

Data Security: Protocols

Integrity

Az üzenethitelesítés (integritásvédelem) feladata az, hogy a vételi oldalon detektálhatóvá tegyük azon eseményeket, amelyek során az átviteli úton az üzenet valamilyen módosulást szenvedett el.

Módszerek:

- kriptográfiai ellenőrző összeg (CBC-MAC)
- kulcsolt hash
- rejtjelezés
- digitális aláírás
- spec.

Data Security: Protocols

Integrity

Külföldön dolgozunk, s alkalmanként szeretnénk rejtetten párbeszédet folytatni otthoni barátunkkal. Tilos azonban rejtjelezni a határon átlépő üzeneteket. Hitelesítő protokollok használata viszont nem tiltott (amikor is egy nyílt szöveg nyílt marad). Van megoldás?

A párbeszéd egy részlete, ahol Feri és Sanyi “beszélgetnek”:

A: Halló, Feri vagyok.

A: *Halló, itt Jóska.*

B: *Szia, Laci.*

B: Szia, itt Sanyi.

A: *Jövő hétfőn továbbmegyek Rómába.*

A: Holnap indulok Athénba.

B: Menjek én is?

B: *Sajnos én nem tudok menni?*

A: *Mindenképp gyere.*

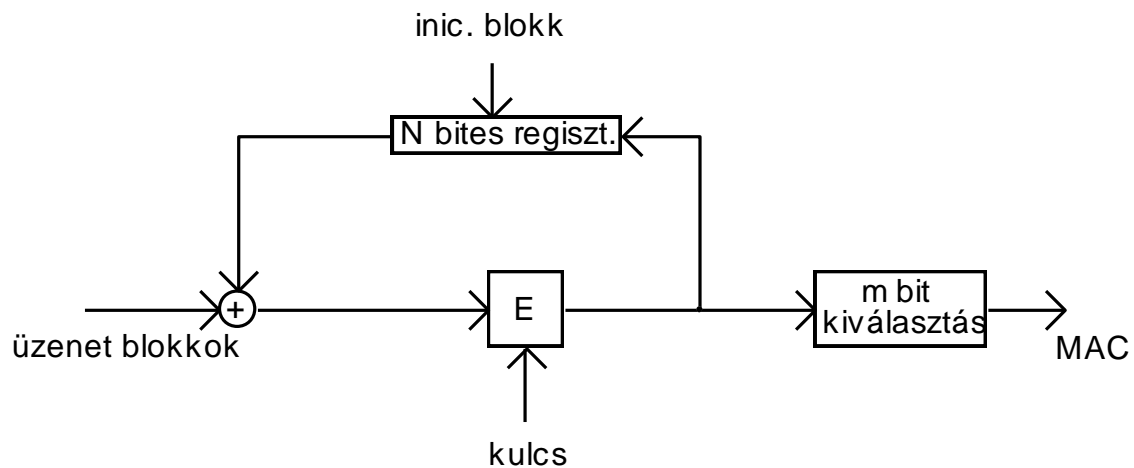
A: Semmiképp ne gyere.

Data Security: Protocols

Integrity - MAC

CBC-MAC (Message Authentication Code)

$[x_1, \dots, x_n, \text{MAC}(x_1, \dots, x_n)]$



$$y_i = E_k(x_i \oplus y_{i-1}), \quad y_0 = \text{inic.}, \quad i=1, \dots, n$$

$$\text{MAC} = g(y_n)$$

Data Security: Protocols

Integrity-MAC

CBC-MAC támadása: adaptívan választott üzenetek technikája

Tf. $IV=0$

1.)

$m_1=m$, $MAC_k(m)$

$m_2=MAC_k(m)$, $MAC_k(MAC_k(m)) = E_k(MAC_k(m)+IV) = E_k(MAC_k(m))$

$m_3=[m,0]$, $MAC_k([m,0]) = E_k(0+MAC_k(m)) = E_k(MAC_k(m)) = MAC_k(MAC_k(m))$

2.)

m_1 , $MAC_k(m_1)=z_1$

m_2 , $MAC_k(m_2)=z_2$

$m_3=m_1|z_1+z_2+y$, $MAC_k(m_3)=z_3$, y tetszőleges bitsorozat

$m_4= z_2+y$, $MAC_k(m_4)= E_k(z_2+y)=z_3$

Data Security: Protocols

Integrity-MAC

Kulcsolt hash

titok prefix: $MAC_k(m) = h(k|m)$

$$\begin{aligned} MAC_k(m | m') &= h(k | m | m') = h((k | m) | m') \\ &= MAC_{h(k|m)}(m') \end{aligned}$$

titok suffix: $MAC_k(m) = h(m|k)$

$$h(m)=h(m') \rightarrow h(m | k)=h(m' | k)$$

(születésnapi paradoxon)

szendvics módszer: $MAC_{k,k'}(m) = h(k | m | k')$

Data Security: Protocols

Integrity-HMAC

HMAC:

$H(\text{key1} \parallel H(\text{key2} \parallel \text{message}))$

Pseudocode of HMAC

function hmac (key, message)

if (length(key) > blocksize) **then** key = hash(key) *// keys longer than blocksize are shortened*

else if (length(key) < blocksize) **then** key = key \parallel zeroes(blocksize - length(key))
// keys shorter than blocksize are zero-padded

end if

opad = $[0x5c * \text{blocksize}] \oplus \text{key}$ *// Where blocksize is that of the underlying hash function*

ipad = $[0x36 * \text{blocksize}] \oplus \text{key}$ *// Where \oplus is XOR*

return hash(opad \parallel hash(ipad \parallel message))

end function

Data Security: Protocols

Integrity

Tekintsük az

$m|E_k(\text{MDC}(m))$

integritásvédő kódolást. Milyen tulajdonsággal kell ez esetben az MDC-nek rendelkeznie?

Ütközés-ellenállónak kell lennie:

Legyen $(m; m')$ MDC-ütközésre vezető üzenetpárt, m "nem gyanús" üzenetre kérve a kódolást, $E_k(\text{MDC}(m)) = E_k(\text{MDC}(m'))$ egyenlőség miatt m' üzenetre is ismert lesz a MAC.

Data Security: **Protocols**

Key exchange

Kriptográfiát alkalmazó rendszer biztonsága nem haladhatja meg a kulcsgondozása biztonságát.

Kulcsgondozási alapfeladatok:

- **kulcsgenerálás**
- **kulcstárolás**
- **kulcscsere (szállítás, megegyezés)**
- **kulcsfrissítés**
- **kulcsvisszavonás**

Data Security: Protocols

Key exchange

Egy kriptorendszer kulcshossza 100 bit. A kulcsot olyan generátorból nyerjük, amely 5 bites blokkokat állít elő. Ezen blokkokból azonos valószínűséggel olyanokat generál, amelyekben az 1 bitek darabszáma mindig kevesebb, mint a 0 bitek darabszáma. Az egymás utáni blokkok függetlenek.

a.) 20 db blokkot használunk kulcsként. Mennyi a tényleges kulcshossz, azaz mennyi az így generált kulcs entrópiája?

b.) Lehetséges-e, hogy 100 bit entrópiájú kulcsot előállítani az adott generátorra támaszkodva?

a.)

blokkfajta: db

3db 1bit 2db 0 bit → 10

4db 1bit 1db 0 bit → 5

5db 1bit 0db 0 bit → 1

Összesen: 16 –féle blokk → $\log_2 16 = 4$ bit /blokk → $100/5 \cdot 4 = 80$ bit

b.) A 16 lehetséges blokkot egyértelműen leképezzük a 0000, ..., 1111 16 db félbájt egyikébe. 25 db blokkot ilyen módon leképezve 100 db pénzfeldobás bitet kapunk.

Data Security: Protocols

Key exchange

A kulcscsere protokollok szolgáltatásai (A fél számára):

Implicit kulcshitelesítés

a protokoll sikeres lefutása után A meg lehet győződve arról, hogy rajta kívül csak a feltételezett másik fél, mondjuk B, és esetleg egy megbízható harmadik fél (pl. a kulcsszerver) férhet hozzá a protokoll során létrehozott kapcsolatkulcshoz

Kulcskonfirmáció

a protokoll sikeres futása után A meggyőződhet arról, hogy a másik résztvevő (B), valóban birtokában van a protokoll futása során létrehozott kapcsolatkulcsnak

Explicit kulcshitelesítés

a protokoll egyszerre biztosítja az implicit kulcshitelesítést és a kulcskonfirmációt

Kulcsfrissesség

a protokoll sikeres futása után A meg van győződve arról, hogy a létrehozott kapcsolatkulcs új, és nem egy korábban már használt (esetleg megfejtett) kulcs

Partnerhitelesítés

a protokoll sikeres futtatása után A meg van arról győződve, hogy a feltételezett másik résztvevő valóban részt vett a protokoll végrehajtásában.

Data Security: Protocols

Key exchange

Kulcscsere protokoll jellemzők:

- Szolgáltatásnyújtás iránya
- Megbízható harmadik fél használata
- Előzetesen szétosztott információk
- Hatékonyság

Data Security: Protocols

Key exchange

Kulcscsere protokollok támadása

Támadás típusok

- passzív lehallgatás
- protokoll üzenetei törlése, módosítása, visszajátszása (replay)
- támadó a protokoll egyik résztvevője (megszemélyesítés, parallel sessions)

A támadó célja

(tipikusan) a protokoll által létrehozott kapcsolatkulcs megszerzése, vagy annak elhitése egy becsületes *A* résztvevővel, hogy *A* a kapcsolatkulcsot egy másik becsületes *B* résztvevővel hozta létre, miközben valójában a támadóval osztja meg azt.

A támadó rendelkezésére álló információk

a támadó rendelkezésére álló információk tekintetében általában azt feltételezzük, hogy a nyilvánosan elérhető információkon kívül a támadó nem rendelkezik további információkkal.

Data Security: Protocols

Key exchange

Tekintsük a Wide Mouth Frog protokoll következő (javított) változatát

$$A \rightarrow S : A, \{“0”, B, K, T_A\}K_{as}$$
$$S \rightarrow B : \{“1”, A, K, T_S\}K_{bs}$$

Itt a “0” és az “1” irányjelző bitek, ahol a “0” jelöli a szervernek küldött üzeneteket és “1” jelöli a szerver által küldött üzeneteket.

- Melyik osztályba tartozik a protokoll?
- Melyik szolgáltatások nyújtja a protokoll A számára?
- Melyik szolgáltatások nyújtja a protokoll B számára?

- kulcs-szállító protokoll
- implicit kulcshitelesítés
kulcsfrissesség
- implicit kulcshitelesítés
explicit kulcshitelesítés
kulcsfrissesség
partnerhitelesítés

Data Security: Protocols

Key exchange

Tekintsük a következő kulcscsere protokollt:

1. $A \rightarrow B : \{K\}K_b$
2. $B \rightarrow A : \{N\}K$
3. $A \rightarrow B : \{\text{sig}_A(N)\}K$

- K_b B nyilvános kulcsa,
- K egy A által generált friss kapcsolatkulcs,
- N egy B által generált friss véletlenszám
- $\text{sig}_A(N)$ jelöli A aláírását N-en

A aláírása ellenére sem lehet B biztos benne, hogy a K kulcsot rajta kívül csak A ismeri!

Rosszindulató X fél meg tudja személyesíteni A-t B felé:

- $A \rightarrow X : \{K\}K_x$
 $X \rightarrow B : \{K\}K_b$
 $B \rightarrow X : \{N\}K$
 $X \rightarrow A : \{N\}K$
 $A \rightarrow X : \{\text{sig}_A(N)\}K$
 $X \rightarrow B : \{\text{sig}_A(N)\}K$

Javítás

1. $A \rightarrow B : \{K\}K_b$
2. $B \rightarrow A : \{N\}K$
3. $A \rightarrow B : \{\text{sig}_A(B, K, N)\}K$

Data Security: Protocols

Key exchange/agreement

1. Verzió:

1. A→B: $ID_A, E_B(R)$

passzív támadó: -

aktív támadó: megszemélyesítés sikeres

2. Verzió:

1. A→B:	ID_A, k_A^p
2. B→A:	ID_B, k_B^p
3. A→B:	$E_B(R1)$
4. B→A:	$E_A(R2)$
5. A: , B:	$k=F(R1,R2)$

Data Security: Protocols

Key agreement

MIM (Man-In-the-Middle), “támadó közepen” támadás sikeres:

C az A és B közé áll:

1'. A→C:	ID_A, k_A^P
C→B:	ID_A, k_C^P
2'. B→C:	ID_B, k_B^P
C→A:	ID_B, k_C^P

3. Verzió (kulcstanusítvány alkalmazása)

1. A→B:	ID_A, k_A^P, C_A
2. B→A:	ID_B, k_B^P, C_B
3. A→B:	$E_B(R1)$
4. B→A:	$E_A(R2)$
5. A: , B:	$k=F(R1,R2)$

Data Security: Protocols

Key agreement

Diffie-Hellman kulcscsere (alapprotokoll)

diszkrét hatványozás (kommutatív egyirányú függvény)

$GF(q)$: egy 'nagy méretű' véges test

g : primitív elem

1. $A \rightarrow B$: g^{R1}
2. $B \rightarrow A$: g^{R2}
3. A: $(g^{R2})^{R1}$
B: $(g^{R1})^{R2}$

kommutativitás: $k=(g^{R2})^{R1}=(g^{R1})^{R2}$

- lehallgató C támadó nem képes az $R1$ illetve $R2$ véletlen elemeket megállapítani
- aktív C támadó képes "támadó a közepén" támadásra

Kulcshitelesség nincs biztosítva! → hitelesítés: pl. SSL (Secure Socket Layer) protokollban

Data Security: Protocols

Public key certificate

ISO X.509 tanúsítvány

Az X.509 tanúsítvány komponensei az alábbiak:

- .Verziószám
- .Sorozatszám
- .Algoritmus azonosító
- .Tanúsítvány kibocsátó (CA) azonosítója
- .Tanúsítvány érvényességi időtartam (Not Before Date, Not After Date)
- .A tanúsítvány tulajdonosának azonosítója
- .A tanúsítandó **publikus kulcs**
- .A **digitális aláírás** (a fenti adatokra)

Data Security: Protocols

Public key certificate

Verzió V3

Sorozatszám 52C8 2013 7C85 A7ED F217 CE82 C845 1673

Aláírási algoritmus md5 RSA

Kiállító Class 2 Public Primary CA

Érvényesség kezdete 1998. 05. 12. 2:00:00

Érvényesség vége 2004. 01. 07. 1:59:59

Tulajdonos Verisign Class 2 CA - Individual Subscriber

Nyilvános kulcs RSA(1024 bit)

```
3081 8902 8181 00B5 CB1A 545E 25B0 2C59 5F09 6BD0 DAD6 4A4B 119D 1A0A 3E7E 2FB7 655F
1763 15E5 2CD0 2000 0CF0 BA6B AA5E 49B1 6893 8325 AC24 5FA2 231C 694D B83B DB7D DA8F
C109 CFA5 583A B64B C4D4 DBD8 AE75 FA86 2299 2201 2860 A5DB D530 DF21 705E 4899 AD21
5491 D1DE 5FFB 3829 531B E27A 5358 C50D 5D13 07B3 50C4 064B 39F8 54AB B98B 6912 1302
0301 0001
```

Kiegészítő adatok:

CRL URL: <http://crl.verisign.com/pca2.1.1.crl>

Ujjlenyomat alg. sha1

Ujjlenyomat: 7B02 312B ACC5 9EC3 88FE AE12 FD27 7F6A 9FB4 FAC1

Data Security: Protocols

Public key certificate

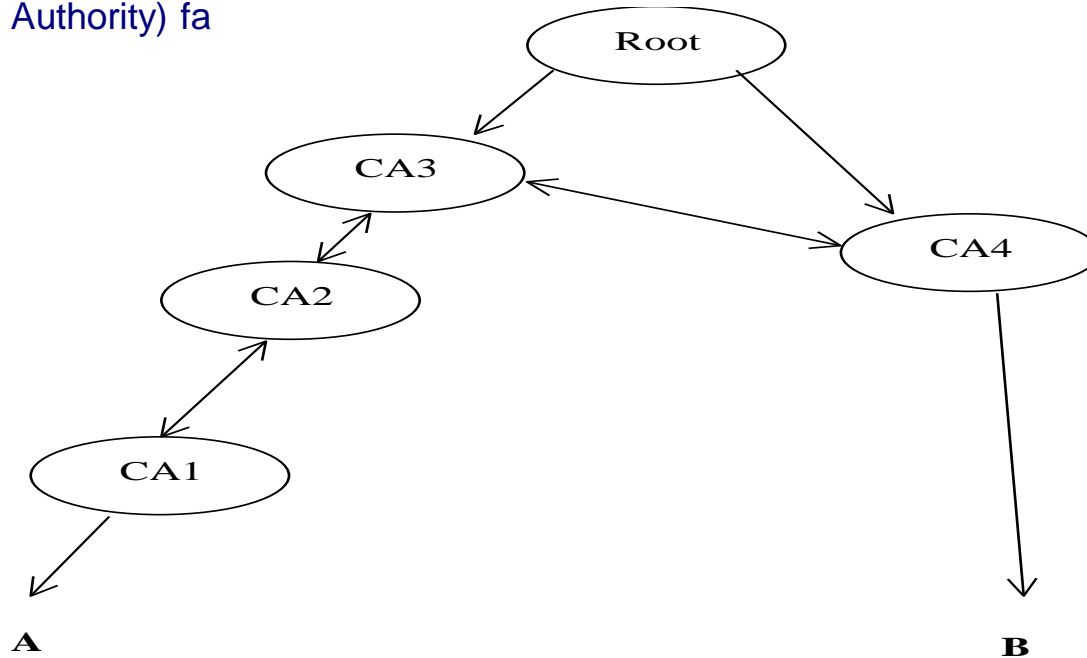
(Base64 kódolású X.509 (*.CER) fájl)

```
-----BEGIN CERTIFICATE-----  
  
MIIDUjCCArugAwIBAgIQusggE3yFp+3yF86CyEUWczANBqkqhkiG9w0BAQQFADBF  
MQswCQYDVQQGEwJVUzEXMBUGA1UEChMOVmVyaVNPZ24sIEluYy4xNzA1BgNVBASt  
LkNsYXNzIDIgUHViG1jIFByaW1hcnkgQ2VydG1maWNhdG1vbiBBdXR0b3JpdHkw  
HhcNOTgwNTEyMDAwMDAwWhcNMDQwMTA2MjM1OTU5WjCBuDEXMBUGA1UEChMOVmVya  
aVNPZ24sIEluYy4xNzA1BgNVBAStFlZlcm1TaWduIFRydXN0IE51dHdvcmsxRjBE  
BgNVBAStPXd3dy52ZXJpc2lnbi5jb20vcmlvbnNpdG9yeS9SUEEgSW5jb3JwLiBC  
eSBSZWYyLExJQUJlUFRFKGMpOTgxNDAYBgNVBAMTK1Zlcm1TaWduIENSYXNzIDIg  
Q0EgLSBjb20wZmVyaVNPZ24sIEluYy4xNzA1BgNVBAStLmN5bDBHbG9uZS9SUEEgSW5jb3JwLiBC  
MIGJAoGBALXG1ReJbAsWV8Ja9DalpLEZ0aCj5+L7d1XxdjFeUs0CAADPC6a6pe  
SbFok4M1rCRfoiMcaU2409t92o/BCC+1WDq2S8TU29iudfqGIpkiAShgpdvVMN8h  
cF5Ima0hVJHR31/7OclTG+J6U1jFDV0TB7NQxAZLOfhUq7mLaRITAgMBAAGjgbQw  
gbEwEQYJYIZIAYb4QgEBBQDAgEGMDUGA1UdHwQuMCwwKqAooCaGJGh0dHA6Ly9j  
cmwudmVyaVNPZ24uY29tL3BjYTIuMS4xLmN5bDBHbG9uZS9SUEEgSW5jb3JwLiBC  
+EUBBwEBMC0wKwYIKwYBBQUHAgEWH3d3dy52ZXJpc2lnbi5jb20vcmlvbnNpdG9y  
eS9SUEEgSW5jb3JwLiBCwIBADALBgNVHQ8EBAMCAQYwDQYJKoZIhvcNAQEE  
BQADgYEAgktjUSLq20lVX/o4zdMkXuH00xi0Saa0jBREETcUj5pSuLKEzt3sF4nV  
dcSKUoEH1FgF2a6fuwjQZJuZg1YBL7/eNKHBiUjXj4tiFsaFkGUI2B35wLdsy0B+  
a0+XIRs0/lrzoMdfcAMg7aKC7Oawb3zTOvcoAihGEEoL5J/DLk=  
  
-----END CERTIFICATE-----
```

Data Security: Protocols

Public key certificate

CA (Certification Authority) fa



- A ismeri CA1, B ismeri CA4 hiteles publikus kulcsát.
- Nyíl iránya a hitelesítés irányát mutat.
- Root CA publikus kulcsa már nem tanúsítvánnyal igazolt, hitelesen ismertnek feltételezett.

Data Security: Protocols

Public key certificate

Amikor A és B egymással egy nyilvános kulcsú kriptográfiára épülő protokollt szeretne futtatni, be kell szereznie a másik fél nyilvános kulcsát igazoló tanúsítványokat.

Például A elküldi a tanúsítványok alábbi listáját B számára:

$$A \rightarrow B: \{k_A^P\}_{CA1}, \{k_{CA1}^P\}_{CA2}, \{k_{CA2}^P\}_{CA3}, \{k_{CA3}^P\}_{CA4}$$

ahol $\{k\}_{CA}$ azt jelenti, hogy k kulcsot CA tanúsítja.

B a listát fordított sorrendben dolgozza fel:

1. A lista utolsó eleme és $CA4$ általa hitelesen ismert publikus kulcsa alapján megállapíthatja $CA3$ hiteles publikus kulcsát.
2. A lista megelőző, harmadik eleme alapján - most már $CA3$ hiteles kulcsa alapján - megállapíthatja $CA2$ hiteles publikus kulcsát.
3. A lista második eleme alapján - most már $CA2$ hiteles kulcsa alapján - megállapíthatja $CA1$ hiteles publikus kulcsát, majd ennek alapján az első tanúsítvány felhasználásával elér a céljához, A publikus kulcsa hitelességének ellenőrzéséhez.