Jelszótárolás, lenyomatok, támadások, megelőzés, védekezés

Dr. Répás Sándor

Bemutatkozás

- Széchenyi István Egyetem Kiberbiztonsági és Hálózati Technológiák Kutatócsoport alapító tagja
- Tanácsadás információbiztonsági, kiberbiztonsági és számítógéphálózati területeken
- Egyetemi docens
  - Széchenyi István Egyetem, Távközlési Tanszék
- Informatika, Ph.D.
  - Széchenyi István Egyetem
- MSc védelmi vezetéstechnikai rendszertervező
  - Nemzeti Közszolgálati Egyetem
- Okleveles közgazdásztanár
  - Budapesti Műszaki és Gazdaságtudományi Egyetem
- Okleveles Villamosmérnök
  - Széchenyi István Egyetem
- Villamosmérnök
  - Óbudai Egyetem
- Közgazdász
  - Budapesti Corvinus Egyetem

- 1985 óta informatika, és első hackelések 😉
- ISO/IEC 27001:2022 LA
- Certified Ethical Hacker (CEH)
- EC-Council Certified Encryption Specialist v2 (ECES)
- Certified Information Systems Auditor (CISA)
- Certified Information Security Manager (CISM)
- Certified in Risk and Information Systems Control (CRISC)
- Certified Data Privacy Solutions Engineer (CDPSE)
- Symantec Certified Antivirus Engineer
- MikroTik Certified Consultant
- Certified Cisco Systems Instructor (CCSI)
- Microsoft Certified Trainer (MCT)

Témakörök

- Alapok
  - Lenyomatok
  - Jelszavak
- NIST jelszóajánlások
- Jellemző támadási módszerek
  - Bemutató
- Megelőzés
  - Jelszó házirendek
  - Jelszavak tárolása szervereken
- Audit

Lenyomatképzés (hash, digest)

- Hash függvények célja a bemeneti szövegre (vagy egyéb információra) jellemző kimenet létrehozása („ujjnyomat”)
- Szempontok:
  - Egyirányú (lenyomatból sosem állítható elő az algoritmus bemenete)
  - Nehéz legyen olyan szöveget előállítani, ami egy előre megadott ujjnyomatot (DIGEST) eredményez (születésnap paradoxon)
    - Könnyen lehetne szöveget hamisítani meglévő aláíráshoz
  - Viszonylag rövid legyen a generált lenyomat (és állandó hosszúságú)
- Rengeteg mindennek alapja:
  - Hitelesítés
    - Jelszavak, OTP-k (még a Google authenticator is 😉)
  - Digitális aláírás
  - Hibák detektálása
  - Összehasonlítás
  - …

Néhány lenyomatképző algoritmus

- NTLM
- MD5
- SHA1, SHA2, SHA3
- Gyors $\leftrightarrow$ lassú?
- Egyre bonyolultabbak és egyre hosszabb lenyomatokat képeznek

NTLM

- Windows NT (New Technology) LAN Manager (NTLM)
- 1993-ban jelent meg a Windows NT 3.1-ben
- LM hash utóda
- MD4-en alapul
- 128 bites lenyomat
  - Ez ma már túl rövid…
- Használata nem ajánlott, de régebbi Windows hálózatokban, vagy vegyes (Linux, Windows) környezetben előfordul
  - Pass the hash, és más támadások
  - Nem „sóz”
  - Nagyon gyors
- A Windwos 2000-től kezdődően Kerberos hitelesítés van a hálózaton, de az Aktív direktoriban (AD) tárolva vannak a felhasználói jelszavak NTLM hashei, tehát a tartományvezérlőről (DC) kinyerhetők

MD5

- Message Digest 5 (MD5)
- 1991 Ron Rivest
- RFC-1321
- Az MD4 javítása
- 128 bites lenyomat
  - Ez ma már túl rövid a születésnap paradoxonon alapuló támadásoknak
- Használata nem ajánlott, de számos esetben előfordul (Pl sok tanúsítványban is)
  - 1996-ban találtak benne egy gyengeséget
  - 2004-ben sikerült ezt kihasználni is
  - Nagyon gyors
  - Nem „sóz”

SHA-1

- Secure Hash Algorithm (SHA)-1
- FIPS PUB 180-1 (1995. április 17.)
- RFC-3174 (2001. szeptember)
- 160 bites lenyomatot képez
- NSA tervezte a DSA (Digital Signature Algorithm, digitális aláírás) részeként
- Az SHA továbbfejlesztése (FIPS 180, 1993. május 11.)
- 2005 óta nem tartják biztonságosnak
- 2017-ben (február 17.) publikáltak azonos lenyomattal rendelkező PDF állományt
- Alkalmazása nem ajánlott. Több rendszer nem fogadja el biztonságosnak.

SHA-2

- FIPS 180-2 (2002. augusztus 1.)
- RFC-6234 (2011. május)
- USA Szabadalom: US6829355B2
- NSA tervezte
- Lényegesen eltér az SHA-1-től, de más problémákkal rendelkezik
- SHA-224
  - Csak később jelent meg: 2004. február 25.
- SHA-256
- SHA-384
- SHA-512
- SHA-512/224, SHA-512/256
  - Csak később jelent meg: 2012. március
  - A 64 bites számítógépek megjelenésével az SHA-512 lényegesen gyorsabban elvégezhető, mint az SHA-256
  - Elvégzi az SHA-512 lenyomatképzést, majd csonkolja az eredményt

SHA-3

- FIPS 202 (2015 augusztus 5. NIST szabvány)
- Keccak algoritmuson alapul, mely megnyerte a nyilvános SHA-3 pályázatot.
  - SHA3-224
  - SHA3-256
  - SHA3-384
  - SHA3-512
  - SHAKE128 (SHAKE: **S**ecure **H**ash **A**lgorithm kombinációja a **KE**CCAK szóval)
  - SHAKE256
- NIST SP 800-185 plusz függvények, 2016. december 22.
  - cSHAKE (customizable SHAKE)
  - KMAC (KECCAK Message Authentication Code)
  - TupleHash (változó hosszúságú lenyomatképző, tuples sorokhoz)
  - ParallelHash (Nagyon hosszú stringek párhuzamosított lenyomatképzése)
- Példa a Wikipediáról (1 bit változás a bemenetben, 50%-os valószínűséggel okoz változást a kimenet minden bitje esetében):

```
SHAKE128("The quick brown fox jumps over the lazy dog", 256)
f4202e3c5852f9182a0430fd8144f0a74b95e7417ecae17db0f8cfeed0e3e66e
SHAKE128("The quick brown fox jumps over the lazy dof", 256)
853f4538be0db9621a6cea659a06c1107b1f83f02b13d18297bd39d7411cf10c
```

Sózás

- Salt
- Véletlenszerű bitek felhasználása a lenyomatképzés során
- Megnehezíti a szótár, és a szivárványtáblán alapuló támadások kivitelezését
- A só általában a hash mellett kerül tárolásra, de lehet külön is

„alma” néhány lenyomata

- NTLM
  feb86de6066741373962fbf12ab770c
- MD5
  ebbc3c26a34b609dc46f5c3378f96e08
- MD5 Crypt
  1bPqKuVne$viFz6XQCjB/r.IuFySdHD0
  De ez is (hiszen eltér a só):
  1tTCqeage$XS49j1jhfuW7UnQLP3a.t1
- SHA1
  5f5ea3800d9a62bc5a008759dbbece9cad5db58f
- Argon2id
  $argon2id$v=19$m=102400,t=2,p=8$SynlPOc8J2RsDYHQOufcew$cJyXRyAMRxl0WQg36TXU3w

Támadási módszerek

- Leginkább brute-force (kimerítő keresés), de:
  - Kiszivárgott jelszavak lenyomatának kiszámítása, és az eredmény összehasonlítása a megszerzett lenyomatokkal
  - Valamilyen szabályrendszer alapján szép sorban végigmenni a lehetséges jelszavakon, lenyomatokat kiszámítani, majd összehasonlítani
    - Például:
    - aaaaaa…zzzzzz
    - Rokonok, barátok, cég, stb neve listába gyűjtve
  - A két módszer kombinálása
- Szivárványtáblák használata
  - Csak régebbi, elavult lenyomatképző algoritmusok esetén

Szivárványtáblák

- Rainbow tables
- 1980 Martin Hellman írta le a kriptoanalízis idő-memória kompromisszumát
  - Sok tárhely felhasználásával nagy sebesség érhető el
- Ezek a módszerek előre kiszámított táblázatokat használnak a jelszólenyomatok (password hashes) feltörésére
- A táblázatokban nem az összes jelszó és lenyomat párosa van, csak olyan adatok, amik segítenek a hashből közelíteni
  - ~Hasonlít a veszteséges tömörítésre 😉
- Nem 100% a találati arány, de 9x% nagyságrend
- Minden hash algoritmushoz külön táblázatok: LM, NTLM, MD5, SHA1
- Ma már elavult módszer, az újabb lenyomatképző függvények, és a sózás miatt nem működik, de a GPU-k számítási kapacitása miatt is túlhaladott
- Ismert eszközök
  - https://ophcrack.sourceforge.io/
    - Windows, Linux, open source
  - https://l0phtcrack.gitlab.io/
    - Windows, open source
  - http://project-rainbowcrack.com

Szólisták

- Rengeteg lista kering az interneten
- Alapvető eszközök a jelszólenyomatok számolásához, próbálgatásához, jelszótöréshez
- Legismertebb a rockyou.txt
  - 2009 környékén feltörték a RockYou nevű cég szerverét, ahol kódolatlanul tárolták a felhasználók jelszavait
  - 14.344.392 darab lehetséges jelszó
  - 134 Mbyte
- A szakemberek, hackerek saját listákat gyártanak/bővítenek, az adott országhoz, feladathoz kapcsolódóan

Eszközök

- Hashcat
  - https://github.com/hashcat/hashcat
- John the ripper
  - https://www.openwall.com/john/
  - Több verzió, ingyenes és fizetős is
- Hashtopolis
  - https://github.com/hashtopolis
  - Clusterbe szervezhető több Hashcat futtató gép
- CPU (a számítógép processzora)
  - Elég lassú
- **GPU (a gépben lévő grafikus kártyák)**
  - **Gyors és univerzális**
- FPGA
  - Viszonylag gyors, de erősen korlátozott, és nagyon drága

Képek forrása: https://filecr.com/windows/hashcat/
https://medium.com/cryptography101/john-the-ripper-c78577cb34f

Hashcat

- https://hashcat.net/hashcat/
- 2015-ben lett nyílt forráskódú (MIT licensz)
- Linux, macOS, Windows
- CPU, Nvidia, AMD, Intel GPU
- 377 féle hash típus
- Támadási módok:

Név	Támadás	Kapcsoló
Dictionary	Szólista minden elemét végigpróbálgatja. Például: rockyou 🤓	-a 0
Combinator	Több szólistában lévő szavakat kapcsolja össze	-a 1
Mask (Brute-force)	A megadott helyen megadott karakterfajtákat próbálja végig	-a 3
Hybrdid	Dictionary+Mask, vagy fordítva	-a 6 és -a 7
Association	Felhasználó neve, állománynév, tipp, vagy bármi más, ami segíthet	-a 9
Rule-based	Különböző szabályokat kombinál a Dictionary, és Hybrid módszerekkel	
Toggle-case	Kisnagybetűk variálása, szabályokkal kiegészítve	

- DEMO 🤓

Kevin Mitnick Crack-In-The-Box

- Az egyik legismertebb hacker, Mitnick Securtiy Consulting
  - https://www.mitnicksecurity.com/
- 2023. április, a red team megbízásokhoz Crack-In-The-Box:
  - 24 darab Nvidia GeForce RTX 4090
  - 6 darab Nvidia GeForce RTX 2080
  - Folyadék hűtéssel
  - Hashtopolis cluster

Képek forrása: https://x.com/kevinmitnick/status/1649421434899275778

NIST jelszó (és hitelesítő) ajánlások

- NIST Special Publication 800-63-3 Digital Identity Guidelines
  - 800-63B Authentication and Lifecycle Management
    - 5.1 Memorized Secrets (password és PIN)
- 8 karakter minimális hossz, ha az ügyfél (ember) választhatja, és 6, ha gép (megfelelő véletlenszámgenerátorral)
- Legalább 64 karakter a maximális hosszt kell lehetővé tenniük
- Minden megjeleníthető ASCII karakter, beleértve a szóközt is (de ajánlott az UNICODE támogatása is)
- Nem csonkolható
- Jelszóerősség mutató, vagy instrukciók megjelenítése
- Legalább 10 próbálkozási lehetőség biztosítása a kitiltás előtt
- Gépelés közben rövid időre felfedési lehetőség biztosítása (pl. csillagok, pontok helyett)
- Csak (MitM támadás ellen) megfelelően titkosított és hitelesített csatornán továbbíthatóak a hitelesítő felé
- A jelszavak tárolása a hitelesítő oldalon, csak offline támadásoknak ellenálló módon ajánlott
  - **salt (legalább 32 bit), cost factor (minél nagyobb, amit a hitelesítő szerver elbír), majd hash képzésével**

- Ellenőrizni kell (5.1.1.2):
  - **Nem kompromittálódott (korábbi kiszivárgások)**
  - Nem elterjedten használt
  - Nem várható
  - Például:
    - szótári szavak
    - ismétlések, egymást követőek, „aaaaaa”, „1234abcd”
    - környezetből következő: szolgáltatás-, cég-, felhasználó név különböző alakjai)
  - Nem megfelelő jelszó esetén, jelezni a felhasználónak, és az indoklást is!
- Ajánlott:
  - Próbálkozások lassítása
  - Beillesztés lehetősége (paste)
  - Gépelés közben az aktuális karakter rövid idejű megjelenítése
- **Nincs:**
  - **Komplexitási szabály**
  - **Lejárat (maximális életkor), rendszeres változtatás kikényszerítése, viszont kompromittálódás esetén javasolt a jelszóváltoztatás**
  - **Emlékeztető**
  - **„Hint” (Például: Első háziállatod neve, Iskolád neve)**
- SMS 2FA-ként használata csak korlátozottan (RESTRICTED)

Forrás: https://pages.nist.gov/800-63-3/sp800-63b.html

Windows

- Jelszó házirend erősen korlátozott képességekkel
- Viszont a Microsoft megengedi a beépülést a jelszóváltoztatás folyamatába
  - passfilt.dll mondja meg, hogy egy jelszó elfogadható, vagy sem
  - Sok termék elérhető, ami ezt le tudja cserélni
  - Nagyon komplex házirendek készíthetők, akár HIBP is ellenőrizhető
  - Viszont nem árulja el, hogy miért nem enged át egy jelszómódosítást
    - **Elég idegesítő tud lenni!**

';--have i been pwned?

- HIBP
- https://haveibeenpwned.com/
- 2013. december 4.
- Troy Hunt
  - Ausztrál biztonsági szakember
  - MS MVP
- Kiszivárgott jelszavak gyűjteménye
  - Folyamatosan frissített
  - Bárki által elérhető
  - Ingyenes
  - A jelszavakhoz nem lehet hozzáférni (Csak lenyomat/HASH)
  - Automatikus értesítés lehetősége, ha bekerülünk az adatbázisba
- Tanulság: Sose tároljuk a szervereken magukat a jelszavakat! Csak biztonságos lenyomatokat!

Forrás: https://en.wikipedia.org/wiki/Have_I_Been_Pwned%3F

Legrosszabb jelszavak (HIBP)

Legsűrűbben használt, kiszivárgott jelszavak:

1. 123456
2. 123456789
3. qwerty
4. password
5. 111111
6. 12345678
7. abc123
8. password1
9. 1234567
10. 12345

Forrás: https://www.troyhunt.com/86-of-passwords-are-terrible-and-other-statistics/

Jelszótárolás szervereken

- Csak olyan módon tároljunk jelszavakat a szervereken, ami akkor is biztosítja védelmüket, ha a szervert feltörnék!
- Kizárólag lenyomatot és azt is sózva!
- A hagyományos lenyomatképző algoritmusok (NTLM, MD5, SHA…) túl gyorsak, ezért csak erre a célra ajánlott lenyomatképző függvénnyel:
  - Argon2id
    - 2015-ben nyerte meg a jelszó hashekre kiírt pályázatot.
    - Többféle védelemmel is rendelkezik a különböző támadási módszerek ellen.
  - scrypt
    - 2009 Colin Percival
    - RFC 7914
    - Régebbi rendszerekben használhatjuk.
    - Több crpypto pénz is alapul rajta:
      - Dogecoin
      - Litecoin
      - Einsteinum
      - Viacoin
      - …

Jelszótárolás szervereken

- bcrypt
  - 1999 Niels Provos, David Mazières
  - Blowfish alapján készült.
  - Használata akkor javasolt, ha Argon2id és scrypt sem áll rendelkezésre.
  - Megadható az input cost, ami azt határozza meg, hogy hány körben történjen a lenyomatképzés. (2 mely hatványa)
  - 72 bytenál hosszabb jelszavakat csonkolja!
- PBKDF2 (Password-Based Key Derivation Function 2)
  - RFC 2898
  - NIST ezt ajánlja, emellett van FIPS-140 tanúsított implementációja.
  - A HMAC-SHA-256 hash általában támogatott (de pl SHA1 is lehet), és NIST is ezt ajánlja.
  - Többszázezer iterációt hajt végre. Ezek növelésével nehezíthető meg a brute force támadás.
- Ha lehetőségünk van rá, a sózás mellett használjunk borsozást is!
  - Több lehetőség, de a peppert nem tároljuk a jelszavak mellett, és minden lenyomatnál egységes.
  - Például a lenyomatok helyett azok (és a pepper) HMAC-át tároljuk.
- Régebbi lenyomatok esetén nem áll rendelkezésünkre az eredeti jelszó. Ilyenkor annak lenyomatáról készíthetünk lenyomatot. (Vagy kötelező jelszóváltoztatás…) Például:
  - bcrypt(md5($password))
  - Az áttérést mielőbb el kell végezni!

Jelszó audit

- „A password audit is the process of checking the strength of passwords.”
- A jelszó házirendek általában nem tudnak minden szempontnak megfelelni
  - Ezért sok mindent csak adminisztratív kontrolokkal tudnak megoldani
  - Ennek betartását is lehet ellenőrizni
- A korábban beállított jelszavak lehet, még nem feleltek meg a ma elvárásainak
- Lehetőségek:
  - Ha van valamilyen jelszómenedzsment eszköz, az képes lehet a benne tárolt jelszavak ellenőrzésére
  - Bizonyos rendszerekhez léteznek jelszó ellenőrző eszközök
  - Ha nincs, akkor viszont csak a jelszó hashek összegyűjtése, és ellenőrzése a megoldás
- Etikai és adatvédelmi kérdésekre figyelni kell!

További eredményes fórumot! 😎

Dr. Répás Sándor

+36(20)971-2371

rsandor@ahol.co.hu

https://repas.hu