Szimmetrikus titkosítás és használata

Javaban

Fülöp Márk, 10.D

2016. május 17.

Tartalomjegyzék

Adatok elrejtése a kommunikáció során

- Adott két szovjet kém, akik távol élnek egymástól az USA-ban. Az egyikük Alice, a másikuk Bob.
  1. Alice egy fontos üzenetet szeretne elküldeni Bobnak

Adatok elrejtése a kommunikáció során

- Adott két szovjet kém, akik távol élnek egymástól az USA-ban. Az egyikük Alice, a másikuk Bob.
  1. Alice egy fontos üzenetet szeretne elküldeni Bobnak
  2. Mivel szovjet kémek, nagyon nem lenne jó, ha bárki is hozzáférne rajtuk kívül a küldött üzenethez, ezért valamilyen olyan megoldásra lenne szükségük, amely segítségével csak Ők férnek hozzá az eredeti adathoz

Adatok elrejtése a kommunikáció során

- Adott két szovjet kém, akik távol élnek egymástól az USA-ban. Az egyikük Alice, a másikuk Bob.
  1. Alice egy fontos üzenetet szeretne elküldeni Bobnak
  2. Mivel szovjet kémek, nagyon nem lenne jó, ha bárki is hozzáférne rajtuk kívül a küldött üzenethez, ezért valamilyen olyan megoldásra lenne szükségük, amely segítségével csak Ők férnek hozzá az eredeti adathoz
- Kérdés: Hogyan tudndánk kivitelezni, hogy az adat nyílt csatornán kézbesítése során senki illetéktelen ne tudjon hozzáférni az üzenet valódi tartalmához?

Adatok elrejtése a kommunikáció során

- Adott két szovjet kém, akik távol élnek egymástól az USA-ban. Az egyikük Alice, a másikuk Bob.
  1. Alice egy fontos üzenetet szeretne elküldeni Bobnak
  2. Mivel szovjet kémek, nagyon nem lenne jó, ha bárki is hozzáférne rajtuk kívül a küldött üzenethez, ezért valamilyen olyan megoldásra lenne szükségük, amely segítségével csak Ők férnek hozzá az eredeti adathoz
- Kérdés: Hogyan tudndánk kivitelezni, hogy az adat nyílt csatornán kézbesítése során senki illetéktelen ne tudjon hozzáférni az üzenet valódi tartalmához?
- Válasz: az adatok látszólagos elrejtésével $\Longrightarrow$ ehhez van szükség a titkosításra

Adatok elrejtése a kommunikáció során

- Adott két szovjet kém, akik távol élnek egymástól az USA-ban. Az egyikük Alice, a másikuk Bob.
  1. Alice egy fontos üzenetet szeretne elküldeni Bobnak
  2. Mivel szovjet kémek, nagyon nem lenne jó, ha bárki is hozzáférne rajtuk kívül a küldött üzenethez, ezért valamilyen olyan megoldásra lenne szükségük, amely segítségével csak Ők férnek hozzá az eredeti adathoz
- Kérdés: Hogyan tudndánk kivitelezni, hogy az adat nyílt csatornán kézbesítése során senki illetéktelen ne tudjon hozzáférni az üzenet valódi tartalmához?
- Válasz: az adatok látszólagos elrejtésével $\Longrightarrow$ ehhez van szükség a titkosításra
- Két fajtája van, amit a gyakorlatban is használnak
  1. szimmetrikus
  2. aszimetrikus

Szimmetrikus titkosítás

- P: az üzenet (plaintext)

Szimmetrikus titkosítás

- P: az üzenet (plaintext)
- K: titkos kulcs (secret key)

Szimmetrikus titkosítás

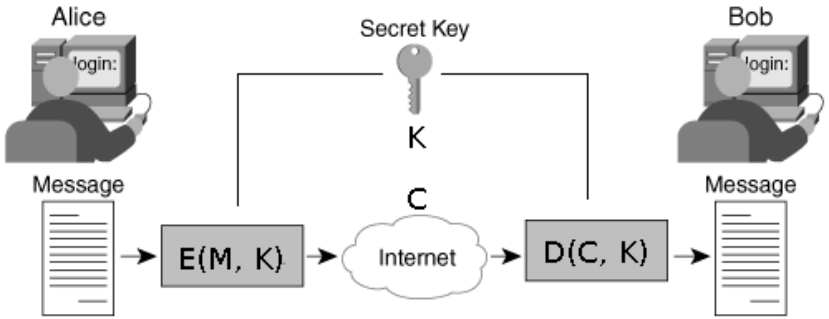
- P: az üzenet (plaintext)
- K: titkos kulcs (secret key)
- E(K, P): titkosító algoritmus (cipher)

Szimmetrikus titkosítás

- P: az üzenet (plaintext)
- K: titkos kulcs (secret key)
- E(K, P): titkosító algoritmus (cipher)
- C: titkosított szöveg (ciphertext)

Szimmetrikus titkosítás

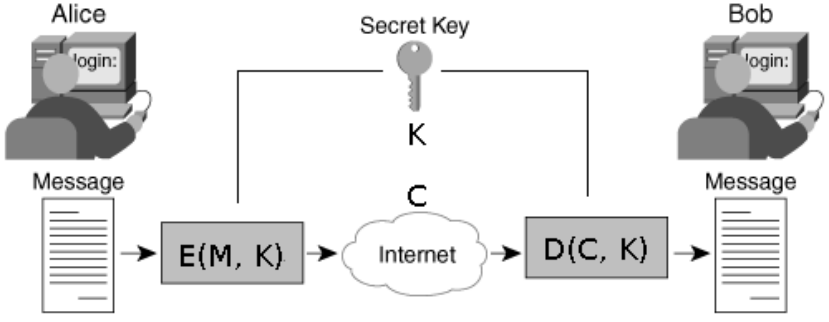
- P: az üzenet (plaintext)
- K: titkos kulcs (secret key)
- E(K, P): titkosító algoritmus (cipher)
- C: titkosított szöveg (ciphertext)
- D(K, C): visszafejtő algoritmus (decipher)



ábra: A szimmetrikus titkosítás működése kommunikáció során *(M=P)*

Szimmetrikus titkosítás

- P: az üzenet (plaintext)
- K: titkos kulcs (secret key)
- E(K, P): titkosító algoritmus (cipher)
- C: titkosított szöveg (ciphertext)
- D(K, C): visszafejtő algoritmus (decipher)



ábra: A szimmetrikus titkosítás működése kommunikáció során *(M=P)*

Definíció

Szimmetrikus titkosításról beszélünk, ha $E_K = D_K$

$$C = E(K, P)$$
$$P = D(K, C)$$

Szimmetrikus titkosítás

- E és D minden modern és biztonságos kriptográfiai rendszerben publikus, mindenki által ismert

[1] jelentős számítási teljesítmény befeketetése nélkül

Szimmetrikus titkosítás

- E és D minden modern és biztonságos kriptográfiai rendszerben publikus, mindenki által ismert
- Bob számára az egyedüli szükséges információ tehát: K, ennek segítségével P bármikor előállítható

[1] jelentős számítási teljesítmény befeketetése nélkül

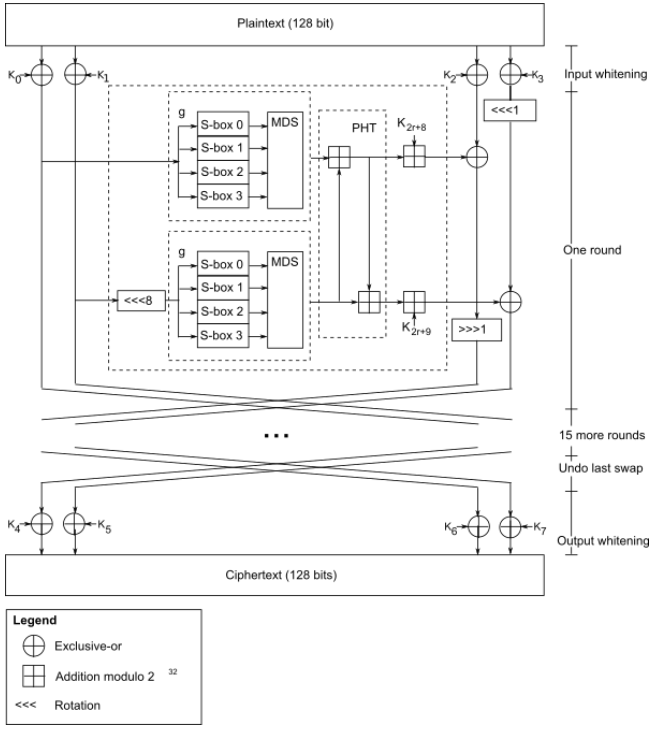
Szimmetrikus titkosítás

- E és D minden modern és biztonságos kriptográfiai rendszerben publikus, mindenki által ismert
- Bob számára az egyedüli szükséges információ tehát: K, ennek segítségével P bármikor előállítható
- Ha a használt algoritmus jó, akkor P nem nyerhető vissza[1], még akkor sem, ha K-n kívül minden adott (nyílván P nem).

[1] jelentős számítási teljesítmény befeketetése nélkül

Blokktitkosítók (block ciphers)

- Nagyon sok létezik belőlük, és NAGYON bonyolultak, pl. kép (Twofish)
- Első publikus: 1975, DES
- Példák blokktitkosító algoritmusokra:
  - a DES és a 3DES ('75, '95 egyik sem biztonságos)
  - az AES (jelenlegi ipari szabvány, ezzel működik a program, nagyon biztonságos)



Példa

1. Alice és Bob megegyezik a paraméterekben, és biztonságos csatornán egyeztetik az algoritmust, és a titkos kulcsot
   - Legyen $M :=$ "Hello Bob!", $K :=$"kortefa", $E :=$AES, így tehát $C :=$AES("Hello Bob!", "kortefa")

Példa

1. Alice és Bob megegyezik a paraméterekben, és biztonságos csatornán egyeztetik az algoritmust, és a titkos kulcsot
   - Legyen $M :=$ "Hello Bob!", $K :=$"kortefa", $E :=$AES, így tehát $C :=$AES("Hello Bob!", "kortefa")
2. Alice elvégzi a titkosítást, ezzel megkapja a titkosított szöveget, $C :=$ ”$97272b719354b8857e4d070da16d22b2$” (hexadecimális)
3. C szabadon továbbítható bármilyen nem biztonságos csatornán

Példa

1. Alice és Bob megegyezik a paraméterekben, és biztonságos csatornán egyeztetik az algoritmust, és a titkos kulcsot
   - Legyen $M :=$ "Hello Bob!", $K :=$"kortefa", $E :=$AES, így tehát $C :=$AES("Hello Bob!", "kortefa")
2. Alice elvégzi a titkosítást, ezzel megkapja a titkosított szöveget, $C :=$ ”$97272b719354b8857e4d070da16d22b2$” (hexadecimális)
3. C szabadon továbbítható bármilyen nem biztonságos csatornán

A Java lehetőségei

- A Java API mint minden máshoz, természetesen a titkosításhoz is nagyon sok segítséget nyújt.
- A `javax.crypto.*` csomag tartalmazza a legtöbb kriptográfiával kapcsolatos dolgot,
- köztük a `javax.crypto.Cipher` osztályt, amely a szimmetrikus titkosítás megvalósításának alapja.

A program felépítése

- A program, ahogy a neve is utal rá, arra való hogy fájlokat titkosítson
- A használt titkosítási eljárás biztonságos, feltörése min. 10^{18}-on évbe telne[3]
  - a használt algoritmus: AES, CBC móddal, 128 bites blokkmérettel
- A program 4+1 csomagból, és 6 osztályból áll.

```
.
|
+filetitok
    |
    +FileTitok.java
    +Constants.java
    |
    +crypto
      |
      +Cryptography.java
    +gui
      |
      +Window.java
    +io
      |
      +FileIO.java
    +misc
      |
      +Util.java
```

[3] Megfelelő bonyolultságú jelszót és megfelelő key derivation funkciót használva a titkosításhoz, illetve egy 2010 körüli szuperszámítógépet használva a feltöréshez

Osztályok feladatainak rövid összefoglalása

Az (f) jelzés azt jelzi, hogy a komponens a frontend munkájában vesz rész, a (b) pedig azt, hogy a backendében.

(f) `filetitok.FileTitok` általános, `main()` metódust tartalmazó osztály

(f) `filetitok.Constants` gyakran használt szöveg-, és számkonstansokat tartalmazó osztály

(f) `filetitok.misc.Util` gyakran végzett, de különböző funkciók

(f) `filetitok.gui.Window` a GUI-ért felelős osztály, továbbá végez alapvető ellenőrzéseket, mielőtt átkerülne a vezérlés a program backend részéhez

(b) `filetitok.io.FileIO` fájlok I/O műveleteit vezérli, az adatokat és az utasításokat a `Window` osztálytól kapja, a `Cryptography` osztállyal kommunikál

(b) `filetitok.crypto.Cryptography` a program lelke, mindenféle kriptográfiai műveletet (titkosítás és visszafejtés) végez