDLEGES_A}$ és $k_{ELSŐDLEGES_B}$.

3) *Hibaellenőrzési szakasz*

Alice és Bob a kialakult elsődleges kulcsból, egy esetleges lehallgatás detektálása érdekében feláldoznak egy bizonyos nagyságú részt. Ezen kulcsrész jelölése legyen $k_{Ellenőrzés}$. Helyes bázisú mérés során kapott hibás bit esetén egyértelmű a lehallgatás ténye, így a kulcsot azonnal elvetik a felek. A sikeres ellenőrzés után kialakul az *egyeztetett* kulcs mindkét oldalon:

$$k_{EGYEZTETETT_A} = k_{ELSŐDLEGES_A} - k_{Ellenőrzés_A}.$$
$$k_{EGYEZTETETT_B} = k_{ELSŐDLEGES_B} - k_{Ellenőrzés_B}.$$

4) *Megerősítési szakasz*

A megerősítési szakasz célja a támadó által esetlegesen megszerzett információ további redukálása. Miután a feláldozott kulcsrészletben nem találtunk hibát, a kialakult egyeztetett kulcson még további, biztonsági ellenőrzéseket hajtunk végre. A kulcsból nem egy adott részt választunk ki, hanem véletlenszerűen egy-egy bitet.

Ebben a szakaszban Alice meghatározza a kommunikáció bithiba-arányát kifejező λ értéket, illetve a γ-vel jelölt biztonsági paramétert [1].

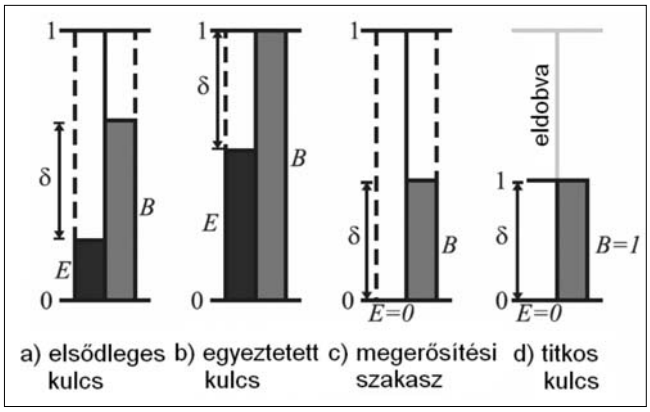
Miután ezen értékek kialakultak, Alice véletlenszerűen kiválaszt $r = n - \lambda - \gamma$ bitet az egyeztetett kulcsból. Azonban az ellenőrzés során ahelyett, hogy a konkrét bitértékeket átküldenék egymásnak, a *paritásokat* vizsgálják [2]. A folyamat során az *n* bites kulcsunkról készítünk egy $n - \lambda - \gamma$ bites lenyomatot, azaz egy véletlenszerű *f* hash függvényt alkalmazunk, a következő leképezést realizálva: $f : \{0,1\}^n \to \{0,1\}^{n-\lambda-\gamma}$, ahol $\gamma > 0$.

Ekkor, annak a valószínűsége, hogy az egyeztetés során egy esetleges lehallgató megszerzi a kulcsunkat, a következőképpen adható meg:

$$P \le \frac{2^{-\gamma}}{\ln 2}.$$

Az előző lépésben történt hibaellenőrzési eljárás nem biztosítja azt, hogy Eve nem juthatott hozzá a kulcsunk bizonyos részeihez. A megerősítési szakasz fő célja tehát ezen rejtett hibák kiszűrése.

A *13. ábrán* a kulcsok méreteinek változását láthatjuk az egyes ellenőrzési szakaszokban.



13. ábra
A kulcsméretek alakulása a kulcskialakítási szakaszokban

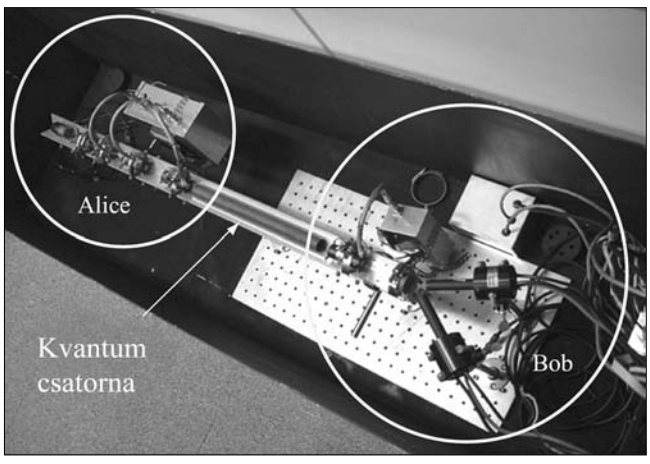
4. A kvantumkriptográfia gyakorlati megvalósítása

Wiesner tanulmánya tehát egy abszolút biztos kommunikációs rendszer létrejöttét segítette elő. A kriptográfusok lelkesen üdvözölték Bennett és Brassard kvantumkriptográfiáját, néhányan azonban úgy tartották, hogy a gyakorlatban megvalósíthatatlan. Bennett és Bassard azonban biztosak voltak a dolgukban. 1988-ban Bennett elkezdte összegyűjteni a kvantumkriptográfia megvalósításához szükséges eszközöket, segítségként maga mellé vette John Smolin kutatót.

Egy fénytől elzárt laboratóriumba vonultak és megpróbáltak polarizált fotonokat küldeni a helyiség egyik pontjáról a másikra. A fotonküldést egy Alice nevezetű számítógép irányította, a vételi oldalon pedig egy Bobnak keresztelt számítógép döntötte el, hogy melyik fotonhoz milyen detektort használ. Alice-nek és Bobnak sikerült fotonokat küldenie és fogadnia, elvetve a helytelenül bemért biteket, így megállapodva egy egyszeri kulcsban. Bennett kísérlete bebizonyította, hogy két számítógép képes abszolút titkosan kommunikálni egymással.

A gyakorlati megvalósítás azonban nem egyszerű feladat, mert a fotonok nehezen közlekednek. Ha Alice levegőn át küld egy bizonyos polarizációjú fotont, akkor az útjában álló levegő molekulái megváltoztatják polarizációjukat. Jobb megoldás a száloptika alkalmazása. A Genfi egyetemnek 1995-ben sikerült száloptika alkalmazásával megvalósítani egy Genf és Nyon közötti 23 km-es távolságon alkalmazni a kvantumkriptográfiát. A szabadtérben megvalósított eddigi legnagyobb távolság 23 km volt, ezt Münchenben hajtották végre. A szabadtéri kvantumkommunikáció azonban egyelőre sokkal lassabbnak bizonyul, mint az optikai szálas kivitelezés.

14. ábra
A kvantumkriptográfia első kísérleti megvalósítása



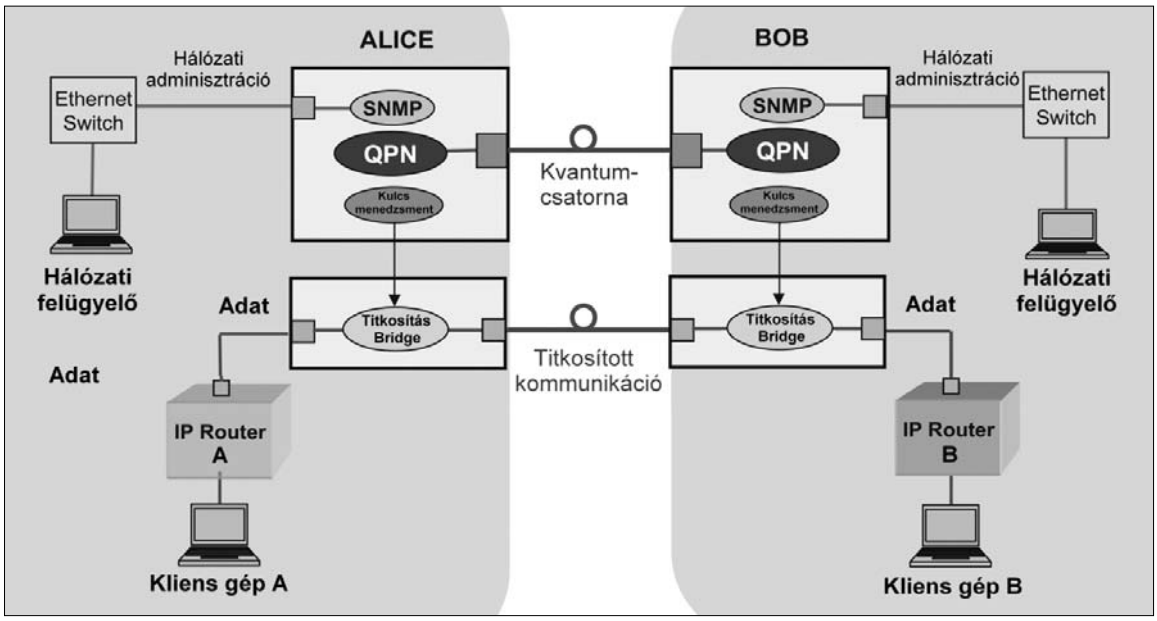
Az első kvantumkriptográfiára épülő banki tranzakciót Ausztriában, Bécsben hajtották végre. Anton Zeilinger laboratóriumából a fejlesztőcsapat egy 3000 eurós átutalást intézett a bank felé, kihasználva a kvantumcsatorna nyújtotta abszolút biztonságot. A kvantumkriptográfia megvalósításához szükséges eszközök már ma is elérhetőek a piacon. A technológia azonban jelenleg még drága, így a potenciális vásárlói kör is meglehetősen szűkre szabott. Az új eszközök elsősorban kutatóintézetek és kormányzati hivatalok, bankok számára jelenhetnek egy fejlettebb, biztonságosabb alternatívát.

A fotonok véletlenszerű viselkedése a kvantummechanikában egy olyan jelenség, amelyet a gyakorlatban több helyen is felhasználhatunk. Így, az előzőekben tárgyalt kvantumkriptográfián kívül alkalmazhatjuk például *valódi véletlenszám-generátorként* is. A foton alapú véletlenszám előállításhoz szükséges eszközök már kereskedelmi forgalomban is elérhetőek, PCI, USB-eszközként, illetve OEM-chipként, egy egyszerű perifériaként illeszthetőek egy klasszikus számítógéphez. Egy klasszikus, determinisztikus működésű számítógéppel csak *álvéletlen-számokat* állíthatunk elő, így az a *valódi véletlenszám-generátort* csak közelíteni képes. A kvantummechanika jelenségeire építve azonban lehetőségünk adódik a valódi véletlenszámok előállítására is, egy klasszikus számítógépes rendszeren belül is.

4.1. Kvantumkriptográfiai eszközök

A jelenleg forgalmazott kvantumtitkosító eszközökkel 80-110 km-es távolságon valósítható meg a tökéletesen biztonságos kommunikáció. Az optikai szál alapú implementációk esetén a detektorok pontatlansága, illetve a különböző zajforrások jelentik a szűk keresztmetszetet. Emellett, jelenleg még nem áll rendelkezésünkre az optikai erősítőhöz hasonlító „kvantumállapot-erősítő” eszköz, így a kvantumbiteket gyenge koherens lézernyalábbal küldjük át a kvantumcsatornán. A kvantumkriptográfia implementációjához szükséges eszközök az adatkapcsolati rétegben működnek, transzparens módon.

15. ábra Kvantumtitkosító berendezés



16. ábra Kvantum-kriptográfia alkalmazása hálózati környezetben

A kvantumtitkosító eszközökkel megvalósított hálózati kommunikáció egy lehetséges implementációja látható a *16. ábrán*, ahol a hálózaton belüli adatkommunikáció titkosítását a kvantumcsatornán kialakított kulccsal hajtjuk végre [8]. A kvantumcsatorna egy szabványos optikai szál segítségével is megvalósítható, így a már kiépített optikai hálózatok tökéletesen alkalmazhatóak a kvantumkriptográfia gyakorlati implementációiban. A kvantumkriptográfia hálózati rendszereken belüli alkalmazása során azonban figyelembe kell vennünk, hogy az üvegszálon csak passzív optikai elemek lehetnek, a foton szintű kommunikáció következtében pedig a modell rendkívül érzékeny a detektor-zajokra [8].

A kvantumtitkosító eszközök LAN, MAN, SAN hálózatokon belül is alkalmazhatóak. A gyakorlati implementációk a fellépő zavarok következtében egyelőre csak limitált távolságon (<100km) képesek garantálni a tökéletes biztonságot. Azonban kaszkádosítással nagyméretű hálózati rendszerek védelme is megvalósítható, így a kvantum-titkosítás által nyújtott biztonság egy-egy hálózat egészére kiterjeszthető.

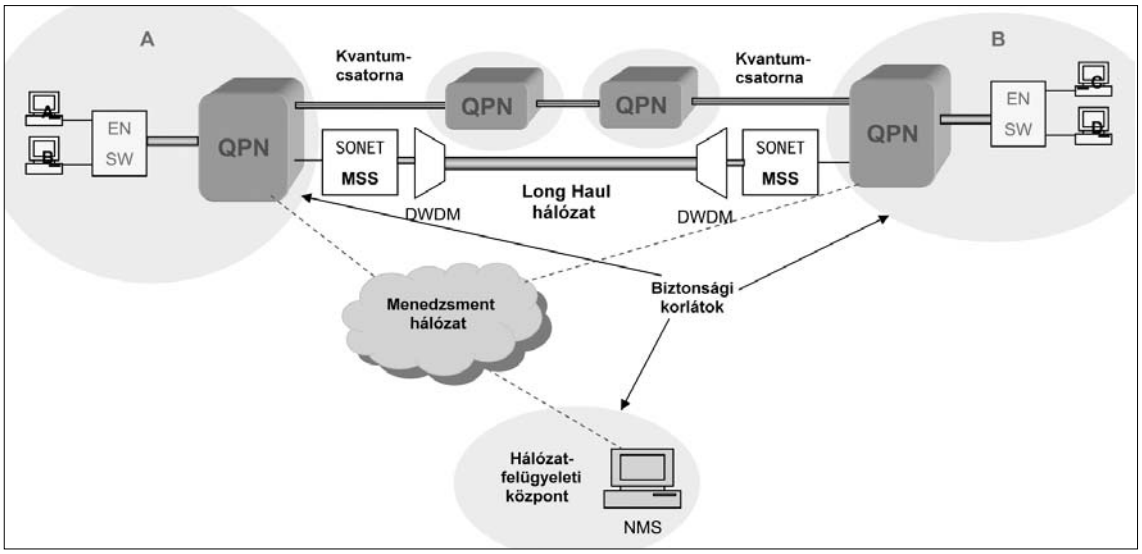
A *17. ábrán* egy „long-haul” hálózati implementáció gyakorlati megvalósításának vázlatát láthatjuk [8].

Az eszközök támogatják az összes fejlett, illetve napjainkban alkalmazott titkosító és hitelesítő algoritmust, így például a 128, 192, 256 bites AES-t valamint a HMAC-SHA1, HMAC-SHA-256 stb. módszereket. Az eszközök legtöbbje beépített véletlenszámgenerátorral rendelkezik, a protokoll lehallgathatatlanságát pedig a beépített intelligens lehallgatás-detektáló rendszer garantálja.

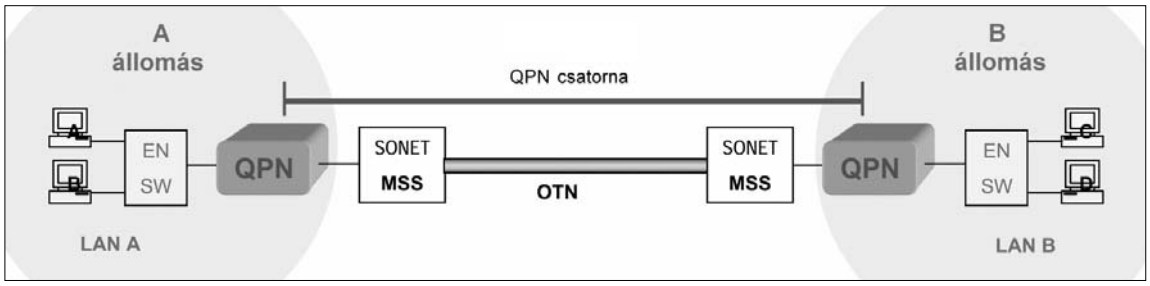
A kvantumkriptográfia által védett kommunikáció kiterjeszthető LAN-ok közötti kommunikációra is, ahogyan azt a *18. ábrán* láthatjuk [8].

Az eszközökkel megoldható Ethernet-hálózatok technikailag vagy logikailag elkülönülő részeinek összekapcsolása is, a hálózaton belüli adatforgalom kvantum-alapú titkosítása mellett. A kvantum-kommunikációhoz szükséges kvantumcsatornát Gigabit Ethernet hálózatok között is felépíthetjük [8].

Összefoglalva, az optikai szál alapú gyakorlati implementációk egyik legfontosabb tulajdonsága, hogy a protokoll a már kiépített optikai hálózatokon keresztül is megvalósítható. A kvantumcsatorna implementálásához mindösszesen egy dedikált optikai üvegszál szükséges a küldő és a vevő között.



17. ábra Long haul megvalósítás



18. ábra LAN-ok közti kvantum-kommunikáció

5. Összefoglalás

Mennyi idő múlva terjedhet el a gyakorlatban a kvantumkriptográfia? Jelenleg nem kínál előnyöket, mert napjaink titkosító algoritmusai révén rendelkezésünkre állnak a gyakorlatban feltörhetetlen kódok [4], azonban, ha a kvantumszámítógépek valósággá válnak, akkor az RSA és a többi modern kriptográfiai eljárás mind használhatatlan lesz, így szükségessé válik a kvantumkriptográfia használata. De vajon a kvantumkriptográfia időben a segítségünkre lesz?

A kvantumkriptográfia nemcsak gyakorlatilag feltörhetetlen kód, hanem abszolút értelemben is az. A kvantumelmélet lehetetlenné teszi, hogy Eve helyesen értelmezze az Alice és Bob közötti megállapodás értelmében kialakult kulcsot. Kijelenthető, hogy ha egy kvantumkriptográfiával titkosított üzenetet valaha is megfejtenének, akkor hibás a kvantumelmélet, ami az egész fizikát alapjaiban döntené össze. A módszer biztonságos kommunikációt garantál a kormánynak, katonaságnak, az üzleti életben, s a nagyközönség számára is.

A szerzőről

GYÖNGYÖSI LÁSZLÓ 2008-ban szerzett kitüntetéses diplomát a BME Villamosmérnöki és Informatikai Kar műszaki informatika szakán, infokommunikációs rendszerek biztonsága szakirányon. Jelenleg PhD hallgató a BME Villamosmérnöki és Informatikai Kar Híradástechnikai Tanszékén. Főbb kutatási területei a kvantuminformatika, a kvantum-kommunikációs protokollok, valamint a kvantumkriptográfia.

IMRE SÁNDOR Budapesten született 1969-ben. 1993-ban szerzett diplomát a BME Villamosmérnöki és Informatikai Karán. 1996-ban dr. univ., 1999-ben PhD, 2007-ben MTA Doktora fokozatott szerzett. Jelenleg a BME Híradástechnikai Tanszékén egyetemi tanár, vezeti a Mobil Távközlési és Informatikai Laboratóriumot, valamint a BME Mobil Innovációs Központjának tudományos kutatási igazgatója. Főbb kutatási területei a korszerű mobil infokommunikációs rendszerek rádiós és hálózati kérdései, valamint a kvantumalapú informatika.

Irodalom

[1] Bennett, Ch.H., Brassard, G., Crépeau, C., Maurer, U.M.: Generalized privacy amplification. IEEE Transactions on Information Theory 41(6), pp.1915–1923., November 1995.

[2] Brassard, G., Crépeau, C.: 25 years of quantum cryptography. SIGACT News 27(3), pp.13–24., 1996.

[3] Imre, S., Balázs, F.: Quantum Computing and Communications – An Engineering Approach. John Wiley and Sons Ltd, 2005.

[4] Diffie, W., M.E. Hellman: New directions in cryptography. IEEE Transactions on Information Theory IT-22(6), pp.644–654., 1976.

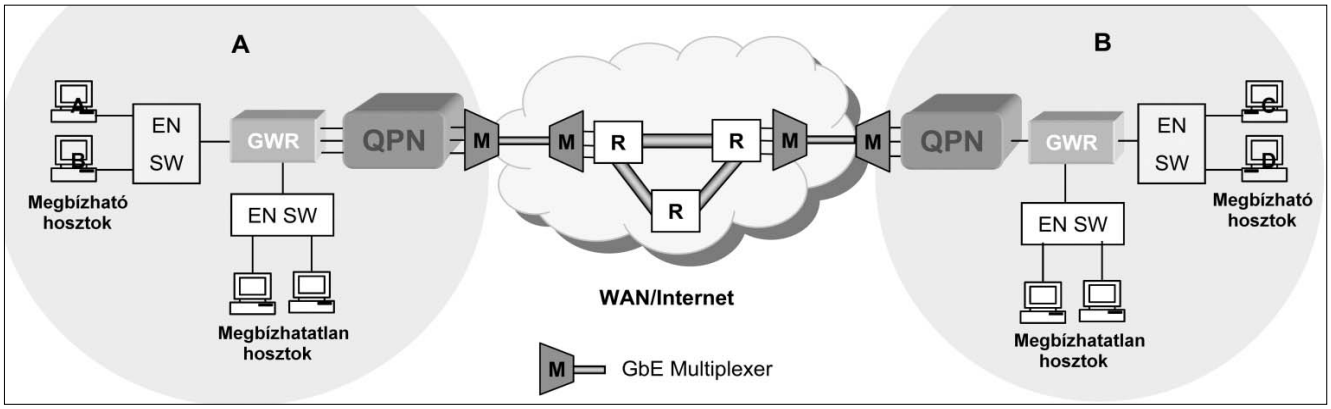
[5] Ekert, A.: Quantum cryptography based on Bell's theorem. Physical Review Letters 67(6), pp.661–663., 1991.

[6] Shor, P.: Algorithms for quantum computation: discrete logarithms and factoring. In: Proc. of 35th Annual Symposium on Foundations of Computer Science (1994)

[7] Wootters, W.K., Zurek, W.H.: A single quantum cannot be cloned Nature 299, p.802 (1982).

[8] Audrius Berzanskis: Applications of Quantum Cryptography in Government, MagiQ Technologies, SC05, November 12-18, 2005.

19. ábra Gigabit Ethernet hálózatok közti kvantumtitkosítás megvalósítása



A TCP/HSDPA rendszer átvitelének analitikus modellje

BODROG LEVENTE, HORVÁTH GÁBOR, VULKÁN CSABA*

Budapest Műszaki Egyetem, Híradástechnikai Tanszék
{bodrog,ghorvath}@hit.bme.hu

**Nokia Siemens Networks, Budapest*
csaba.vulkan@nsn.com

Lektorált

Kulcsszavak: TCP átvitel, HSDPA, sorbanállási hálózat, Markov modell

E cikkben a TCP átvitelét adjuk meg mobil, adatforgalmat nyújtó, HSDPA környezetben a Padhye modell alapján, a TCP csomagvesztési valószínűsége és a körbefordulási ideje segítségével. E két paramétert meghatározandó megalkottuk a HSDPA-t leíró sorbanállási hálózatot, amely tartalmazza a torlódási pontokat és protokollrétegeket, amelyek hatással vannak a vesztésre és a késleltetésre. Ennek a sorbanállási hálózatnak a megoldását részletezzük.

1. Bevezetés

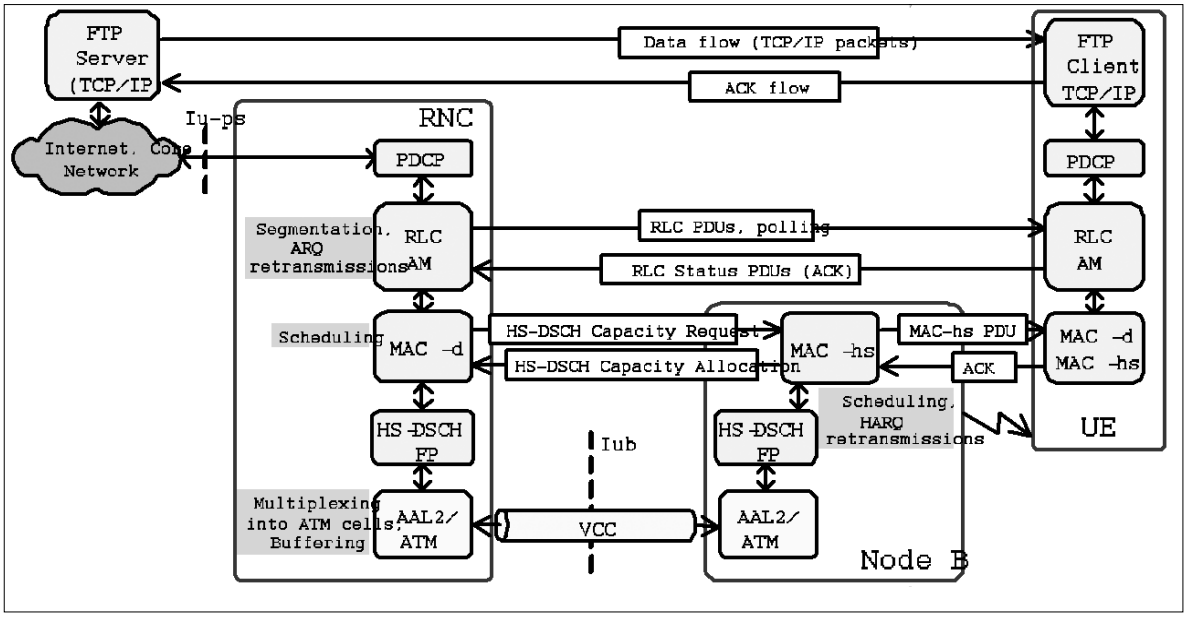
A HSDPA (High Speed Downlink Packet Access) nagysebességű (akár több megabit másodpercenként) csomagkapcsolt, letöltésirányú szolgáltatást nyújt UMTS (Universal Mobile Telecommunications System) felett [6].

Hagyományos UMTS esetén az adatkapcsolati rétegbeli protokollok – mint például Radio Link Control (RLC) és Medium Access Control (MAC) – a rádióshálózat-vezérlőben (RNC, Radio Network Controller) végződnek. A rádiós interfészt megvalósító protokollok az RNC-vel az Iub interfészen kapcsolódó bázisállomásban (3. generációs mobilhálózatok esetén ez a Node B) vannak megvalósítva. Nyugtázott módban (AM) az RLC felelős a hibamentes, sorrendhelyes átvitelért, amelyet az ARQ (Automatic Repeat Request – automatikus újraadás) mechanizmussal érnek el, ami azonban növeli a második rétegbeli körbefordulási időt, így TCP időtúllépéshez vezethet.

HSDPA esetén új protokollréteget – a MAC-hs réteget – vezettek be a bázisállomásban *(1. ábra)*.

Ennek segítségével a bázisállomás képes gyorsan alkalmazkodni a rádiós interfész aktuális állapotához modulációs és kódolási sémaváltással, gyors ütemezéssel és újraadással. Ez utóbbit a HARQ (Hybrid ARQ) mechanizmus valósítja meg. Ezek a megoldások segítik csökkenteni a második rétegbeli körbefordulási időt, ha az újraadást a hibás rádiós interfész feletti átvitel okozta. Habár ezzel a bázisállomás kezeli az újraadást, az RLC rétegbeli újraadás is megmaradt a Rel'99-es megoldásokkal való kompatibilitás, illetve a rendszeren belüli hívásátadás-vezérlés megtartása érdekében. Az RLC ugyanakkor továbbra is kezeli az újraadást, ha a MAC-hs újraadások száma elért egy megengedett legnagyobb számot, vagy ha a szállítási rétegben dobás volt. Így HSDPA esetén is növelheti az RLC a körbefordulási időt.

Ezen megoldások azt is eredményezik, hogy a TCP képtelen megállapítani és kezelni a torlódást, csak ha már lejárt a TCP időzítője, vagy ha a csomag elérte az RLC újraadások legnagyobb megengedett számát is és azt az RLC eldobta.



1. ábra A HSDPA protokollcsalád áttekintése

A rádiós interfész kezelésének az elosztása az RNC és a bázisállomás között egy áramlásvezérlési algoritmus beiktatását is maga után vonta – HSDPA áramlásvezérlés [8]. Ennek az algoritmusnak a lényege, hogy a bázisállomás határozza meg az RNC által az egyes felhasználóknak küldött adat mennyiségét, úgy, hogy a puffereket optimális szinten tartsa, azaz ne legyen sem a késleltetés túl nagy, de ne vesztegesse a rádiós interfész kapacitását sem. Ezt leggyakrabban a sorhossz mintavételezésével és az időegység alatt küldött csomagok (PDU, Packet Data Units) számának mérésével érik el.

Jól mutatja a HSDPA által nyújtott szolgáltatás színvonalát az elérhető TCP átvitel. Vizsgálták már a TCP teljesítményét HSDPA felett [1-3], ahol a szerzők szimuláció alapú modell adtak. Ebben a cikkben mi az analitikus modelljét adjuk ugyanennek.

A cikk további része a következőképpen épül fel. Elsőként megadjuk a rendszer szűk keresztmetszeteit jelentő pufferelési pontokat és a belőlük felépített sorbanállási hálózatot, majd összefoglaljuk a közelítő átvitelszámítást és részletesen ismertetjük a sorbanállási hálózat megoldását, végül pedig összefoglaljuk az eredményeinket.

2. A rendszer áttekintése és sorbanállási hálózatmodellje

A várakozási sorok lényeges alkotóelemei a HSDPA rendszernek, ezért természetesnek tűnik a TCP körbefordulási idő – mivel az nagy hatással van a TCP teljesítményére – modellezésére egy megfelelő sorbanállási modell alkalmazása. Ennek megfelelően a rendszer letöltési irányú késleltetését számottevően befolyásoló szűk keresztmetszeteit kell meghatározni (mobilszolgáltatások esetén a felhasználók jellemzően letöltenek, így leggyakrabban letöltés irányban szenvednek el nagyobb késleltetést). A kidolgozott modellben is a letöltési irány teljesítményére koncentráltunk, ahol a feltöltési késleltetést állandónak tekintettük. Csomagvesztés (p) épp ezeknél a várakozási soroknál, telített pufferek esetén fordulhat elő, vagy az újraadások legnagyobb számának elérésekor.

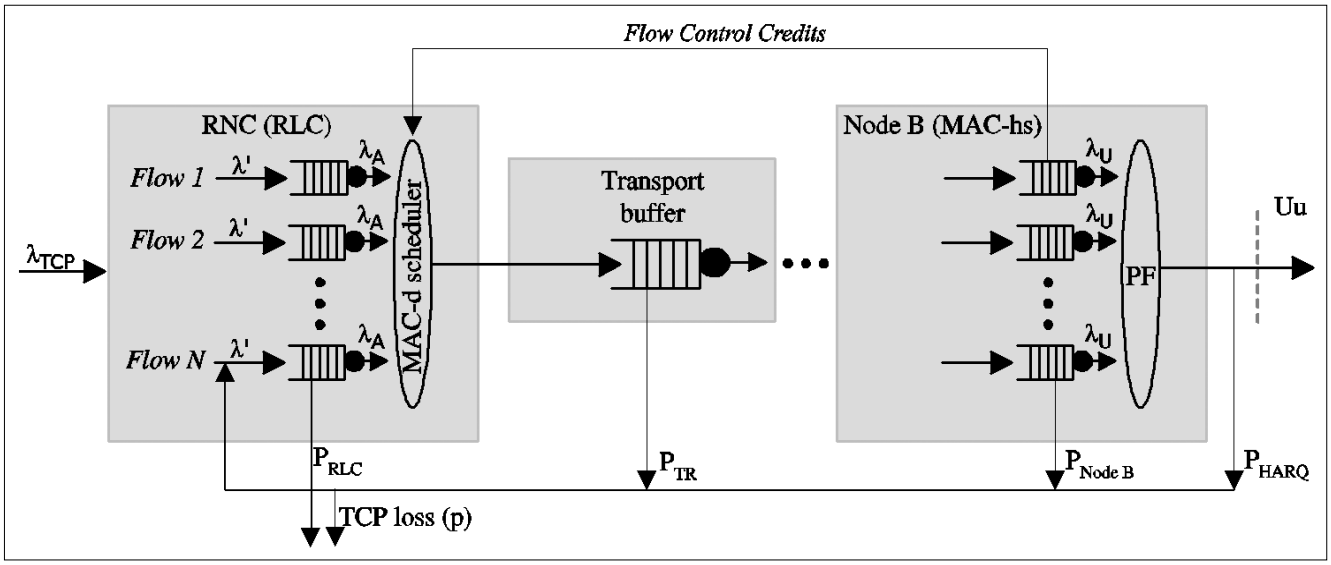
Három ilyen pontja van a rendszernek:

- *Az RLC réteg pufferei,* ahol a felhasználói csomagok részekre bontásával nyert RLC csomagokat tárolja a rendszer nyugta érkezéséig, vagy az újraadások legnagyobb számának elérése után eldobja azokat. A pufferbeli csomagok ütemezését a MAC-d réteg vezérli a bázisállomás MAC-hs rétege által biztosított kreditekre támaszkodva. A kreditek úgy vannak meghatározva, hogy a rádiós interfész átvitele a lehető legnagyobb legyen, nem feltétlenül figyelembe véve az Iub interfészen való torlódást, ezért is lehetséges, hogy az RLC túlterhelheti a szállítási réteget. E modellben azt feltételeztük, hogy feltöltés irányban nincs késleltetés,
- *Az AAL2/ATM szállítási hálózat pufferei.* Minthogy a szállítási hálózaton a felhasználók osztoznak és véges kapacitású, itt is előfordulhat torlódás, ami a csomagok késleltetéséhez, illetve azok eldobásához vezethet. A modellben ezt egy várakozási sorral vettük figyelembe, tekintettel a szűk keresztmetszetet jelentő ATM összeköttetésre.
- *A bázisállomásbeli MAC-hs pufferek.* Itt a rendszer szintén felhasználónként puffereli a csomagokat. A 2 ms alatt küldhető csomagok számát a CQI (Channel Quality Indicator) – a rádiós összeköttetést leíró mennyiség – határozza meg. Amennyiben egy csomag elvész, azt a HARQ mechanizmus működésének megfelelően a rendszer újraküldi, amíg el nem éri az újraküldések legnagyobb számát, amikor is az RLC ARQ veszi át a PDU kezelését.

A sorbanállási hálózati modell a *2. ábrán* látható a különböző rétegekben elhelyezkedő pufferekkel.

Az RLC felhasználónként egy-egy pufferben tárolja a csomagokat, amiket a bázisállomástól kapott kreditek

2. ábra A rendszer sorbanállási hálózatmodellje



alapján egymástól függetlenül ütemez. Egy PDU akkor vész el, ha a puffer túlcsordul, vagy ha elérte az újraadások legnagyobb számát.

Az átviteli hálózatot egy, a szűk keresztmetszetet jelentő összeköttetést reprezentáló pufferrel modelleztük. Telített puffer esetén az ATM cellák elvesznek.

A bázisállomásban is van minden felhasználónak egy-egy puffere, amelyek közül egyet egy PF (proportional fair – arányosan igazságos) ütemező szolgál ki minden 2 ms hosszú időrésben. Dobás esetén a csomagot újra adja a bázisállomás MAC-hs rétege az újraadások legnagyobb számának erejéig.

3. A TCP átvitelének számítása

A TCP átvitelét a rendelkezésre álló lehetőségek közül a legnépszerűbb – Jitendra Padhye és társai által [9]-ben kidolgozott – modellel számoljuk. Ez lényegében egy egyszerű kifejezést ad a TCP átvitelére (B) a csomagvesztési valószínűség (p) és a körbefordulási idő (RTT) függvényében:

$$B(p,RTT)=\begin{cases}\dfrac{\frac{1-p}{p}+E[W]+\hat{Q}(E[W])\frac{1}{1-p}}{RTT\left(\frac{b}{2}E[W_u]+1\right)+\hat{Q}(E[W])T_0\frac{f(p)}{p}}, & E[W_u]<W_{\max}\\ \dfrac{\frac{1-p}{p}+W_{\max}+\hat{Q}(E[W])\frac{1}{1-p}}{RTT\left(\frac{b}{8}W_{\max}+\frac{1-p}{pW_{\max}}+2\right)+\hat{Q}(W_{\max})T_0\frac{f(p)}{1-p}}, & \text{egyébként.}\end{cases} \quad (1)$$

A kifejezésben p a csomagvesztést, b az egyszerre nyugtázott csomagok számát (e cikkben végig b=1-et feltételezünk), T_0 a TCP időzítését (mi T_0=1,5 mp-et feltételeztünk), RTT a körbefordulási időt, W_{max} a legnagyobb torlódási ablakméretet jelöli. $E[W_u]$ a korlátlan ablakméret várható értéke

$$E[W_u]=\frac{2+b}{3b}+\sqrt{\frac{8(1-p)}{3bp}+\left(\frac{2+b}{3b}\right)^2}.$$

$\hat{Q}(w)$ annak a valószínűsége, hogy w ablakméret esetén az időzítő lejárta okozta a vesztést:

$$\hat{Q}(w)=\min\left(1,\frac{\left(1-(1-p)^3\right)\left(1+(1-p)^3\right)\left(1-(1-p)^{w-3}\right)}{1-(1-p)^w}\right).$$

Végül $f(p)$ egy egyszerűsítés:

$$f(p)=1+p+2p^2+4p^3+8p^4+16p^5+32p^6.$$

Azaz a TCP átvitele két paramétertől függ, a körbefordulási időtől (RTT) és a vesztési valószínűségtől (p). Az eddigiek alapján jogos a rendszert egy sorbanállási hálózattal modellezni, hiszen az RTT jelentős részét a különböző sorokban való késleltetés teszi ki, illetve csomagvesztés is vagy ezeknek a pufferekeknek a telítettsége miatt, vagy a rádiós interfész hibái miatt van.

Cikkünkben a TCP forgalmat állandó intenzitású folyamként vesszük figyelembe, ezzel is egyszerűsítve a modellt. Így válunk képessé, hogy a Padhye modell paramétereit a következő alszakaszokban leírt sorbanállási hálózat segítségével számoljuk ki. Amint e két paraméter ismert, az átvitel számolható, ami azonban nem feltétlenül felel meg a kezdeti feltételként megadott intenzitásnak. Ebben az esetben a kezdeti intenzitást korrigáljuk az eredménynek megfelelően és az átvitelt újra kiszámoljuk addig, amíg az egyensúlyi intenzitáshoz nem jutunk. Ez az a B^* átvitelnek megfelelő intenzitás, amelyre – ha ez a bemeneti intenzitás – pontosan olyan körbefordulási idő és vesztési valószínűség jön ki, hogy a Padhye modell a $B^*=B(p,RTT)$ átvitelt adja.

3.1. A számítási algoritmus áttekintése

Ahogyan azt írtuk, a TCP átvitelét HSDPA felett úgy számoljuk, hogy a hálózat terhelése (λ_{TCP}) épp olyan körbefordulási időt (RTT) és csomagvesztési valószínűséget (p) eredményez, amely paraméterekkel a Padhye modell épp ugyanekkora intenzitásnak megfelelő átvitelt ad, azaz $B(p,RTT)=\lambda_{TCP}$.

Ezt az egyensúlyi értéket például intervallumfelezéssel kaphatjuk. Ezt foglaltuk össze az *1. algoritmusban*.

Az intervallum alsó határának természetes kezdeti értéke 0, és mivel az átvitel nem lehet nagyobb, mint a rádiós interfész átlagos átvitele, ezért az intervallum felső határának kezdeti értéke épp ez ($E[S_{NodeB}]$) lesz.

Kiszámoljuk a csomagvesztést és az átlagos körbefordulási időt minden lépésben a 2. ábrán látható sorbanállási hálózat segítségével. Beállítjuk az intervallum alsó és felső határát a legutóbbi TCP átvitel (λ_{TCP}) és az épp kiszámolt Padhye-átvitel ($\lambda_{PADHYE}=B(p,RTT)$). kapcsolatának függvényében.

1. algoritmus A TCP átvitelének számítása

INPUT: sysparam//a rendszerparaméterek az 1. táblázatban
OUTPUT: γ //a TCP átvitel

1: $a_1=0$ //az intenzitás-intervallum alsó határa
2: $a_2=E[S_{NodeB}]$ // felső határ a rádiós interfész átvitelével egyezik
3: **while** $|\lambda_{TCP}-\lambda_{old}|>\varepsilon$ **do** //az intervallumfelező ciklus
4: $\lambda_{TCP}=\frac{a_1+a_2}{2}$
5: $(p,RTT)=$ QN Analysis(λ_{TCP})
6: $\lambda_{PADHYE}=B(p,RTT)K\frac{f_T}{f_M}$ //a Padhye-modell alkalmazása
7: **if** $\lambda_{PADHYE}>\lambda_{TCP}$ **then**
8: $a_1=\lambda_{TCP}$
9: **else**
10: $a_2=\lambda_{TCP}$
11: $\lambda_{old}=\lambda_{TCP}$
12: **return** $\gamma=\lambda f_M$ //mértékegységváltás

leírás	jelölés	érték
HSDPA felhasználók száma	K	16
RLC puffer mérete [PDU]	L	1000
az átviteli hálózat pufferének mérete [ATM cella]		2000
Node B puffer mérete [PDU]		1000
RLC (újra)adások legnagyobb száma	R	6
HARQ (újra)adások legnagyobb száma	M	3
közösen nyugtázott TCP csomagok száma	b	1
TCP időzítő hossza	T_0	1,5 mp
legnagyobb TCP torlódásvezérlő ablak mérete	$W_{\max}$	48 KB
blokkhiba-valószínűség a rádiós interfészen	P_e	0,01
két egymást követő hibás küldés valószínősége	P_s	0,001
a rádiós interfész kiszolgálási eloszlása	$P(\hat{S}=k)$	file-ból
TCP csomagméret	f_T	1500 byte
MAC-d és RLC PDU mérete	f_M	336 bit
pontosság	ε,ε'	1
szállítási réteg csatornakapacitása	C	~

1. táblázat A sysparam file tartalma

Az ábrán látható sorbanállási hálózat vizsgálatával kapjuk meg a csomagvesztést, illetve a körbefordulási időt. A felhasználókat azonosnak tekintjük és a számítást egy adott felhasználóra végezzük el. Ennek megfelelően a megjelölt felhasználó szempontjából a sorbanállási hálózat három várakozási sort tartalmaz: az RLC puffert, a többi felhasználóval közös szállítási (ATM) puffert és a bázisállomásban a MAC-d puffert. Ennek a sorbanállási hálózatnak nincs egzakt megoldása, ezért a forgalom felbontásán alapuló, közelítő megoldását számoltuk [4]. A vizsgálat során minden sornak megadjuk a minket érdeklő teljesítménymutatókon kívül a kimeneti folyamatát is, hiszen ez táplálja a következő sort. Az RLC-vesztést egy visszacsatoló ággal vettük figyelembe, mintha az elveszett, majd újraadott csomagok ismét a sorba érkeznének.

Emiatt a sorbanállási hálózatot csak iteratívan lehet megoldani (lásd *2. algoritmust*): kezdetben azt feltételezzük, hogy nincs visszacsatolt forgalom és kiszámoljuk az elvesző csomagok számát, majd a következő lépésben ezt tekintjük a visszacsatoló ág forgalmának, majd ezt addig csináljuk így, amíg az utolsó két érték közti különbség meghalad egy előre meghatározott pontosságot.

3.2. Az RLC puffer

Az RLC réteg modelljének (a 2. algoritmus 3. sorának `solve rlc` függvénye) lényegét az a megfigyelés adja, hogy a távozó forgalmat (egyben a szállítási hálózat érkező forgalmát) a HSDPA áramlásvezérlési algoritmusa szabályozza. A rádiós interfész hatékony használata érdekében a bázisállomás úgynevezett krediteket biztosít minden felhasználónak, melyek értékét a csatornaminőség és az adott felhasználó átlagos átvitelének függvényében adja. A MAC-d ütemező minden kör során a krediteknek megfelelő mennyiségű csomagot visz át. Mi 10 ms-os köridőt feltételeztünk (ez egy szokásos érték), azaz az ütemező csomagokat TTI_{RLC}=10 ms-onként küldi.

A küldhető csomagok számának meghatározásakor feltételezzük, hogy a bázisállomás ismeri a rádiós interfész állapotát, azaz a 2 ms-os HSDPA TTI alatt küldhető csomagok számának eloszlása ismert (lásd a 3.4. alszakaszt). E feltételezéssel a 10 ms alatt a MAC-d ütemező által átvitt csomagok száma $S_{RLC}=\sum_{1}^{5} S_{NodeB}$.

Az RLC puffer *érkezési folyamata* két részből áll, a rendszerbe belépő (λ_{in}/K intenzitású) forgalom és az RLC által újraadott csomagok (ez a λ_{FB} intenzitású visszacsatoló ág, ahogy a csomagvesztést modellezzük):

$$\lambda' = \frac{\lambda_{in}}{K} + \lambda_{FB}$$

Az RLC pufferbe 10 ms alatt érkező csomagok számának eloszlása így

$$P(A_{RLC}=k)=\frac{(\lambda' TTI_{RLC})^k}{k!}e^{-\lambda' TTI_{RLC}} \quad \forall k=0;1;2;... \quad (2)$$

Gyakorlatban az eloszlást úgy csonkoltuk (*N*-nél), hogy az eldobott farokrész valószínűsége már elhanyagolhatóan kicsi.

A *sorhossz* alakulását minden TTI_{RLC} végén egy diszkrét idejű Markov lánccal (DTMC) modelleztük, amelynek sorhossza a következőképpen alakul

$$X_{n+1}=(X_n+A_{n+1}-S_{n+1})^+,$$

2. algoritmus $(p,RTT)=$ `QN Analysis` (λ_{in})

INPUT: λ_{in} //a TCP források jelentette terhelés
OUTPUT: (p,RTT) //csomagvesztés és átlagos körbefordulási idő

1: $\lambda' = \frac{\lambda_{in}}{K}$ //a megfigyelt TCP felhasználó átvitele
2: **while** $|\lambda' - \lambda'_{old}| > \varepsilon'$ **do** // λ' egyensúlyi értékét kereső ciklus
3: $(P_{RLC}, E[T_{RLC}], D_{RLC}) =$ `solve rlc` (λ') //3.2. alszakasz
4: $(P_{Tr}, E[T_{Tr}], D_{Tr}) =$ `solve tr` (C, D_{RLC}) //3.3. alszakasz
5: $(P_{NodeB}, E[T_{NodeB}], \lambda_U) =$ `solve node b` (S, D_{Tr}) //3.4. alszakasz
6: $p_L \Leftarrow (P_{Tr}, P_{NodeB}, P_{HARQ})$ //a visszacsatoló ág valószínűsége (15)
7: $\hat{p} = \frac{\sum_{k=R+1}^{R}(1-p_L)^{k-1}p_L}{\sum_{k=1}^{R}(1-p_L)^{k-1}p_L}$ //RLC újraküldés valószínűsége
8: $\lambda' = \frac{\lambda_{in}}{K} + \hat{p}p_L\lambda_A$
9: $\lambda'_{old} = \lambda'$
10: $(D_u, D_s) \Leftarrow (P_{Tr}, P_{NodeB}, P_{HARQ}, E[T_{RLC}], E[T_{Tr}], E[T_{NodeB}])$ //(19)
11: $RTT = \sum_{k=1}^{R} \frac{p_L^{k-1}(1-p_L)}{1-p_L^R}((k-1)D_u + D_s)$ //a (20)-ban kapott RTT
12: $p = 1 - \frac{\lambda_U}{\lambda}$ //a (21)-ben megadott TCP vesztési valószínűség

ahol X_{n+1} a sorhossz, A_{n+1} az érkező csomagok száma és S_{n+1} a kiszolgált csomagok száma az n+1-ső időrésben. $(\cdot)^+$ max(0,·)-t jelöli.

A DTMC egylépéses állapotátmeneti mátrixának (P) ij-dik eleme:

$$p_{ij} = \begin{cases} \sum_{k=0}^{\infty} P(A_{RLC}=k)P(S_{RLC}=j-i+k), & i < L-N \\ \sum_{k=0}^{L-i} P(A_{RLC}=k)P(S_{RLC}=i-j+k)+ \\ +P(S_{RLC}=L+1-j)\sum_{k=L-i+1}^{N} P(A_{RLC}=k) & , \ i \geq L-N. \end{cases}$$

Az RLC puffer mérete L, az érkezési eloszlás tartója a [0,N] intervallum. Az első esetben a kiindulási sorhossz olyan rövid, hogy az érkező csomagok nem veszhetnek el, azaz az állapotváltási valószínűség megegyezik annak a valószínűségével, hogy j–i-vel több csomagot szolgált ki a rendszer, mint amennyi érkezett. A második esetben a kifejezésnek két tagja van, az első tag esetében nincs, a másodikéban van dobás.

A DTMC határeloszlását a következő lineáris egyenletrendszer megoldása adja $\pi P = \pi$

$$\pi h = 1,$$

ahol h a megfelelő méretű, csupa egyesekből álló oszlopvektor.

A határeloszlás ismeretében a csomagvesztési valószínűséget a TTI_{RLC}=10 ms alatt elvesző és az ugyanezen idő alatt érkező csomagok átlagos számának hányadosaként számoljuk:

$$P_{RLC} = \frac{\sum_{i=0}^{L}\pi_i \sum_{j=0}^{N}\max(0,i+j-L)P(A_{RLC}=j)}{\sum_{i=0}^{L}\pi_i \sum_{j=0}^{N} jP(A_{RLC}=j)}. \quad (3)$$

A csomagok rendszeridejét az RLC rétegben Little tételével számoljuk

$$E[T_{RLC}] = \frac{E[X_{RLC}]}{(1-P_{RLC})E[A_{RLC}]}TTI_{RLC} + \frac{1}{2}TTI_{RLC}, \quad (4)$$

ahol az átlagos sorhosszt $E[X_{RLC}]$-vel jelöltük. Mivel a modell diszkrét idejű és a csomagok folyamatosan érkeznek, a modell nem tesz különbséget az időrés elején és végén érkező csomag között. Ezt a TTI alatt egyenletesen elosztott érkezési pillanatokkal vettük figyelembe, vagyis a DTMC-ből kiszámolt rendszeridőhöz hozzáadunk fél TTI-t – az érkezési pillanat várható értékét.

Az RLC puffer *távozási folyamatát* szintén megadjuk, mivel a sorbanállási hálózatban ez a szállítási puffer érkezési folyamata. Azt feltételezzük, hogy a távozások független azonos eloszlásúak, ahol a TTI_{RLC} alatt távozó csomagok számának eloszlása

$$P(D_{RLC}=k) = \sum_{i=0}^{L}\pi_i \sum_{j=k+1-i}^{\infty} P(A_{RLC}=j)P(S_{RLC}=k) + \\ + \sum_{i=0}^{L}\pi_i P(A_{RLC}=k-i)\sum_{j=k}^{\infty} P(S_{RLC}=j) \quad (5)$$

A kifejezést két tag összege alkotja. Az első megfelel annak az esetnek, amikor van annyi csomag a pufferben, ahányat a kiszolgáló kiszolgálna, míg a második tag annak felel meg, amikor a kiszolgáló több csomagot szolgálna ki, mint ami a pufferben rendelkezésére áll.

3.3. A szállítási puffer

E cikkben AAL2/ATM szállítási réteget feltételezünk (ennek a modellje, illetve megoldása jelenik meg a 2. algoritmus 4. sorában). Az AAL2 réteg multiplexálja az egyes felhasználók forgalmát egy C kapacitású, állandó sebességű (CBR, Constant Bit Rate) VCC-be.

A MAC-d és a MAC-hs ütemezőkkel ellentétben az ATM kapcsoló folytonos időben működik, ennek ellenére úgy döntöttünk, hogy diszkrét idejű modellt dolgozunk ki, hogy elkerüljük a folytonos és a diszkrét idejű modellrészek keverését. Az RLC puffer TTI_{RLC}=10 ms-onként küld, míg a bázisállomásbeli PF ütemező TTI_{NodeB} = 2 ms-onként. Ez utóbbi kisebb értékűt választottuk időegységül az ATM diszkrét idejű modelljében, mert így valamivel finomabb felbontását nyerjük a folytonos időnek. További egyszerűsítő feltételezés, hogy a szállítási puffer RLC csomagokat továbbít, nem pedig ATM cellákat. Lévén, hogy az RLC PDU az adategység a hálózat többi részén, ezzel is jelentősen egyszerűsödik a modell megoldása.

Az időrésenként *érkező* csomagok számának eloszlását az RLC távozási folyamatából (D_{RLC}) vezetjük le. Ez azonban 10 ms-onként adott, amíg az előzőeknek megfelelően a szállítási puffer időegysége 2 ms. Azaz első lépésként végre kell hajtanunk az átalakítást a két eloszlás között, aholis a TTI_{RLC} ötször nagyobb TTI_{Tr}-nél. Binomiális feltételezéssel élve

$$P(D^{2ms_{tr}}=k) = \sum_{i=k}^{\infty} P(D_{RLC}=i)\binom{i}{k}\left(\frac{1}{5}\right)^k\left(1-\frac{1}{5}\right)^{i-k},$$

ahol $P(D^{2ms_{tr}}=k)$ annak a valószínűsége, hogy TTI_{Tr} idő alatt k csomag érkezett, ha TTI_{RLC} alatt i, más szóval hogyan tudunk kiválasztani k-t i-ből 1/5 valószínűséggel – tudniillik ez a két TTI aránya.

A szállítási puffer érkezési eloszlásának számításakor összegeznünk kell az összes felhasználó forgalmát, hiszen itt a teljes forgalmat egy VCC-be multiplexálja az ATM $A_{Tr} = \sum_{1}^{K} D^{2ms_{tr}}$. Itt K a HSDPA felhasználók száma.

Az RLC csomagok *kiszolgálási idejét* a szállítási pufferben így számoljuk:

$$D = \frac{\text{RLC csomagméret fejlécekkel}}{C}.$$

A fejléceket a következőképpen vesszük figyelembe:

$$\underset{\text{fejlécekkel}}{\text{RLC csomagméret}} = \underbrace{f_M\left(\frac{53}{47}\frac{f_M+24}{f_M}\frac{E[D_{RLC}]f_M+72}{E[D_{RLC}]f_M}\right)}_{\text{fejlécek}}$$

A fejlécek az ATM fejlécből (40 bit) plusz a 8 bites CPS PDU kezdeti mezőből (Start Field – 53/47), a 24 bites CPS csomag fejlécből csomagonként $(\frac{f_M+24}{f_M})$ és végül a 72 bites HS-DSCH FP keret fejlécből áll, ami $E[D_{RLC}]$ RLC csomagot szállít átlagosan.

Ebben a TTI_{Tr} időegységű diszkrét modellben a kiszolgáló vagy $F=\left\lfloor \frac{TTI_{Tr}}{D} \right\rfloor$ vagy F+1 csomagot szolgál ki

$$P(S_{Tr}=F)=1-\left(\frac{TTI_{Tr}}{D}-F\right)$$

$$P(S_{Tr}=F+1)=\frac{TTI_{Tr}}{D}-F \quad \text{valószínűséggel.}$$

A *sorhosszt* az RLC pufferéhez hasonló DTMC modellezi, azaz

$$X_{n+1}=(X_n+A_{n+1}-S_{n+1})^+, \tag{6}$$

ahol X_{n+1} a sorhossz, A_{n+1} az érkező és S_{n+1} a kiszolgált csomagok száma az n+1-edik időrésben.

Az érkezési és a kiszolgálási eloszlás ismeretében az RLC-hez hasonlóan építhetjük fel a DTMC egylépéses állapotátmeneti mátrixát:

$$p_{ij}=\begin{cases}\sum_{k=0}^{\infty}P(A_{Tr}=k)P(S_{Tr}=j-i+k), & i<L-(N-F)\\ \sum_{k=0}^{L-i}P(A_{Tr}=k)P(S_{Tr}=i-j+k)+ & \\ +P(S_{Tr}=L+1-j)\sum_{k=L-i+1}^{N}P(A_{Tr}=k), & i\geq L-(N-F)\end{cases}$$

A csomagvesztési valószínűséget is az RLC-éhez hasonlóan fejezhetjük ki:

$$P_{Tr}=\frac{\sum_{i=0}^{L-F}\pi_i\sum_{j=0}^{N}\max(0,i+j-L)P(A_{Tr}=j)}{\sum_{i=0}^{L-F}\pi_i\sum_{j=0}^{N}jP(A_{Tr}=j)}. \tag{7}$$

A számláló az elvesző, a nevező pedig az érkező csomagok várható száma.

A szállítási puffer rendszeridejét a Little formula segítségével számolhatjuk (ugyanúgy, mint (4)-ben):

$$E[T_{Tr}]=\frac{E[X_{Tr}]}{(1-P_{Tr})E[A_{Tr}]}TTI_{Tr}+\frac{1}{2}TTI_{Tr}. \tag{8}$$

A távozási folyamat eloszlása szintén az RLC azonos paraméteréhez hasonlóan számolandó:

$$P(D_{Tr}=k)=\sum_{i=0}^{L-F}\pi_i\sum_{j=k+1-i}^{\infty}P(A_{Tr}=j)P(S_{Tr}=k)+ \\ +\sum_{i=0}^{L-F}\pi_iP(A_{Tr}=k-i)\sum_{j=k}^{\infty}P(S_{Tr}=j) \tag{9}$$

3.4. A MAC-hs puffer

E cikkben azt feltételeztük, hogy a MAC-hs pufferek tartalmát arányosan igazságos (PF, Proportional Fair) algoritmus alapján ütemezi az ütemező, amely a pillanatnyi csatornaminőség és a felhasználók átlagos átvitelének alapján, a lehető leghatékonyabb erőforráskihasználást szem előtt tartva nyújt kiszolgálást a felhasználóknak.

Az ütemező minden körben kiválaszt egy felhasználót, aki adhat (minden TTI_{NodeB} = 2 ms-ban). A bázisállomás által meghatározott csatornaminőség-mutató (CQI, Channel Quality Indicator) meghatározza a kódolási sémát és ezzel együtt az egy TTI alatt küldhető csomagok számát. Mivel a csatornaminőség gyorsan változhat, időlegesen előfordulhat puffer túlterhelés is a bázisállomásban. Az érkező csomagokat a felhasználónkénti MAC-hs pufferek tárolják.

Nem tartozik e cikk céljai közé a rádiós interfész modellezése, ezért az Eurane projectből [5] vett MATLAB programmal állítottuk elő az egy TTI alatt átvihető csomagok számának eloszlását ($P(\hat{S}=k)$). Az eloszlás készítésekor telített puffereket feltételeztünk és nem vettük figyelembe a HARQ mechanizmust [7].

A MAC-hs puffer *kiszolgálási folyamatához* először is a HARQ-ot vettük figyelembe. [1]-ben és [2]-ben a szerzők megadják annak az eloszlását, hogy j-edikre sikeres az átvitel

$$P_j=\begin{cases}1-P_e, & j=1\\ P_e^{j-1}P_s^{j-2}(1-P_eP_s), & j>1.\end{cases}$$

A két paraméter (P_e és P_s) jelentését korábban, az *1. táblázatban* foglaltuk össze. Figyelembe véve, hogy az újraadások legnagyobb száma M, az (újra)adások várható száma

$$E[H]=\sum_{j=1}^{M}jP_j+M\left(1-\sum_{j=1}^{M}P_j\right),$$

és annak a valószínűsége, hogy egy időrés elvész HARQ vesztés miatt

$$P_{tl}=1-\frac{1}{E[H]}.$$

Végül az egy TTI alatt átvihető csomagok számának eloszlása (figyelembe véve a HARQ vesztéseket is):

$$P(S_{NodeB}=k)=\begin{cases}(1-P_{tl})P(\hat{S}=k)+P_{tl}, & k=0\\ (1-P_{tl})P(\hat{S}=k) & k\neq 0.\end{cases} \tag{10}$$

A MAC-hs pufferbe *érkező* csomagszám *eloszlásának* meghatározásakor feltételeztük, hogy a szállítási hálózatból érkező csomagok közül 1/K paraméterű binomiális eloszlás szerint k-an tartoznak a megfigyelt felhasználóhoz:

$$P(A_{NodeB}=k)=\sum_{i=k}^{\infty}P(D_{Tr}=i)\binom{i}{k}\left(\frac{1}{K}\right)^k\left(1-\frac{1}{K}\right)^{i-k}.$$

Ellentétben a másik két csomóponttal a MAC-hs puffer *sorhosszának* alakulása

$$X_{n+1}=(X_n-S_{n+1})^+ + A_{n+1}.$$

Ez azt jelenti, hogy csak azokat a MAC-d csomagokat szolgálja ki a PF ütemező, amelyek a TTI kezdete előtt érkeztek, azaz az állapotátmeneti mátrix ij-dik eleme

$$p_{ij}=\begin{cases}\sum_{k=0}^{j-1}P(A_{NodeB}=k)P(S_{NodeB}=i-j+k)+ & \\ +P(A_{NodeB}=j)\sum_{k=i}^{\infty}P(S_{NodeB}=k), & i<k_m\\ \sum_{k=0}^{\infty}P(A_{nodeB}=k)P(S_{NodeB}=j-i+k), & i\geq k_m.\end{cases}$$

A határeloszlás meghatározása után a csomagvesztési valószínűséget az elvesző és az összes érkező PDU számának hányadosaként kapjuk:

$$P_{NodeB}=\frac{\sum_{i=0}^{L}\pi_i\sum_{j=0}^{\infty}\max(0,i+j-L)P(A_{NodeB}=j)}{\sum_{i=0}^{L}\pi_i\sum_{j=0}^{\infty}jP(A_{NodeB}=j)}. \tag{11}$$

P_{NodeB} a puffer telítettsége miatt bekövetkezett dobási valószínűséget jelöli. Ugyanakkor nem ez az egyetlen módja a csomagvesztésnek a bázisállomásban. Ha a rádiós interfész rossz minőségű és a HARQ sem tudja már újraadni, a MAC-hs figyelmen kívül hagyja a csomagot és ha az újraadások száma eléri a legnagyobb megengedett értéket (M), akkor ismét az RLC réteg felelőssége lesz az újraadás. Ennek a valószínűsége

$$P_{HARQ} = 1 - \sum_{j=1}^{M} P_j. \tag{12}$$

A MAC-hs puffer rendszerideje a Little formula segítségével

$$E[T_{NodeB}] = \frac{E[X_{NodeB}]}{(1-P_{NodeB})E[A_{NodeB}]} TTI_{NodeB} + \frac{1}{2} TTI_{NodeB}, \tag{13}$$

ahol $E[X]$ az átlagos sorhosszt jelöli és a hozzáadott fél TTI magyarázata ugyanaz, mint az RLC és a szállítási puffer modelljeinek esetében.

A sorbanállási hálózat vizsgálatához szükség van még a bázisállomás pufferéből távozó csomagok intenzitására. A TTI_{NodeB} alatt távozó csomagok száma a kiszolgálható, illetve a pufferben lévő csomagok számának minimumával egyenlő. Így a távozó csomagok intenzitása

$$\lambda_U = \frac{1}{TTI_{NodeB}} \sum_{i=0}^{L} \pi_i \sum_{k=0}^{\infty} P(S_{NodeB} = k) \min(i,k) \tag{14}$$

3.5. A visszacsatoló ág

Azt feltételeztük a sorbanállási hálózatmodellünkben, hogy a hálózat különböző pontjain elveszett csomagok az RLC pufferbe újraadásra ismét belépnek. A 2. ábra visszacsatoló ága ezeket a csomagokat „gyűjti” össze. Ebben az alszakaszban ennek az összeköttetésnek a forgalmát fogjuk kiszámolni. Ezt a (Poissonnak feltételezett [4]) forgalmat adjuk hozzá az RLC puffer bemeneti forgalmához a hálózat vizsgálata során.

Legelőször kiszámoljuk annak a valószínűségét, hogy a PDU az RLC puffer elhagyása után (bármilyen okból) elveszett. Ezt jelölje p_L.

$$p_L = P_{Tr} + (1-P_{Tr})P_{NodeB} + (1-P_{Tr})(1-P_{NodeB})P_{HARQ} \tag{15}$$

Az is megtörténhet, hogy egy újraadott PDU elvész. Egy adott számú újraadás után – ez az RLC újraadások legnagyobb száma (R) – az RLC réteg figyelmen kívül hagyja az adott csomagot, ami TCP-szinten vesztést eredményez. Ezesetben a PDU nem lép be újra az RLC pufferbe (mígnem egy magasabb rétegbeli protokoll azt újra nem adja).

Annak a valószínűsége, hogy egy elveszett PDU még nem érte el az újraadások legnagyobb számát, azaz növeli az RLC puffer terhelését:

$$\hat{p} = \frac{\sum_{k=1}^{R} (1-p_L)^{k-1} p_L}{\sum_{k=1}^{R+1} (1-p_L)^{k-1} p_L}, \tag{16}$$

ahol az újraadások számát csonkolt geometriai eloszlásúnak feltételeztük.

A fentieket figyelembe véve a visszacsatoló ág forgalma:

$$\lambda_{FB} = \hat{p} p_L \lambda_A, \tag{17}$$

ahol λ_A az RLC pufferből való átlagos távozási intenzitást jelöli.

3.6. A TCP vesztés és a körbefordulási idő

Ebben az alszakaszban a pufferenkénti teljesítményjellemzők (részletekért lásd (12),(3),(4),(7),(8),(11) és (13) egyenleteket) alapján kiszámoljuk a TCP teljesítményét.

Az a TCP csomag, amely nem vész el

$$D_s = E[T_{RLC}] + E[T_{Tr}] + E[T_{NodeB}] \tag{18}$$

késleltetést szenved el.

Ha azonban valahol elveszett, akkor az átlagos csomagkésleltetést

$$D_u = P_{Tr} E[T_{RLC}] + (1-P_{Tr}) P_{NodeB} (E[T_{RLC}] + E[T_{Tr}]) + \\ + (1-P_{Tr})(1-P_{NodeB}) P_{HARQ} (E[T_{RLC}] + E[T_{Tr}] + E[T_{NodeB}])$$

adja. (19)

Egy k-szor (újra)adott csomag átlagos körbefordulási ideje a k-1 sikertelen és a sikeres küldés késleltetésének az összege. Geometriai eloszlású (újra)adásszámot feltételezve

$$RTT = D_{UL} + \sum_{k=1}^{R} \frac{p_L^{k-1}(1-p_L)}{1-p_L^R} ((k-1)D_u + D_s), \tag{20}$$

ahol D_{UL} az állandónak feltételezett feltöltési irányú késleltetést jelöli, ahogy valóban UTRAN-ban jellemzően nincs torlódás ebben az irányban.

A TCP vesztési valószínűsége egyszerűen 1 mínusz a sikeresen átvitt csomagok hányada:

$$p = 1 - \frac{\lambda_U}{\lambda}. \tag{21}$$

4. Összefoglalás

Ebben a cikkben egy közelítő modelljét adtuk a TCP-nek HSDPA felett. Azonosítottuk a rendszer lényeges torlódási pontjait, amelyek számottevően befolyásolják a TCP átvitelét és megadtuk ezek Markov-i modelljeit, hogy kiszámoljuk a rendszer teljesítményjellemzőit. A rendszer sorbanállási hálózatmodelljének egy iteratív megoldási módját adtuk.

A szerzőkről

BODROG LEVENTE a BME Villamosmérnöki és Informatikai karán diplomázott 2005-ben, illetve ugyanitt végzi doktori tanulmányait, Telek Miklós vezetésével. Érdeklődési körébe tartoznak a sztochasztikus modellek, különösképpen a sorbanállási rendszerek, a sztochasztikus folyamatok, illetve mindezek távközlési alkalmazásai. E témakörökben már több folyóirat- és konferenciacikke jelent meg.

Irodalom

[1] Mohamad Assaad, Badii Jouaber, Djamal Zeghlache, Effect of TCP on UMTS-HSDPA System Performance and Capacity.

In Global Telecommunications Conference, GLOBECOM '04, Dallas, TX, USA, November 2004. IEEE, Vol. 6, pp.4104–4108.
[2] Mohamad Assaad, Djamal Zeghlache, Cross-layer Design in HSDPA System to Reduce TCP Effect. IEEE Journal on Selected Areas in Communications, 24(3):614–625, March 2006.
[3] Mohamad Assaad, Djamal Zeghlache, TCP Performance Over UMTS-HSDPA Systems. Auerbach Publications, Boston, MA, USA, 2006.
[4] Gunter Bolch, Hermann de Meer, Stefan Greiner and Kishor S. Trivedi, Queueing Networks and Markov Chains: Modeling and Performance Evaluation with Computer Science Applications. Wiley-Interscience, August 1998.
[5] Eurane. The Eurane project, 2004. http://www.ti-wmc.nl/eurane/
[6] H. Holma, A. Toscala, HSDPA/HSUPA for UMTS. John Wiley & Sons, 2006.
[7] G. Horváth, Cs. Vulkán, Throughput Analysis of the Proportional Fair Scheduler in HSDPA. In Jan Sykora, editor, Proc. European Wireless 2008 (EW2008).
[8] P.J. Legg, Optimised Iub Flow Control for UMTS HSDPA. Vehicular Technology Conference, VTC 2005-Spring, 30 May–1 June 2005. IEEE 61st, Vol. 4, pp.2389–2393.
[9] Jitendra Padhye, Victor Firoiu, Don Towsley and Jim Kurose, Modeling TCP Throughput: a Simple Model and its Empirical Validation. Proc. of the ACM SIGCOMM '98 Conference on applications, technologies, architectures and protocols for computer communication, New York, Sept. 1998, ACM Press, pp.303–314.

Hírek

A **Cisco Magyarország** immár tizenegyedik alkalommal rendezte meg november 19-20. között a Cisco Expót, az év hálózati konferenciáját és kiállítását, az iparág szakembereinek és döntéshozóinak legjelentősebb hazai szakmai fórumát. Az idei Expo kiemelt témái – a hálózatokhoz kapcsolódó felhasználói trendeknek megfelelően – a videó, a virtualizáció és a kollaboráció voltak. Az Európa Kongresszusi Központban a két nap alatt több mint 60 előadás várta a résztvevőket. A kiállítás keretében több mint 10 standon jelentek meg a cég eszközeire épülő különböző – így például videó- és érintőképernyős – megoldások, emellett idén is felépült a Cisco City, amelyben a látogatók valósághű környezetben tekinthették meg és próbálhatták ki a legmodernebb hálózati megoldásokat egy bankfióktól kezdve az irodai környezeten keresztül az otthoni felhasználásig.

•

A **T-Systems és a Cisco** olyan közös innovatív technológiai megoldást dolgoztak ki, amelynek köszönhetően a most bejelentett „Compleo" szolgáltatási konstrukció új megközelítésbe helyezi az informatika alkalmazását a kis- és középvállalatok számára, mivel szélessávú internetet, IP telefonszolgáltatást, alközponti és számítógépes hálózatüzemeltetést, biztonsági funkciókat és modern készülékeket nyújt kezdeti beruházás nélkül.

Évek óta sokat ismételt tény, hogy a kis- és középvállalatok hatékonyságának javításában az informatikának kulcsszerepe lehet, azonban a beruházás kezdeti költségigénye, az informatikai szakemberek hiánya és a jelentős szervezési erőforrásigény miatt a fejlesztések legtöbbször nem valósulnak meg. Mindezekre együtt adhat megoldást a T-Systems Cisco technológián alapuló új megoldása, melylyel a cégek néhány hét alatt hozzájuthatnak egy azonnal használatba vehető, komplett kommunikációs rendszerhez, havi általánydíjas formában, a meglevő megoldás költségeinél körülbelül 20%-kal olcsóbban, hozzávetőlegesen munkaállomásonként 5-15 000 forintért a rendszer különböző paramétereitől függően. Mindezért az előfizető egy olyan egységes IP-alapú üzleti kommunikációs megoldást kap, amely magában egyesíti a szimmetrikus (2-10 Mbs) szélessávú internetkapcsolatot, a telefóniát, az egységes üzenetküldést, a hangpostát, az ügyfélkapcsolati alkalmazásokat, az audio- és videolehetőségeket, az interaktív konferenciamegoldásokat, illetve a jelenléti és mobilitási megoldásokat. A szolgáltatás már akár néhány alkalmazottal működő cég számára is hatékony megoldást biztosít.

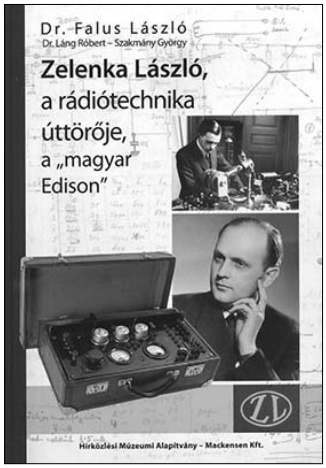
Az egységes, IP alapú műszaki háttérből adódó további előny, hogy nem merülhetnek fel kompatibilitási problémák az egyes egységek között, a távfelügyelet révén csökken a rendszer leállásából adódó kiesés, a szolgáltatás rugalmasan módosítható a szervezeti változásoknak megfelelően és nem utolsó sorban az üzleti folyamatok által igényelt adatbiztonsághoz, védelemhez és szabályozáshoz szükséges mélységi védelmet nyújtja. A szolgáltatás alapját jelentő Cisco UC 500 egységes kommunikációs rendszer lehetőséget teremt a cégek integrált hang-, video- és adathálózatának kialakítására. A szolgáltatás részeként telepített IP-telefonok segítségével nemcsak a hagyományos telefonszolgáltatások érhetőek el, hanem számos többletfunkció is, így például a személyes telefonkönyv, a hangposta, vagy a hívócsoportok kialakítása.

A korszerű IP-technológia lehetőséget nyújt további IP-alapú alkalmazások bevezetésére, mint a vállalati címtár-integráció, hangposta- és e-mail integráció, tárcsázás adatbázisból, táv- és csoportmunka, videotelefonálás.

Könyveket ajánlunk

Dr. Falus László, Dr. Láng Róbert, Szakmány György:

Zelenka László, a rádiótechnika úttörője, a „Magyar Edison”



A Hírközlési Múzeumi Alapítvány és a Mackensen Kft. kiadásában jelent meg az idei, immáron 79. Ünnepi Könyvhét újdonsága, a magyar híradástechnika és műszeripar egyik legnagyobb, méltatlanul elfeledett úttörőjének, Zelenka Lászlónak (1902-1960) a monográfiája.

A Műegyetemen gépészmérnöki diplomát szerzett feltaláló 1931-ben céget alapított és 15 éven keresztül készítette rádiótechnikai műszereit. A ZL Rádiólaboratórium sikerét jól példázza, hogy termékeit a Magyar Királyi Honvédség és a Magyar Rádió éveken át használta. Találmányai iránt behatóan érdeklődött a világhírű Marconi és a Philips cég, amely kísérletet tett a ZL Rádiólaboratórium felvásárlására. Bár cégét a II. világháború után államosították, a műegyetemi adjunktus Zelenka László állami vállalatokban fáradhatatlanul dolgozott tovább és olyan találmányok köthetők a nevéhez, mint az olvasókészülék vakok részére vagy a gumikifáradást mérő műszer. Bizton állíthatjuk, hogy csak a háború, az államosítás és diktatúra évei gátolták meg abban ezt a kivételes üzleti érzékkel is felvértezett, páratlan termékenységű alkotót, hogy Edisonéhoz hasonló világhírre és üzleti sikerre tegyen szert.

A könyvben dr. Falus László tanulmánya áttekintést ad Zelenka László munkásságáról és találmányainak sorsáról, olvashatunk benne egy visszaemlékezést dr. Láng Róbert tollából, aki ifjú korában másfél évet töltött műszerészinasként a ZL Rádiólaboratóriumban és a Zelenka Lászlóról festett képet a feltalálónak négy, az 1920-as évekből származó ifjúkori naplója teszi még színesebbé. A 120 oldalas, minőségi papírra készült, keménykötésű könyvben 88 darab soha nem publikált fénykép is látható Zelenka László találmányairól, a kiváló feltalálóról és laboratóriumáról, egykori lakásáról, tervrajzairól és naplóoldalairól.

Ez a hiánypótló kiadvány nemcsak a magyar híradástechnika története iránt érdeklődő lelkes amatőrök számára szép ajándék, hanem egyetemi szakkönyvként is használható, hisz dr. Falus tanulmánya pontos műszaki adatokkal szolgál Zelenka találmányairól, a lábjegyzetekben aprólékosan feltüntetett forrásanyaggal pedig további kutatásokat is lehetővé tesz a magyar híradás- és elektrotechnika, valamint műszeripar területén.

A könyv bolti ára: 2990 Ft.

A beszélő újságtól a rádióig – Puskás Tivadar és a Telefonhírmondó

Az elmúlt esztendőben volt 125 éve, hogy hazánkban helyszíni közvetítés jött létre a Nemzeti Színházból a Vigadóba, annak pedig 115 éve, hogy Puskás Tivadar benyújtotta „Új eljárás telefonújság szervezetére és berendezésére” című szabadalmi bejelentését, amely a Telefonhírmondó létrehozására vonatkozott. Ebből az alkalomból 2007. szeptember 20. és október 3. között a Magyar Szabadalmi Hivatalban (MSZH) kiállítással emlékeztek meg a telefonhírmondóról és feltalálójáról, Puskás Tivadarról, aki korát mintegy negyedszázaddal megelőzve, először valósította meg a kötött program szerinti közösségi információ- és műsorszórást.

A sokoldalú feltaláló munkásságát bemutató, a Postamúzeumtól, a diósdi Rádió- és Televíziómúzeumtól, a pesti Rádiómúzeumtól, az Országos Műszaki Múzeumtól, a Magyar Nemzeti Filmarchívumtól, a Puskás Tivadar Távközlési Technikumtól, valamint néhány magángyűjtőtől kölcsönkapott korabeli tárgyakat, szabadalmi és egyéb dokumentumokat, valamint hang- és filmfelvételeket az MSZH munkatársainak többségén kívül több mint száz külső érdeklődő és több iskolai osztály is megtekintette. A kiállított tárgyak között volt egy aranyozott fülhallgató-pár is, amelyen még I. Ferenc József hallgatta a Telefonhírmondó műsorát a Millenniumi kiállításon, valamint egy eredeti fejhallgató a 20-as évekből.

A kiállítás létrehozói gondoltak az igazoltan távol maradott kollégákra is, így az MSZH vezetőinek támogatása mellett elkészítették „A beszélő újságtól a rádióig – Puskás Tivadar és a Telefonhírmondó” c. kötetet. Az ajánlott kiadvány bemutatja a Telefonhírmondó történetét, szorosan követve a jelzett kiállítás tematikáját.

A kiadvány 1200 Ft/db áron megvásárolható az MSZH Ügyfélszolgálatán.

Ajánlotta: Sipos László

Könyveket ajánlunk

Németh József:

Műegyetemtől a világhírig – Képes egyetemtörténet

E könyv olvasója bizonyságot talál arra, hogy a Műegyetem tanárai és tanítványai hogyan vettek részt a magyar gazdaság fejlesztésében, hogyan járultak hozzá a világ műszaki fejlődéséhez. A könyv a Budapesti Műszaki és Gazdaságtudományi Egyetem közel 225 éves történetét, nemzetközi hírű mérnök-tanárait, több, ma már világhírű egykori tanítványát – akik közül hárman Nobel-díjat kaptak –, és a mai Műegyetemen folyó oktató, kutató munkát mutatja be magyar és angol nyelven, több mint 350 képpel. Elődeinktől kapott örökségünk arra kötelez bennünket, hogy a Műegyetem továbbra is hazánk vezető felsőoktatási intézménye, valamint Európa aktív, jelentős műszaki, természet- és gazdaságtudományi oktató-kutató központja maradjon. A kötet egyszerre emlékeztet a régiekre és bátorít az új keresésére.

A Budapesti Műszaki és Gazdaságtudományi Egyetem és a Műegyetemi Kiadó reprezentatív kiadványa a Műegyetem történetét, nagyjait, a világnak adott találmányait mutatja be. A minőségi kivitelű, nagyformátumú, sok képet és illusztrációt tartalmazó, kétnyelvű könyv a BME 225 évének rövid története mellett bemutatja azokat a nagy elődöket, a Nobel-díjasokat és feltalálókat, akiknek fontos szerepük volt a mérnökképzésben, a technikai fejlődésben, akik munkásságukkal jelentős mértékben járultak hozzá Magyarország hírnevének öregbítéséhez. A történeti áttekintés után bemutatja a 21. század elejének Műegyetemét, alkotó műhelyeit, oktatási és tudományos eredményeit, valamint az egyetem ipari kapcsolatait és az ebből hasznosuló fejlesztéseket.

Ez az album egyszerre szép, reprezentatív ajándék és tartalmas, információt hordozó kiadvány, mely azoknak készült akik fontosnak érzik, hogy a hazai műszaki nagyságok, eredmények jelentős részét bemutató album ott legyen a könyvespolcukon.

A szerző négy évtizede oktatja és kutatja a technika és a mérnökség magyarországi történetét. Amikor ennek szolgálatára szegődött, úgy vélte, a múlt, az egykori híres mérnökelődök történetének megismertetése erősíti egyetemünk hallgatóinak és a mindenkori olvasónak is az identitását. Erre különösen itt, a Kárpát-medencében és Európa új útjait kereső világunkban van nagy szükség. Németh József a mérnöki alkotómunka szépségét szerette volna bemutatni eddig megjelent könyveiben, tanulmányaiban és konferenciákon elhangzott előadásaiban is, így nem lehet más a célja a Műegyetem képes történetének összeállításával sem, amelyhez szerencsére sok segítőre talált munkája során.

A könyv bolti ára: 6990 Ft.

Ingyenes tankönyv a Microsoft PowerShell technológiáról

A Microsoft TechNet gondozásában jelent meg *Soós Tibor és Szerényi László* magyar nyelvű tankönyve, amely a Microsoft PowerShell szkripting technológiáját mutatja be a rendszergazdák szemszögéből, több mint 400 oldalon. A rendkívül részletes könyv segítségével az informatikai szakemberek gyakorlati példákon keresztül, az alapoktól kezdve sajátíthatják el a Microsoft parancssori környezetének működését.

A Windows alapú rendszerek üzemeltetői már régóta vágytak egy olyan eszközre, amellyel könnyen lehet automatizálni a gyakran ismétlődő feladatokat, de a korábbi lehetőségek vagy túl sok programozást igényeltek (VBScript, WSH), vagy csak egy szűk területet fedtek le (parancssori eszközök, pl. netsh parancs). A PowerShell nagyszerűen egyesíti magában a hatékony parancssori környezet és az objektumorientált programnyelvek legfontosabb jellemzőit, amelyekkel a rendszergazdák nagyon tömör, rövid, logikus felépítésű szkriptekkel könnyíthetik meg a munkájukat. Ebben kíván segíteni ez a könyv.

A .NET keretrendszerre épülő PowerShell lehetővé teszi, hogy a rendszergazdák a gyakran ismétlődő vagy sok műveletből álló feladatokat (pl. több száz postafiók létrehozása, adatbázismentés stb.) automatizálják. A PowerShell segítségével olyan műveletek is elvégezhetők, amelyek a grafikus felügyeleti eszközökkel nem vagy csak nagyon nehezen kivitelezhetők. A kliens- és szerveroldalon egyaránt használható PowerShell 1.0 a Windows Vistában és a Windows Server 2008-ban már opcionális komponensként megtalálható, de akár Windows XP-re is telepíthető.

A PowerShell fontos komponense a Microsoft legújabb generációs szerverszoftvereinek is, például az Exchange Server 2007 levelező- és az SQL Server 2008 adatbázisszervernek, valamint a System Center rendszerfelügyeleti alkalmazásoknak. Ezen szoftverek már mind támogatják a PowerShell segítségével megvalósított automatizálást és parancssori felügyeletet. Az új szerverszoftverek közös jellemzője, hogy funkcionalitásuk teljes egészében elérhetők PowerShell szkriptek használatával és maga a grafikus felület is erre a rétegre épül. A grafikus felületen a leggyakrabban szükséges feladatok könnyen elvégezhetők, de ha szükséges, a PowerShell használatával sokkal több lehetőség tárul fel a rendszergazdák előtt.

A technológia 2.0-ás verziója a Windows 7-be és a Windows Server 2008 R2-be is bekerül. Ebben debütál majd a továbbfejlesztett, natív Active Directory kezelés és a távoli gépek szkriptelése is lényegesen egyszerűbbé válik majd.

A tankönyv szabadon letölthető(!) a Microsoft TechNet portálról.

Hírek

A Novell bejelentette, hogy elérhető a Novell ZENworks Network Access Control terméke, amely a vállalat végpontbiztonsági és felügyeleti megoldásainak körét bővíti.

A ZENworks termékcsalád legújabb tagja a heterogén hálózati környezetek biztonságára felügyel: a javítócsomagoktól a tűzfalbeállításokig terjedő, szigorú biztonsági tesztek alapján létrehozott házirendekkel határozzák meg az eszközök hozzáférését vagy éppen annak megtiltását a hálózathoz. A legújabb ZENworks megoldás az alkalmazottak hatékonyságának csökkenése nélkül teszi lehetővé a vállalatok számára a hálózati hozzáférésvezérlés (Network Access Control – NAC) kockázatainak csökkentését, valamint a HIPAA, a PCI DSS és más szabályozások, illetve a belső biztonsági házirendek előírásainak való megfelelést.

A Novell új terméke ideális választás a heterogén hálózati környezetekben, mivel lehetővé teszi a vállalatok számára, hogy a hálózati hozzáférésvezérlést további frissítések és hálózati elemek beszerzése nélkül valósítsák meg. Emellett az új megoldás könnyen telepíthető az egyes eszközök és csoportok esetében előre meghatározott tesztek, valamint a fázisokra bontott telepítési lehetőségek révén, amelyeknek köszönhetően a bevezetés során nincs szükség az informatikai tevékenységek megszakítására.

A ZENworks Network Access Control a biztonság érvényesítésének kulcsfontosságú eszköze. Egyszerűen definiálható házirendek segítségével biztosítja az eszközök megfelelőségét, automatikus tesztfrissítésekről gondoskodik az új javítócsomagokhoz és a folyamatos felügyeletnek köszönhetően kivédi a nulladik napi támadásokat.

Egyre több vállalat telepíti át mainframe kiszolgálóiról az üzletkritikus IT megoldásokat HP Integrity szerverekre.

A HP megoldásával jelentősen csökkenthető a hardverek működtetési költsége, illetve megtakarítható a mainframe-hez szükséges szoftverlicencek árai. A HP szerint a következő 12 hónap során mintegy 125 európai vállalat fog az átköltözés mellett dönteni.

A HP Integrity szerverek az ügyfelek szerint is a megfelelő alternatívái a mainframek-nek, amelyet a vállalat magas, egyúttal tovább növekvő piaci részesedése is bizonyít. Az IDC szerint a HP-nek származik a legnagyobb bevétele az Európát, Közel-Keletet és Afrikát magában foglaló EMEA régióban az üzleti szerverek területén: a vállalat 32,7%-os piaci részesedést ért el 2008 második negyedévében. Az EMEA régióban a 2008-as pénzügyi év harmadik negyedévében a HP Integrity szerverek adták a cég üzletkritikus rendszerei bevételének legnagyobb részét, összesen 78%-át.

A HP Európában létrehozta a „HP Mainframe Áttelepítő Központját”, amely célja, hogy segítse az ügyfelek növekvő áttelepülési igényeinek magas szintű kiszolgálását. A szervezet központi irodája Madridban van, míg Bukarestben egy további iroda található. Az irodák munkatársai speciális tudásuk és tapasztalataik segítségével támogatják valamennyi EMEA ország szakembereit, akik az ügyfelek kiszolgálásának minden fázisában számíthatnak a specialisták tudására, és esettanulmányok segítségével közösen alakíthatják ki az adott ügyfél számára legoptimálisabb megoldást. A magyar szakembereknek ezen felül a FreeSoft Nyrt nyújt támogatást napi munkájuk során – legyen szó a rendszer kiválasztásáról, üzemeltetésről, vagy költségelemzésről.

Átadták a Budapesti Műszaki Főiskola Tanárképző és Mérnökpedagógiai Központjában a Microsoft Magyarország támogatásával létrehozott Innovatív Tanári Kompetenciaközpontot.

Ez a harmadik szakmai műhely, amit Magyarországon a Microsoft Társ a Tanulásban programjának keretein belül létrehoztak. A központ célja, hogy jó példákat és ötleteket, tippeket és trükköket mutasson be a tanár szakos hallgatóknak arról, miként tudja a számítógép támogatni a tanulás-tanítás folyamatát. A központ célja egyben az is, hogy olyan kutatások helyszíne legyen, ahol az IKT hatékonyságát vizsgálják a hallgatók.

A Microsoft Magyarország a „Társ a Tanulásban” program keretein belül csak idén 40 millió forintot költ a különböző programokra, melyekből számos már le is zárult a tanév első felében. Ezek közül érdemes kiemelni a több mint 400 számítástechnika tanárnak és iskolai rendszergazdának tartott több napos, ingyenes képzést és a valamennyi iskolába térítésmentesen eljutó, a Microsoft referenciaiskolák által írt szakkönyveket.

Az informatikai szakkönyvek mellett számos olyan hiánypótló tananyag került idén kiadásra a Microsoft gondozásában, melyek nemcsak a klasszikus értelemben vett oktatást, hanem az iskolából kikerülve például a diákok életrevaló felkészítését is segíti. Ilyen például az „Életrevaló – fiataloknak” című könyv, melynek célja, hogy a végzett középiskolásokat hasznos, életszerű, de nem tanult ismeretekkel gazdagodjanak. Hasonló megfontolásból támogatta a cég a „130 Starttipp kezdő vállalkozóknak” című könyvet, vagy azt, amely éppen a 21. századi modern iskola ismérveit foglalja össze iskolaigazgatók számára.

Software security

Keywords: software security, buffer overflow, fuzzing, software testing, static analysis, verification

Today's techniques of software development leave many programming bugs in our systems, which rarely appear during normal use. However, these bugs, which seem to be harmless, often hide possibilities for a malicious attacker to abuse the system.

The importance of the problem and the scale of the threat are increased by the fact that it is enough to find only one of these bugs for the attacker to circumvent the protection mechanisms and have control over a system by exploiting the found bug. Since the existence of these security flaws expose our systems to very serious threats, the protection against them and the prevention is vital.

Introduction to the world of botnets

Keywords: robot network, botnets, darknet, honeypot

The botnet is an army of computers driven by an attacker. The computers do not belong to the attacker itself, but the botnet consists of common home PCs infected with malicious code. The botnets are the most widespread and most dangerous use of malicious code nowadays. The average user does not know much about the working mechanism of the botnets and the defense methods against them even several years after their first appearance. The aim of this paper is to bring this knowledge closer to the reader by summarizing the working mechanisms of the botnets and give information about the possible countermeasures.

DRM technologies

Keywords: copyright problems, protection, Digital Rights Management

The protection of the author's rights is an important problem in the society. In the analog world the problem was simpler, due to the fact that during a content copy procedure the quality of the content degraded. In that world people paid for the quality. However, in the digital word things have changed and the copy procedure does not impact the quality anymore. In order to eliminate copyright problems, the content should be protected and the rights related to the content should be controlled. This protection and control is called Digital Rights Management (DRM). The purpose of this article is to describe the basics of DRM technologies and their usage.

Quantum cryptography based info-communication systems

Keywords: cryptography, quantum communication, quantum computation

In the information age that we live in, more than ever in history, we are faced with the problem of exchanging data quickly and accurately. Although there exist classical cryptographic schemes in theory which are unconditionally secure, the unconditional security in practice is impossible in classical modern cryptosystems. At the same time, the extrapolation of Moore's Law leads us to conclude that we will be able to transcribe a single bit of information on an atom to 2017. As follows, the world of the quantum is no longer a theoretical curiosity, the quantum-mechanical phenomena can be effectively exploited for the storage, manipulation and exchange of information.

The quantum systems and quantum computers have remarkable properties. The factoring quantum algorithm would allow us to decrypt with ease communication encrypted with today's modern state of the art encryption techniques. Recent interest in quantum cryptography has been stimulated by the observation that quantum algorithms threaten the security of classical cryptosystems. The quantum cryptography has been shown to be unconditionally secure against all attacks in an information-theoretic setting. The quantum cryptographic protocols are designed with the intention that their security is guaranteed by the laws of quantum physics, therefore the security of the quantum-protocols will not be compromised by future developments in quantum computing.

Analytical TCP/HSDPA throughput model

Keywords: TCP throughput, HSDPA, Markov model, queueing network

In this paper, an approximate, Padhye model based TCP throughput calculation method is presented for mobile data services over HSDPA. The Padhye model is defining the TCP throughput based on two input parameters: the packet loss probability and the TCP Round Trip Time. In order to provide the input parameters for the TCP throughput calculation, an equivalent queueing network model of the HSDPA system is created, which includes the congestion points and protocol layers that are having dominant impact on the delay and packet drop. The solution of the queuing network model is described in detail.

Summaries • of the papers published in this issue

Contents

Szerkesztőség
HTE Budapest V., Kossuth L. tér 6-8.
Tel.: 353-1027, Fax: 353-0451, e-mail: info@hte.hu

Hirdetési árak
Belív 1/1 (205x290 mm) FF, 120.000 Ft + áfa
Borító II-III (205x290mm) 4C, 180.000 Ft + áfa
Borító IV (205x290mm) 4C, 240.000 Ft + áfa

Cikkek eljuttathatók az alábbi címre is
Szabó A. Csaba, BME Híradástechnikai Tanszék
Tel.: 463-3261, Fax: 463-3263
e-mail: szabo@hit.bme.hu

Előfizetés
HTE Budapest V., Kossuth L. tér 6-8.
Tel.: 353-1027, Fax: 353-0451
e-mail: info@hte.hu

2008-as előfizetési díjak
Közületi előfizetők részére: bruttó 32.130 Ft/év
Hazai egyéni előfizetők részére: bruttó 7.140 Ft/év
HTE egyéni tagok részére: bruttó 3.570 Ft/év

Subscription rates for foreign subscribers:
12 issues 150 USD,
single copies 15 USD

www.hte.hu
Felelős kiadó: NAGY PÉTER
Lapmenedzser: DANKÓ ANDRÁS

HU ISSN 0018-2028
Layout: MATT DTP Bt. • Printed by: Regiszter Kft.