1 **Adatvédelem, Adatbiztonság**

A DES utódjai, az AES, A kulcs disztribúció problémái, Kétkulcsos rendszerek: RSA, PGP, WLAN hálózatok biztonsága

2 **A DES Feltörése utáni idő**

- Masszív fejlesztési láz
- Több helyettesítő algoritmus is született:
  - RC5, később RC6
  - Blowfish
  - Triple DES
  - IDEA
  - FEAL
- Egyik sem váltotta fel szabvány szinten a DES-t
  -

3 **Pár szó a DES alternatíváiról**

- Nagyjából mind 64 bites blokkokban dolgozik
- Növelt kulcstér: 128 -> 2048 bites kulcs
- Elterjedés hiánya:
  - Nem megfelelő támogatottság (marketing)
  - 1-2 esetben részletes kriptoanalízis hiánya
  - AES szabvány kidolgozása
  - Némelyik a mai napig igen jónak bizonyult
  - Sok algoritmus továbbfejlesztett változata ma is használatos

4 **A DES leváltása, az AES megszületése**

- Gondok a DES algoritmussal
  - 56 bites kulcsok
  - Főként cél hardware-re lett tervezve
  - Szoftveresen jóval lassabb
  - Triple DES megoldotta a kulcstér problémát, de nem volt hozzá cél hardware
  - Szoftverből igencsak lassú volt 1997-1998 környékén.

5 **A DES leváltása, az AES megszületése**

- 1997 január 2: az Amerikai Szabványyhivatal (NIST) bejelenti, hogy a DES-t le fogják váltani AES néven.
- Alternatíva kidolgozása helyett verseny hirdetése 1997 szeptember 12.-én
- 9 hónap ált rendelkezésre a fejlesztőknek
- Számos algoritmus született

6 **A DES leváltása, az AES megszületése**

- Az algoritmusokkal szemben támasztott elvárások a következőek voltak:
  - 128 bites blokkméretben dolgozzon
  - 128, 192 és 256 bites kulcsméretet támogasson
- Ezen feltételeknek megfelelő algoritmusok igencsak ritkák voltak abban az időben.

7 **A nyertes kiválasztása**

- A beérkezett algoritmusokat egy szakértő csoport vizsgálta a következő szempontok

alapján:
- Biztonság
- Algoritmus gyorsasága különböző körülmények mellett:
  - Kevés memória
  - Lassú CPU
  - Eltérő architektúrák
  - Cél HW építése FPGA chip-ek segítségével

8 **A nyertes kiválasztása**
- A beküldött algoritmusok egy része a biztonság ponton vérzett el.
- Konferenciák a nyertes algoritmus kiválasztásáról:
  - AES1: 1998 Augusztus
  - AES2: 1999 Március
- Konferenciák eredményeképpen sikerült kiválasztani a döntős algoritmusokat

9 **A nyertes kiválasztása**
- Döntős algoritmusok:
  - MARS
  - RC6
  - Rijndael
  - Serpent
  - Twofish

10 **Döntős algoritmusok szavazatai**

11 **A szabvány létrejötte**
- 2000. október 2: A NIST bejelenti, hogy az AES szabvány algoritmusa a Rijndael lesz.
- 2001. november 26: Az AES szabvány elfogadásra került.
-

12 **Az algoritmus**
- 2 belga matematikus munkájának eredménye
- Joan Daemen és Vincent Rijmen
- Az algoritmus neve az ő nevük kombinálásából született meg.
-

13 **Az algoritmus hatékonysága**
- 2002-ben az NSA elfogadta első nyílt forráskódú algoritmusként amely alkalmas szigorúan titkos információk védelmére.
- Nagyon biztonságos.
- Elméletben lehetséges törni, de a mai számítási teljesítmény mellett lehetetlen.
- Elvileg 10-20 év múlva lesznek csak olyan gépek, amelyek belátható időn belül végeznek a feltöréssel.

14 **Elméleti támadások**
- 128 bites kulcs esetén:
  - 2^126,1 lehetséges kulcs
- 192 bites kulcs esetén:
  - 2^189,7 lehetséges kulcs

- 256 bites kulcs esetén:
  - 2^254,4 lehetséges kulcs.

15 **Ez alapján feltörés ideje**

- Egy 4 magos CPU-val, amely magonként 3 millió kulcsot próbál végig egy 128 bites kulcs esetén:
  - Nagyjából 2,4 x 10^23 év kellene a megtöréshez.
  - Ilyen CPU jelenleg nincs, mivel feltételeztük, hogy 1 órajel ciklus alatt 1 kulcsot kipróbál a rendszer.
  - Valóságban ennél 2,3x lassúbbak a processzorok
  - GPU sem elég gyors a célra

16 **Az AES működése**

- Nem hasonlítható a korábbi titkosítási algoritmusokhoz, mivel azok működése alapvetően soros volt.
- Az AES ezzel szemben mátrix transzformációk sorozata.
- 128 bites blokkokban dolgozik. Kulcsméret: 128 bit, 192 bit, 256 bit.
- Elvben a kulcsbitek száma a végtelenségig növelhető, a blokkméret viszont csak 256 bit-ig.

17 **Az algoritmus leírása**

1. Kulcs bővítés – körönként használt kulcsok előállítása Rijndael kulcs ütemezéssel
2. Első kör
   1. Körben használt kulcs XOR Adat
3. Körök
   1. Byte csere – minden byte cserélődik egy nem lineáris fv. alapján meghatározott másik bájtra.
   2. Sor eltolás körönkénti más értékkel
   3. Oszlopok cseréje kettesével úgy, hogy közben értékeik összeadódnak
   4. Következő kör kulcs XOR adat

18 **Az algoritmus leírása**

4. Utolsó kör (nincs oszlopcsere)
   1. Byte csere – minden byte cserélődik egy nem lineáris fv. alapján meghatározott másik bájtra.
   2. Sor eltolás körönkénti más értékkel.
   3. Következő kör kulcs XOR adat.

19 **Kulcs ütemezés**

- Az algoritmus „lelke”
- Mátrix műveletek sorozata
- Egyirányú folyamat, visszafelé nem működik.
- Eltérő a kimenet mérete kulcsmérettől függően:
  - 128 bit esetén: 11 x 128 bit mátrix
  - 192 bit esetén: 13 x 128 bit mátrix
  - 256 bit esetén: 15 x 128 bit mátrix
-

20 **Byte Csere lépés**

- S nemlineáris függvény x bájtot y bájttal helyettesít.
- S értékei optimalizálhatók egy 256 elemű gyorsító táblázatban

21 **Sor eltolás**

- Minden egyes sor elemét más értékkel tolja el.
- Első sor változatlan marad
- Második sor 1 byte eltolás
- Harmadik sor 2 byte eltolás
- Negyedik sor 3 byte eltolás

22 **Oszlop csere lépés**

1
- Egyirányú lineáris függvény
- Bemenete és kimenete is 4 bájt (1 oszlop)
- A kimeneti oszlop mátrix szorzás eredményeképpen áll elő.

23 **Kör kulcs hozzáadása**

- Egyszerű XOR művelet az adat és a kulcs mátrix között.
- 128 bit esetén 11,
  192 bit esetén 13,
  256 bit esetén 15 kör van, eltérő kulcsokkal.

24 **AZ AES Biztonsága**

- Többmagos CPU-k terjedésével a törési kísérletek új lendületre kaptak.
- Matematikusok készítettek olyan algoritmust, amely 8 925 512 év alatt fel tudná törni a 128 bites titkosítást.
- Ez jóval kevesebb, mint ami kellene a 128 biteshez, azonban így is elméleti csak a támadás.

25 **AZ AES Biztonsága**

- Mivel a kulcs komplexitása a végtelenig növelhető, így ha mondjuk feltörik a 128 bites titkosítást, akkor egyszerűen csak növeljük a kulcsméretet.
- A kulcsméret duplázása 128 bitről 256 bitre 3,4 * 10^38 lehetőséggel bővíti a lehetséges kulcsok számát.
- Az USA jelenleg 256 bites titkosítást használ a szigorúan titkos dokumentumokhoz.

26 **AZ AES, mint szabvány**

- Nyílt szabvány, amit bárki szabadon implementálhat.
- Implementációtól függően eltérő lehet a biztonsága az egyes csomagoknak.
- Az implementáláshoz a NIST (National Institute of Standards and Technology) publikált egy tesztcsomagot is, amellyel tesztelhető a különböző implementációk biztonsága.
- Éles körülmények között érdemes ezzel tesztelni használat előtt.
-

27 **Implementációk biztonsága**

- Mivel a publikált teszt nem teljes körűen átfogó, ezért az NIST bevizsgálást is kínál.
- Egy ilyen bevizsgálás akár 30.000$ vagy több is lehet.
- Ebből kifolyólag nem sok implementáció rendelkezik az NIST által elfogadott minősítéssel.

28 **Implementációk biztonsága**

- Miért fontos ez ?
- 2011: OpenBSD által fejlesztett implementáció körül kétségek merültek fel egy lehetséges hátsó kapu sebezhetőséggel kapcsolatban.
- A kivizsgálás még most is folyamatban van.
- A gond ezzel a BSD Licenc, ami miatt rengeteg termék használja ezen implementációt.

29 **A BSD Licenc gondjai**

- Nyílt forráskódú licenc.
- Lényegében azt írja le, hogy:
  - Az alkotó lemond a szoftver tulajdonjogáról és a közösségre ruházza át, akik azt csinálnak vele, amit akarnak.
  - Nincs megkötés, hogy ha módosítja valaki a programot, akkor publikálnia kell a módosított forrást.
  - Az eredeti alkotót sem kötelező megnevezni.

30 **AZ AES Elterjedése**

- Rengeteg programban és hardverben megtalálható.
- Pl:
  - ZIP tömörítés
  - WPA2 WLAN titkosítás
  - Teljes lemezt titkosító megoldások
  - Processzorok
  - Telefonok

31 **Az AES sebessége**

- Jól optimalizálható.
- Rengeteg műveletsor gyorsító táblázatból megoldható.
- Pentium Pro processzor esetén 1 byte titkosítása 18 órajel ciklust vesz igénybe. (120 MHz mellett ez 6,6 MB/s sebesség)
- Újabb processzorok HW szinten gyorsítottak AES titkosítással.

32 **AES a processzorokban**

- Új utasítás készlet: AES-NI
- Összes Intel Core i5 és i7 processzor tartalmazza.
- Összes AMD Buldozer CPU tartalmazza.
- 5x-6x gyorsabb titkosítást tesz lehetővé a tisztán szoftveres megoldásokhoz képest.
- Programnak támogatnia kell ezen utasításkészletet.

33 **Népszerű programok amik támogatják**

- TrueCrypt
- BitLocker
- WinZip
- WinRar
- 7zip

34 **Kétkulcsos rendszerek**

RSA

35 **Kétkulcsos rendszerek**

- A kétkulcsos rendszerek, más néven a nyílt kulcsú rendszerek.
- Ilyen rendszerek esetén az ember 2 kulccsal rendelkezik.
  - Titkos kulcs: titokban tartandó
  - Nyilvános kulcs: szabadon terjeszthető.
  - A kulcsok összefüggnek, azonban csak a nyilvános ismeretében nem határozható meg a titkos kulcs.

36 **Kétkulcsos rendszerek**

- Titkosításra a nyilvános kulcsot használjuk.
- A kész üzenet visszafejthetetlen a titkos kulcs ismerete nélkül.
- Ilyen rendszerek az RSA és a PGP

37 **Probléma a szimmetrikus kulcsú rendszerekkel**

- Hiába jó az algoritmus, ha a kulcs gyenge
- A kulcs ismeretében nem csak titkosítani tudok, hanem vissza is tudom azt fejteni.
- Emiatt a kulcs eljuttatása macerás 2 fél között anélkül, hogy egy 3. fél ne tudjon róla.
- Titkosítatlan csatornán további titkosítást, rejtést igényel: Pl szteganográfiai módszerekkel.

38 **KÉTkulcsos rendszerek**

- Elv: a titkosító kulcs és titkosítást feloldó kulcs nem azonos.
- Elnevezések:
  - Titkos kulcs: Ez a titkosítás feloldó kulcs
  - Publikus kulcs: Ez a titkosító kulcs
- A kulcsok összefüggenek.

39 **Az RSA megszületése**

- 1978-ban találta ki 3 kriptográfus:
  - Ron Rivest
  - Adi Shamir
  - Leonard Max Adleman
- A név a nevük kezdőbetűiből jön.
- Az algoritmus prímszámokat és a prím tényezős felbontás problémáját használja fel.

40 **Prím tényezős bontás problémája**

- Elv: Minden szám felbontható prímtényezők szorzatára.
- Gond: prímszámok megtalálása.
- A probléma halmozottan fent áll nagy számoknál.
- Konkrétan a legjobb algoritmus ideje egy N számra, amely b biten írható le:

41 **AZ RSA működése**

1. 2db egymástól távol álló prím választása: p és q
   1. p és q véletlenszerűen választott és nagyjából azonos bithosszúsággal lehet őket reprezentálni bináris formában
2. n = p * q kiszámítása

42 **AZ RSA működése**

3. $\varphi(n) = (p-1)(q-1)$ számítása. A φ Euler függvény, amely megadja, hogy egy N számhoz mennyi nála kisebb relatív prímszám tartozik. *a* és *b* relatív prím, ha az 1-en

és −1-en kívül nincs más közös osztójuk.

4. Véletlenszerű e szám választása, amelyre igaz: $1 < e < \varphi(n)$ és $\gcd(e, \varphi(n)) = 1$;
5. e lesz a publikus kulcs kitevője, publikus kulcs: n^e

43 **Az RSA Működése**

6. d meghatározása az alábbi formában:

➨

➨d nem más, mint a moduláris multiplikatív inverze a e mod $\varphi(n)$ számnak.

➨d szám lesz a privát kulcs kitevője. Privát kulcs: n^d

44 **Az RSA Működése**

➨Titkosítás adott n^e publikus kulcs segítségével:

➨m üzenet titkosításához az üzenetet számmá kell alakítani, méghozzá úgy, hogy $0 < m < n$. Erre egy előre egyeztetett visszafordítható sémát alkalmazunk un. helykitöltés

➨A titkos c üzenet ezután az alábbi módon áll elő:

45 **Az RSA működése**

➨Titkosításfeloldás ismert n^d titkos kulcs esetén:

46 **RSA Biztonsága**

➨Kulcstér: 1024 bit és 4096 bit között van tipikusan

➨Ennél kisebb és nagyobb kulcstér is használható

➨Kis kulcstér esetén törhető az algoritmus, mivel ekkor:

➨e és d értéke próbálkozással kiszámítható, mivel nincs keverés a kimenetben.

47 **RSA Biztonsága**

➨Nagy kulcstér esetén azonban biztonságos.

➨Biztonsági kockázat továbbá a helykitöltésünk jósága is. Ezért célszerű alkalmazni az úgynevezett PKCS1 rendszert, amely biztonságos.

➨További gond, ha a véletlenszám előállító algoritmusom hibás.

48 **Gondok a nem igazán véletlen számokból**

➨2008: Debian OpenSSL ügy

➨Valamelyik fejlesztő „véletlenül” kitörölte a véletlenszám generátor inicializálásáért felelős kódrészletet.

➨Ezáltal az összes ezen rendszeren generált RSA kulcs törhetővé vált.

49 **RSA a mindennapi életben**

➨Számos helyen alkalmazott:

➨SSL protokoll titkosítás

➨HTTPS

➨SSH

➨Számos kereskedelmi megoldás, amit az RSA Laboratories fejleszt. Köztük: HW titkosított USB kulcsok, védelmi chip-ek (Playstation 3), stb…

50 **A HTTPS működése vázlatosan**

➨Kliens: privát - és publikus kulcs.

➨Szerver: privát - és publikus kulcs.

➨Kliens felveszi a kapcsolatot a szerverrel, közli, hogy milyen titkosítást támogat.

➨Szerver erre elküldi az ő publikus kulcsát, majd a kliens is az ő publikus kulcsát.

- Kommunikáció ezután indul meg.

51 **A HTTPS működése vázlatosan**

- A kapcsolat akkor megbízható, ha a felhasználó valóban azzal kommunikál, akivel szeretne.
- Ezért kellene rendelkeznie a szervernek egy tanúsítvánnyal, ami az identitásának helyességét és a kapcsolat biztonságát garantálja.
- Ezt egy úgynevezett megbízható 3. félnek kellene kiállítania.

52 **A HTTPS működése vázlatosan**

- Sok szerver nem rendelkezik ilyennel
- Egy ilyen aláírás nem olcsó mulatság.
- 1 éves érvényességgel 500$ és 1500$ dollár között mozognak a tanúsítványárak szolgáltatótól függően.
- Újabb böngészők (Firefox 2.0 óta kb) figyelmeztetnek a nem megfelelően aláírt tanúsítványok esetén. NEM KELLENE FIGYELMEN KÍVÜL HAGYNI!

53 **Az RSA Elterjedése**

- Jogdíjas szabvány volt 1983 és 2000 között.
- A szabvány lejárta után igencsak kezdett terjedni a rendszer.
- Kezdetben csak üzenetek titkosítására használták.
- Mára inkább már digitális aláírásokban használt.
- Titkosításban helyette inkább a PGP használt

54 **Digitális aláírás RSA-val**

- Adott x publikus kulccsal, és adott y, aki küldeni akar neki üzenetet.
- Mivel x publikus kulcsa mindenki által ismert, ezért x nem tudhatja, hogy valóban y akar-e vele kapcsolatba lépni, vagy más, aki y-nak adja ki magát.
- Itt jön be a digitális aláírás.

55 **Digitális aláírás RSA-val**

- y ezért az üzenetből egy hash-t képez, ezután a tikos kulcsát (n^d) felhasználva a hash értéket kódolja (mintha a hash lenne a titkos üzenet), majd ezt az üzenethez csatolja
- X szintén hash számítást végez az üzeneten és y publikus kulcsát használva kódolja a hash-t. Ennek eredményeképpen megkapja az y által számított hash értéket, amit ellenőrizni tud a saját hash értékével.

56 **A PGP**

57 **PGP**

- Pretty Good Privacy szavakból jön a neve
- 1991-ben készítette el Phil Zimmermann
- Szabad szoftver
- Több technológia kombinálásából jött létre
- Az RSA-t is felhasználta
- Mára a legnépszerűbb e-mail titkosítási rendszer

58 **A PGP megszületése**

- 1980-as évek második felében a hidegháború erősödni látszott.

- Phil Zimmermann ebben az időben igen sok tüntetésre járt
- Egy atombomba ellenes csoport tagja is volt. + 1 pici üldözési mániája is volt.
- Ezért merült fel benne az ötlet, hogy létrehoz egy titkosítási rendszert, amit az NSA sem tud megtörni.

59 **A PGP Megszületése**

- Ebben az időben a DES volt a titkosítási etalon.
- A DES-el az 56 bites kulcstér mellett probléma volt az, hogy az NSA tervezte.
- Összeesküvéselméletek szerint van benne egy rejtett kapu, amivel az NSA ki tud bontani minden titkosított üzenetet a jelszó ismerete nélkül.

60 **A PGP Megszületése**

- 1991-ben született meg a program.
- Mivel nem volt internet kapcsolata, ezért egyik barátját kérte meg, hogy tegye fel az internetre.
- Rohamosan elkezdett terjedni a rendszer.
- Eredetileg fizetős programként akarta kiadni, de aztán meggondolta magát és ingyenes maradt.

61 **A PGP Megszületése**

- 2 hibát követett el a PGP létrehozásakor:
  - Nem kért licenszjogot az RSA használatára
  - Az amerikai törvények egy kalap alá veszik a kódoló/dekódoló programokat az atombombákkal, rakétákkal és ezek exportja fegyverkereskedelemnek számít.
    - Abban az időben a 40 bitnél erősebb titkosítási algoritmusokra vonatkozott ez a törvény.
    - A PGP már ekkor nem használt 128 bitnél kevesebbet ☺

62 **A PGP Megszületése**

- Következmények:
  - 1993: FBI letartóztatja fegyverkereskedelem vádjával
  - Az RSA is beperelte
  - Később az RSA visszavonta a pert, mivel ekkor már igencsak elterjedt volt a rendszer
  - 1996-ban felmentette az FBI is a fegyver kereskedelem vádja alól

63 **A PGP Megszületése**

- Folyamatos FBI piszkálás miatt, a forráskódot neten nem lehetett terjeszteni.
- Ezt úgy játszották ki, hogy a forráskódot könyv formájában árulták ☺
- A könyvek exportja már nem számított fegyverkereskedelemnek. Sőt, a könyv exportjának a tiltása az alkotmány szólásszabadság jogába ütközött volna ☺

64 **A PGP megszületése**

- Ezután apróbb hibák merültek fel az algoritmusban, így Zimmermann megnyitotta a rendszer forráskódját.
- RFC 4880
- 6.5.1-es változata óta van nemzetközi változat és amerikai. (a szoftverszabadalmak miatt)

65 **Szálak kuszálódása a PGP körül**

- A PGP-t megvették, majd létrejött a PGP Corporation

- OpenPGP Alliance megszületik
- Létrejön egy szervezet is, amit PGP international-nek neveznek

66 **A PGP működése**

- Több technológiát használ fel, folyamatosan fejlődik.
- Blokkvázlatban a működése:

67 **A PGP Működése**

- Minden lépésben több algoritmust támogat
- Emiatt az egymással PGP titkosítással kommunikáló feleknek meg kell egyezniük a használt PGP beállításokon.
- Mivel folyamatosan fejlődik, így az újabb PGP verzióval készített fájlok nem bonthatóak ki régebbi PGP változatokkal, még a jelszó ismeretében sem.

68 **A PGP alkalmazási területei**

- Teljes lemez titkosítás
- E-mail titkosítás
- E-mail aláírás

69 **E-mail titkosítás PGP-vel**

- Joggal merülhet fel a kérdés, hogy miért szükséges, mikor régóta van HTTPS és TLS?
- Az ok: SPAM levelek terjedése
- Magyarországon és nemzetközileg is bevett gyakorlat az, hogy az Internetszolgáltató cégeknek szűrniük kell a kéretlen leveleket.
- Ez szép és jó, azonban ez azt is magával vonja, hogy titkosítottan nem lehet levelet küldeni.

70 **E-mail titkosítás PGP-vel**

- Nem lehet, mivel egy TLS titkosított csatornán keresztül menő levélbe nem tud belenézni a SPAM szűrő.
- Ezért általában blokkolva van a titkosított e-mail küldési lehetőség.
- Itt jön be a PGP alkalmazása.

71 **A PGP Biztonsága**

- Talán a legbiztonságosabb rendszer
- Nincs ismert eset arról, hogy valaki jelszó hiányában fel tudta volna törni.
- Csak olyan algoritmusok vannak benne, amelyek önmagukban is biztonságosak lennének.
- Emiatt csak és kizárólag Brute Force módszerrel törhető fel.

72 **A PGP Biztonsága**

- Több bűnügyben bebizonyosodott, hogy az FBI sem tudja feltörni a PGP-vel titkosított üzeneteket.
- Egy 2006-os incidens kapcsán az Amerikai Vámügyi Hivatal megjegyezte, hogy majdhogynem lehetetlen feltörni a PGP titkosított fájlokat.

73 **A PGP Biztonsága**

- Az amerikai alkotmány és számos más alkotmány alapján egy ember sem vehető rá, arra, hogy kiadja a jelszavát.
- Ez ütközik az alkotmány azon feltételezésével, hogy mindenki ártatlan, míg be nem

bizonyítják bűnösségét.
- Érdemes használni.

74 **Az ACTA és a SOPA, valamint a jövő**
- Talán a jövőben mindenki rá lesz kényszerítve a titkosításra, köszönhetően ezen zseniális törvényeknek.
- Ezekről későbbi előadáson lesz szó.
- Azonban ha valósággá válnak gyakorlatban is, akkor hatalmas öngól lesz ez a kormányoknak.
- A viszonylag elhanyagolható titkosított forgalom helyett minden titkosított lesz.

75 **Használati leírások**
- Elsősorban a GPG program használatát érdemes átnézni.
- Ez egy GNU PGP implementáció
- Főként Unix és Linux rendszerekre van fejlesztve elsősorban
- Azonban van Windows változat is, de ennek használata macerásabb
- http://moser.cm.nctu.edu.tw/gpg.html

76 **WLAN hálózatok biztonsága**

77 **Biztonság tesztelése**
- Igencsak szürke övezet jogilag, mivel országonként mások a törvények.
- Emiatt interneten is korlátolt tartalom a témáról. Hozzáértő emberek nem igazán beszélnek arról, hogy mit mivel érnek el.
- Van egy univerzális eszköz azonban.

78 **Tesztelő eszköz**
- Kali (BackTrack) Linux.
- Majdnem univerzális tesztelő eszköz
-

79 **Kali Linux Linux**
- Első kiadás: 2006-ban (BackTrack néven)
- Jelenlegi legfrissebb változat: 2
- Nagyjából évente van nagyobb kiadás, közben karbantartási kiadások.
- Telepíteni nem kell, mivel Live CD.
- Használatához elég egy Pendrive is.
- Számos eszközt tartalmaz.

80 **BackTrack Linux**
- Főbb alkalmazási területei:
  - Offline támadások Windows rendszerek ellen: Registry és Jelszó szerkesztés
  - WLAN hálózatok támadása
  - N+1 alkalmazási terület még ezen kívül ☺

81 **WLAN hálózatok biztonsága**
- Előtte azonban néhány fontos dolog:
  - IEEE 802.11 alatt lett szabványosítva 1997-ben
  - A szabvány 1991-ig nyúlik vissza. Ekkor kezdődött meg a kifejlesztése.
  - OSI modell szerint: Adatkapcsolati és fizikai réteget definiál

- 2,4Ghz és 5Ghz vivő frekvenciák
-

82 **WLAN hálózatok biztonsága**

- Titkosítás
  - WEP
  - WPA
  - WPA2
- Hozzáférés korlátozás:
  - MAC cím alapján.

83 **WEP**

- Wired Equivalent Privacy
- Eredetileg 24 bites titkosító kulcs.
- Később fel lett bővítve 64 bitre
- Minden csomag XOR - olva van a kulcs segítségével lényegében
- Kellő adatmennyiség lehallgatása után a kulcs visszafejthető.
- 2003-ban hivatalosan is elavulttá vált.

84 **WPA**

- 1999-ben jelent meg legelőször
- WEP Utódja.
- Kezdetben nem nagyon terjedt el, mivel a titkosításhoz jóval több erőforrásra van szüksége.
- Emiatt a kezdetleges WLAN eszközökön nem használható.

85 **WPA újdonságai**

- 2 módja van:
  - Kezdeti kulcs megadása.
  - Radius szerver azonosítás
- Kulcskeverés:
  - Kiinduló kulcs alapján generál kulcsot, amivel a titkosítás történik.
- Csomag sorrend figyelés.
- 64 Bites hash a csomagra

86 **Miért biztonságos a WPA?**

- Csomag sorrend figyelés:
  - Ha egy csomag sorszáma alapján hamarább érkezik meg, mint az előtte lévő, akkor azonnal új titkosító kulcs lép életbe.
- Csomag hash figyelés:
  - 2db eltérő hash 60mp-en belül: új kulcs bevezetését eredményezi.
- Ezek szinte lehetetlenné teszik a kriptoanalízist.

87 **„Szinte lehetetlenné teszik"**

- Algoritmus hibájából fakadóan 12 bit / óra adat módosításra van lehetőség észrevétlenül ☺
- 2004-ben ezért jelent meg a WPA2, amely TKIP titkosítás helyett 128 bites AES titkosítást használ.

88 **WLAN hálózat megtörése**

89 **WLAN hálózat megtörése**

- Mikor lehetséges gyakorlatilag is:
  - Ha van a támadónak elég ideje
  - Ha van a támadott Acces Pointon kellő adatforgalom
  - Ha WEP vagy WPA/WPA2 PSK titkosítást használ a rendszer.
  - A támadó rendelkezik megfelelő eszközzel.

90 **Megfelelő eszköz**

- Laptop WLAN adapterrel
- Törő program
- WLAN adapterhez packet injection Driver
- WLAN adapternek támogatottnak kell lennie. Legtöbb eszköz támogatott, de érdemes utána nézni.
- A törő program és a driver adott BackTrack alatt.

91 **A törés menete**

- Mivel nem volt időm felkészülni a hatályos jogszabályokat illetőleg konkrét példa nem lesz bemutatva.
- Helyette azonban a menet el lesz mondva

Használat csak és kizárólag saját felelősségre

-

92 **A törés menete**

- Először is elérhető Acces Pointok megkeresése
- Ha több is van, akkor WEP vagy WPA1 titkosításúakkal érdemes kezdeni a próbálkozást.
- Ezután passzív lehallgatás (Aircap program) kezdése. Nagyjából 1-2GB adatot kellene letölteni, de a több jobb.
- Ezután törés
-

93 **A törés menete**

- Mítosz: SSID sugárzásának kikapcsolása esetén csak az tud csatlakozni, aki ismeri az SSID-t.
- Gond ezzel az, hogy rádió hullámokról beszélünk, ami lehallgatható.
- A rejtett hálózatok is felderíthetőek számos programmal.
- Pl:NetStumbler (Windows), InSSIDer (Windows, android)

94 **A törés menete**

- WEP esetén az algoritmus hibájából és a jelszó bonyolultságától függően (2-120mp) megszerezhető a jelszó.
- WPA és WPA2 esetén csak BruteForce próbálgatásokra van lehetőség.
- Ilyenkor érdemes kezdeni a törést általában használt jelszavakkal
- Ilyen listák a neten széles körben elérhetőek.

95 **A törés menete**

- Jelszó listák összeállítására számos program érhető el.
- Ha megvan a jelszó, akkor tudunk csatlakozni az eszközhöz, amennyiben nincs MAC

cím korlátozás az eszközön.

- MAC cím korlátozás esetén is passzív lehallgatással és a jelszó ismeretében rengeteg információhoz juthatunk.

96 **A törés menete**

- Jelszó lista generálása:
  - makepasswd --chars=[jelszó karakterek] --count=[jelszó darab] > [kimeneti fájl]
  - apg -m [minimum jelszó hossz] -x [maximum jelszó hossz] -n [jelszó darab] > [kimeneti fájl]
- Sok időt és tárhelyet is igénybe vehet.

97 **A törés menete**

- A tényleges törés az Aircrack programmal végezhető el, melynek paraméterezése a hálózat titkosítási módtól függően változó.
- WPA/WPA2 esetén 8 karakteres jelszó esetén, ami az angol abc karaktereit tartalmazza, 8^26-on lehetőség van. Ez konkrétan számban: 302 231 454 903 657 293 676 544 ☺

98 **A törés menete**

- 4 magos CPU esetén, ha magonként 1 millió póba/s teljesítményünk van, akkor csupán 2 395 924 141,486 év kell legrosszabb esetben a jelszó feltöréséhez. ☺
- De átlagban ennél jóval kevesebb kell, mivel sok esetben nem alkalmaznak bonyolult jelszavakat a védelemre.
- Sok esetben a hálózat SSID a jelszó.

99 **A törés menete**

- Sok esetben a jelszó a router alapbeállítása.
- Amennyiben ismerjük a router típusát, akkor ezt is érdemes kipróbálni.
- Továbbá érdemes a router konfigurációs felületének saját erős jelszót választani
- Mivel sok esetben WLAN-on is elérhető a konfigurációs felület.

100 **A törés menete**

- Konkrétabb információk és mit hogyan:
  - http://www.sans.org/reading_room/whitepapers/auditing/wifi-backtrack_2038

! Használat csak és kizárólag saját felelősségre !

101 **Köszönöm a figyelmet**

Kérdések ?