ÁROP – 2.2.21 Tudásalapú közszolgálati előmenetel

Biztonsági Technológiák Alkalmazása

egyetemi jegyzet
második, átdolgozott kiadás

Nemzeti Közszolgálati Egyetem

DR. LEITOLD FERENC

MAGYARY PROGRAM

Budapest, 2019

Szerzők:
Dr. Leitold Ferenc PhD

Hatályosítást 2019-ben végezte:
Dr. Leitold Ferenc PhD

A hatályosított kézirat lezárásának dátuma:
2019. július 8.

Kiadja:
© NKE, 2019

Felelős kiadó:
Prof. Dr. Kis Norbert
Dékán

TARTALOMJEGYZÉK

BEVEZETÉS

Az informatikai eszközök használata mára teljesen hétköznapivá vált. Ahhoz, hogy ezen eszközök használatában megbízhassunk, az informatikai biztonság kérdéseire is oda kell figyelnünk, legyünk akár a területen jártas szakértők, vagy akár laikus felhasználók.

Manapság az informatikai rendszerekkel szembeni fenyegetések többsége az internetről érkezik. Ebben a tananyagban a teljesség igénye nélkül a legáltalánosabb fenyegetési formákat tekintjük át, és a lehetséges védekezési módszereket összegezzük. A kommunikáció többszörösen jelenik meg az informatikai biztonság esetén. Egyrészt manapság a védendő adat általában része valamilyen kommunikációnak, a fenyegetések többsége is kommunikáción alapul, és a hatékony védekezési megoldások jelentős része is a kommunikációt használja.

Az első fejezetben az informatikai támadások rövid történeti áttekintését követően a legelterjedtebb támadási formákkal foglalkozunk: Windows-alapú rendszerek veszélyforrásai, interneten terjedő kártevők, célzott támadások, APT-k, illetve hálózati támadások.

A második gondolati egységben a biztonsági technológiákat vesszük számba, elsősorban a hálózati kommunikáció védelmére vonatkozóan, de nem maradnak ki a sorból a végpontokon futó védelmi lehetőségek sem.

A harmadik fejezetben pedig a biztonsági technológiák vizsgálatához szükséges legfontosabb módszereket, elvárásokat részletezzük, annak érdekében, hogy támpontul szolgáljon egy informatikai infrastruktúra védelmi rendszereinek a kiépítéséhez, a védelmi rendszerek kiválasztása ugyanis nem egyszerű feladat.

1. INFORMATIKAI RENDSZEREK TÁMADÁSI LEHETŐSÉGEI

Manapság az informatikai rendszereket veszélyeztető egyik legnagyobb veszélyforrás a programozott kártevők köre, melyek legnagyobb csoportját az olyan kártékony programok alkotják, melyek képesek önmaguktól terjedni, támadást végrehajtani, számítógépekhez távoli hozzáférést biztosítani, vagy akár az áldozat számítógépen tárolt dokumentumait eltulajdonítani.

Az internet elterjedésével az informatikai rendszerek biztonságának egyik sarkalatos kérdésévé vált azok biztonsága az internetről érkező támadásokkal szemben**. A hálózatbiztonság témakörébe tartozik minden olyan biztonsági kérdés, amin nem javíthatunk csupán titkosítással.** Manapság a legtöbb informatikai rendszert ért támadás az internetről származik. Az internet 1990-es évektől kezdődő rohamos fejlődése a XXI. század első évtizedének a végére termékeny melegágyat teremtett az informatikai támadások számára. Már 2008-ban is az internetről érkezett a támadások 92%-a, és csupán 8% vett igénybe valamilyen adathordozót a bejutáshoz. Az internetről érkező veszélyeket alapvetően két csoportba sorolhatjuk. Egyrészt az interneten elérhető és onnan érkező kártevők köre, illetve a célzott támadásokkal kapcsolatos biztonsági kérdések. A célzott támadások esetén a támadó kifejezetten a megtámadott informatikai infrastruktúrába szeretne behatolni, hogy ott a felhasználók tudta nélkül tevékenykedjen. Ebben a fejezetben a teljesség igénye nélkül a támadási formák fejlődését, majd a Windows alatti veszélyforrásokat, illetve az internetről érkező veszélyeket tekintjük át. Az egyes informatikai rendszereket potenciálisan veszélyeztető célzott támadási lehetőségekkel, a legújabb APT támadási módszerekkel, illetve a hálózati támadások eljárásaival egy-egy alfejezetben külön foglalkozunk.

1.1. KÁRTEVŐK TÖRTÉNETE, FEJLŐDÉSE

A számítógépes kártevők alapötlete az 1940-es évekre vezethető vissza. Neumann János ugyanis nemcsak a mai számítógépek (tárolt programú automaták) működését tervezte meg, hanem az "Önreprodukáló automaták elmélete" című tanulmányában a terjedésre képes számítógépes kártevők működését is leírta ([5]).

A szobányi területet elfoglaló nagygépes rendszereken jelentek meg az első féregprogramok. (Az egyik első ilyen kártevő a *Morris féreg* volt az 1970-es években.) Az operációs rendszer réseit használták ki, viszony főként csak az adott nagygépes rendszerben voltak képesek terjedni, hiszen akkoriban nem volt kapcsolat ezen számítógépek között. Céljuk általában az információszerzés (például jelszótáblák) volt. Nem irtották őket, ehelyett inkább kijavították az operációs rendszer hibáit, így nem tudtak elterjedni.

Az első IBM PC-kompatibilis számítógép megjelenését követő néhány éven belül, 1986-ban jelent meg az első PC-s vírus, a Brain, mely floppy-lemezek, illetve a merevlemez boot szektorát fertőzte meg. Néhány évvel később Magyarországon is megjelentek az első vírusok: 1988 őszén bukkantak fel a Cascade, illetve a Vienna programvírusok első változatai.

A 90-es évek elején a legelterjedtebb operációs rendszer a DOS volt, a Windows programrendszert is DOS alól kellett indítani. A Microsoft 1995-ben jelentette meg a DOS nélküli Windows operációs rendszert, illetve ekkor jelent meg az Office programrendszer is. A kártevők készítői hamar felfedezték az új rendszerekben rejlő lehetőségeket, és új irányokat kerestek. 1994-ben jelent meg az első Word alatti makróvírus kísérleti példánya (DMV), illetve a következő évben a Concept makróvírus fertőzte végig a világ számítógépeinek a nagy részét.

A makróvírusok megjelenéséhez hasonló áttörést jelentett az internet megjelenése is. A 2000-es évektől már a legtöbb fertőzést az interneten terjedő kártevők, elsősorban férgek (például Lovegate, Melissa) jelentették. A férgeken túlmenően egyre inkább tért hódítottak a phishingek, spamek, hoaxok is.

Az internet nem csupán a terjedés eszköze lett, hanem egyre inkább a támadások célpontját is jelentette. A kártevők készítői hamar felismerték, hogy hatékonyan vihetik végbe támadási törekvéseiket, ha a megtámadott számítógépekből hálózatot (botnet) alakítanak ki, és e számítógépeket együttesen veszik igénybe a támadás céljának megfelelően. Az internet ugyanakkor egyre bővülő reklámfelületet jelentett és jelent, amit spamüzenetek küldésére, reklámablakok megjelenítésére használnak ki. A nagy volumenű támadások mellett ugyanakkor egyre kifinomultabb módszereket használnak specializált kémtevékenységre is.

Egy futtatható kártékony kódot tartalmazó kártevő célja minden esetben az, hogy a kártékony kód a megtámadandó számítógépen végrehajtódjon. A számítógépek felépítése szerint a futtatandó programkódnak a memóriában kell lennie és a processzornak végre kell hajtania azt. Egy kártevőt a számítógép memóriájából a legegyszerűbben a kikapcsoló gomb használatával távolíthatunk el. Ekkor ugyanis a számítógép memóriájának tartalma és ezzel együtt a kártevő kódja is törlődik. A kártékony programoknak – annak érdekében, hogy működőképességüket megőrizzék – az a céljuk, hogy képesek legyenek arra, hogy a későbbiekben (például a számítógép következő elindulásakor) aktivizálódjanak. Ennek érdekében a futtatható kódot tartalmazó kártevőknek két lényeges feladatot kell elvégezniük. Egyrészt a saját programkódjukat olyan helyre kell tenniük, aminek a tartalma a számítógép kikapcsolását követően is megmarad, másrészt gondoskodniuk kell arról, hogy ez a programterület a számítógép későbbi működése során automatikusan vagy valamilyen esemény hatására végrehajtódjon.

1.2. ÖNREPRODUKÁLÓ KÁRTEVŐK (VÍRUSOK, FÉRGEK)

A számítógépes kártevők egy jelentős csoportját képviselik azok a kártékony programok, melyek képesek arra, hogy önmagukat másolva szaporodjanak. Amennyiben egy támadó nem célzott támadást szeretne végrehajtani, az önmagát másoló mechanizmust tudja kihasználni annak érdekében, hogy a kártékony kód sok számítógépre eljusson. Ilyenkor a kártékony program kódja általában két fő részből áll: egyrészt a programkódot megvalósító részből, másrészt pedig a kártékony tulajdonságot megvalósító programrészből. Ez utóbbi nem kötelező része a kártékony kódnak, csupán opcionális. Az önmagukat másoló, szaporodásra képes kártevők egy jelentős része ugyanis a szaporodáson kívül semmi egyebet nem tesz. Ez viszont nem jelenti azt, hogy az ilyen programkódok ne lennének kártékonyak. A szaporodási eljárás ugyanis önmagában kártékony, hiszen igénybe veszi a számítógép erőforrásait.

Az önmagukat másoló, „szaporodni” képes kártevők két csoportját különböztethetjük meg aszerint, hogy szaporodásukhoz szükséges-e hordozóprogram, vagy sem. A hordozóprogramot igénylő kártevőket *vírusoknak*, a hordozóprogram nélkülieket pedig *férgeknek* nevezzük.

1.3. VÍRUSOK TÍPUSAI, MŰKÖDÉSI ELVÜK

A számítógépes vírusok olyan kártékony programok, melyek képesek arra, hogy önmagukat másolva szaporodjanak. A szaporodási mechanizmus során először keresnek egy programterületet, majd a saját kódjukat ehhez a programterülethez illesztik. Gondoskodnak továbbá arról, hogy ha az operációs rendszer vagy a felhasználó a megfertőzött programot végre szeretné hajtani, akkor a kártékony kód is lefusson.

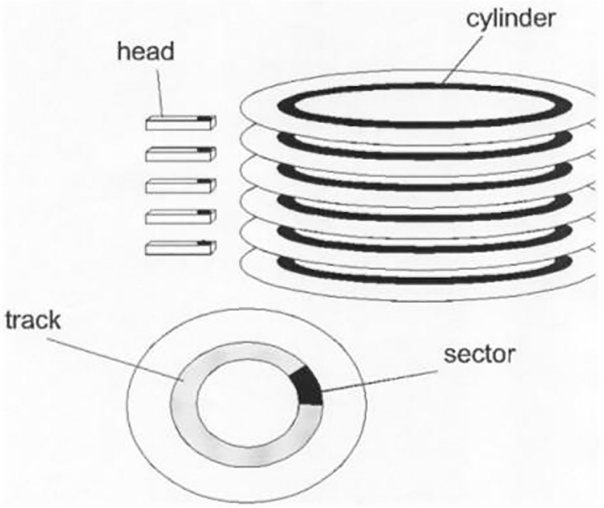
A számítógépen belül a vírusok által célba vett programterületek sokfélék lehetnek, aszerint, hogy a programkód végrehajtását az operációs rendszer vagy esetleg valamely alkalmazás végzi. Különbséget jelent továbbá, hogy a végrehajtás valamilyen esemény hatására automatikusan történik-e, vagy felhasználói beavatkozást igényel.

A számítógépes vírusok fertőzésük során a számítógép háttértárának valamely programterületét fertőzik meg. A vírusok működésének a megértéséhez alapvető szükség van a háttértárolók belső felépítésének a megismerésére.

A diszkek fizikai felépítése

A diszkek kétdimenziós felületű lemezekből épülnek fel. A floppy-diszkek egy lemezt, a merevlemezek általában több lemezt tartalmaznak. A lemezeknek mind a két oldalát használják, mind a két oldalon találhatók író/olvasó fejek (head). Előfordulhat azonban, hogy a merevlemez egy lemezoldalát szinkronizálási célokra használják. A lemezoldalak kétdimenziós felépítésűek: sávoknak (track, vagy cylinder) nevezett körgyűrűkre, a körgyűrűk pedig szektorokra vannak osztva. Egy szektor mérete a diszk formázásának a függvénye, a legelterjedtebb operációs rendszer alatt általában 512 byte.

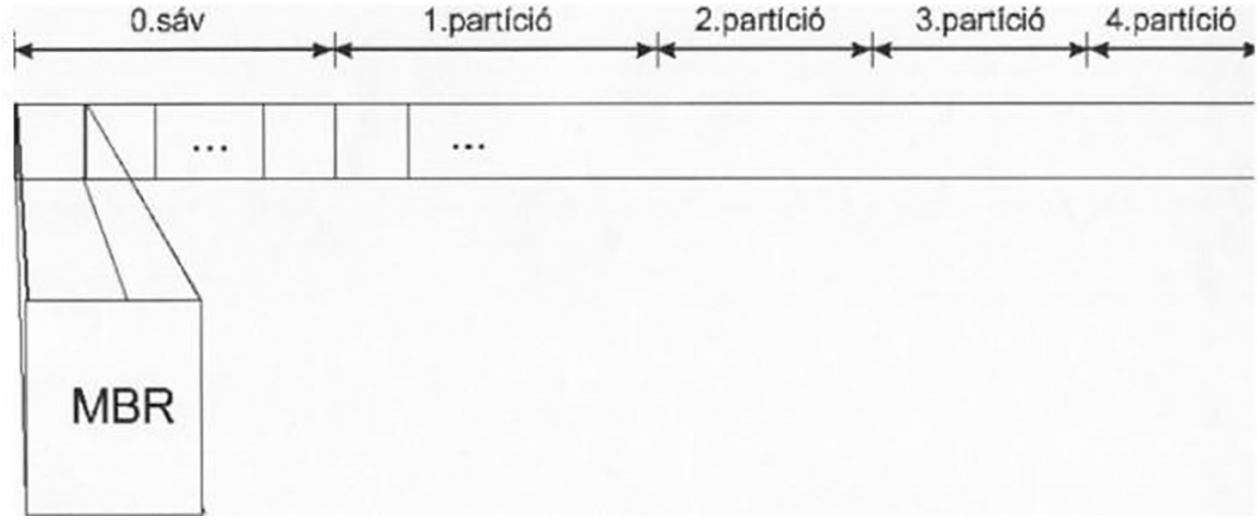
Mind az egyes lemezoldalak, azaz az író-/olvasófejek (head), mind a sávok (cylinder), mind pedig a sávokon belüli szektorok (sector) sorszámozottak. A szektorszám 1-től, a cylinder- és a head-szám pedig 0-tól kezdődik. Így egy szektort a „cylinder, head, sector” hármassal azonosíthatunk. A merevlemez szektorainak ezt az azonosítási módját fizikai címzésnek nevezzük. A merevlemez szektorai logikailag egy meghatározott sorrendet alkotnak. Ebben a sorrendben az első helyen a 0. cylinder 0. head-jének szektorai állnak (1-től kezdődően). Ezt követik a 0. cylinder 1. head-jének szektorai, majd a 0. cylinder utolsó head-jének szektorai után az 1. cylinder 0. head-jének szektorai következnek. Így a merevlemez egy szektorára a szektor sorszámával is hivatkozhatunk. A szektoroknak ezt az azonosítási módját abszolút címzésnek nevezzük, ahol tehát az 1-es abszolút című szektor fizikai paraméterei: 0. cylinder, 0. head, 1. sector. A merevlemezek paramétereit (cylinder-szám, head-szám, szektor-szám) a CMOS-memória tartalmazza.



1. ábra: Lemezek fizikai felépítése

Diszkek logikai felépítése

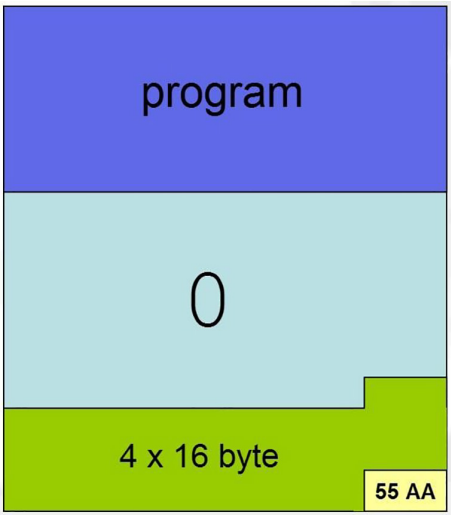
A diszkeket kezelő operációs rendszerek a merevlemezeket logikai egységekre, partíciókra osztják. A partíciókra osztás operációs rendszertől, hálózati rendszertől független. Az egyes partíciók információit (kezdet, vég, bootolható vagy sem) a partíciós táblázat tartalmazza, amely a Master Boot Recordban (MBR) található. Az MBR a partíciók információin, a partíciós táblázaton túlmenően egy programrészletet is tartalmaz, amely a gép bekapcsolásakor, illetve minden bootoláskor hajtódik végre.



2. ábra: Lemezek logikai felépítése: merevlemezek partíciókra osztása

Mint az az ábrán látható, a 0. cylinder 0. head 1. szektora az MBR, a 0. cylinder 0. head-jének többi szektora pedig nem tartozik egyetlen partícióhoz sem. Általában igaz ugyanis, hogy az első partíció nem az MBR-t követő szektorral kezdődik, hanem a 0. cylinder 1. head 1. szektorral.

A partíciós táblázat maximum négy partíció információit tartalmazza. Az operációs rendszerek ennél több logikai partíciót is képesek kezelni. Ehhez úgynevezett kiterjesztett (Extended) partíciót hoznak létre, amelyet logikailag további egységekre osztanak.

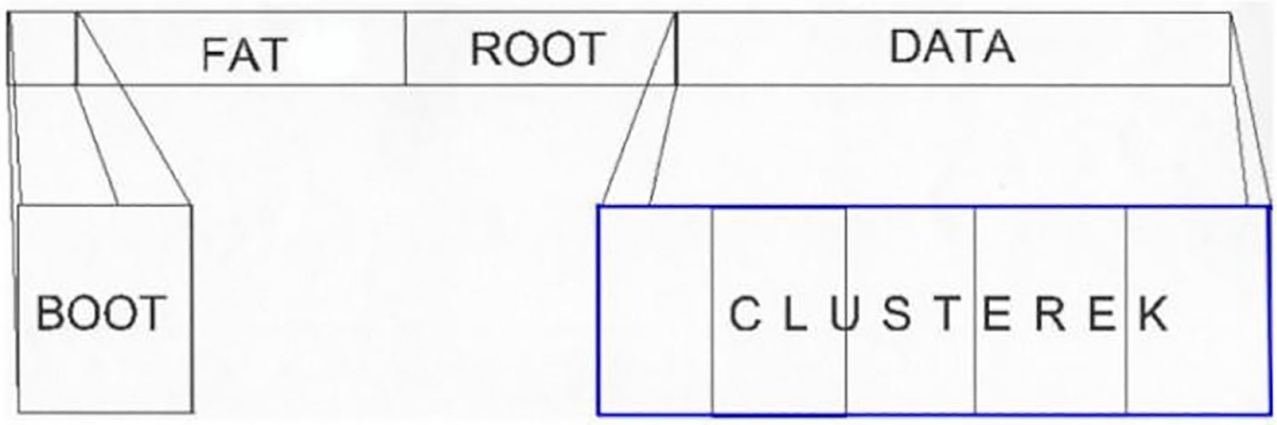


3. ábra: Az MBR felépítése

A floppy-lemezek felépítése a merevlemezektől annyiban tér el, hogy a floppy-lemezeken nincsenek partíciók, nincs szükség MBR-szektorra sem. A floppy-lemezeken összesen egy partíció található, ami pontosan ugyanúgy épül fel, mint a merevlemez egy partíciója.

Partíciók felépítése

Az operációs rendszer a partíciókat, mint logikai egységeket a partíció típusától függően további részekre osztja. Egy FAT- (FAT12, FAT16, FAT32) partíció első szektora tartalmazza a partíció boot-szektorát. Ezt követi a File Allocation Table (FAT, File Allokációs Tábla) két példányban, majd a Root Directory (Főkönyvtár), és végül a Data Area (Adatterület), amely a partíción elhelyezett alkönyvtárakat, illetve fájlokat tartalmazza. A boot-szektor felépítése operációs rendszerenként más és más, azonban közös jellemzőjük, hogy bootolható partíció esetén tartalmaznak egy programrészletet, mely az operációs rendszer betöltését hívatott elindítani.

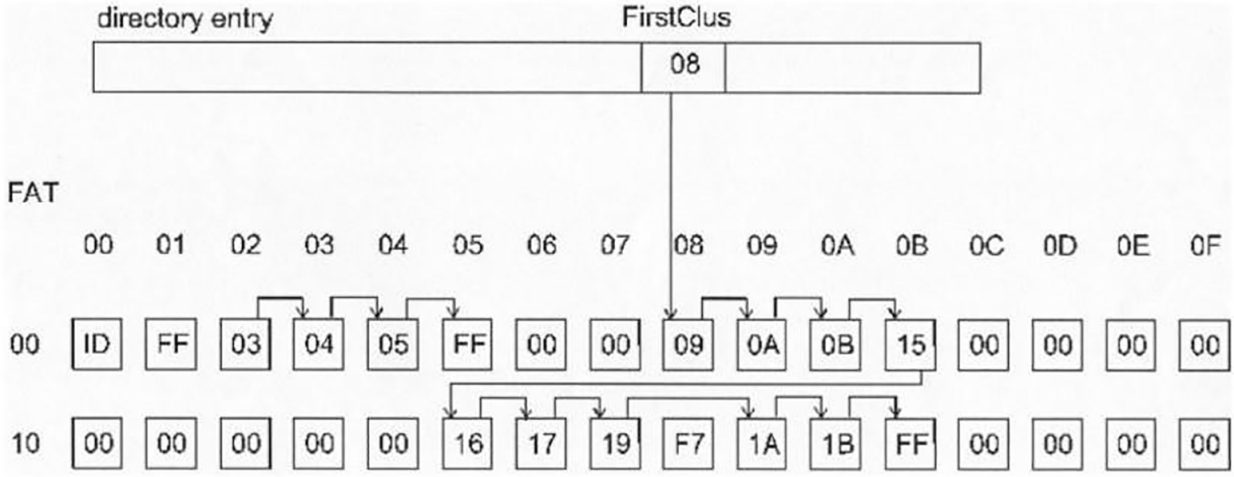


4. ábra: A FAT-partíció felépítése

A FAT-partíción lévő fájlok és alkönyvtárak a partíció adatterületén találhatók. Az adatterület clusterekre osztódik, melyek 2-től kezdődően sorszámozottak. Egy-egy cluster a szektorméret egész számú, a 2 valamely hatványának megfelelő többszöröse. A clusterek tartalmazzák az alkönyvtárak és a fájlok információit. Minden alkönyvtárhoz és minden fájlhoz (ha az nem 0 byte hosszúságú) egy cluster-sorozat tartozik. Alkönyvtárak esetén ezek a clusterek tartalmazzák az alkönyvtárban elhelyezett további alkönyvtárak, illetve fájlok információit. Minden fájlhoz és alkönyvtárhoz tartozik egy könyvtárbejegyzés (directory entry), amely tehát a "tartalmazó" alkönyvtár cluster-sorozatában, míg a főkönyvtár esetén a főkönyvtárnak fenntartott helyen található. Ez utóbbi független a clusterektől, egy fix hosszúságú szekvenciális szektorsorozatot jelent.

A könyvtárbejegyzés tartalmazza a fájlok, illetve alkönytárak cluster-sorozatának első elemét (FirstClus). A cluster-lánc további tagjait a FAT írja le (5. ábra). A FAT-tábla minden egyes clusterhez tartalmaz ugyanis egy értéket, ami a következő cluster-számot jelenti. A FAT-bejegyzések a partíció cluster-számának függvényében 12, 16 vagy 32 bitesek lehetnek. A FAT-bejegyzések 0-tól sorszámozottak. Az első két FAT-bejegyzés speciális jelentéssel bír. Ennek első bájtja a lemezegység médialeíró bájtja, melynek meg kell egyeznie a boot-szektor médialeíró bájtjával. A többi bájt (12 bites FAT esetén 2 byte, 16 bites FAT esetén 3 byte) értéke FFh. A FAT-bejegyzések a láncoláson túlmenően speciális tartalommal is rendelkezhetnek. Ezek a következők:

(0)000h üres cluster.
(F)FF0h - (F)FF6h fenntartott cluster.
(F)FF7h hibás cluster.
(F)FF8h - (F)FFFh a cluster-lánc vége.



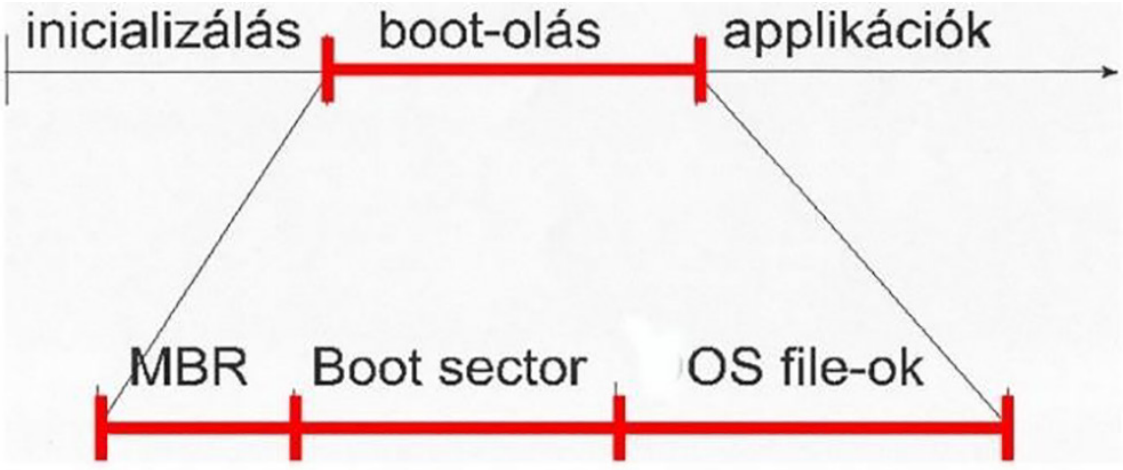
5. ábra: Clusterek láncolása, a FAT-lánc
(Az ábrán szereplő példában az egyszerűség kedvéért a FAT 8 bit méretű.)

Indítási, bootolási folyamat

A számítógép bekapcsolását követően, a bootolás során a ROM-ban lévő BIOS-program indul el, amely elvégzi a szükséges inicializálásokat, teszteléseket, majd betölti a boot-lemez fizikailag első szektorát (0. head, 0. cylinder, 1. sector), és a betöltött szektor elejére adja a vezérlést. Floppy-lemezek esetén (ahol egy partíció található), a partíció első szektora hajtódik végre. Merevlemezek esetén az MBR töltődik be, melynek programja megkeresi partíciós táblázatban az első bootolható partíciót, betölti annak első szektorát és annak az elejére adja a vezérlést. A boot-szektorban lévő program már operációsrendszer-függő, az operációs rendszert tölti be.

Boot-vírusok

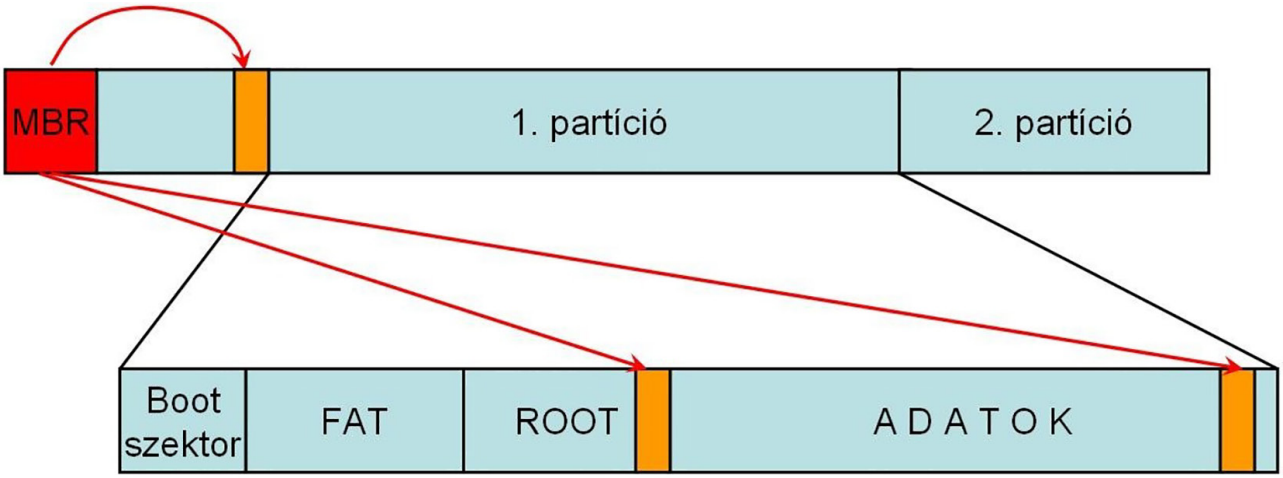
A boot-vírusok közös jellemzője, hogy azelőtt aktivizálódnak, mielőtt a felhasználó – bármilyen interaktivitása révén – elindíthatna programokat, alkalmazásokat. A boot-vírusok a merevlemezek MBR-jét, illetve a floppy- és a merevlemezek partícióinak boot-szektorát fertőzhetik meg.



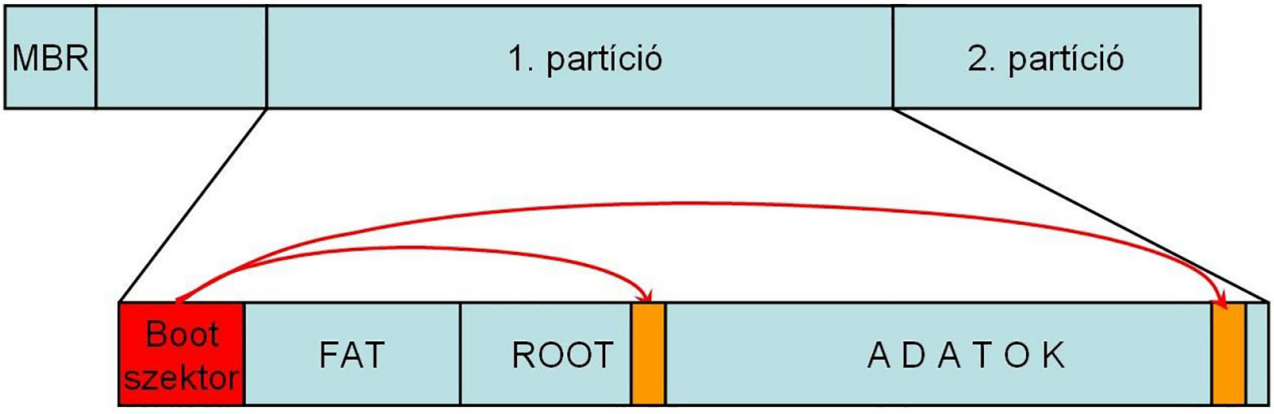
6. ábra: Boot-vírusok lehetséges betöltődése

A fertőzés során felülírják a boot-folyamatnak az adott szektorban lévő programrészletét. Annak érdekében azonban, hogy a felhasználó ne vegye észre azonnal a fertőzést, gondoskodnia kell arról is, hogy az operációs rendszer elindulása zavartalanul megtörténjen. Ezt két módon oldhatja meg:

- Elláthatja az eredeti programkód feladatát. Ezt főként az MBR fertőzéseként teheti meg, ugyanis az MBR-ben lévő programrészlet nagyon kicsi és szabványos feladatot lát el. (Az MS-DOS 3.3-astól kezdődően az MBR ugyanazt az egyszerű programkódot használja.)
- Elmentheti az eredeti szektort, amit aztán, ha végzett a tevékenységével, betölt és lefuttatja az abban lévő kódot. Az eredeti szektort általában az alábbi helyekre mentheti el:
    - Merevlemezeken az MBR-t követő, az első partíció előtti szektorokba.
    - Floppy- és merevlemezek utolsó szektoraiba, bízva abban, hogy a lemez nem fog megtelni és az operációs rendszer nem írja felül. A fájlrendszer ismeretében bejegyezheti (FAT-es partíció esetén a FAT-be), hogy a felhasznált szektor hibás, azt az operációs rendszer ne használja.
    - Floppy- és merevlemezek partícióiban a főkönyvtárnak fenntartott területbe. Ez a terület alapértelmezés szerint 512 bejegyzést képes tárolni, azonban ennyi bejegyzés ritkán fordul elő egy főkönyvtárban.



7. ábra: Az eredeti MBR mentési lehetőségei



8. ábra: Az eredeti boot-szektor mentési lehetőségei

Fájlvírusok

A fájlvírusok olyan önmagukat másolni képes kártevők, melyek az operációs rendszer által közvetlenül végrehajtható állományokat fertőznek meg. Az operációs rendszerek különböző formátumú végrehajtható állományokat kezelnek. A fájlvírusok az egyes formátumokhoz különböző fertőzési módszereket használnak. Az alábbiakban példaként a legelterjedtebb módszereket tekintjük át. A legegyszerűbb végrehajtható formátumú állományok a DOS és a Windows operációs rendszerek által

kezelt .COM fájlok. A .COM állományok pontosan azt a bitsorozatot tartalmazzák, ami a futtatás kezdetén a memóriába kerül és a processzor az állomány első bájtján kezdődő utasítással kezdi a végrehajtást. A fájlvírusok fertőzésének legáltalánosabb módszere szerint a kártevő a saját programkódját a megfertőzendő állomány elejére, közepére, illetve a legtöbb esetben a végére írja. Amennyiben a kártevő kódja nem az állomány elejére kerül, akkor gondoskodnia kell arról, hogy a program indításakor a saját kódja hajtódjon végre elsőként. Ezt úgy éri el, hogy miután elmentette az eredeti állomány első néhány bájtját, felülírja a fájl elejét egy olyan ugróutasítással (JMP), ami a vezérlést a kártékony kód elejére adja. Miután a kártékony kód lefutott, az előzőleg elmentett eredeti első néhány bájtot visszamásolja eredeti helyére, és a végrehajtást ott folytatja.

9. ábra: .COM állomány fertőzése (a kártevő a fájl végére írja magát)

10. ábra: .COM állomány fertőzése (a kártevő a fájl elejére írja magát - I)

11. ábra: .COM állomány fertőzése (a kártevő a fájl elejére írja magát - II)

12. ábra: .COM állomány fertőzése (a kártevő a fájl közepére írja magát)

A fájlvírusok fertőzési módszerei között olyanok is léteznek, melyek az állomány speciális formátumát, esetleg az operációs rendszer valamely speciális kezelési módját használják ki. Ilyen módszerek az alábbiak:

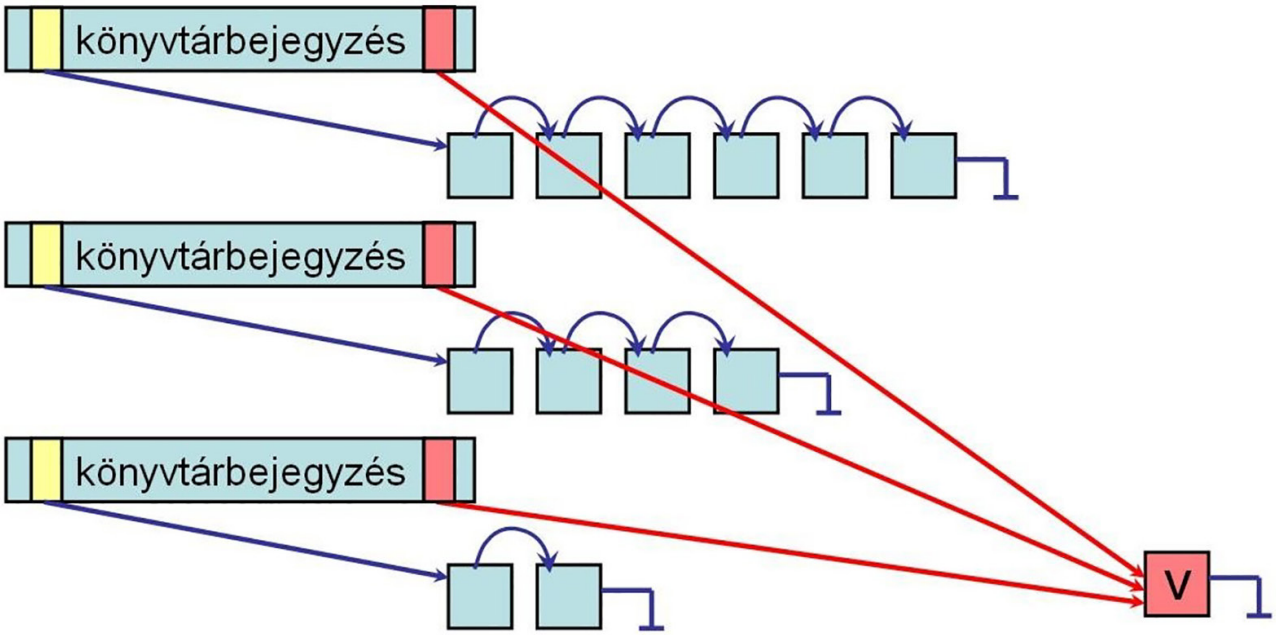
- A DOS és a Windows operációs rendszerek által értelmezett .EXE fájlok egy speciális fejrészt tartalmaznak a formátumra jellemző speciális adatokkal, illetve egy változó méretű táblázattal. Ugyanakkor számos .EXE állományban nincs vagy csak nagyon kicsi táblázat található a fejrészben, ami ilyenkor 0 értékű bájtokkal van feltöltve. Az ezeket az állományokat fertőző vírusok

megtehetik azt, hogy a saját programkódjukat az ilyen “csupa 0” területre írják, majd ha végeztek a tevékenységükkel, akkor a területet 0 bájtokkal feltöltve visszaállítják az eredeti állapotot. Ezzel a módszerrel a vírus fertőzése nem okozza az eredeti állomány méretének a növekedését.

A módszer nem csupán az .EXE formátumú állományok esetén használható, hanem bármilyen olyan program esetén is működőképes, ahol az állományban egymást követően elegendő mennyiségű azonos bájt található. Számos vírus a DOS, illetve a Windows COMMAND.COM állományát képes megfertőzni hasonló módon.

13. ábra: .COM állomány fertőzése méretnövekedés nélkül

– A CEB vagy más néven “companion” vírusok az operációs rendszer (DOS, Windows) automatikus programvégrehajtási sorrendjét használják ki. Ha a számítógép felhasználója ugyanis kiterjesztés nélkül indít el a DOS-ban vagy a Windows parancssorában egy programot, akkor az operációs rendszer megvizsgálja, hogy az aktuális könyvtárban létezik-e egy a program nevének megfelelő .COM állomány. Ha létezik, akkor elindítja, ha nem létezik, akkor keres tovább, előbb az .EXE, majd a .BAT kiterjesztéssel rendelkezőket. Ha az aktuális könyvtárban nem jár sikerrel, akkor az elérési út (path) könyvtárai következnek, a kiterjesztések ugyanilyen (.COM, .EXE, .BAT) sorrendjében. Ezt a program-végrehajtási sorrendet számos vírus kihasználja. Egyszerű esetben az .EXE állomány mellé, ugyanabban a könyvtárban létrehoznak egy .COM kiterjesztésű állományt, így a program kiterjesztés nélküli indítása esetén ez fog elindulni. A .COM fájl tartalmazza a kártevő kódját, majd a működését követően pedig elindítja az .EXE kiterjesztésű fájlt is. Bonyolultabb esetben a CEB-vírusok az elérési utat is figyelembe veszik (path companion vírusok) és egy sorban korábbi könyvtárba helyezik el a kártevőt tartalmazó fájlt.
– A fájlrendszer sajátosságát használják ki a FAT-vírusok. Az eredeti állományokat nem változtatják meg, sőt a CEB-hez hasonlóan újabb állományokat sem hoznak létre. Ezzel szemben a FAT-vírusok a floppy- vagy merevlemezre egyetlen példányban kerülnek fel. A fájlrendszer végén, az adatterületen belül néhány clustert foglalnak el, itt helyezik el saját kódjukat, és természetesen a FAT-nyilvántartásba bejegyzik, hogy ezek a clusterek foglaltak, azokat az operációs rendszer ne is próbálja meg használni. A kártevő kódját tartalmazó cluster-láncot aztán a “megfertőzendő” állományhoz csatolják oly módon, hogy a könyvtárbejegyzésnek az első clusterre mutató mezőjét módosítják. Így az összes megfertőzött állomány bejegyzése ugyanoda fog mutatni, indításukkor a kártevő kódja indul el. Elindulást követően a vírus bekerül a memóriába és minden könyvtárművelet esetén visszaállítja a könyvtárbejegyzést eredeti állapotába. Így ha a vírus aktív, akkor minden állomány az eredeti formájában olvasható, ha nem aktív, akkor minden fertőzött fájl csak a vírus kódját tartalmazza.



14. ábra: FAT-vírusok fertőzési módszere

Makróvírusok

A makróvírusok olyan, önmagukat másolni képes kártevők, melyeket nem az operációs rendszer indít el közvetlenül, hanem az operációs rendszeren futó valamely alkalmazás értelmez és hajt végre. Ilyenkor a vírus kódja az alkalmazás adatállományában található az alkalmazás valamilyen saját makrónyelvén, melyet csak az adott alkalmazás, esetleg az adott formátumot feldolgozni képes más alkalmazás tud értelmezni és végrehajtani. A makróvírusok közös jellemzője, hogy csak a makrónyelv lehetőségein belül mozoghat, azonban a legelterjedtebb Microsoft Office alkalmazások alatti makróvírusok esetén a makró nyelve semmilyen kötöttséget nem tartalmaz. A Microsoft Word, Excel, Powerpoint, … alkalmazásokban a makróprogram bármit megtehet, amit egy, az operációs rendszer alatt futó végrehajtható program megtehet. A makróvírusok fertőzésének a célterülete mindezek alapján az alkalmazások adatállományai, melyek makróprogramokat is tartalmazhatnak.

A makróvírusok történetében a legelső alkalmazás, ami alá makróvírusok készültek, a Microsoft Word volt (a DMV, majd a Concept makróvírusok). A Word alapvetően kétfajta adatállományt kezel: a *dokumentumot* és a *sablont*. A dokumentum tartalmazhat szöveget, ábrát, formátumokat, illetve a stílusok (betű, bekezdés, keret, tabulátor, …) is itt találhatók. A sablon mindezeken felül makrókat, gyorsszövegeket, illetve menü-, eszköztár- és gyorsbillentyű-beállításokat is tartalmazhat. Látszólag azt mondhatjuk, hogy egy dokumentum nem is tartalmazhat makrót, így makróvírus sem lehet benne. Ez így is van, egy dokumentum típusú állományban nem lehet makró. Azonban a Word a kezelt állomány típusát NEM annak kiterjesztése (.DOC vagy .DOT) alapján dönti el, hanem annak tartalma alapján. A két típust viszont csak egyetlen bit különbözteti meg egymástól és ráadásul ezt az egy bitet a makróprogramok is felülírhatják. Így viszont egy makróvírus megteheti azt, hogy átdefiniálja az adatállomány típusát, majd beleírja saját kódját a dokumentumba, amely immár sablon típusú lesz, de a .DOC kiterjesztése megmarad. A Word makróvírusok elsődleges célpontja a NORMAL.DOT állomány. Ez a sablon minden Word indításkor betöltődik, az ebben lévő makróprogramok automatikusan aktivizálódnak. A Word makróvírusok tehát az adatállományok makróterületeire mentik el saját kódjukat. Aktivizálódásukat viszont a Word számos beépített lehetőségével, esetleg lehetőségeivel oldják meg:

- Az automakrók lehetőséget adnak arra, hogy a Word bizonyos esemény hatására egy makróprogramot lefuttasson. Néhány automakró:
  - AutoExec: a Word indításakor mindig lefut;
  - AutoExit: a Wordből való kilépéskor mindig lefut;
  - AutoNew: új dokumentum nyitásakor fut le;
  - AutoOpen: létező dokumentum lefutásakor hajtódik végre;
  - AutoClose: egy dokumentum bezárásakor hajtódik végre.

Az automakrók közül a leggyakrabban használt az AutoClose. Ennek oka, hogy az AutoClose már a dokumentum bezárásakor, annak mentésekor fut le. Ekkor a felhasználó számára kevésbé feltűnő, ha esetleg a rendszer az írásvédelem eltávolítását kéri, hiszen a felhasználó maga is szeretné elmenteni a dokumentumot, így kevésbé fog gyanakodni.

- A Wordben lehetőség van a belső parancsok (például FileSave) átdefiniálására is, de módosíthatók a leütött billentyűkhöz, az eszköztár gombjaihoz, illetve az egyes menüpontokhoz tartozó programok is.
- Egy makróvírus megteheti azt is, hogy a dokumentumba helyez el egy űrlapmezőt, amihez hozzárendeli a saját programkódját. Ebben az esetben a makróvírus kódját maga a felhasználó aktivizálja.

A Microsoft Wordnek számos változata létezik. Az első makróvírusok a Word 2.0-ás verzió idején jelentek meg. Az Office 97-ben szereplő Word 97 volt az első, amiben már az addigi WordBasic helyett VBA-t (Visual Basic for Applications) használtak. A VBA a Word verzióiban különböző utasításokkal rendelkezett, nyelvfüggő függvényhívásokat tartalmazott. A Microsoft annak érdekében, hogy a régebbi, WordBasic alatti makróprogramokat is lehessen használni az új környezetben, készített egy automatikus konvertáló eljárást, amelyet minden Word 97-es verzióba beleépített. Ennek az lett az eredménye, hogy a Word 97 a régi WordBasic-es vírusokból automatikusan a VBA alatt is működőképes vírusokat generált, sőt minden egyes WordBasic-es vírushoz több tucat különböző változatú VBA-s példányt hozott létre. A hivatalosan kiadott Office 97 már rendelkezett vírusvédelemmel, viszont a védelem egyrészt a béta változatból kimaradt, másrészt összesen 21 darab makróvírust ismert az akkor létező kb. 1200-ból.

A Word alatti makróvírusok a makrónyelv lehetőségei között mozoghatnak, azonban a makrónyelv mindent megenged. Így a makróvírusok is végezhetnek könyvtár- és fájlműveleteket, közvetlenül elérhetik a háttértárat is. Amakróvírusok által leggyakrabban megvalósított károkozási lehetőségek azonban magához a Word alkalmazáshoz kötődnek. Például üzeneteket, ábrákat jelenítenek meg, oldalakat nyomtatnak, a szövegben betűket, szavakat cserélnek, esetleg jelszóval védik le a dokumentumot.

A Word mellett az Office más alkalmazásai, illetve más makrónyelvet is kezelő alkalmazások hasonlóan védtelenek a makróvírusokkal szemben. Az Office-hoz tartozó Excel- ben az 5.0-ás verziótól a Word-höz hasonló VBA található, itt is megvannak az automakró-lehetőségek és még az adatállomány típusával sem kell trükközni, minden állomány (.XLS, .XLT) ugyanúgy tartalmazhat makróprogramot.

1.4. VÍRUSOK TULAJDONSÁGAI

A számítógépes vírusok fejlődésével a tulajdonságaik is folyamatosan fejlődtek. A kártevők készítői ugyanis nemcsak újabb és újabb vírustípusokat indítottak útjukra, hanem újabb tulajdonságokkal, képességekkel is felruházták programjaikat.

Parazita vírusok

A parazita vírusok terjedésük során, ha a kódjuk vezérléshez jut, akkor keresnek egy olyan programterületet, amit megfertőzhetnek, és megfertőzik azt. Ezt akkor teszik meg, ha a felhasználó elindít egy fertőzött programot. Ha a program végrehajtása befejeződik, akkor nem aktívak, nem kerülnek be a memóriába.

Rezidens vírusok

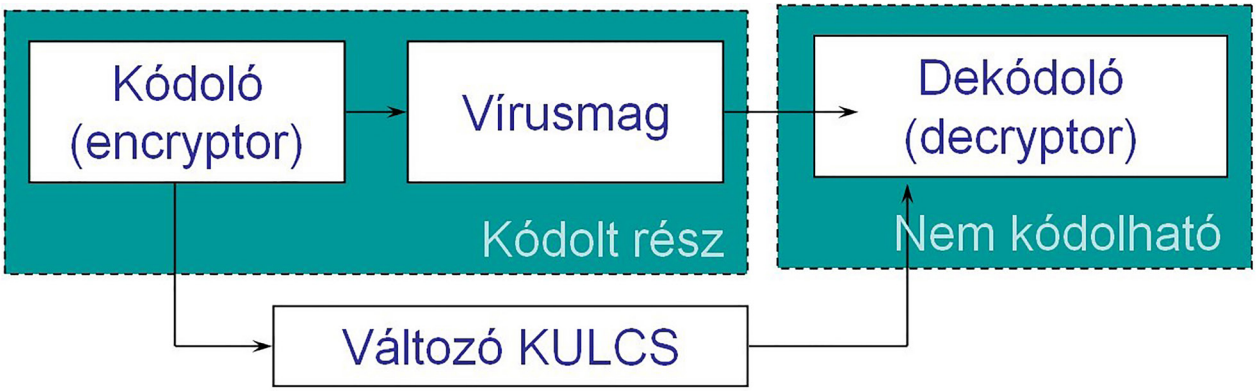
A rezidens vírusok működésük során bekerülnek a memóriába és bizonyos esemény hatására aktivizálódnak. Ilyen esemény lehet a programindítás, fájlok másolása, vagy akár egy fájllista lekérése valamely könyvtárról. Az aktivizálódás során végzik el fertőző tevékenységüket: ekkor keresnek olyan programterületet, amit megfertőzhetnek.

Lopakodó vírusok

A lopakodó vírusok célja, hogy elhitessék a felhasználóval, hogy a vírus nincs jelen a rendszerben. Fájlvírusok esetén az egyszerű lopakodó vírusok a megfertőzött állományt az eredeti mérettel mutatják, teljes lopakodó esetben pedig az eredeti fájl tartalmát is képesek szimulálni. Boot-vírusok esetén a lopakodó tulajdonsággal rendelkező kártevők a megfertőzött (MBR- vagy boot-) szektort az eredeti állapotában mutatják, ha a felhasználó vagy az operációs rendszer szeretné elolvasni. Makróvírusok esetén például a Microsoft Word alatt a makróvírusok több módon is képesek a felhasználó átverésére. Egyrészt képesek arra, hogy eltüntessék az Eszközök/Makrók menüpontot, másrészt megtehetik azt is, hogy az Eszközök/Makrók menüpontot meghagyva a makrókról egy üres listát mutatnak. Minden lopakodó vírusnak folyamatosan figyelnie kell a rendszer eseményeit ahhoz, hogy tevékenységét megvalósítsa, azaz rezidensen a memóriában kell lennie.

Mutációs vírusok

A számítógépes vírusokat egyszerű algoritmussal felismerhetjük, ha fertőzésük során mindig ugyanaz, vagy közel ugyanaz a bájtsorozat kerül a megfertőzendő állományok végére. A vírusok készítői hamar rájöttek arra, hogy alaposan megnehezíthetik a védekezést, ha a vírus kódja (akár fertőzésről fertőzésre) változtatja magát. Ennek érdekében olyan mutációs technikákat dolgoztak ki, amelynek során az eredeti állományhoz kerülő bájtsorozat mindig más és más. Ezt úgy érik el, hogy a kártevő kódját valamilyen titkosító algoritmussal titkosítják, amihez egy változó kulcsot használnak, és egy olyan dekódoló algoritmust (ez általában egy rövid ciklus) készítenek, mely képes a kódolt terület visszaállítására. Ily módon a kártevő kódjának nagy része mindig más és más lesz.



15. ábra: Dekódoló ciklussal rendelkező mutációs vírus felépítése

A mutációs vírusoknak három fő csoportját különböztethetjük meg:
- Oligomorf vírusok: Olyan mutációs vírusok, melyekben a dekódoló ciklus változatlan.
- Polimorf vírusok: Olyan mutációs vírusok, melyekben a dekódoló ciklus is változik, amihez az alábbi módszerek a legelterjedtebbek:
  - Hatástalan töltelékutasítások beszúrása.
  - Változók, regiszterek funkcióinak cseréje.
  - Azonos feladatot ellátó utasításblokkok változtatása.
  - Programkód elhelyezése több darabban.
- Metamorf vírusok: Titkosítást, dekódolást nem végeznek, az egész víruskódot úgy változtatják, mint ahogy a polimorf vírusok változtatják a dekódoló ciklusukat.

A mutációs vírusok elsősorban a fájlvírusokra jellemzőek, azonban a makróvírusoknál is előfordulnak.

1.5. FÉRGEK, MŰKÖDÉSI ELVÜK, TULAJDONSÁGAIK

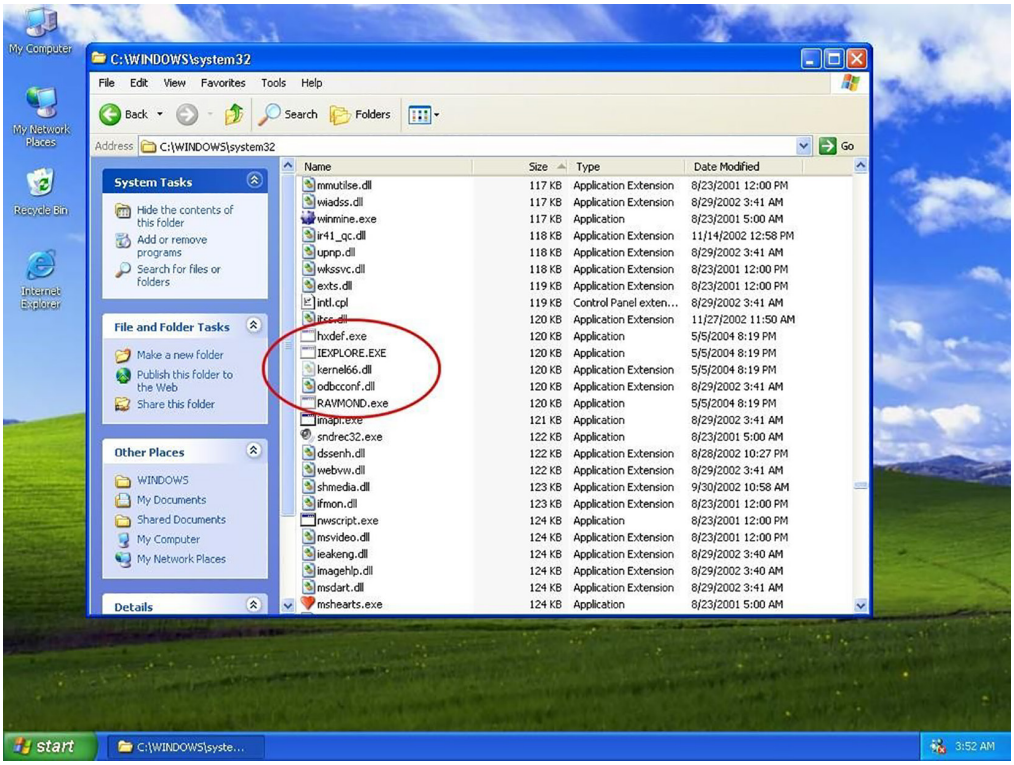
A férgek a vírusokhoz hasonlóan képesek arra, hogy önmagukat másolva szaporodjanak, azonban a vírusoktól eltérően terjedésükhöz nem szükséges hordozóprogram. A férgek nem másik programba épülnek be, céljuk egy másik rendszerbe történő beépülés. A vírusokhoz hasonlóan a férgeknek is el kell helyezniük a másik rendszerben a kódjukat, és gondoskodniuk kell arról, hogy a kód a számítógép bekapcsolásakor végre is hajtódjon. Az alábbiakban a Win32/Lovgate.Z féreg példáján tekintjük át a férgek leggyakoribb terjedési módszereit, tulajdonságait.

A Win32/Lovgate.Z egy féregprogram, mely a lokális hálózatokon és az interneten egyaránt képes terjedni. Több változata is ismert, melyek egy vagy több röptömörítővel (futtatható állományt képes tömöríteni) vannak csomagolva. Tömörített mérete kb. 120 kbyte, tömörítetlenül 206336 byte.

Telepítés

Futtatásakor a féreg bemásolja magát a Windows rendszerkönyvtárába (alapértelmezés szerint c:\windows\system32), a Windows könyvtárába (c:\windows), illetve az elérhető egységek főkönyvtárába. A Windows rendszermappájába a teljes féreg több néven is bekerül:

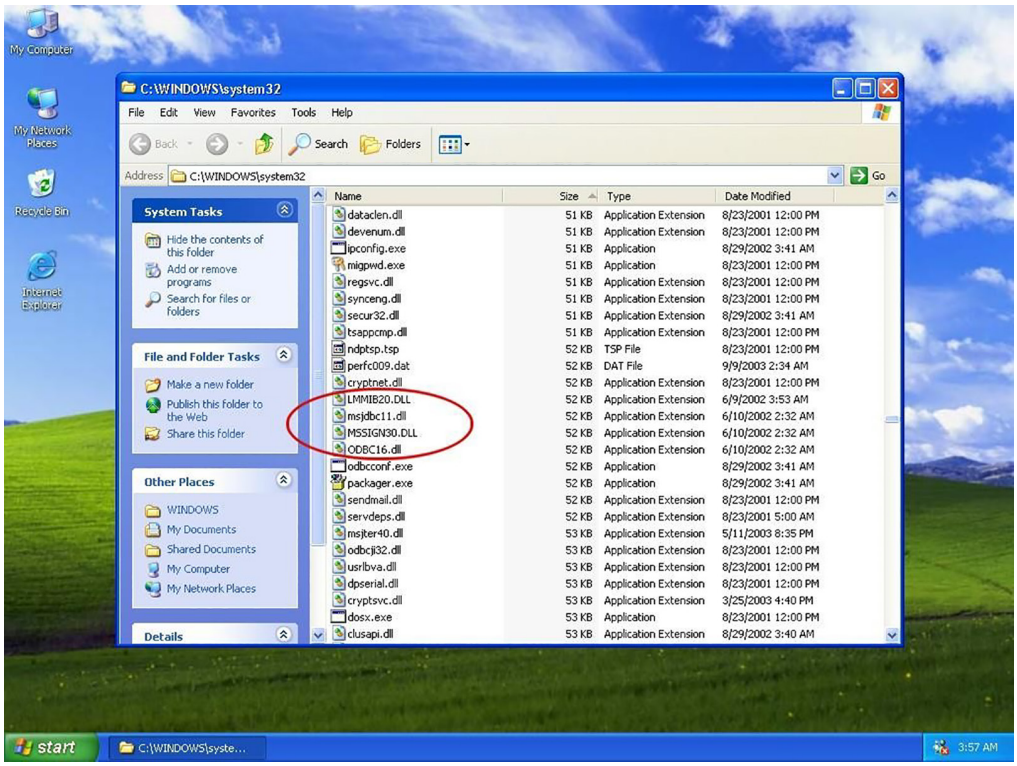
kernel66.dll
iexplore.exe
hxdef.exe
ravmond.exe



16. ábra: a Win32/Lovgate.Z féreg a Windows könyvtáraiban

Ugyanebbe a könyvtárba a féreg elhelyezi egy komponensét is (mérete 53248 byte):

odbc16.dll
msjdbc11.dll
mssign30.dll
lmmib20.dll

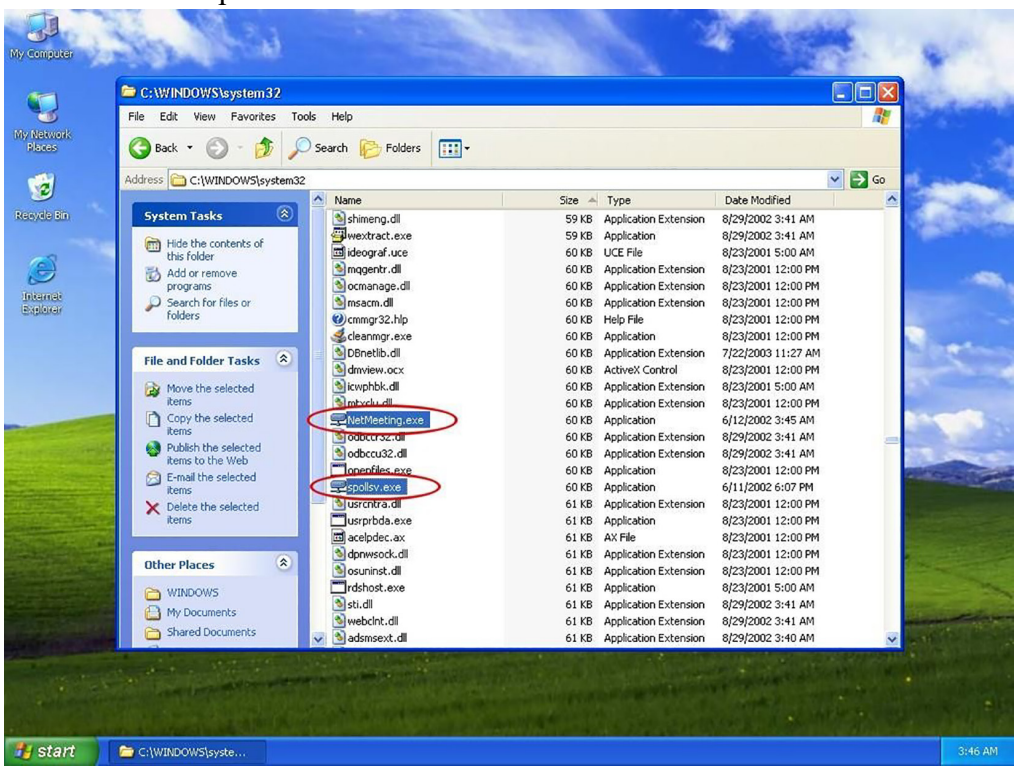


17. ábra: a Win32/Lovgate.Z féreg .DLL komponense a Windows könyvtáraiban

A féreg egy további .EXE kiterjesztésű komponenst is létrehoz (mérete 61440 byte):

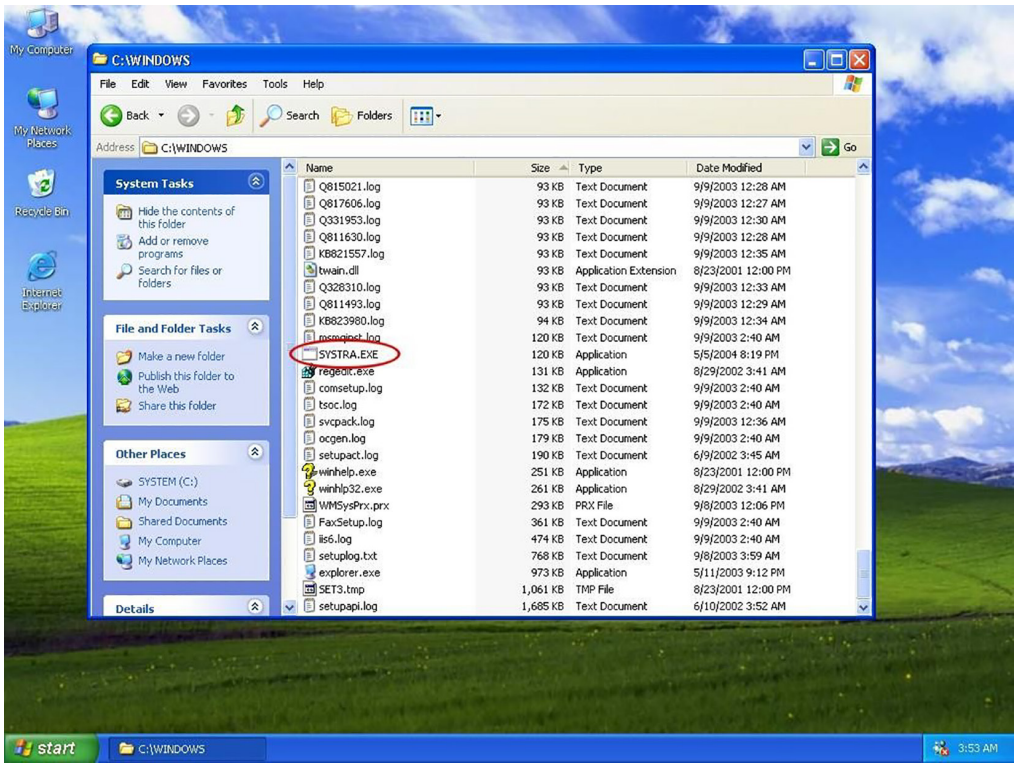
Netmeeting.exe

spoolsv.exe



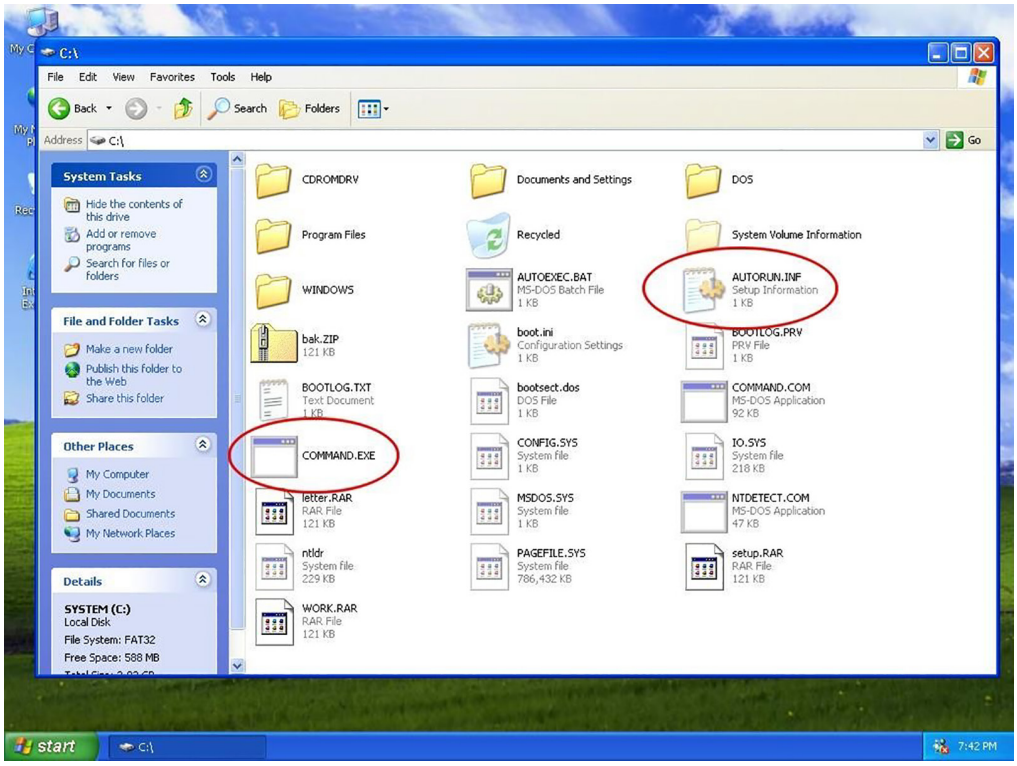
18. ábra: a Win32/Lovgate.Z féreg .EXE komponense a Windows könyvtáraiban

A Windows könyvtárába *systra.exe* néven is bemásolja magát.



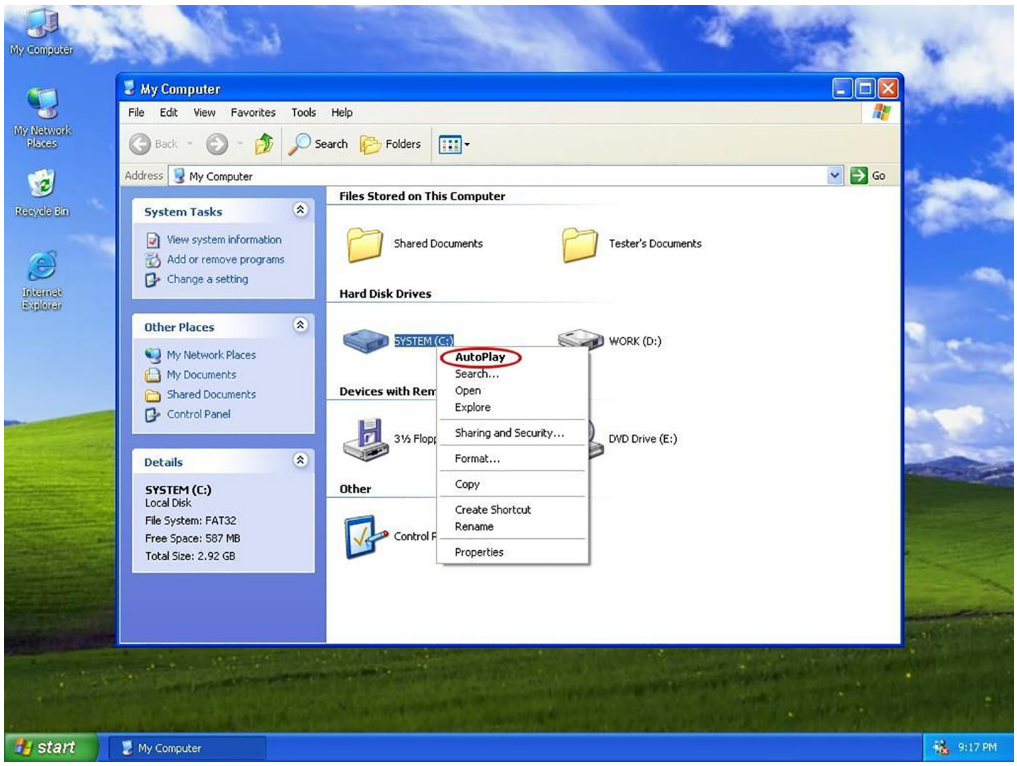
19. ábra: a Win32/Lovgate.Z féreg a Windows könyvtáraiban

Az elérhető lemezegységek főkönyvtárába *command.exe* néven kerül be. Létrehoz továbbá egy AUTORUN.INF nevű állományt, mely a command.exe-t indítja.



20. ábra: a Win32/Lovgate.Z féreg a lemezegységek főkönyvtárában

Az AUTORUN.INF állomány létrehozásával eléri, hogy a Sajátgép ablakban az egyes egységekre kattintva automatikusan a féreg induljon el.



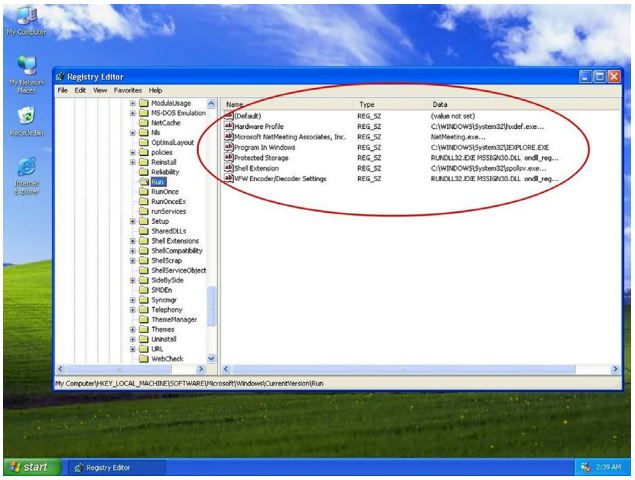
21. ábra: a Win32/Lovgate.Z féreg a lemezegységek főkönyvtárának megnyitásakor

A féreg a rendszerleíró adatbázisba is számos helyre bejegyzi magát, ezzel is biztosítva, hogy a Windows elindulásakor legalább egy példány elinduljon belőle. A

HKLM\Software\Microsoft\Windows\CurrentVersion\Run

kulcs alatt több változót hoz létre:

”Hardware Profile”= ”C:\WINDOWS\System32\hxdef.exe”
”Microsoft NetMeeting Associates, Inc.”= ”NetMeeting.exe”
”Program In Windows”= ” C:\WINDOWS\System32\IEXPLORE.EXE”
”Protected Storage”= ” RUNDLL32.EXE MSSIGN30.DLL ondll_reg”
”Shell Extension”= ” C:\WINDOWS\System32\spoolsv.exe”
”VFW Encoder/Decoder Settings”= ” RUNDLL32.EXE MSSIGN30.DLL ondll_reg”



22. ábra: a Win32/Lovgate.Z féreg a rendszerleíró adatbázisban I.

Módosítja a

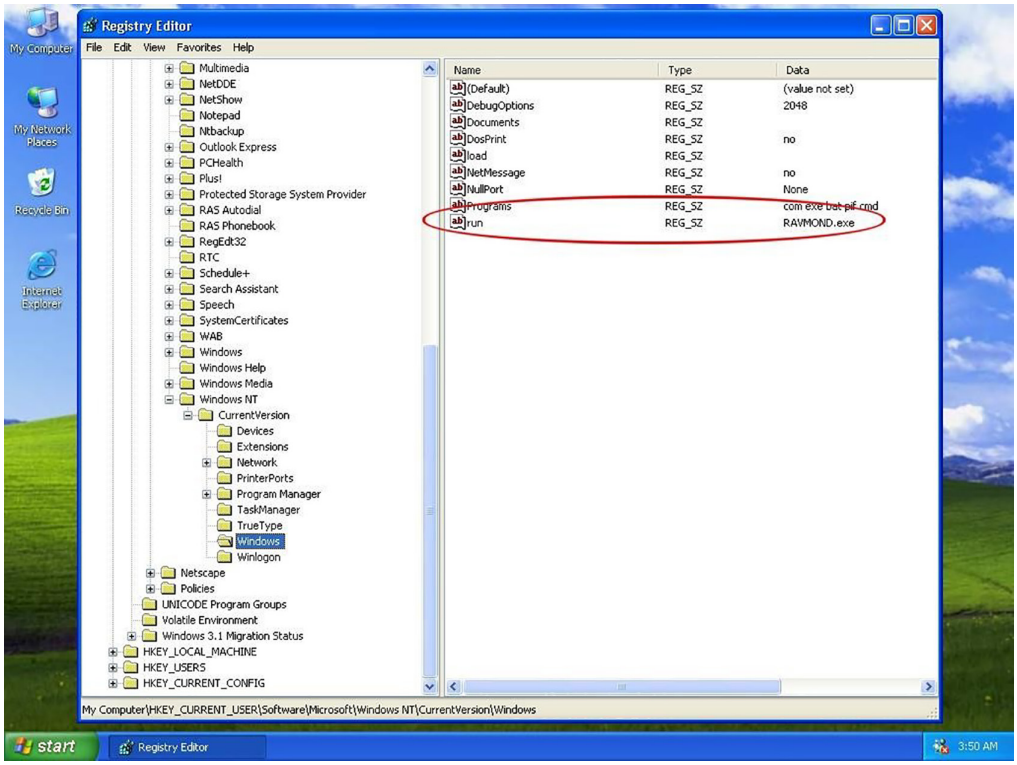
HKLM\Software\Microsoft\Windows NT\CurrentVersion\Windows

kulcs alatti változókat, illetve a

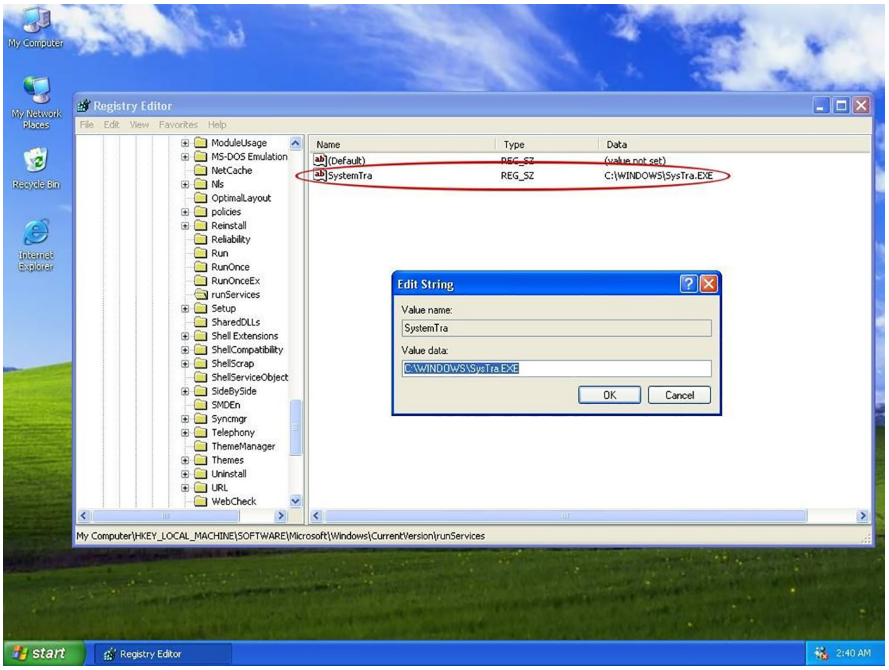
HKLM\Software\Microsoft\Windows\CurrentVersion\RunServices

kulcs alatt is bejegyzi magát:

”SystemTra”=”Systra.exe”



23. ábra: a Win32/Lovgate.Z féreg a rendszerleíró adatbázisban II.



24. ábra: a Win32/Lovgate.Z féreg a rendszerleíró adatbázisban III.

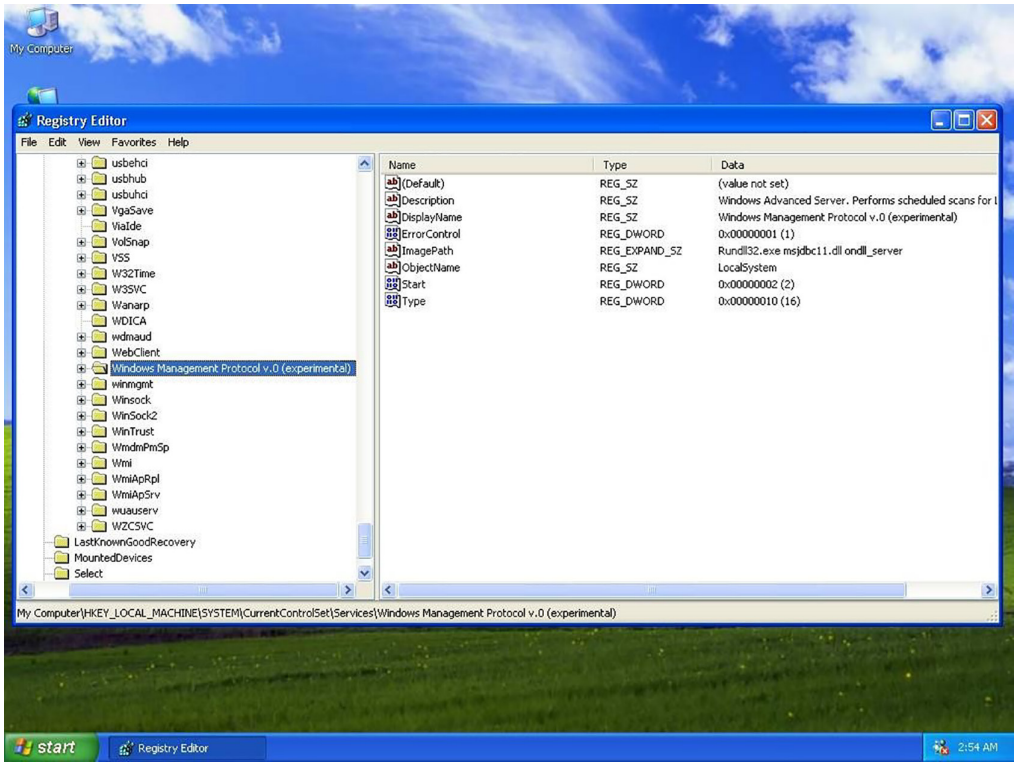
A féreg létrehozza a következő változót, jelezve ezzel a jelenlétét:

HKLM\Software\Microsoft\Windows\CurrentVersion\ZMXLIB1

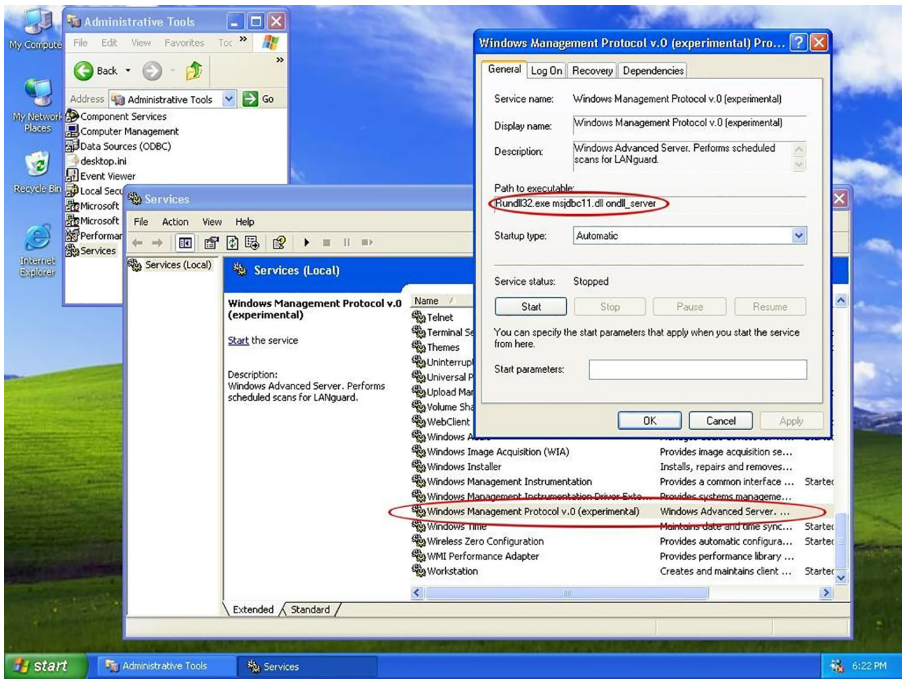
Szolgáltatásként regisztrálja továbbá magát a

HKLM\SYSTEM\CurrentControlSet\Services\Windows Management Protocol v.0 (experimental)

kulcs alatt is:



25. ábra: a Win32/Lovgate.Z féreg által létrehozott szolgáltatás I.



26. ábra: a Win32/Lovgate.Z féreg által létrehozott szolgáltatás II.

Terjedés a lokális hálózaton

A féreg képes magából másolatot készíteni az elérhető lemezeken az alábbi nevek valamelyikén:

WinRAR.exe
winhlp32.exe
i386.exe
client.exe
findpass.exe
Cain.pif
mmc.exe
Internet Explorer.bat
xcopy.exe
autoexec.bat
MSDN.ZIP.pif
Support Tools.exe
Windows Media Player.zip.exe
WindowsUpdate.pif
Microsoft Office.exe
Documents and Settings.txt.exe

A féreg megosztja a Windows *Media* könyvtárát a lokális hálózaton, *Media* néven. Megpróbál becsatlakozni a hálózaton elérhető lemezmegosztásokra *Administrator* vagy *Guest* felhasználóként, az alábbi, beépített jelszólistát használva:

0
1
7
11
12
111
123
321
1234
2003
2004
2600
12345
54321
111111
121212
123123
123456
654321
666666
888888
1234567
11111111
12345678
88888888
123456789
!@#$

!@#$%
!@#$%^
!@#$%^&
!@#$%^&* 123abc 123asd
a aaa abc
abc123 abcd abcdef abcdefg Admin admin admin123
administrator alpha
asdf asdfgh computer database enable god
godblessyou guest
home Internet login Login love mypass
mypass123 mypc mypc123 oracle owner pass passwd Password password pc
pwd root secret server sex sql super sybase temp
temp123 test test123 win
xp xxx yxcv zxcv

Ha a becsatlakozás sikerül neki, bemásolja magát a távoli gép Windows-könyvtárába és elindítja magát a távoli gépen. Az elindításhoz a távoli gépen szolgáltatásként (service) regisztrálja magát (a távoli gépen való folyamatindításnak a Windows-ban ez az egyetlen ismert módja).

Terjedés e-mail-üzenetekben

A program elküldi magát .COM, .EXE, .SCR vagy .PIF kiterjesztésű csatolmányként a beállított levelezőkliensben talált e-mail-címekre. Ezen túlmenően a merevlemezeken is keres e-mail-címeket, a .WAB, .PL, .ADB, .TBB, .DBX, .ASP, .PHP, .SHT, .HTM kiterjesztésű fájlokban. Az e-mail-címek keresése során értelmezi, ha az e-mail-címben a "@" karakter helyett valamilyen helyettesítő karaktersorozat található: "(at)", "(@)", "@ ", " @", " ". Az e-mail-üzenetek tárgy mezője az alábbiak egyike:

Error hi
hello
Mail Delivery System
Mail Transaction Failed
Server Report
Status test

A csatolmányok neve a következők közül kerül ki:

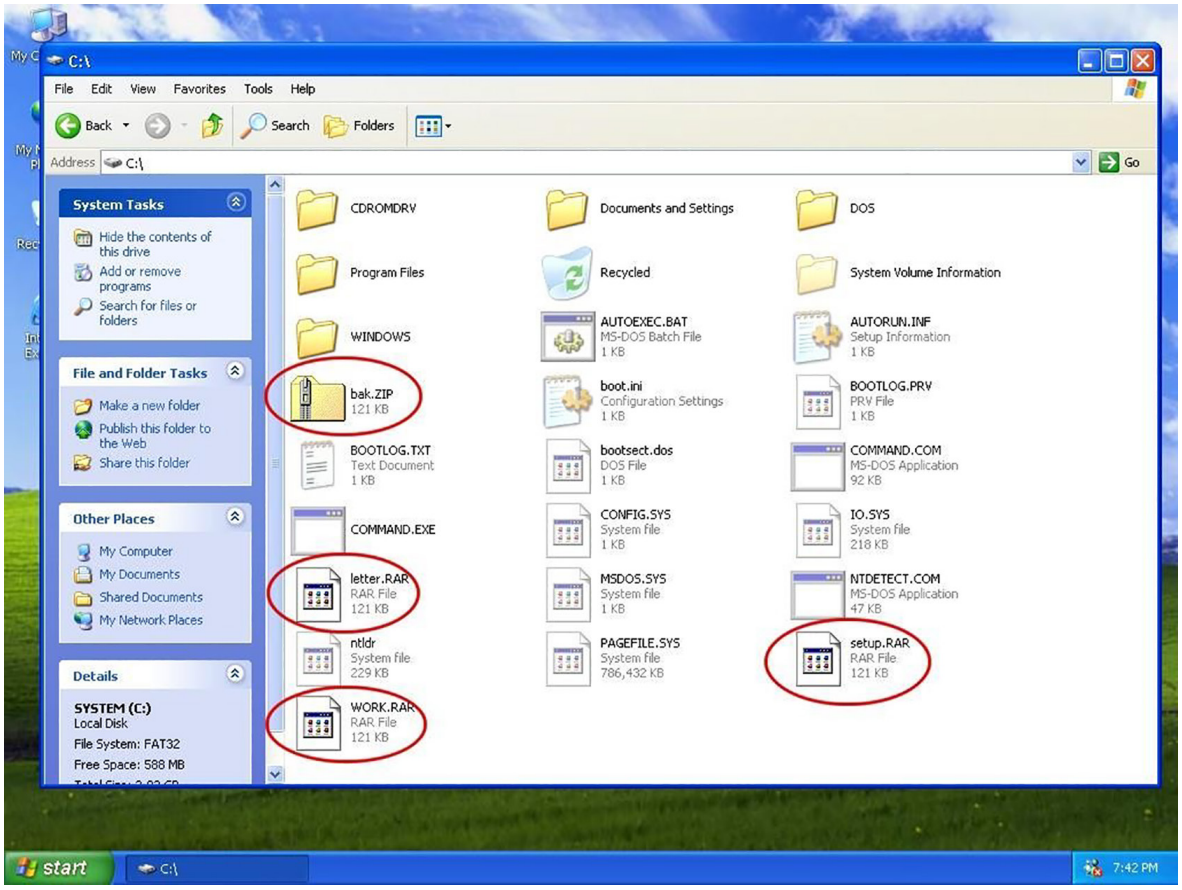
I am For u.doc.exe
Britney spears nude.exe.txt.exe
joke.pif
DSL Modem Uncapper.rar.exe
Industry Giant II.exe
StarWars2 - CloneAttack.rm.scr
dreamweaver MX (crack).exe
Shakira.zip.exe
SETUP.EXE
Macromedia Flash.scr
How to Crack all gamez.exe
Me_nude.avi.pif
s3msong.MP3.pif
Deutsch BloodPatch!.exe
Sex in Office.rm.scr
the hardcore game-.pif

Az e-mail elküldéséhez a helyi domainnevet az alábbi előtagokkal kombinálva megpróbálja átalakítani:

gate.
ns.
relay.
mail1.
mxs.
mx1.
smtp.
mail.
mx.

Fertőzés

Az elérhető egységek főkönyvtárában a Windows újraindításakor újabb és újabb ZIP, illetve RAR tömörítvényeket készít, melyben a saját kódját helyezi el.



27. ábra: a Win32/Lovgate.Z féreg által létrehozott tömörített fájlok a lemezegységek főkönyvtárában

Az összes merevlemezen – C:-től Z:-ig – .EXE kiterjesztésű állományokat keres. Átnevezi őket .ZMX kiterjesztésűre, rejtett és rendszerattribútumokat ad nekik, majd az eredeti fájl nevén eltárolja saját magát (CEB- /companion/ vírus viselkedés).

Folyamatok

A féreg megpróbálja leállítani az alábbi szolgáltatásokat:
Rising Realtime Monitor Service
Symantec Antivirus Server
Symantec Antivirus Client

Windows NT (illetve ezzel kompatibilis) operációs rendszer alatt megpróbálja leállítani azokat a folyamatokat, melyek nevében szerepelnek a következők:
rising
SkyNet
Symantec
McAfee
Gate Rfw.exe
RavMon.exe
kill
NAV
Duba
KAV
KV

Hátsóajtó

A féreg egy véletlenszerű, 1024-nél nagyobb portcímen elindít egy FTP-szerver-szolgáltatást, melyhez nem szükséges felhasználónév, illetve jelszó, valamint nyit egy hátsóajtót a 6000-es TCP-porton is. Annak érdekében, hogy a féreg készítője is tudomást szerezzen a sikeres támadásról, a féreg a megfertőzött számítógépről gyűjtött információkat elküldi a *hello_zyx@163.com* e-mail-címre.

1.6. INTERNETEN TERJEDŐ KÁRTEVŐK

Az interneten terjedő kártevők célja, hogy egy másik számítógépre vagy egy másik felhasználóhoz jussanak el. Ezek a kártevők tehát üzenet küldésén alapulnak, a cél lehet egy felhasználó vagy lehet egy alkalmazás.

Amennyiben a cél a felhasználó, akkor a kártevő egy üzenetküldési szolgáltatást használ, mint például e-mail, skype, MSN, ICQ, IRC vagy akár a közösségi oldalak (például Facebook) megosztásai. A felhasználóknak címzett üzenetek két fő csoportját különböztethetjük meg:

- Ha **az üzenet tartalmaz** olyan **programkódot**, amely a számítógép számára bármilyen módon értelmezhető, akkor a kártevő célja az, hogy ez a programkód a másik számítógépen végrehajtódjon. A féreg valamilyen módon megpróbálja rávenni a felhasználót arra, hogy az átküldött kódot végrehajtsa (rákattintson a mellékletre).

Ehhez többféle, social engineering-en alapuló módszert használhat:

- A csatolt állomány nevében a dupla kiterjesztés használatával elérheti, hogy a Windows alapértelmezés szerint elrejtse a fájl valódi típusát.
- A csatolt fájl nevében a valódi kiterjesztés elé sok szóköz karakter elhelyezésével a Windows nem képes megjeleníteni az állomány teljes nevét.
- Webcímre emlékeztető nevet használ: Például ha a www.myparty.com karaktersort egy csatolmány nevében látjuk, akkor az nem egy webcím, hanem egy .COM kiterjesztésű állomány.
- Az üzenet feladójának meghamisításával elérheti, hogy úgy tűnjön, mintha az üzenet a címzett valamely ismerősétől érkezett volna.
- Az üzenet szövegével vagy a csatolmány nevével felkelti a felhasználó figyelmét, érdeklődését.

A gyanútlan felhasználónak címzett üzenetek az említett social engineering-alapú, a felhasználó mint emberi tényező átverésére vonatkozó technikák mellett ugyanakkor megcélozhatják az üzenetet kezelő alkalmazást is. Az üzenetkezelő alkalmazás valamely biztonsági problémáját kihasználva a kártevő elérheti, hogy az alkalmazás automatikusan végrehajtsa az elküldött mellékletet.

- Ha **az üzenet nem tartalmaz** semmilyen **programkódot**, amely a számítógép számára bármilyen módon értelmezhető lenne, akkor a kártevő célja az, hogy a felhasználó valamilyen tevékenységet hajtson végre. Ilyen tevékenységek általában az alábbiak:
  - Ha az üzenet egy webcímet tartalmaz, akkor a kártevő célja, hogy a felhasználó a webcímet nyissa meg a böngészőjében. Ez továbbviheti a felhasználót egy olyan weboldalra, ahonnan kártékony kód települ a számítógépére, vagy egy phishing-oldalra, ahol megpróbálják a személyes adatait megismerni.
  - Ha az üzenet nem tartalmaz webcímet, akkor a kártevő célja az lehet, hogy a felhasználó olvassa el az üzenetet (kéretlen reklámüzenet, spam), vagy az a kártevő célja, hogy a felhasználót használja a terjesztés “motorjaként” (lánclevél,hoax).

Amennyiben a cél egy alkalmazás, akkor a kártevő az alkalmazás egy biztonsági rését használja ki. Ilyenkor a biztonsági rés révén a kártékony kód automatikusan elindul. A biztonsági rés kapcsolódhat az *operációs rendszerhez*, az operációs rendszeren futó *alkalmazáshoz*, esetleg valamely *protokollhoz* is. A biztonsági rések egy szűkebb körét valamely fájlformátumhoz tartozó sérülékenységnek is szokták nevezni, például az SWF-, JPG- vagy MP3-fájlok esetén. Az ilyen esetekben azonban a probléma SOHA NEM magához a file formátumához kapcsolódik, hanem az adott fájlformátumot kezelő alkalmazáshoz.

Az interneten terjedő kártevők esetén elmondhatjuk, hogy hatásos terjedésük két fő tényezőre vezethető vissza. Terjedésükhöz egyrészt az emberi tényező, másrészt a biztonsági rések is hozzájárulnak. A biztonsági rések viszont szintén az emberi tényezőre vezethetők vissza. Egyrészt a biztonsági rések a programok, alkalmazások fejlesztése során, a nem megfelelő tervezésből, programozásból, kódolásból adódnak. Másrészt a biztonsági rések nagy részét a megfelelő, és időben végrehajtott frissítésekkel el lehet kerülni. Így leszűrhetjük azt a végkövetkeztetést, hogy az interneten terjedő kártevők terjedése elsősorban az emberi tényezőre vezethető vissza.

1.7. CÉLZOTT TÁMADÁSOK

Célzott támadásoknak nevezzük az olyan fenyegetéseket, melyeket a támadók kifejezetten egy adott célpont (személy vagy szervezet) ellen használnak. Egy számítógépes vírushoz képest a fenyegetés "megalkotója" ebben az esetben nem arra törekszik, hogy a kártékony kód minél jobban elterjedjen, hanem arra, hogy a kiszemelt célpont eszközére, eszközeire bejusson.

Célzott támadások már a 2000-es évek elején is léteztek, sőt néhány esetben Magyarországon is megjelentek. Az egyik ilyen hazai esetről szóló esettanulmány ([1]) szerint 2001-ben egy magyarországi intézmény, mely mintegy 200 számítógéppel rendelkezett, vált célzott támadás áldozatává. A tanulságos történet szerint a vezetés a rendszergazda elbocsátását követően néhány hónappal vette észre, hogy kezdik sorozatosan elveszteni a tendereket. Megbíztak egy informatikai biztonsággal foglalkozó vállalkozást a szervezet átvilágítására, melynek során a számítógépeket is szúrópróbaszerűen megvizsgálták. Miután a második olyan számítógépet is megtalálták, amelyen egy sehova nem köthető kis programot találtak, a vizsgálatot kiterjesztették a szervezet valamennyi számítógépére. Összesen 11 számítógépen került elő ez a kis program, valamennyi a szervezet működéséhez szükséges legfontosabb pontokon volt, beleértve a vezetők hordozható számítógépeit is. A kis program elemzése során kiderült, hogy az alkalmas arra, hogy információkat küldjön a szervezeten kívülre, másik számítógépre telepedjen. Mintegy 16 különböző módszerrel rendelkezett a külső kommunikáció érdekében, a kapcsolatot pedig számtalan úgynevezett proxy-szerveren keresztül valósította meg, ami meggátolta a támadó kilétének az azonosítását.

1.8. APT-K

Az APT-k (Advanced Persistent Threat – magas szintű, folyamatos fenyegetést jelentő támadások) a 2010-es évek elején világosan bebizonyították, hogy képesek a hagyományos védelmi technológiák mellett is rendszerekbe behatolni és ott hosszú ideig észrevétlenül maradni, valamint gondoskodni értékes információknak a szervezeten kívülre történő küldéséről, azok eltulajdonításáról ([3]).

Az APT-k működésük során több különböző támadási lehetőség módszereit egyesítik, úgymint social engineering-módszereket a felhasználók átverésére ([2]), vírusterjedési módszereket újabb számítógépek felderítésére és megfertőzésére, illetve hálózati kommunikációs módszereket a kártékony kód távirányítására és az adatok kijuttatására.

28. ábra: Egy tipikus célzott támadás és az APT (Advanced Persistent Threat) felépítése

A 28. ábra egy tipikus APT működését mutatja be, az alábbi legfontosabb részekre bontva:

1. **Intelligens felderítés (Intelligence Gathering):** Az első lépés során a támadó elsősorban nyilvános forrásból információkat gyűjt a leendő áldozatról, általában mint szervezetről, illetve az ott dolgozó munkatársakról. Nem csak és kizárólag a szervezetre vonatkozó adatok érdeklik, hanem bármilyen személyes információ, amellyel akár egy alkalmazott bizalmába lehet férkőzni. A nyilvános források köre tehát nem csak a szervezet weboldala, hanem például az alkalmazottak közösségi médiaoldalakon (például LinkedIn, Facebook) lévő profiloldalai. Az információk összegyűjtésével, a szervezet és alkalmazottainak a felderítésével előkészítheti az egyedi, testreszabott támadást. Például elég lehet, ha egy alkalmazottról kiderül a Facebook-oldalán, hogy szereti a macskákat, máris megvan egy közös téma, amire hivatkozva kapcsolatot építhet ki vele a támadó.
2. **Bejutás (Point of Entry):** Egy támadás során a támadó a kártékony kódjának bejuttatására több módszer közül is választhat. A lehetőségeket azonban két lényeges csoportba oszthatjuk: egyrészt azokra a bejutási formákra, melyek nem igényelnek felhasználói interaktivitást, illetve azokra, amelyeknél szükség van a felhasználó beavatkozására. Az előbbi esetben a kiszemelt infrastruktúra valamely elemére vonatkozó sérülékenységet használja ki a támadó, mely általában az operációs rendszer, valamelyik alkalmazás vagy esetleg a valamely alkalmazáson belül egy kiegészítő biztonsági problémájához köthető. A másik csoportba azok a bejutási módszerek sorolhatók, melyek során a felhasználónak például egy üzenetben érkezett csatolmányt meg kell nyitni, vagy esetleg egy linkre kell kattintani. Az ilyen esetekben a felhasználót social engineering-módszerekkel kell a támadónak becsapnia, rávennie, hogy a kívánt cselekedetet végrehajtsa. Nyilvánvaló, hogy ez utóbbi esetben van különös jelentősége az intelligens felderítés során megszerzett információknak.
3. **C&C kommunikáció (C&C – Command and Control Communication):** Amennyiben a támadó sikeresen bejuttatta a kiszemelt környezetbe a kártékony kódját, a következő lépés során a támadó irányítása alá vonja a megtámadott eszközöket. Ennek érdekében egy saját C&C-szervert állít fel, melyen keresztül a megtámadott számítógépeket irányíthatja: parancsokat adhat, állományokat tölthet fel és le, lényegében bármit megtehet a megtámadott számítógépen akár a háttérben is.
4. **„Oldalazó mozgás" (Lateral Movement):** A bejutást és a sikeres kapcsolatfelvételt követően a támadás következő lépésében a helyi hálózaton belüli további számítógépek és eszközök felderítése, majd az azokba történő behatolás következik. Ennek az a célja, hogy további hozzáférési adatokat szerezzen meg, növelje a privilégiumszintet, illetve, hogy biztosítsa a hálózat folyamatos felügyeletének a lehetőségét. A helyi hálózaton történő terjedést a támadó nyilván az 1. és 2. pont szerint akár kívülről is végrehajthatná, azonban egy infrastruktúra védelmi elemeinek jelentős része a külső kapcsolódási pontokon helyezkedik el, azaz egy belső számítógépről, eszközről történő behatolás egy másik számítógépre, eszközre sokkal egyszerűbb és a támadó szempontjából kevésbé kockázatos. Természetesen a támadó itt is bevethet social engineering trükköket, vagy akár biztonsági réseket is kihasználhat. A megtámadott számítógépek, eszközök irányítására a már használt C&C-szervert is igénybe veheti.
5. Értékes adatok, információk felderítése (Asset/Data Discovery): A belső hálózaton történő terjedés során a támadó célja nem egyszerűen más számítógépek és eszközök felderítése, hanem az ezeken az eszközökön tárolt értékes adatok és információk elérését biztosító szerverek és szolgáltatások azonosítása is. Ezt megteheti például a hálózati forgalom vizsgálatával.
6. **Adatok kiszivárogtatása (Data Exfiltration):** Miután az érzékeny adatokat, információkat a támadó azonosította, az adatokat általában először egy olyan belső számítógépre juttatja el, ahol egy elsődleges elemzés, válogatás és tömörítés történik, majd az így összeállított információk kijuttatása egy külső szerverre, amelyet a támadó közvetlenül elér.

1.9. HÁLÓZATI TÁMADÁSOK

A CERT (Computer Emergency Readiness Team) szerint **az incidens az a cselekedet, amelynek során megsértenek egy explicit vagy implicit biztonsági házirendet**, például:

- Sikeres vagy sikertelen kísérlet arra, hogy jogosulatlanul hozzáférést szerezzenek egy rendszerhez vagy annak adataihoz.
- Egy szolgáltatás nem kívánt megszakítása vagy megtagadása.
- Egy rendszer jogosulatlan használata adatok kezelése vagy tárolása céljából.
- A rendszer hardverének, firmware-ének vagy szoftverjellemzőinek a megváltoztatása a tulajdonos tudomása, utasítása vagy jóváhagyása nélkül.

Egy hálózati támadás első lépése szinte minden esetben a megcélzott számítógépen futó operációs rendszer és esetleg az azon futó alkalmazások megismerése. Ehhez kifinomult módszerek és eszközök állnak rendelkezésre. Ezt követhetik azok az eljárások és módszerek, amelyek már a konkrét támadást jelentik. Az alábbiakban a teljesség igénye nélkül ezeket a módszereket részletezzük.

1.9.1. OPERÁCIÓS RENDSZER MEGISMERÉSE, BIZTONSÁGI RÉSEK KIHASZNÁLÁSA

Az operációs rendszer megismerése (OS fingerprinting) egy metodológia annak meghatározására, hogy milyen operációs rendszer fut egy számítógépen. Gyakran ez a legelső lépés egy támadás során. Ha ugyanis egy támadó megtudja, milyen operációs rendszerrel fut a távoli, célba vett számítógép, akkor már könnyű feladat találni hozzá egy megfelelő támadó eszközt (exploitot).

Az operációs rendszer megismerésére a legegyszerűbb módszer a Telnet-munkamenet indítása, ugyanis a sok Telnet-szerver rengeteg információt ad az operációs rendszerről. A munkaállomások, szerverek telepítésekor, installálásakor azonban az egyik legelső lépés a Telnet-szolgáltatás leállítása, hiszen ez a protokoll mindenfajta védelem nélkül teremt kapcsolatot számítógéppel, így a támadások szempontjából veszélyforrást jelenthet.

A TCP (Transmission Control Protocol – [10]) egy kapcsolatorientált protokoll, azaz a két kommunikáló fél összeköttetést létesít, majd adatokat továbbíthatnak egymáshoz, végül lebontják a kapcsolatot. A TCP a kommunikációt full duplex (kétirányú) módon valósítja meg, azaz a küldés és fogadás egy időben történik. A kommunikáló feleknek nem kell törődniük a küldendő adatmennyiséggel, a hibakezeléssel; ezt mind megoldja a TCP.

A Telnet híján az operációs rendszer és a hozzá tartozó környezet feltérképezésére, profilkészítésre a TCP használható. Az operációs rendszerek ugyanis különbözőképp valósítják meg a TCP-t, mely különbségek lehetőséget adnak az operációs rendszer azonosítására.

A TCP-t használó módszer szerint speciális csomagokat küldenek a TCP-szerverhez és figyelik a szerver válaszát. Az interneten számos eszköz szabadon elérhető, amely komplex módon megvalósítja ezt a stratégiát. Ilyen pl. az egyik legelterjedtebb eszköz, az Nmap ([6]).

Az operációs rendszer felderítésére a TCP például az alábbi módszerekkel használható:

- **FIN-próba:** A TCP-ben a FIN-csomag a kapcsolat lezárására szolgál, azaz létező kapcsolathoz kell tartozzon. A támadó csupán egy FIN-csomagot küld a célszámítógép egy nyitott portjára, anélkül hogy kapcsolattal rendelkezne. A TCP-t definiáló RFC 793 szerint ([10]) erre nem kell válaszolni, de a Windows-megvalósítások válaszolnak.

- **ACK-érték:** A TCP-ben minden elküldött csomagot sorszámmal látnak el. A sorszámot a küldő fél folyamatosan, növekvő sorrendben generálja. A sorszámokat használják a megérkezett csomagok nyugtázására is (ACK-érték). A csomagok sorszámozásához az operációs rendszerek azonban különböző tartományban lévő értékeket használnak.
- **ICMP-hibaüzenetek:** A támadó a TCP segítségével olyan üzeneteket küld, amelyek a TCP eljárásában hibát eredményeznek. Ekkor a megcélzott szerver hibaüzeneteket küld, a kiküldött hibaüzenetek száma azonban korlátozott, és az operációs rendszertől függ.

A felsorolt módszereken túlmenően számos további lehetőség is kínálkozik a TCP-ben és a TCP-re vagy az UDP-re épülő további alkalmazási rétegek protokolljaiban az operációs rendszer felderítésére.

Az Nmap-program az operációs rendszer felderítését több módszer szerint is elvégzi, majd az eredményeket összehasonlítja az egyes operációs rendszerekhez eltárolt sémákkal, így a tapasztalt viselkedéssel történő összevetéssel a legvalószínűbb operációs rendszer is meghatározható.

Az operációs rendszer felderítését követően egy támadó már pontosan tudja, hogy az adott operációs rendszer milyen biztonsági hibákkal rendelkezik, rendelkezhet. Természetesen arra is következtethet, hogy melyek a leggyakoribb alkalmazások az adott operációs rendszer alatt, melyeknek szintén lehetnek kihasználásra alkalmas biztonsági hibái. A biztonsági hibákat kihasználó eljárások (exploitok) egyik legnagyobb gyűjteménye a Metasploit-alkalmazás. A Metasploit folyamatosan frissülő adatbázisa lehetőséget ad a legújabb sérülékenységek kihasználására is.

1.9.2. A KOMMUNIKÁCIÓS PROTOKOLLOK TÁMADÁSA

Az interneten történő támadások egyik elterjedt módja a szolgáltatás megtagadásával (DoS – Denial of Service) járó támadás. Ezen módszerek használatával a támadó eléri, hogy a szerver túlterhelődjön, és ne legyen képes kiszolgálni a hozzá beérkező további kéréseket. A módszer lényege, hogy nagyszámú feladat feldolgozása elé állítják a szervert, melynek a kiszolgálása időigényes, lényegesen több időt vesz igénybe, mint a "feladatot megfogalmazó" kérés előállítása és elküldése.

A teljesség igénye nélkül néhány internetprotokoll támadására szolgáló módszer:

- **SYN Flooding:** A SYN Flooding (SYN-elárasztás) támadások célpontja a TCP kapcsolatfelépítési eljárása. A kapcsolat felépítésének az első lépése a kapcsolat felépítésére vonatkozó kérés (SYN-csomag) elküldése. A támadó nagyszámú SYN-csomagot küld a távoli számítógépre és nem foglalkozik a válaszokkal. A SYN-csomagok előállítása sokkal kisebb erőforrást (idő és tárhely) igényel, mint azok feldolgozása és kezelése, ezért a szerver könnyen túlterhelhető, előbb-utóbb nem lesz képes újabb kérések kiszolgálására.
- **Smurfing támadások:** Az ICMP (Internet Control Message Protocol) egy gyengeségét használja ki. Az ICMP-protokoll segítségével „echo" (visszhang) csomagokkal ellenőrizhető, hogy vajon él-e, működik-e egy távoli állomás. Az „echo" csomagokban a küldő megadhatja, hogy milyen IP-címre kéri a választ. A támadó hamisított IP-címekkel készít „echo" csomagokat, ahol a hamisított IP-cím az áldozaté, így a kérések elküldésével az áldozatot árasztják el a válaszok. Az ICMP-szabványt 1999-ben módosították, hogy ellensúlyozzák az ilyen jellegű támadásokat.

A szolgáltatásmegtagadásos támadásokat (DoS) a támadó a támadás elosztásával tovább fokozhatja. Az elosztott szolgáltatásmegtagadás-támadásokat (DDoS – Distributed Denial of Service) a leggyilkosabb támadási formának tekinthetjük. A módszer lényege, hogy egy támadó nagyszámú gép felett veszi át az ellenőrzést, és megszokott támadószoftvereket telepít rájuk. Egy adott jelre aztán a támadó szoftver üzenetekkel, csomagokkal kezdi bombázni a célba vett számítógép(ek)et.

2. VÉDEKEZÉSI ESZKÖZÖK

Matematikailag bizonyított, hogy a számítógépes kártevők, fenyegetések ellen tökéletes, 100 százalékos biztonságot jelentő védekező eljárás nem létezik. A védelmi szoftverek elsősorban a *már ismert* kártevők, támadási módszerek ellen tudják felvenni a harcot. Léteznek persze olyan megoldások, melyekkel a fenyegetésekre jellemző viselkedéseket próbálják azonosítani, de ezek hatékonysága messze elmarad az ismert kártevők, fenyegetések elleni védekezés hatékonyságától.

A számítógépes fenyegetések az első kártékony kód megjelenésétől kezdve folyamatosan újabb és újabb technikákat fejlesztenek ki annak érdekében, hogy a létező védelmi módszereket ne, vagy csak nagyon nehezen lehessen alkalmazni. Ebből a szempontból a fenyegetések készítői mindig előnyben vannak, hiszen a "fejlesztésük" során a létező védelmi rendszereken tesztelhetik a kártékony kódot, és módosításokkal elérhetik, hogy a védelmek ne legyenek hatékonyak.

A védekezés szempontjából a lehetőségeket két oldalról közelíthetjük meg:

1. Egyrészt lehetőségünk van a védekezésre a védendő eszköz által használt kommunikációs csatornák felügyelésével is. Ekkor természetesen a hálózati kapcsolatok felügyeléséről beszélhetünk, de nyilván nincs ilyen lehetőség a cserélhető adathordozók felügyeletére.
2. Másrészt a védelem elhelyezkedhet magán a védendő eszközön, azaz általában egy olyan végponton, amely képes arra, hogy egy támadás során áldozattá váljon; amely képes arra, hogy a fenyegetéshez tartozó programkódot értelmezze és végrehajtsa. Természetesen a végponton lévő védelem felügyelheti az adott eszköz kommunikációs csatornáit is az 1. pontnak megfelelően.

Mind a két megközelítésnek megvannak az előnyei és a hátrányai. A védendő eszközön lévő védelem előnyösebb, mert sokkal mélyebb vizsgálatokat végezhet, akár egy gyanús kód valós idejű futása során is figyelheti annak tevékenységét. Nem beszélve a titkosított kommunikációs csatornákról (például VPN), amelyek használata esetén a végponton mindenképpen megtörténik az eredeti tartalom visszaállítása. Ugyanakkor erre a védelemre csak akkor számíthatunk, ha egy potenciális támadás már elérte a védendő eszközt. A kommunikációs csatorna felügyelete nyilván azért előnyösebb, mert az ellenőrzés és blokkolás már azelőtt megtörténik, mintsem a támadás elérte volna a védendő eszközt. Ugyanakkor hátránya a módszernek, hogy a potenciálisan kártékony kód átvitelét látja csupán, annak végrehajtását, működését nem, illetve, hogy titkosított kommunikáció esetén annak felügyelete is körülményes.

Az alábbiakban a különböző védelmi lehetőségeket tekintjük át a kommunikáción alapuló megközelítés alapján. Ide tartoznak a tűzfalak, a behatolásérzékelő és -megelőző rendszerek, de külön alfejezetben foglalkozunk az APT-k elleni védekezés problémájával is.

2.1. TŰZFALAK

A tűzfal két hálózat között elhelyezkedő olyan védelmi eszköz, amely mindkét hálózat irányába haladó forgalom vonatkozásában az előzetesen beállított biztonsági szabályok alapján tényked ik. Természetesen a tűzfal csak a rajta keresztülhaladó forgalmat képes szabályozni. A tűzfalakat működésük alapján több csoportba oszthatjuk, aszerint, hogy a kommunikáció során az adatforgalom mely részének az elemzésére képesek.

Csomagszűrő tűzfalak

Az 1980-as évek végén megjelenő első tűzfalak a csomagszűrő tűzfalak csoportjába tartoztak. A csomagszűrő tűzfalak az OSI-modell 3. és 4. rétegében (hálózati és szállítási réteg) végzik munkájukat. Ez a TCP/IP-modell internet és szállítási rétegének felel meg, ahol tipikusan az IP- (internetréteg-), illetve a TCP- és az UDP- (szállításiréteg-) protokollok találhatók. A csomagszűrő tűzfalak a feljebb lévő rétegek adataival nem foglalkoznak, így egy TCP-szegmens adatterülete már nem befolyásolja a döntést. A csomagok vizsgálatánál a csomagok (IP), illetve szegmensek (TCP, UDP) fejlécét vizsgálják meg. Az IP-szint minden esetben kiértékelésre kerül, azaz a döntést befolyásolja a csomag forrás- és célcíme, esetleges darabolási adatai, illetve egyes esetekben még az IP-fejléc egyéb paraméterei is. A legtöbb implementáció esetében kiértékelésre kerül a TCP és UDP fejléce is. A csomagszűrő tűzfalak döntési mechanizmusa szabálylistákon alapszik, mely szabályok feltételrendszereket és tevékenységeket írnak le. A feltételek teljesülése esetén a tűzfal a szabályhoz rendelt tevékenységet hajtja végre, azaz a csomag továbbítását engedélyezheti, vagy eldobja azt.

A csomagok vizsgálata során a tűzfal az előre megadott szabályokat megpróbálja az adott csomagra illeszteni. A csomagszűrő tűzfalak esetén általában beállítható, hogy alapértelmezésként milyen tevékenységet hajtsanak végre (engedélyezés vagy eldobás). A szabályok illesztését a csomagszűrő tűzfalak egy meghatározott sorrendben végzik, és az első illeszkedő szabály szerinti tevékenységet alkalmazzák.

A beállított szabályok tehát mintegy láncként kapcsolódnak egymáshoz, a szabályok sorrendje ezért nagyon fontos a tűzfal működésének hatékonysága és a tűzfal terhelésének szempontjából. Praktikus a szabályrendszert úgy felépíteni, hogy az egyszerű tiltó vagy engedélyező szabályok a döntési lánc elején legyenek. A finomabb feltételeket megfogalmazó szabályokat a lánc alsóbb szintjein célszerű elhelyezni. A csomagszűrők felépítésükből, és működésükből adódóan nem alkalmasak bonyolult igények megvalósítására, például az átmenő forgalom összetett szűrésére, kapcsolatok követésére. A beérkező csomagokat mint egymástól különálló adatokat kezeli a tűzfal, melynek eredményeként nincs lehetőség a TCP-kapcsolatok állapotának megbízható nyomon követésére. Előfordulhatnak továbbá olyan hálózati alkalmazások is, melyek esetében a kommunikáció során változhat a TCP portszáma (például FTP). Többek között az ilyen esetekben a csupán csomagszűrést alkalmazó tűzfalak nem tudják megoldani a feladatot. A csomagszűrő tűzfalak lehetőségei tehát korlátozottak, egyre ritkábban találunk tisztán csak csomagszűrő technológiát alkalmazó megoldásokat.

Áramkörszintű tűzfalak

Az áramkörszintű tűzfalak a második generációs tűzfalak körébe tartoznak. A kapcsolatok vizsgálatát az OSI-modell 5. rétegben, illetve a TCP/IP-modell 4. rétegében végzik. Egy kapcsolat megnyitása előtt ellenőrzik a kapcsolat kiépítésének folyamatát. Ha a kapcsolat létrejött, megindulhat a kommunikáció a csatornán. Az áramkörszintű tűzfalak egy táblázatban tartják nyilván a legális kapcso-

latokhoz tartozó adatokat. Egy kapcsolat felépítését követően az áramkörszintű tűzfalak általában a következő adatokat tárolják el a kapcsolatról:

- A kapcsolat egyedi azonosítója, melyet a tűzfal ad neki.
- A kapcsolat állapota: felépítés folyamatban, létrejött, vagy lezárás alatt.
- A forrás IP-címe, ahonnan az adatok érkeznek.
- A cél IP-címe, ahová az adatokat továbbítani kell.
- A fizikai interfész, melyen keresztül az adatok érkeznek.
- A fizikai interfész, melyen keresztül az adatok távoznak.

A legtöbb áramkörszintű tűzfal megvalósítása a SOCKS-protokollon keresztül történik. A SOCKS egy hálózati protokoll, mely kliens-server alapú. Amennyiben egy alkalmazás kapcsolódni szeretne egy külső szerverhez, akkor a kliensszámítógépen futó SOCKS-kliens elküldi az általa támogatott azonosítási eljárások listáját a szervernek. Ha a tűzfalban lévő SOCKS-szerver támogatja ezek közül az azonosítási eljárások közül valamelyiket, akkor válaszában közli ezt a klienssel. A kliens ezután azonosítja magát a megadott eljárások valamelyikével. Ha nincs ilyen közös azonosítási eljárás, akkor a kapcsolatfelépítési kérést a szerver elutasítja. Az azonosítás után a kliens közli a szerverrel, hogy hová szeretne kapcsolódni. Ha engedélyezett a kapcsolat, akkor a SOCKS-szerver kapcsolódik a kliensprogram által kijelölt szerverhez, és kiépíti közöttük a virtuális kapcsolatot. A kapcsolódás csak akkor lehetséges, ha ez a kérés nem tiltott kapcsolat kiépítését kéri. Ezek után a szerver értesíti a klienst, hogy a kapcsolat létrejött, elkezdődhet az adatok továbbítása. Egy áramkörszintű tűzfal a kliens-szerver viszonyban tulajdonképpen a kliens és a szerver között helyezkedik el, két kapcsolatot tart fent: egyet a kliens felé, míg a másikat a kért szerver felé. A kapcsolat kiépülése után az adatforgalom a kliens és a szerver között a tűzfalon keresztül történik, ekkor a csomagok már csak egy egyszerűsített ellenőrzésen esnek át. Az ellenőrzés ekkor már csak annyiból áll, hogy a küldött, illetve fogadott adatok megfelelnek-e a kapcsolat létrehozásakor rögzített adatoknak. Ahhoz, hogy a tűzfal érvényesnek minősítsen egy kapcsolatot, a kapcsolat kiépítésének minden fázisát ellenőrzi. Csak azok a csomagok juthatnak át a tűzfalon, amelyekhez tartozik érvényes kapcsolati bejegyzés a táblázatban. Amennyiben egy kapcsolat lezárul, akkor a tűzfal törli a kapcsolathoz tartozó bejegyzését a táblázatából. Ez a működési elv nagyon gyors működést eredményez.

Az áramkörszintű tűzfalakat szokás még SOCKS proxy tűzfalaknak is nevezni, abból adódóan, hogy a kliens felé a szervert, a szerver felé a klienst személyesítik meg, mintegy átjáróként szerepelnek a kommunikációban.

Alkalmazásszintű tűzfalak

Az alkalmazásszintű tűzfalak a harmadik generációs tűzfalak körébe tartoznak, az OSI- és a TCP/IP-modell alkalmazási rétegében ellenőrzik a csomagokat. Minden adatcsomagot alkalmazásszinten vizsgálnak, a kapcsolat állapotának nyilvántartása mellett képesek a csomagok, szegmensek adatterületében egyéb ellenőrzések végrehajtására is.

A legtöbb alkalmazásszintű tűzfal minden egyes alkalmazásszintű protokollhoz egy-egy speciális alkalmazást, proxyt futtat. Ezek a proxyk kezelik a tűzfalon áthaladó forgalmat a hozzájuk rendelt protokoll tekintetében, pl. HTTP, FTP. Mivel ezek az alkalmazások speciálisan a kezelendő protokollhoz készülnek, így a teljes forgalom elemzésére is képesek. Minden ilyen proxy két részből áll: egy proxy szerverből és egy proxy kliensből. A kliensek és a kiszolgálók között nem épül fel közvetlen kapcsolat, hanem mindketten a tűzfalon futó proxy alkalmazás megfelelő részével kommunikálnak. A proxy szerver fogadja a belső hálózat felől érkező kéréseket. A szerver megvizsgálja, hogy a kapcsolat nincs-e tiltva és, hogy a csomagok megfelelnek-e a protokoll szabványának. Ha mindent rendben talál, akkor a proxy szerver kapcsolódik a proxy klienshez. A proxy kliens ezek után felépíti a tényleges kapcsolatot a klienssel. A visszafelé irányuló kommunikáció is hasonlóan történik. A külső számítógép felveszi a kapcsolatot a proxy kliensével. A kliens ellenőrzés után továbbítja a proxy

szervernek a kérést, a szerver pedig felveszi a kapcsolatot a belső hálózaton lévő számítógéppel. Mivel minden protokollhoz külön proxy-ra (belső speciális alkalmazásra) van szükség, ezért ennek a módszernek a használata esetén rendelkezni kell az összes használni kívánt protokollhoz megfelelő proxy-val. Ez nagyszámú protokoll esetén komoly költséget jelenthet. A protokollonkénti proxy-k alkalmazásának viszont az az előnye, hogy egy adott proxy-n csak az adott protokoll specifikációjának megfelelő kommunikáció folyhat. A proxy tűzfalak esetén nem okoz problémát a több portot használó protokollok kezelése sem, mivel a proxy az alkalmazásszintből minden információval rendelkezik az újabb kapcsolatok megnyitásához. A további csatornákon történő kommunikációt a proxy szintén ellenőrizheti.

Dinamikus vagy állapottartó csomagszűrő tűzfalak

A dinamikus csomagszűrő tűzfalak a tűzfalak negyedik generációjának tagjai. A csomagszűrő tűzfalakhoz hasonlóan működnek, megkísérlik azok gyengeségeit kiküszöbölni. A hatékonyabb elemzés érdekében a tűzfalnak szüksége van arra, hogy azonosítani tudja a kapcsolat kezdetét és végét, valamint a kettő között zajló adatforgalmat. Ha erre képes, akkor ki tudja szűrni a kapcsolatba nem illő csomagokat, amiket potenciálisan veszélyesnek ítél. A feladatot állapottartással oldják meg, azaz a beérkező csomagokat a tűzfal addig tárolja, amíg a döntéshez szükséges információkat össze nem gyűjti. A csomagszűrő tűzfalakhoz hasonlóan a dinamikus tűzfalak is szabályláncokat használnak. A döntések meghozatalánál azonban a tűzfal nem csupán a csomagok és a szegmensek fejlécei alapján hozza meg döntéseit, hanem a csomagok közötti kapcsolatokat is figyelembe veszi. Ezt a módszert kihasználja mind a kapcsolatorientált TCP-protokoll, mind a kapcsolatnélküli UDP esetén is. Kapcsolatorientált protokoll esetén a protokoll szabályrendszerét felhasználva képes a kapcsolatok állapotának nyomon követésére akár a több porton folyó kommunikáció esetén is, mint például az FTP. A csomagok megkülönböztetése során a tűzfal figyelembe veszi, hogy adott csomag csak adott helyen jelenhet meg a kommunikációban, így egy adatokat tartalmazó csomag nem előzheti meg a kapcsolat felépítését, és nem érkezhet a kapcsolat lezárását követően sem. A dinamikus tűzfalak módszere korlátozottan képes a kapcsolat nélküli UDP-protokoll szűrésére is. A tűzfal a beérkező UDP-kérésekhez felépít egy virtuális kapcsolatot. A válaszcsomag megérkezése esetén a csomagot továbbengedi. Ha a válaszcsomag nem érkezik meg egy adott időn belül, a virtuális kapcsolatot érvénytelennek tekinti. A módszer használatával kéretlen UDP-csomagok nem tudják elárasztani a védett hálózatot.

Moduláris proxy tűzfalak

A moduláris proxy tűzfalak a tűzfal-technológia legfejlettebb módszerei közé tartoznak, a csomagok legteljesebb elemzésére képesek. Csomagszűrő képességekkel is rendelkeznek, de képesek az alkalmazásszintű elemzésre. Az alkalmazásszintű tűzfalak és a moduláris tűzfalak közötti leglényegesebb különbség, hogy míg az alkalmazásszintű tűzfalak minden protokoll értelmezésére, elemzésére különálló tűzfalkomponenssel rendelkeznek, amelyek nemképesek együttműködésre, addig a moduláris proxy tűzfal részei, moduljai képesek együttműködni. A különböző protokollokhoz tartozó proxy-k csak a saját protokollspecifikus feladatukat látják el, a kapcsolatok kiépítését például egy másik, közös modul végzi. Ha ez a modul mindent rendben talál, továbbadja a kapcsolatot egy olyan proxynak, amelyikhez tartozik (például FTP, HTTP, POP3 stb.). A modulok egymástól függetlenül, különállóan, mégis egymással összedolgozva végzik tevékenységüket. A módszer kifejezetten előnyös az összetett alkalmazásszintű protokollok esetén. Például a HTTPS-protokoll – mely egy SSL-protokollba bújtatott HTTP-protokoll – esetén külön modul kezeli az SSL-specifikus részt és külön modul a HTTP-specifikus részt. Az SSL-modul kitömöríti a kódolt csomagokat, ellenőrzi a protokoll szerinti megfelelősséget, majd további feldolgozás céljából átadja a HTTP-modulnak. A moduláris felépítésből adódóan a tűzfalnak minden protokoll kezeléséhez különálló modullal, proxy-val kell rendelkeznie. A módszer segítségével lehetőség nyílik a protokollok transzparens elemzésére, az összetett protokollok elemzésére, szűrésére, valamint lehetőség van nem protokollspecifikus modul alkalmazására is. A moduláris

proxy tűzfalak alkalmazásszintű jelenlétükből kifolyólag elvileg képesek lehetnek a teljes átmenő adatforgalom elemzésére és befolyásolására. Ehhez egyrészt rendelkezniük kell egy olyan modullal, amely ismeri a protokoll összes szabványos utasítását és metódusát, másrészt képesnek kell lenniük a protokollban átvitt adat elemzésére. Az előbbit hívjuk mélyprotokoll-elemzésnek, míg az utóbbit a tartalomelemzésnek. Ha a tűzfal ismeri a protokoll összes szabványos utasítását és képes a tartalom ellenőrzésére, így képes például a szabványt sértő kommunikáció azonosítására, tartalomszűrésre, kártékony tartalmak keresésre.

2.2. BEHATOLÁSÉRZÉKELŐ RENDSZEREK (IDS)

A behatolásérzékelő rendszerek (IDS – Intrusion Detection Systems) története az 1980- as évek elejére nyúlik vissza. Ezek a rendszerek a hálózati, illetve a számítógépes erőforrásokon olyan események nyomai után kutatnak, amelyek rosszindulatú tevékenységek, támadások jelei lehetnek. A behatolásérzékelő rendszerek célja, hogy felismerjék a számítógépeket ért támadásokat, visszaéléseket, illetve értesítsék a megfelelő személyeket, esetleg válaszlépéseket tegyenek. A behatolásérzékelő rendszerek működésük során figyelik a számítógépek, illetve hálózatok folyamatait, forgalmait, és a gyanúsnak vélt események észlelése esetén riasztanak, esetleg beavatkoznak. Saját szabályrendszerük alapján képesek eldönteni, hogy egy adott tevékenység nem megengedett, illegális tevékenységnek minősül-e. A legtöbb behatolásérzékelő rendszer nemcsak a támadás felismerésére képes, hanem valamilyen válaszlépést is képes megtenni, például megszakítja a kapcsolatot, kilépteti a felhasználót, vagy akár átkonfigurálja a védelmi rendszert.

Az IDS-rendszereket két csoportba sorolhatjuk aszerint, hogy a kiértékelendő információt honnan gyűjtik össze. Azokat a rendszereket, melyek a hálózat forgalmát gyűjtik, monitorozzák, hálózatalapú behatolásérzékelő rendszereknek (NIDS – Network-based Intrusion Detection Systems) nevezzük. A másik nagy csoportot a hoszt-alapú behatolásérzékelő rendszerek (HIDS – Host-based Intrusion Detection Systems) jelentik, melyek munkaállomásokon futnak, és az operációs rendszer, illetve a futó alkalmazások viselkedését figyelik.

Host-alapú behatolásérzékelő rendszerek (HIDS)

A hoszt-alapú behatolásérzékelő rendszerek (HIDS) számítógépes végpontokra feltelepített rendszerek. Működésükhöz az információt a felügyelt számítógépes végpontból veszik, ezáltal nagyon precíz és megbízható elemzésre képesek. A számítógépen futó folyamatok, tevékenységek paraméterei, részletei mind az elemzés rendelkezésére állnak, ezáltal pontosabb és precízebb riasztásra képesek. Ellentétben hálózatalapú behatolásérzékelő rendszerekkel (NIDS), képesek megfigyelni a támadás eredményét, célját.

A hoszt-alapú behatolásérzékelő rendszerek (HIDS) egy speciális alcsoportja az alkalmazásalapú behatolásérzékelő rendszerek, melyek a különböző felhasználói programok működése során keletkező naplófájlok elemzésével foglalkoznak. A felhasználói programmal való közvetlen kapcsolat a lehető legpontosabb elemzést teszi lehetővé a behatolásérzékelő rendszerek számára.

A hoszt-alapú behatolásérzékelő rendszerek az 1980-as és 1990-es évek vírusvédelmi megoldásaiból nőtték ki magukat. Az 1980-as évek végén még víruskereső programok képezték a végpontok védelmét. Ezeket a programokat a felhasználó tudta elindítani, és az átvizsgálta a számítógépet, de folyamatos védelemre nem voltak képesek. Az 1990-es években már a folyamatos védelmek is megjelentek, később ez egyre fontosabbá vált a Microsoft Windows térhódításával. A hoszt-alapú behatolásérzékelő rendszereknek a következő nagy lökést az internet fejlődése adta. A végpontvédelmi rendszerekbe ekkor kerültek bele hálózati forgalom felügyeletét biztosító megoldások, kezdetben

egyszerű tűzfalak, később pedig a különböző behatoláskezelő lehetőségek, és kialakultak az úgynevezett Internet Security-megoldások.

Az alábbiakban a végpontokhoz köthető védelmi rendszerek legáltalánosabb hagyományos módszereit részletezzük, amelyeket a védelmek használnak a kártevők felismerésére, azonosítására:

- A legegyszerűbb módszer a *bájtsorozat alapú keresés*, melynek során a kártevő kódjából választanak egy, a kártevőre jellemző bájtsorozatot, és amennyiben ezt megtalálják egy programterületen, akkor az adott kártevő jelenlétét jelzik. Általában legalább 30-100 bájt hosszúságú bájtsorozatot praktikus választani, csökkentve ezzel a vakriasztásokat. Az Intel 80x86 processzor alapú számítógépeken az eljárást meg lehet valósítani oly módon, hogy az algoritmus sebessége gyakorlatilag ne függjön a minták számától, csupán az ellenőrizendő állomány hosszától. A módszer azonban csak abban az esetben használható, ha egyrészt a kártevő ismert, és az nem változtatja a kódját. Mutációs vírusok esetén csak az oligomorf vírusok dekódoló ciklusának a felismerése oldható meg bájtsorozat alapú kereséssel, polimorf, metamorf kártevők esetén pedig a módszer nem használható.
- Manapság már nem létezik olyan megbízható vírusvédelmi megoldás, ami ne rendelkezne beépített emulátorral, amely tulajdonképpen egy hardver-szoftver környezetet szimulál. Ebben a virtuális környezetben a védelmi rendszer azt vizsgálja, hogy mit tenne, milyen tevékenységeket végezne a vizsgálandó kód, ha elindulna. Ilyenkor a vizsgált kód első néhány ezer, tízezer, százezer utasítását az emulált környezetben végrehajtják, és az elvégzett tevékenységeket az előre adatbázisba elmentett tevékenységi „mintákkal” hasonlítják össze. Az ilyen *emulátor alapú felismerési algoritmus* a mutációs vírusok ellen is alkalmazható, ugyanakkor lényegesen lassabban valósítható meg.
- A számítógépes kártevők jelentős része nem képes más programkódba beépülni, csak önmagában, változtatás nélkül terjed. Ilyenek például az e-mailek mellékletében terjedő férgek is. A saját kódjukat nem változtató kártevők felismerésére az *ellenőrző összeg számítása* hatékony lehetőséget kínál. A módszer alapján az ismert kártevők kódjáról egy ellenőrző összeget számolnak, ez kerül az adatbázisba, majd a védelem a vizsgált kód ellenőrző összegét hasonlítja össze az adatbázisban lévővel.
- Előfordulhat olyan speciális kártevő, amelyre sem a bájtsorozat alapú keresés, sem az emulátor alapú felismerési algoritmus, sem pedig az ellenőrző összeg számítása nem használható. Az ilyen esetben általában specializált, külön az adott kártevőre vagy a kártevők egy szűkebb csoportjára készített *speciális keresési algoritmus* használható.
- A felsorolt módszerek mindegyike csupán a már ismert, azaz a védelmi rendszer gyártója számára rendelkezésre álló kártevő felismerésére, azonosítására szolgál. A fenti módszerek közül azonban a *bájtsorozat alapú keresés és az emulátor alapú felismerési algoritmus* (főleg az utóbbi) könnyen használható a védelmi rendszerek gyártói számára ismeretlen kártevő felismerésére is. Ilyen esetben az algoritmusok nem a konkrét, ismert kártevők mintáihoz hasonlítják a vizsgált kódot, hanem csupán kártevőkre utaló jeleket keresnek. Ezt a módszert *heurisztikus keresésnek* nevezik.
- Az ismeretlen kártevőkre utaló jeleket azonban nemcsak a védelem virtuális környezetében, hanem a valós, működő rendszerben is folyamatosan keresheti a védelem. Ez esetben a módszert *viselkedésalapú algoritmusnak* hívják.

A fenti módszereket a végpont egy védelmi rendszere különböző események alkalmával használja:
- A felhasználó által indított ellenőrzés alkalmával. Ezt *on-demand* ellenőrzésnek is szokták nevezni.
- Abban az esetben, ha az operációs rendszer az ellenőrzendő állományt bármilyen okból megnyitja. Ez történhet például másolás, letöltés, végrehajtás hatására, és a védelmi rendszer akár különböző mélységű vizsgálatokat is végezhet a különböző esetekben. Például letöltés esetén az újonnan érkező állományt tüzetesebb vizsgálatnak veti alá, mint másolás esetén. Az ilyen típusú ellenőrzést *on-access* ellenőrzésnek is nevezik.
- Abban az esetben, ha az operációs rendszer vagy valamely alkalmazás az ellenőrzendő állományt értelmezi és végrehajtja. Ebben az esetben a védelem az adott programkód tevékenységét vizsgálja, az általa végrehajtott műveletek révén. Ilyen esetben *dinamikus* ellenőrzésről beszélünk.

Hálózatalapú behatolásérzékelő rendszerek (NIDS)

A hálózatalapú behatolásérzékelő rendszerek (NIDS) a hálózati forgalom ellenőrzésével végzik feladatukat. Figyelik a hálózat egészén vagy annak egy részén áthaladó kommunikációt, és védik a védett hálózatnak tekintett hálózati részen elhelyezkedő munkaállomásokat. A hoszt-alapú behatolásérzékelő rendszerek (HIDS) helyi megfigyelésével szemben a hálózatalapú behatolásérzékelő rendszerek a hálózati csomagok begyűjtésével (packet-sniffing) és elemzésével végzik védelmi tevékenységüket. Ezek általában speciális szenzorok, hálózati eszközök, esetleg egy számítógépen futó programot jelentenek. Ezek a rendszerek tehát figyelik és kiértékelik a hálózat forgalmát, azonosítják az esetleges támadásokat, mely támadások tényét továbbítják egy központi felügyelő rendszernek. Az összegyűjtött adatokat gyakran összehasonlítják a már ismert támadási sémákkal, ezáltal kideríthető, hogy az adott események kártékony vagy veszélytelen tevékenységet jelentenek-e. Tekintettel arra, hogy a támadási lehetőségek folyamatosan változnak, bővülnek, ezért a megfelelő működéshez elengedhetetlen a sémákat tartalmazó adatbázis folyamatos frissítése, karbantartása, illetve egyes hálózatalapú behatolásérzékelő rendszerek esetén arra is van lehetőség, hogy a megismert, kártékonynak ítélt eseményeket felhasználva – tanulási folyamatként – egy új sémát építsenek fel. Amennyiben valamilyen károsnak vélt folyamatot érzékel a rendszer, akkor valamilyen riasztás, illetve a káros folyamat esetleges leállítása következhet. A hálózatalapú behatolásérzékelő rendszerek általában nem végeznek más tevékenységet, így könnyen elrejthetők a hálózaton, ami egy esetleges betörés esetén megnehezíti létezésük és helyük felderítését. Mivel a hálózati kommunikációt figyelik, nem pedig egy munkaállomás megfigyelését végzik, ezáltal különösen alkalmasak a hálózaton kívülről érkező támadások azonosítására.

A stack-alapú behatolásérzékelő rendszerek a hálózatalapú behatolásérzékelő rendszerek egy alcsoportjának tekinthetők. A technológia megvalósítása kifejezetten az egyes gyártókra jellemző megoldásokat tartalmaz, azonban általánosan jellemző, hogy többségében hardverközeli megoldást valósítanak meg, figyelik és elemzik a hálózati csomagokat a különböző hálózati rétegek szintjein, ahogy egyre feljebb kerülnek, még mielőtt elérnék az operációs rendszert vagy a felhasználói alkalmazást.

Hibrid rendszerek

A hoszt-alapú és a hálózatalapú behatolásérzékelő rendszerekben alkalmazott módszerek ugyan jelentősen eltérnek egymástól, azonban a két rendszer hatékonyan képes kiegészíteni egymást. A teljeskörű védelem érdekében mindkét rendszerre szükség van, együttműködésükkel sokkal hatékonyabb védelem valósítható meg. A hoszt-alapú behatolás érzékelő rendszerek pontosabb elemzést tesznek lehetővé, ugyanakkor a hálózatalapú behatolásérzékelő rendszerek viszont nagyon hatékonyak lehetnek a teljes hálózat monitorozásában, felügyeletében.

Behatolásérzékelő rendszerekben alkalmazott technológiák

A behatolást érzékelő rendszerek két fő technológiát használnak az események analizálására, a támadások észlelésére. A visszaélést érzékelő módszer az ismert kártékony viselkedéseken, a rendellenességet érzékelő módszer az ismert normális viselkedéseken alapul.

A **visszaélést érzékelő módszer**en alapuló behatolásérzékelő rendszerek esetében tehát az ismert támadások és sebezhetőségek lenyomatait, ismérveit tárolja a rendszer, és ha olyan adatokat észlel, amik a tárolt adatokkal egybeesnek, akkor riaszt. A visszaélést érzékelő módszer főbb működési elvei:

- **A szakértői rendszerek**en alapuló behatolásérzékelő rendszerek szabályalapon működnek, egy-egy szabály egy rendellenes viselkedés leírását és a végrehajtandó tevékenységet tartalmazza.
- **A modellalapú következtetés** módszere egy magasabb absztrakciós szinten végzi a kereséseket, mint a csak mintaillesztő szakértői rendszerek. A támadási mintákat egyéb információkkal egészítik ki, melyeket az elemzés során a rendszer felhasznál. Ez a technika hasznos az olyan támadások kiderítésére, melyek lenyomatai nagyon hasonlóak, de mégsem azonosak.
- Az állapotátmeneti elemzés módszere az ismert támadások állapotátmeneti modellje alapján dolgozik. A rendszer a modell kezdeti állapotában tartózkodik alapállapotban, és ha a támadó tevékenységei folytán a rendszer a köztes állapotokon keresztül egy veszélyes állapotba kerül, akkor a rendszer a támadást észlelve riasztást küld.
- A neuronhálókon alapuló behatolásérzékelő rendszerek egy hatékonyabb, kevésbé összetett, jobban átlátható, jobb hatékonyságú rendszert eredményeznek, azonban széleskörűen még nem terjedtek el.

A **rendellenességet** érzékelő **módszer** a hálózaton vagy számítógépen bekövetkező nem normális jelenségeket, anomáliákat figyeli. Az ilyen módszert alkalmazó rendszerek működésének módja azon a feltételezésen alapul, hogy a számítógép működése különbözik a támadások során és a normál felhasználás esetén. Ezek a rendszerek a normális működésre jellemző sémákat definiálnak a számítógép vagy hálózat működésére vonatkozóan, a megtanult sémáktól eltérő viselkedés esetén pedig a rendszer riasztást küld. A rendellenességet érzékelő modell főbb működési elvei:

- **A küszöbérték figyelésén** alapuló módszer szerint a rendszer küszöbértékeket rendel a normális tevékenységekhez, a küszöbérték elérését pedig támadásként értékeli és riasztást küld.
- **A felhasználói viselkedési séma figyelése esetén** a rendszer minden egyes felhasználóhoz létrehoz egy felhasználói sémát, amiben a felhasználó szokásos tevékenységei, az általa használt programok, a felhasználótól várható események találhatóak. Ha a rendszer a séma és az aktuális események között jelentős eltérést tapasztal, akkor riasztást küld.
- **A csoportos séma figyelése** hasonló a **felhasználói viselkedési séma figyeléséhez** azzal a különbséggel, hogy a rendszer a felhasználók csoportjainak sémáin végzi a vizsgálatokat, nagyban csökkentve ezzel a karbantartandó sémák számát.
- **Az erőforrások sémáinak figyelése esetén** a rendszer a teljes környezet használatáról készít és menedzsel egy sémát, ami többek között tartalmazza a felhasználók, programok, háttértárak, protokollok, kommunikációs portok használatával kapcsolatos azon eseményeket, melyek a normális működésre jellemzőek.
- **A futtatható állományokról készített séma használata esetén a védelem** a védendő rendszer futtatható állományairól és azok használatáról, működéséről készít sémát. A védelem ilyenkor elsősorban azokra az alkalmazásokra figyel, amelyek a felhasználó beavatkozása nélkül, önállóan is képesek tevékenykedni. Ha a védelem a sémában szereplő megszokottól eltérő használatot észlel, akkor természetesen riaszt.

A hatékony védelem megvalósítása érdekében a sémákat célszerű naprakészen tartani, folyamatosan frissíteni. Ez a legtöbb ilyen módszert használó behatolást érzékelő rendszerek esetén automatikusan is megtörténik, egyes esetekben pedig lehetőség van arra is, hogy a felhasználó a saját magára vonatkozó sémát frissítse.

2.3. BEHATOLÁSMEGELŐZŐ RENDSZEREK (IPS)

Az 1990-es években megjelenő támadások és hálózaton terjedő kártevők ellen fejlesztették ki az első behatolásmegelőző rendszereket. A behatolásérzékelő rendszerek kezdetben teljesen alkalmasnak bizonyultak az akkor még kisszámban előforduló támadások kezelésére, azonban a támadások többszöröződésével egyre inkább fokozódott az igény a támadások megelőzésére is, nemcsak azok jelzésére.

A behatolásmegelőző rendszerek célja a támadás blokkolása, sikeres befejezésének a megakadályozása. Ezek a rendszerek proaktívak, nem várják meg, míg a támadás kialakul vagy befejeződik. A behatolásmegelőző rendszerek a behatolásérzékelő rendszerekhez hasonlóan képesek a támadás észlelésére, viszont a tűzfalakhoz hasonlóan lehetőségük van a támadás blokkolására. A behatolásérzékelő rendszerekhez hasonlóan a védelmi rendszereknek itt is két fő csoportja létezik: hoszt-alapú behatolásmegelőző rendszerek és hálózatalapú behatolásmegelőző rendszerek.

A hoszt-alapú behatolásmegelőző rendszerek – ugyanúgy, mint a hoszt-alapú behatolásérzékelő rendszerek – a védett számítógépen helyezkednek el. Az operációs rendszerhez, annak szolgáltatásaihoz, illetve a számítógép alkalmazásaihoz közvetlenül hozzáférnek. Figyelik, monitorozzák a rendszerhívásokat, a be- és kimenő hálózati forgalmat, a futó folyamatokat és igyekeznek megakadályozni a támadásokat. Ha a védelmi rendszer támadást érzékel, akkor blokkolja a támadást, sőt bizonyos esetekben megpróbálja megelőzni a támadásokat, például puffer-túlcsordulási támadás esetén a roszszindulatú kód figyelésével és futtatásának blokkolásával.

A hálózatalapú behatolásmegelőző rendszerek már nemcsak figyelik a hálózat egy adott pontján áthaladó forgalmat, hanem képesek arra, hogy a hálózati forgalomba beavatkozzanak. Ez azt is jelenti, hogy a hálózati forgalom is rajtuk keresztül kell, hogy folyjon. Az áthaladó adatokat a rendszer elemzésnek veti alá, és támadási nyomokat keres, csakúgy, mint ahogy ezt a hálózatalapú behatolásérzékelő rendszerek teszik. A különbség csupán annyi, hogy a behatolásérzékelő rendszerekkel ellentétben itt nemcsak riasztás, hanem hatékony beavatkozás is történik, jellemzően a kártékony tevékenység blokkolásával.

Hálózatalapú behatolásmegelőző rendszerekben alkalmazott technológiák

Hálózatalapú behatolásmegelőző rendszerek esetén a védelmi rendszer legfontosabb feladata, hogy képes legyen a hálózati forgalom, a hálózati kommunikáció elemzésére, analizálására. Ehhez alapvető fontosságú, hogy egy adott porton zajló azonosított kommunikáció esetén el tudja dönteni, hogy ott milyen protokoll szerint zajlik a hálózati forgalom. A védelmi rendszert ebben a protokollfelismerési és -azonosítási technikák segítik. Amennyiben egy kommunikáció esetén sikerült azonosítani a használt protokollt, akkor az információáramlás elemzésére a forgalomelemzési technikák szolgálnak.

Egy adott kommunikációs csatornán (porton) folyó hálózati forgalom elemzése előtt szükség van magának a protokollnak a beazonosítására. Ha a protokoll azonosítása sikertelen vagy hibás, akkor nagyon sok hamis pozitív vagy hamis negatív riasztást kaphatunk. Többféle azonosítási technika alkalmazása pedig pontosabb eredményhez vezethet. A legtöbb hálózatalapú behatolásmegelőző rendszer az alábbi protokollazonosítási technikákat alkalmazza:

- **A port** és **protokoll** összerendelése **történhet a szabvány szerint, mely** az egyik legegyszerűbb módja a folyamat által használt port beazonosításának. A módszer azonban önmagában megbízhatatlan, de előzetes azonosításként mindenféleképpen felhasználható.
- **Heurisztikus módszerek használatával a védelmi rendszer a** protokollnak valamilyen egyedi tulajdonságát használja ki. Vannak ugyanis olyan protokollok, melyek nem egy előre definiált porton keresztül kommunikálnak, hanem tetszőleges nyitott portot felhasználhatnak az adatcserére, és ebben az esetben általában ez az egyetlen módszer az azonosításra.
- **A védelmi rendszer a port-követés** módszerével az előzőleg beazonosított kommunikációs csatorna által megnyitott további portokat is figyeli. Az ilyen alkalmazások általában megnyitnak egy adott portot, hogy felvegyék a kapcsolatot a másik számítógéppel, azonban a tényleges adatátvitel nem ezen a csatornán folyik, hanem további megnyitott portokat használ a rendszer.
- **A protokollok teljes körű azonosítása érdekében a védelmi rendszerek nem kerülhetik meg a protokollbújtatás felismerését.** A protokollbújtatás technikájának alkalmazása során az egyik protokollba egy másik protokoll adatfolyamát helyezik el.

A forgalom elemzése csak azután történhet, miután a protokollok megfelelő azonosítása megtörtént. Ha a protokollazonosítás nem sikerül, esetleg hibás, a védelmi rendszer nem tudja, hogy milyen adatokra számítson, azokat mely protokoll szerint értelmezze. A forgalomelemzési technikák esetén is igaz, hogy többféle módszer használata megbízhatóbbá teszi az elemzést. A legtöbb hálózatalapú behatolásmegelőző rendszer az alábbi protokollazonosítási technikákat alkalmazza:

- A **protokollanalízis** egy széleskörűen használt módszer a hálózatalapú behatolás megelőző rendszerekben, mind az ismert, mind pedig az ismeretlen támadások megakadályozására. A protokollanalízis az OSI modell 2. rétegétől felfelé végzi az elemzést, és ha egy hálózati csomagot nem elfogadhatónak minősít a rendszer, a védelmi rendszer blokkolja a forgalmat.
- A védelmi rendszer az **RFC szerinti kompatibilitásellenőrzéssel** folyamatosan ellenőrzi a kommunikációs csatornán folyó csomagok RFC-szabvány szerinti megfelelőségét, tulajdonságait.
- **A védelmi rendszereknek képesnek kell lenniük a szétdarabolt csomagok újraegyesítésére, annak érdekében, hogy** a több kisebb csomagra darabolt adatfolyamot is ellenőrizni tudják. A támadók szívesen használják a hálózati csomagok szétdarabolásának a technikáját a védelmi rendszerek megkerülésének érdekében. A védelmi rendszernek képesnek kell lennie az ilyen csomagok elemzésére azok továbbküldése előtt.
- Az **adatfolyam-megfigyelés** módszere hasonló a csomagok újraegyesítésének technikájához, de a védelmi rendszer ebben az esetben a folyamat egészét elemzi, szemben a csomagok elemzésével, azaz itt egyesíteni kell a folyamathoz tartozó különálló csomagokat.
- **A statisztikai küszöbértékek analízise** a hálózati rendellenességek érzékelésén és blokkolásán alapul. A védelmi rendszer a működésének kezdetén egy adott időszakban megfigyeli a szokásos hálózati forgalmat és a normális működéshez küszöbértékeket rendel. Az olyan hálózati forgalmat, amely ezen határértékeken kívül esik, a védelem blokkolja.
- A **mintaillesztés** az egyik leggyakrabban használt elemzési módszer. Az eljárás hasonló a behatolásérzékelő rendszereknél bemutatott módszerhez. A behatolás megelőző rendszerek esetén az áthaladó hálózati forgalmat a védelmi rendszer összehasonlítja a saját adatbázisában lévő, előre definiált támadási sémákkal. Ha egyezést talál, akkor blokkolja a forgalmat.

2.4. VÉDEKEZÉS A CÉLZOTT TÁMADÁSOK ELLEN

A hálózaton, átjárókon és végpontokon lévő hagyományos védekezési módszerek vitatható szerepet játszanak egy szervezet adatainak védelmére, illetve működésének a fenntartására vonatkozóan. A célzott támadások, az APT-k fejlődésével bebizonyosodott, hogy a hagyományos védelmi rendszerek a szignatúra alapú, illetve az elterjedtségi adatbázis frissítéseitől függő korlátokkal rendelkeznek a fenyegetések azonosítása terén, és valójában alkalmatlanok egy valós célzott támadás, illetve APT időben történő azonosítására. Ezek a támadások azért tudnak sikeresek lenni, mert a 0. napi (zero day) támadások a hagyományos védelmi rendszerek számára – azok tervezéséből adódóan – láthatatlanok, illetve egy támadó manuálisan végrehajtott tevékenységei vagy detektálatlanok, vagy pedig nagyon nehezen felismerhetők, mivel a különböző védelmi rendszerek (például tűzfalak) naplóállományainaka legmélyén találhatók.

A védelmi rendszerek gyártói nagyon keveset vagy szinte semmit sem tettek az aktuális termékeik korlátait vagy új technológiák fejlesztését illetően. Néhány kezdő vállalkozás új innovatív ötletekkel próbálkozott, elsősorban a 0. napi (zero day) támadások azonosítását megcélozva. Ezek a próbálkozások azonban szinte kivétel nélkül a Microsoft operációs rendszereken futó kártevőkre vonatkoznak, nem foglalkoznak a támadó tevékenységeinek az azonosításával, és további gondot jelent, hogy más védelmi rendszertől teljesen függetlenül működnek, nem veszik figyelembe azok jelzéseit. Ezen túlmenően az újgenerációs tűzfalak, IPS-ek és egyéb hálózati védelmi eszközök gyártói szintén megpróbálták felvenni a harcot az APT-kkel szemben egy új technológiájú felhőalapú virtuális környezet beépítésével a jelenlegi védelmi rendszereikbe. Ez azonban nem jelentett áttörést az APT-kkel szemben.

Annak ellenére, hogy az APT-k nagyarányú növekedése során a módszer a legelterjedtebb támadási formává vált, a védelmi rendszerek gyártói valójában nem tudtak hatékony segítséget adni egy informatikai infrastruktúra védelméért felelős vezető kezébe.

Egy informatikai infrastruktúra védelemért felelős vezetője esetében az APT-k elleni védekezés megköveteli, hogy a biztonsági technikák alkalmazása mellett a hálózati forgalom minél teljesebb körű monitorozására is nagyobb figyelmet fordítson. Ezt ugyan a korszerű védelmi rendszerek megteszik, azonban egy hatékony felügyelet nagyban segíthet a legújabb, még ismeretlen támadások elleni védekezésben. Mindemellett a technikai védelmet nagyon fontos kiegészíteni a felhasználók biztonsági tudatosságának a növelésével is, mely megvalósulhat a teljes intézményt érintő vagy célzott biztonsági oktatással is.

3. VÉDELMI RENDSZEREK VÁLASZTÁSA

A védelmi rendszerek kiválasztása nem egy egyszerű feladat. Alapvetően három módszer közül választhatunk:

1. Hagyatkozunk a gyártókra, hogy mit állítanak termékeikről.
2. Saját magunk próbálkozunk meg a védelmek vizsgálatával, tesztelésével.
3. Független tesztelő szervezet elemzéseire bízzuk választásunkat.

Az első eset nem feltétlenül járható, mivel minden gyártó a saját termékeit dicséri, hiszen azt szeretné eladni. A másik két lehetőség alapja a gyártótól független vizsgálat. Azonban ennek a kivitelezése sajnos nem kevés erőforrást igényel, így egy korrekt vizsgálatot nem feltétlenül tud egy átlagos felhasználó elvégezni. Ebben a fejezetben a védelmi rendszerek vizsgálatának, minősítésének legfontosabb problémáit tekintjük át.

3.1. VÉDELMEK VIZSGÁLATÁNAK PROBLÉMÁI

A védelmi rendszerek tesztelése, vizsgálata a szoftvertesztelésnek egy teljesen különálló területévé nőtte ki magát, ez az „általános” szoftvertesztéléshez képest egy teljesen más terület. Ezen a területen ugyanis számos speciális körülmény nehezíti a vizsgálatot.

Egyrészt talán a védelmi rendszereken kívül nincs még egy olyan szoftverterület, ahol olyan gyorsan jelennének meg az új verziók. Ennek oka, hogy – mint matematikailag is bizonyított – általános vírusvédelem nem létezhet. A vírusvédelmek gyártói így az ismert kártevőkkel, támadási lehetőségekkel foglalkoznak elsősorban, ellenük próbálnak 100%-os védelmet biztosítani. Természetesen voltak és vannak is próbálkozások, kísérletek arra vonatkozóan, hogy bizonyos módszerekkel ismeretlen kártevőket, fenyegetéseket is azonosítsanak. De mitől lesz egy kártevő ismert? Az “ismertség” itt arra vonatkozik, hogy a vírusvédelem gyártója a konkrét kártevő (vagy esetleg egy csoportjuk) ismeretében készíti el a kártevő ellenszerét (felismerési, azonosítási, eltávolítási vagy blokkolási algoritmusát). Így viszont az újabb és újabb kártevők megjelenésével újabb és újabb verziójú védelmeket kell kibocsátani. Az 1980-as évek végén, az 1990-es évek elején még havonta, negyedévente jelentek meg a védelmi rendszerek újabb verziói. 2005-2010 környékén a védelmi rendszerek fejlődése elérte azt a szintet, melynek során átlagosan kevesebb, mint 10 percenként (!) adtak ki egyes gyártók egy új verziót. A 2010-es évek elejére nyilvánvalóvá vált, hogy a szinte folyamatos frissítés sem jelent már megoldást. A kártékony programok, url-ek olyan gyorsan jelennek meg, hogy ha ezekkel egy védelem lépést szeretne tartani, akkor a felhőtechnológiát (cloud technology) kell segítségül hívnia. Ez azt jelenti, hogy a védelmek a folyamatos internetes kapcsolat révén a gyártó által felállított központtól

segítséget kérve hozzák meg döntéseiket. Így viszont a védelem működése különböző időpontokban más és más lehet, ami maga után vonja, hogy a védelmi rendszerek tesztelését, vizsgálatát nem lehet megismételni, reprodukálni.

További lényeges különbséget jelent, hogy a kártevők nagy száma miatt rengeteg tesztkörnyezetet feltételezhetnénk. Tegyük fel ugyanis, hogy létezik 100 millió kártevő. Elméletileg (de hangsúlyosan csak elméletileg) egy kártevő vagy jelen van egy tesztelési környezetben vagy nem, azaz így $2^{100.000.000}$ db tesztelési környezetet képzelhetnénk el. Ez természetesen kezelhetetlen, de még akkor is gondot okoz a mennyiségi probléma, ha csak azokat az eseteket vesszük alapul, ahol csak egyetlen kártevő van jelen (ez nyilván 100 millió a példa szerint).

Nagyon nehéz definiálni, hogy egy védelmi rendszernek milyen elvárásoknak kell megfelelnie, mi az a működés, ami a védelem szempontjából korrekt, megbízható működésnek tekinthető. Az sem egyértelmű – és ebben is vannak különbségek a gyártók között –, hogy mely programokat tekintünk kártevőknek és melyeket nem. Sok védelem ugyanis olyan programokat is kártevőként azonosít, melyek esetén kérdéses a megítélés. Például egyes védelmek valamely szoftverhez licenszkulcsot generáló ún. feltörő (crack) programot is kártevőként azonosítanak. Hasonlóan gondot jelent, hogy egyre-másra jelennek meg az olyan, gyakorlatilag hivatalosnak tűnő alkalmazások, melyek mögött teljesen legális cégek állnak. Ezek a cégek aztán mindent megtesznek azért, hogy a védelmek gyártóinak adatbázisából kikerüljön a termékük.

Az AMTSO-t (Anti-Malware Testing Standards Organization) 2008-ban alapították elsősorban védelmi szoftvereket és hardvereket fejlesztők, de a munkában számos tesztelést végző szervezet is részt vesz. Az elsődleges célkitűzések között szerepel, hogy a védelmi megoldások tesztelésére vonatkozóan ajánlásokat, oktatási anyagokat dolgozzon ki, tesztelést segítő eszközöket bocsásson rendelkezésre, illetve az érintett felek között fórumot biztosítson.

3.2. VIZSGÁLATI MÓDSZEREK

Védelmi rendszerek esetén a fent vázolt problémák miatt elképzelhetetlen, hogy annak minden végrehajtási ágát vizsgáljuk. Bizonyos szempontokat – melyek megfelelnek az AMTSO ajánlásainak is – azonban célszerű betartani.

A tesztelési eljárásnak célszerűen nyíltnak és átláthatónak kell lennie. Ez vonatkozik egyrészt a tesztelési metódus és a konkrét tesztelési folyamatok átláthatóságára is. A nyíltság és átláthatóság követelménye nem feltétlenül jelenti a reprodukálhatóság biztosítását. Ennek legfőbb oka a dinamikus internethasználat, mely mind a védelmek (fenti 2. probléma), mind a kártevők, fenyegetések részéről egyaránt jelen van. A nyíltságot és átláthatóságot azonban a dokumentáltság szintjének emelésével javíthatjuk. Ehhez célszerű a tesztelt állapotot (installált rendszert), a naplóállományokat, hálózati forgalmat elmenteni, illetve a problémás szituációkról képernyőképet és videót is rögzíteni.

Mint azt a fenti problémák esetén láthattuk, a vírusvédelmek valamennyi folyamatát nem tesztelhetjük. Bizonyos szempontokat vizsgálhatunk “csak”, célszerűen azokat, melyek a legfontosabbak a felhasználó szempontjából:

Hatékonyság, azaz a védelem milyen biztonsági szintet tud biztosítani (kártevők, fenyegetések korrekt ismerete – felismerés, eltávolítás, blokkolás). Ez a szempont kifejezetten a védelem tudásbázisára vonatkozik: Milyen kártevők, fenyegetések elleni védelemre készítették fel a védelmet? Milyen területeket képes vizsgálni? Ide tartoznak a különböző háttértár-területek vizsgálati lehetőségei, fájlformátumok, tömörítők és egyéb, fájlok tárolására használható formátumok vizsgálati lehetősége, illetve a hálózati protokollok ismerete (vagyis mely protokollokat képes azonosítani és kezelni). Ebbe a csoportba tartozik természetesen a téves riasztások vizsgálata is.

Megbízhatóság (stabil, hibamentes működés). Képes-e a védelem úgy működni, hogy folyamatosan ellássa feladatát? A stabilitás vizsgálatánál a tesztelés során hangsúlyozottan kell törekedni a reprodukálhatóságra.

Teljesítmény (sebesség, számítógép lassítása, bootolás időigénye). Általában a teljesítménnyel szembeni felhasználói elvárás akkor jelenik meg, ha védelem képes hibamentesen megfelelő biztonsági szintet nyújtani. Nem véletlen, hogy az AMTSO alapelvei között szerepel az, hogy a hatékonyságot és a teljesítményt együtt, kiegyensúlyozottan kell vizsgálni. Ennek oka az, hogy ha például egy védelem nem vizsgál meg bizonyos dolgokat, nyilván gyorsabban képes végrehajtani feladatát. De hiába gyorsabb, ha nem véd bizonyos problémákkal szemben.

Vírusvédelmi rendszerek tesztelése esetén – a kártevők ismeretének vizsgálata során – kulcskérdés, hogy milyen és mennyi mintát használunk. A minőségre vonatkozóan az AMTSO több dokumentumot is elfogadott, mely a használt minták kérdéskörével, például azok validálásával foglalkozik. Eszerint a vizsgálat során olyan kártevőmintákat, olyan fenyegetéseket kell alkalmazni, melyek egyrészt működőképes kódot tartalmaznak, illetve valóban képesek a kártékony tevékenységet végrehajtani. Ennek biztosítása/bizonyítása nem egyszerű feladat. Mint az a fenti problémák között is szerepelt, nem egyértelmű, hogy mi tekinthető kártevőnek és mi nem. Hasonlóképpen számos védelmi rendszer olyan állományt is kártevőként azonosít, amelyet semmilyen környezetben nem lehet futóképessé tenni, azaz végrehajtani, például tönkretett futtatható állományok. A vizsgálat szempontjából a mennyiség is nagyon fontos kérdés, azonban a használt minták mennyiségét (mint a vizsgálati metódus egy lényeges elemét) a vizsgálat céljának kell meghatároznia. A vizsgálat kiterjedhet egyrészt a létező kártevők, fenyegetések összességére. Ebben az esetben nyilván többmilliós nagyságrendű készlettel lehetne statisztikailag helyes következtetésre jutni (itt viszont a korrekt validálás okozna nagy problémát). A szűkítésre két lehetőség kínálkozik: egyrészt vizsgálhatunk bizonyos típusú kártevőket, fenyegetéseket, amivel egy adott területre fókuszálva vonhatunk le következtetéseket a védelmek tudására vonatkozóan. Másrészt figyelembe vehetjük az elterjedtségi adatokat is. A létező kártevők, fenyegetések mennyiségéhez képest az elterjedt kártevők és fenyegetések száma több nagyságrenddel kevesebb. Ezen támadási lehetőségek körében már statisztikailag is helyes következtetést vonhatunk le néhány ezres mintakészlettel. Felmerülhet még továbbá olyan vizsgálatnak a lehetősége is, ahol nem az a cél, hogy a felismerési képesség alapján rangsoroljuk a védelmeket, hanem hogy egy-egy konkrét biztonsági probléma kezelésére következtessünk, azaz például válaszoljunk olyan kérdésekre, amelyek arra terjednek ki, hogy valamely felmerült probléma jelent-e biztonsági kockázatot. Ilyen esetben néhány minta is elegendő lehet a vizsgálathoz.

A védelmi rendszerek számos tevékenységgel azonosíthatnak egy kártevőt. Amennyiben a felhasználó indítja el interaktív módon a vizsgálatot, on-demand eljárásról beszélünk. Az azonosítást a folyamatosan figyelő ún. on-access védelem is megteheti. Ez utóbbi felfedezheti a kártevőket a fájlhoz történő hozzáféréskor, de előfordulhat, hogy bizonyos kártevőket a védelem csak akkor képes azonosítani és persze blokkolni, ha azokat elindítjuk. Ez utóbbi – proaktívnak hívott – működés egyre gyakoribb a védelmek esetén. A vizsgálati folyamat során tehát célszerű proaktív vizsgálatot végezni. A módszer természetéből adódóan ezt viszont csak úgy szabad elvégezni, ha az operációs rendszer bootolását követően minden egyes alkalommal csak egyetlen kártevőpéldánnyal tesztelünk. Az ily módon történő vizsgálat ezért rendkívül időigényes. Gondoljunk csak bele: Ha egy számítógépet használunk egy védelmi rendszerhez és egyetlen kártevő tesztelése 5 percet vesz igénybe – amibe beletartozik az eredeti, biztosan kártevőmentes operációs rendszer visszaállítása is –, akkor egymillió minta teszteléséhez mintegy 9 és fél évre lenne szükség. Proaktív vizsgálatot mindezek alapján tehát nagy mennyiségű mintán belátható időn belül nem végezhetünk. További nehézséget jelent, hogy egyes kártevők az azonosításukhoz szükséges tevékenységeket esetenként internetes kapcsolat révén biztosítják. Ebben az esetben viszont bonyolult feladat annak biztosítása, hogy a kártékony kód ne kerülhessen ki a vizsgálati környezetből és mindemellett a védelem rendelkezzen internetkapcsolattal.

Fenyegetések ismeretének a vizsgálata (felismerés): Alapfeltétel, hogy hoszt-alapú védelmi rendszer esetén a vizsgált megoldás képes legyen valamilyen naplóállományt készíteni a vizsgálatról. A felhasználó által indított on-demand ellenőrzés során a kártevők találatainak az összesítése történik. A folyamatosan figyelő on-access védelmek esetén a vizsgálat kicsit bonyolultabb. Itt nemcsak a naplóállományokat elemezzük, hanem megvizsgáljuk, hogy a kártevők másolása esetén mi történik a forrással és mi lesz a "célban". Természetesen itt olyan beállításokat kell alkalmaznunk, ami arra utasítja a védelmet, hogy kártevő esetén valamilyen módon gátolja meg a másolást. Tekintettel arra, hogy a védelmi rendszerek jelentős része ma már proaktív védelemmel is rendelkezik, az olyan kártevők esetén, amiket a másolás során nem azonosított a védelem, szükség van a proaktív védelem vizsgálatára. Ehhez a kártevőket el kell indítani a tesztkörnyezetben, és vizsgálni a védelmet, hogy meggátolja-e a kártevő működését. Ehhez azonban minden egyes kártevő vizsgálatát megelőzően kártevőmentes környezetre van szükség. A fenti módszerek kiegészítéseként vizsgálhatjuk a védelmeket abból a szempontból, hogy a kártékony kód rendszerbe kerülését mennyire tudják megakadályozni. Például egy weboldalról történő letöltéssel vagy egy e-mailben történő érkezéskor. A fentiek alapján a hoszt-alapú védelmi rendszerek különböző védelmi szintjei akár külön-külön is vizsgálhatók.

Hálózatalapú védelmi rendszer esetén a fenyegetéseknek a végpontokra történő bejutása, illetve bonyolultabb támadás esetén a támadás egyes fázisai vizsgálhatóak. A vizsgálatnak célszerűen ki kell terjedni az észlelésre és a blokkolásra is.

Fenyegetések ismeretének a vizsgálata (eltávolítás – helyreállítás): Hoszt-alapú védelmi rendszer esetén a kártevők kódjának eltávolításának, az eredeti kártevőmentes állapot visszaállításának vizsgálata mind az on-demand, mind az on-access, illetve a proaktív esetben is elvégezhető. Mindegyik esetben arra terjed ki a vizsgálat, hogy a védelem működése előtti állapot hogyan változott meg. A változásokat természetesen a védelmi rendszer naplóállományával is összevetjük.

Téves (false positive) riasztások vizsgálata: Hoszt-alapú védelmi rendszereken az on-demand ellenőrzések esetén a védelmi rendszerek által generált naplóállományok vizsgálata alapján, on-access esetben pedig a naplóállományok, illetve a forrás és a célterületek alapján történik azon fertőzésmentes állományok körének a meghatározása, amelyek esetén a védelem – tévesen – kártevőt jelez. Hálózatalapú védelmi rendszer esetén az ilyen típusú vizsgálatot a szokásos hálózati forgalommal vizsgálhatjuk.

Sebesség ellenőrzése kártevőmentes környezetben: Hoszt-alapú védelmi rendszer esetén a sebesség ellenőrzését alapvetően kártevőmentes környezetben célszerű elvégezni. Kártevő(k) találata esetén ugyanis már másodlagossá válik a sebesség, a felhasználók számára sokkal fontosabb lesz a biztonságos helyreállítás. Másrészt pedig a sebesség nagyban függ a találat esetén elvégzendő akciótól (törlés, eltávolítás, karanténba helyezés, …). Hoszt-alapú védelmi rendszer esetén, kártevőmentes környezetben a sebességet nagyban befolyásolja a hardver- és szoftverkörnyezet is. On-demand esetben az indítás és befejezés közti időtartam jelenti az ellenőrzés idejét. On-access esetben ez egy kicsit bonyolultabb. Ekkor ugyanis, ha például másolással történik a vizsgálat, akkor – a korrekt összehasonlíthatóság érdekében –, az indítás és befejezés között eltelt időtartamot csökkentenünk kell a védelem nélküli rendszerben ugyanezen feladat elvégzéséhez szükséges időtartammal. Így kapjuk meg ugyanis tisztán a védelem időszükségletét. A sebességellenőrzések mindegyikét – a hiba mértékének csökkentése érdekében –- legalább 20-szor végezzük el. A vizsgálat eredményét ezen időtartamok statisztikai jellemzői adják (minimum, maximum, átlag, szórás).

Hálózatalapú védelmi rendszer esetén nyilvánvalóan a hálózati szolgáltatás paramétereinek a vizsgálatát célszerű elvégezni. Ez a védelemmel ellátott és védelem nélküli hálózat paramétereinek az összevetését jelenti.

A fentiekből is látható, hogy a védelmi rendszerek vizsgálata, kiválasztása nem egy egyszerű folyamat. Különösen, ha egy informatikai infrastruktúra több, különböző típusú védelmi rendszeréről legyen szó. Ez esetben előre nagyon nehéz megmondani, hogy a kiépített, több komponensből álló védelmi rendszer milyen védekezésre lesz képes, milyen biztonságot tud nyújtani.

4. ÖSSZEFOGLALÁS

Ebben a tananyagban a legelterjedtebb biztonsági technológiákat tekintettük át a legelterjedtebb támadási lehetőségek, fenyegetések fényében. Az itt leírt módszerek és megoldások mellett számos további fejlesztés igyekszik újabb és újabb technológiákat kidolgozni a védekezésre. A támadási formák, megoldások azonban folyamatosan változnak, óránként több ezer kártékony kódot tartalmazó weboldal tűnik fel az interneten, melyek ellen a védelmeknek folyamatosan lépést kell tartaniuk, ami nem egy egyszerű feladat.

FELHASZNÁLT IRODALOM

[1] SaveAs Kft.: Esettanulmány egy felfedezett poloska programról, 2002. https://doksi.hu/get.php?lid=395 (Letöltés ideje: 2019. 07. 10.)

[2] Mitnick, Kevin D. – Simon, William L.: A legendás hacker – A megtévesztés művészete. Perfact-Pro Kft., Budapest 2003.

[3] Trend Micro: The Custom Defense Against Targeted Attacks, A Trend Micro White Paper http://www.trendmicro.com/cloud-content/us/pdfs/business/white-papers/wp_custom-defense-against-targeted-attacks.pdf

[4]: Kyas, O.: Számítógépes hálózatok biztonságtechnikája, Kossuth Kiadó, Budapest, 2000

[5]: von Neumann, John: *The Theory of Self-reproducing Automata*, A. Burks, ed., Univ. of Illinois Press, Urbana, IL, 1966

[6]: Nmap Network Scanning, Chapter 8. Remote OS Detection, www.insecure.org/nmap/nmap-fingerprinting-article.html

[7]: Norton, P. – Stockmann, M.: A hálózati biztonság alapjairól, Kiskapu Kiadó, Budapest, 2000

[8]: Ször Péter: A vírusvédelem művészete, SZAK Kiadó, 2010

[9]: Tanenbaum, Andrew S.: Számítógép-hálózatok, NOVOTRADE Panem, Budapest, 1999

[10]: RFC 793, TRANSMISSION CONTROL PROTOCOL,
http://www.ietf.org/rfc/rfc793.txt

A Nemzeti Közszolgálati Egyetem kiadványa.

Kiadó:
Nemzeti Közszolgálati Egyetem;
Államtudományi és Közigazgatási Kar
www.uni-nke.hu

Felelős Kiadó:
Prof. Dr. Kis Norbert Dékán

Címe:
1083 Budapest, Üllői út 82.

Kiadói szerkesztő:
Vörösss Ferenc

Tördelőszerkesztő:
Friebert Máté

ISBN 978-963-498-097-1 (elektronikus)

Nemzeti Fejlesztési Ügynökség
www.ujszechenyiterv.gov.hu
06 40 638 638

MAGYARORSZÁG MEGÚJUL

A projekt az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósul meg.