Információs rendszerek biztonságtechnikája

Kriptográfia

Kriptográfiai hash algoritmusok

Vakulya Gergely

Hash mint adatstruktóra

- A hash adatszerkezet lényege, hogy elemeket egy k db rekeszt tartalmazó tárolóban elhelyezzünk úgy, hogy azokhoz azonnal (keresés nélkül) hozzáférjünk.
- Ehhez minden elem értéke alapján egy bélyeget (hash-t) generáljunk. Ez adja a rekesz sorszámát.
- Célok:
  - A tárolni kívánt elemszámhoz minél kisebb (k szmú) rekesz kelljen.
  - Ne kerüljön egy rekeszbe egynél több elem (kerüljük el a **hash ütközést**).
- Számos módszer született (például osztásos, szorzásos, Fibonacci hasítás)
- Az adott célra használhatunk saját algoritmust is.
- A hash függvény k rekeszhez egy $\lceil log_2(k) \rceil$ bites számot rendel.
- Akár a tárolandó adat első byte-ja is használható hasz-ként.

A kriptográfiai hash algoritmusok

- A kriptográfiai hash algoritmus is egy azonosítót (hash-t, **üzenet pecsétet**) rendel a bejövő adathoz.
- Itt is fontos szempont, hogy **ne legyen hash ütközés**.
- A hash-nek **egyirányúnak** kell lennie, azaz belőle ne lehessen következtetni az bemenő adatra, vagy annak egy részére.
- Nem cél, hogy egy b bites hash esetén a virtuálisan létrejövő $k = 2^b$ db rekeszt minél jobban kihasználjuk.
- A gyakorlatban előforduló felhasználási módok:
  - Üzenetek (fájlok) **integritásának** ellenőrzése.
  - A **digitális aláírás** egyik fontos eszköze.
  - **Jelszavak** tárolása
- Fontosabb kriptográfiai hash algoritmusok: MD-5, SHA-1, SHA-2. Láthatáron az SHA-3.

Az MD-5 hash algoritmus

- MD-5: Message Digest 5.
- Ronald Rivest, 1992 (ő az a Rivest, aki az RSA-ból az R betű).
- 128 bites üzenet pecsétet állít elő.
- Sokáig ez volt de facto standard, de ma már elavultnak számít.
- A 2004-es kiadású Számítógép hálózatok könyvben még biztonságosnak említik.
- Azóta már gyorsan generálhatóak szándékosan ütköző file-ok.
- https://github.com/corkami/collisions
- Linux parancs: `md5sum`

Az SHA-1 hash algoritmus

- Secure Hash Algorithm 1.
- 1995-ben publikálták.
- Az AES-hez hasonlóan ez is amerikai kormányzati megbízásból készült.
- Belső felépítése hasonló az MD-5-éhez, de 160 bites üzenet pecsétet állít elő.
- Biztonságosabb, mint az MD-5, de megfelelő anyagi ráfordítással sebezhető.
- Gyengesége miatt egyre kevésbé támogatott.
- Linux parancs: `sha1sum`

Az SHA-2 hash algoritmuscsalád

- Secure Hash Algorithm 2.
- Az NSA publikálta 2001-ben.
- Több különböző méretű hash-t is elő tud állítani
  - 256 és 512 bites fő működési módok (SHA-256 és SHA-512).
  - A 256 bites kimenet 224 bitre csonkolható kis mértékű belső változtatással (SHA-224).
  - Az 512 bites kimenet 224, 256 és 384 bitre csonkolható kis mértékű belső változtatással (SHA-512/224, SHA-512/256 és SHA-384)
- Linux parancsok: `sha224sum`, `sha256sum`, `sha384sum`, `sha512sum`, `shasum`

Kriptoanalízis

A lehetséges támadások típusai

- Egy adott hash-hez tartozó tartalom (fájl) előállítása: Közel lehetetlen.
- Egy adott fájlhoz tartozóval megegyező hash-sel rendelkező másik fájl előállítása: Lényegében azonos az előzővel, így közel lehetetlen.
- Két kölönböző (tetszőleges) fájl előállítása, amik ugyanazt a hash-t adják: lehetséges (születésnap támadás, elég lassú)
- Két megadott fájlkezdethez olyan kiegészítések előállítása, hogy a kiegészített fájlok ugyanazt a hash-t adják: lehetséges (prefix/append támadások, néhány napos számítási idő)
- Két különböző, de megadott megjelenést adó fájl előállítása, amik ugyanazt a hash-t adják: lehetséges (másodperces nagyságrendű számítási idő)

A születésnap támadás (birthday attack)

Születésnap paradoxon

- Meglepően kis létszámú (23 fős) csoportban már legalább 50% a valószínűsége annak, hogy két ember ugyanazon a napon ünnepli a születésnapját.
- A párok száma: $\frac{23 \cdot 22}{2} = 253$
- A lehetséges napok száma: 365.
- Minden esetben $\frac{1}{365}$ a valószínűsége annak, hogy azonos napon születtek.

Következmény a hash-ekre nézve

- Egy n bites hash algoritmust alapul véve $2^{\frac{n}{2}+1}$ darab fájl esetén valószínűleg lesz a halmazban ütközés.
- Hasonlóan egy $2^{\frac{k}{2}}$ elemszámú és egy másik, szintén $2^{\frac{k}{2}}$ elemszámú halmaz elemei között valószínűleg találunk azonos hash-t adó párt.

A születésnap támadás kihasználása

- Tegyük fel, hogy egy üzleti partnerrel szeretnénk szerződést kötni úgy, hogy mi lényegesen jobban járjunk, mint partnerünk.
- Elkészítünk egy „fair” szerződésből $2^{\frac{2}{2}}$ kinézetre azonos, de bit szinten különböző példányt, illetve egy „kizsákmányoló” verzióból is ugyanennyit.
- Keresünk egy olyan párt, a „fair” és a „kizsákmányoló” verzió hash-e azonos.
- Elküldjük a „fair” verziót, valamint annak hash-ét partnerünknek.
- Elküldjük a „kizsákmányoló” verziót annak (megegyező) hash-ével az ügyvédünknek.
- Ennek jelentőségét majd a digitális aláírásoknál fogjuk megérteni.

Jelszavak tárolása hash formájában

- A hash algoritmusok jelszavak kezelésére is alkalmasak
- A különböző rendszerek (webalkalmazások, operációs rendszerek) nem magát a jelszót tárolják, hanem annak hash-ét.
- Valójában nem magát a hash-t tárolják, hanem a jelszót először kiegészítik egy véletlenszerű szöveggel és ennek a hash-ét tárolják (sózott jelszó).
- Sőt leggyakrabban több körös sózás – hash-elés kombinációt alkalmaznak (PBKDF2, Password Based Key Derivation Function 2)

Jelszótárolás kriptoanalízise, szivárványtábla

- A gyakori / egyszerű jelszavak hash-ei előre legenerálhatóak (szivárványtábla, rainbow table)
- A rendszerünkből kiszivárgott hash-eket gyorsan rá lehet próbálni a szivárványtáblában szereplőkre.
- Sózott jelszó esetén minden kipróbált jelszóhoz generálni kell a hash-t (lassabb).
- Többszáz körös sózás – hash-elés (PBKDF2) esetén többmillió potenciális jelszó hash-ének generálása igen lassúvá válik.