Kriptográfia és Információbiztonság
9. előadás

MÁRTON Gyöngyvér

Sapientia Egyetem, Matematika-Informatika Tanszék
Marosvásárhely, Románia
mgyongyi@ms.sapientia.ro

2023

Miről volt szó az elmúlt előadáson?

- a diszkrét logaritmus (DL) probléma, a DL feltételezés
- hatványok és generátor elemek
- a Diffie-Hellman kulcscsere
- DL problémán alapuló digitális aláírások:
  - az ElGamal digitális aláírás
  - a DSA (Digital Signature Algorithm) vagy DSS (Digital Signature Standard)
- elliptikus görbéken alapuló kriptográfia

Miről lesz szó?

- a kommunikáció biztonsága - megoldások
- kriptográfiai kulcsok kezelése
  - a szimmetrikus kulcs megosztása
  - publikus kulcsok megosztása
- ál-véletlenszám generátorok
  - a Blum-Blum shup generátor
  - Goldreich-Levin generátor

A kommunikáció biztonsága - megoldások

- link titkosítás (link encryption)
- végpontok közötti titkosítás (end-to-end encryption)

A kommunikáció biztonsága - megoldások

link titkosítás (link encryption):

- a teljes hálózati adatforgalom titkosítása, ezért minden elosztási ponton az adatokat vissza kell fejteni, majd újra titkosítani azért, hogy a célpontokat meg lehessen határozni
- általában a hálózati protokollok része, a szolgáltató kezdeményezi
- a legismertebb szolgáltatás az SSL/TLS protokoll
- az ember-szerver típusú kommunikációban használják

A kommunikáció biztonsága - megoldások

végpontok közötti titkosítás (end-to-end encryption):

- a fejlécet leszámítva az adatok visszafejtése csak a célponton valósul meg
- az alkalmazások szintjén valósul meg, a felhasználó kezdeményezi
- csak a kommunikációban résztvevő felek tudják a továbbított adatokat elolvasni/titkosítani/visszafejteni
- nincsen lehallgatásra lehetőség: az internet szolgáltatók, a távközlési rendszerek, a kommunikációt kiszolgáló rendszerek sem tudnak a titkosított adatokhoz hozzáférni
- a felhasználók adatait/üzeneteit a szolgáltatók nem tudják senkinek sem kiadni
- a valóságban sok alkalmazás "hátsó ajtón" (back door) keresztül, a felhasználók tudta nélkül képes a kommunikációs adatok lehallgatására, kiadására
- a rendszer sérülékeny pontja a végpont, azaz a végponton elhelyezkedő eszköz és személy; megoldás: az eszközök távolról való irányítása

A kommunikáció biztonsága - megoldások

- kommunikáló felek adatainak a titkosításához szükség van egy szimmetrikus kulcsra, amelyet csak a kommunikáló felek ismernek
- a szimmetrikus kulcs megosztása, cseréje, tárolása, "eldobása" fontos kriptográfiai feladat
- szesszió-kulcs (session key): K_S
  - a tulajdonképpeni szimmetrikus kulcs, vagy egy abból származtatott érték,
  - a kommunikáció két végpontján található eszköz/felek adatainak biztonságos megosztására használják, a teljes kommunikáció titkosítását végzik a szesszió kulccsal,
  - egy kapcsolat idejére használják, utána "eldobják"
- mester-kulcs (master key): K_m
  - a szesszió-kulcs megosztására használják, tulajdonképpen a szesszió kulcsokat titkosítják a mester-kulcs segítségével,
  - a mester kulcsok megosztására többféle technika is létezik, például fizikai úton

Kriptográfiai kulcsok kezelése

- a szimmetrikus kulcsok megosztása
  - szimmetrikus titkosítási módszerrel
  - aszimmetrikus titkosítási módszerrel
- a publikus kulcsok megosztása
  - publikus közlemény segítségével (public announcement)
  - publikusan elérhető könyvtár segítségével (publicly available directory)
  - publikus hatóság segítségével (public-key authority)
  - publikus tanúsítványok segítségével (public-key certificates)

A szimmetrikus kulcs megosztása

1. az A eszköz/fél kiválaszt egy kulcsot és fizikai úton juttatja el a másik eszközhöz/félhez
2. egy harmadik megbízható fél választja ki a kulcsot és fizikai úton juttatja el az A és B eszközökhöz/felekhez
3. az A és B eszközök/felek korábban használt kulcsa alapján egy új kulcsot lehet származtatni
4. ha az A és B eszköz/fél és egy harmadik megbízható C eszköz/fél között rendre létezik egy titkos kommunikációs csatorna, akkor a C eszköz/fél ezen a titkos csatornán keresztül el tudja juttatni a kulcsokat úgy az A, mint a B eszköz/fél számára

A szimmetrikus kulcs megosztása

Megoldások és problémák a szimmetrikus kulcs megosztásakor:

- fizikai kézbesítés
  - link encryption esetében kivitelezhető, mert minden eszköz csak a link másik oldalán elhelyezkedő partnerével kell megoldja a kulcscserét
  - end-to-end encryption esetében nehezen kivitelezhető, mert:
    - hálózati szinten minden kommunikáló fél között szükséges a szimmetrikus kulcsot megosztani, ez n hálózati csomópont esetében $\frac{n \cdot (n-1)}{2}$ kulcs kezelését jelenti,
    - alkalmazás szinten egy hálózatban lehet több ezer csomópont, mindegyik csomópontban pedig több tízezer felhasználó, ekkor tízmillió szimmetrikus kulcs megosztása komoly nehézséget fog okozni
- a 3. pont szerinti szimmetrikus kulcsmegosztás alkalmazható úgy a link encryption mind az end-to-end encryption esetében, de ha egy támadó hozzáfér egy kulcshoz, akkor az összes korábbi kulcsot is megtudja határozni
- a 4. pont szerinti szimmetrikus kulcsmegosztás különböző változatait használják úgy link encryption, mind az end-to-end encryption esetében

A szimmetrikus kulcs megosztása

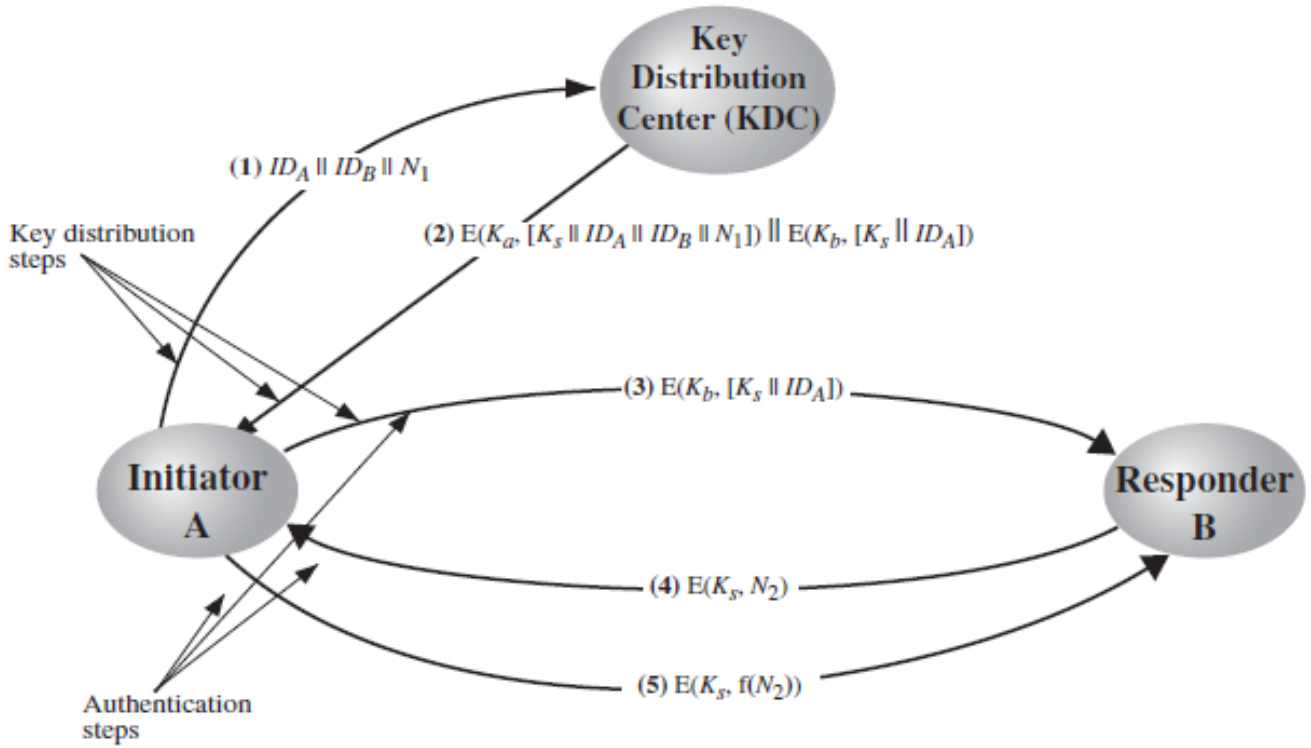
Szimmetrikus titkosítási módszerrel, egy központi KDC használatával

- feltételezzük, létezik egy megbízható KDC (key distribution center) és hogy minden felhasználó rendelkezik egy mester kulccsal, ami korábban került megosztásra, és amit csak a felhasználó és a KDC ismer: K_a, K_b
- N eszköz esetében a rendszerben megosztott mesterkulcsok száma N, ezek megosztása már nem olyan problématikus
- N_1, N_2 nonce (number use at once) értékek: időpecsét, számláló, random szám, amely minden felhasználó esetében egyedi, nehezen kitalálható
- ID_A, ID_B az A illetve B eszközök/felek azonosítói, például a hálózati cím
- K_S a megosztott kulcs
- f egy egyszerű átalakító függvény, pl. a bemenethez hozzáad 1-et.
- $E(K_b, [K_S \mid\mid ID_A])$: a $K_S \mid\mid ID_A$ üzenet a K_b kulccsal szimmetrikus kriptográfiával titkosított értéke

A szimmetrikus kulcs megosztása

Szimmetrikus titkosítási módszerrel, egy központi KDC használatával, folytatás

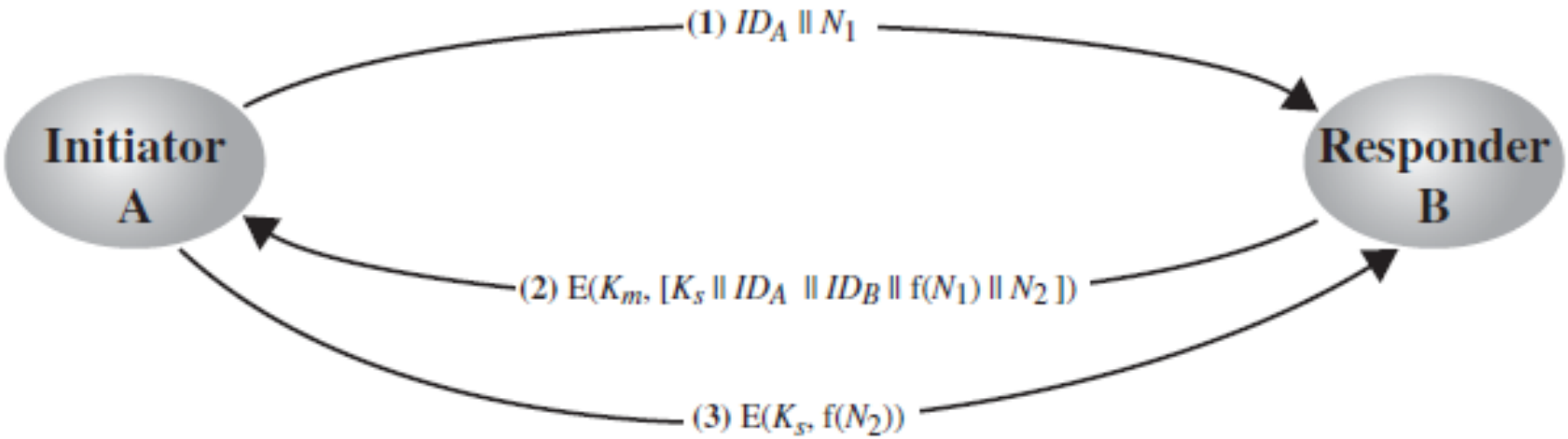
- K_a, K_b: a megosztott mesterkulcsok
- K_S: a megosztott szimmetrikus kulcs



A szimmetrikus kulcs megosztása

Szimmetrikus titkosítási módszerrel, központi KDC használata nélkül

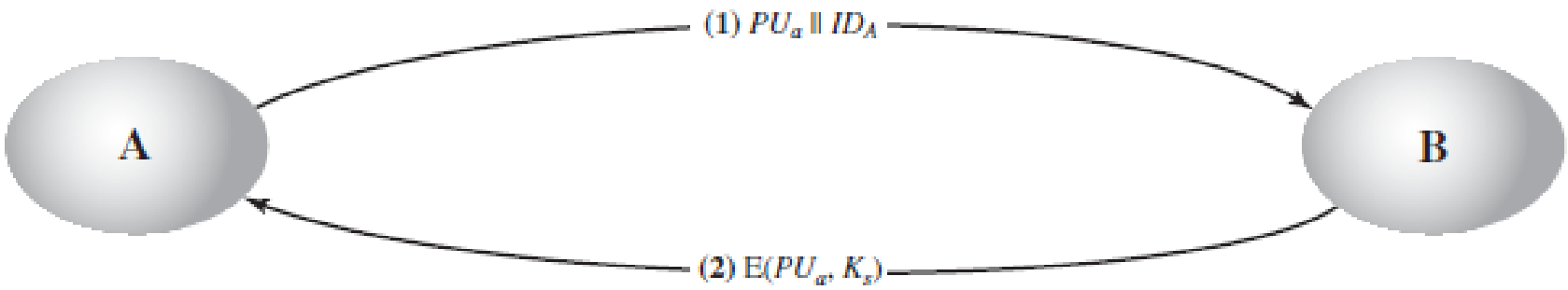
- Nagy számú felhasználó esetén nem praktikus, általában lokális hálózatokban használják, a kulcsok száma miatt, mely n felhasználó esetében $n \cdot (n-1)/2$.
- K_m: korábban megosztásra került, mindkét fél által ismert mester kulcs
- K_S: a megosztott kulcs



A szimmetrikus kulcs megosztása

Aszimmetrikus titkosítási módszerrel, autentikáció nélkül:

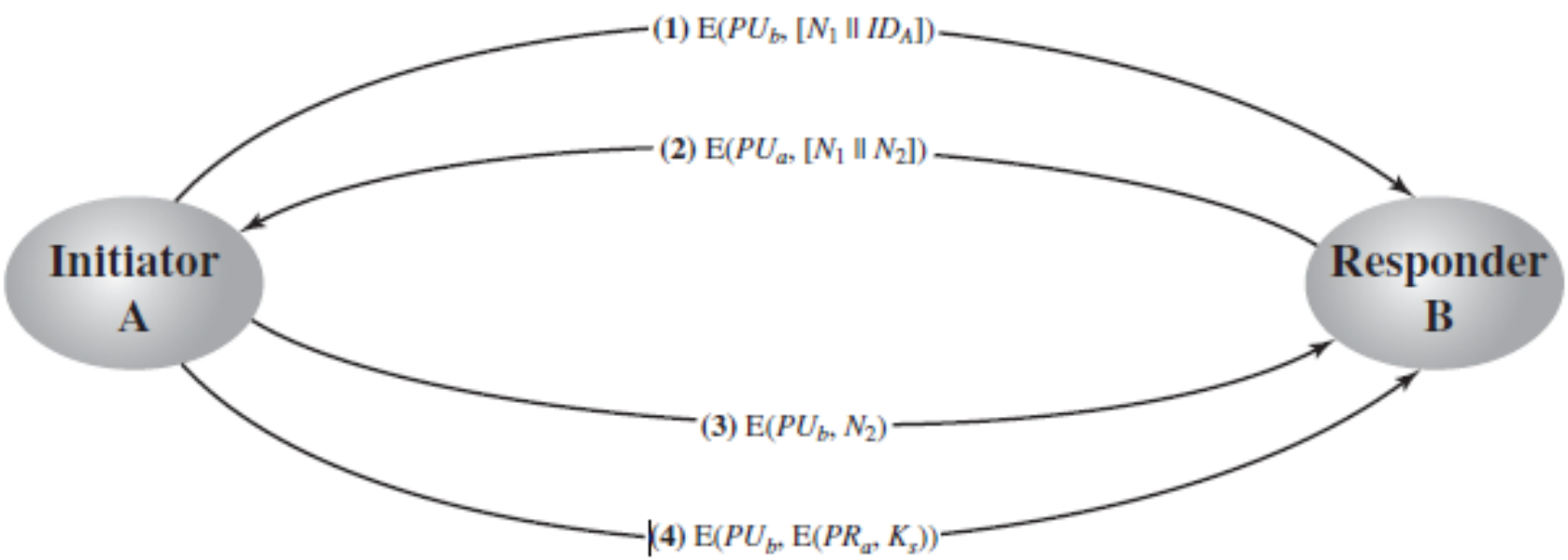
- a legegyszerűbb kulcsmegosztási technikát 1979-ben Merkle dolgozta ki
- a *man in the middle* típusú támadással szemben nem biztonságos
- az A egység, a kommunikáció megkezdése előtt generál egy PU_a, PR_a publikus/privát kulcspárt, ahol a PU_a értéket megosztja egy publikus csatornán B-vel, a PR_a értéket pedig titokban tartja



A szimmetrikus kulcs megosztása

Aszimmetrikus titkosítási módszerrel, autentikációval:

- Feltételezzük, hogy a felek korábban kigenerálták a megfelelő PU_a, PR_a, illetve PU_b, PR_b publikus/privát kulcspárokat és a publikus kulcsaikat korábban már megosztották.
- A megosztott kulcs a K_S lesz.



Publikus kulcsok megosztása

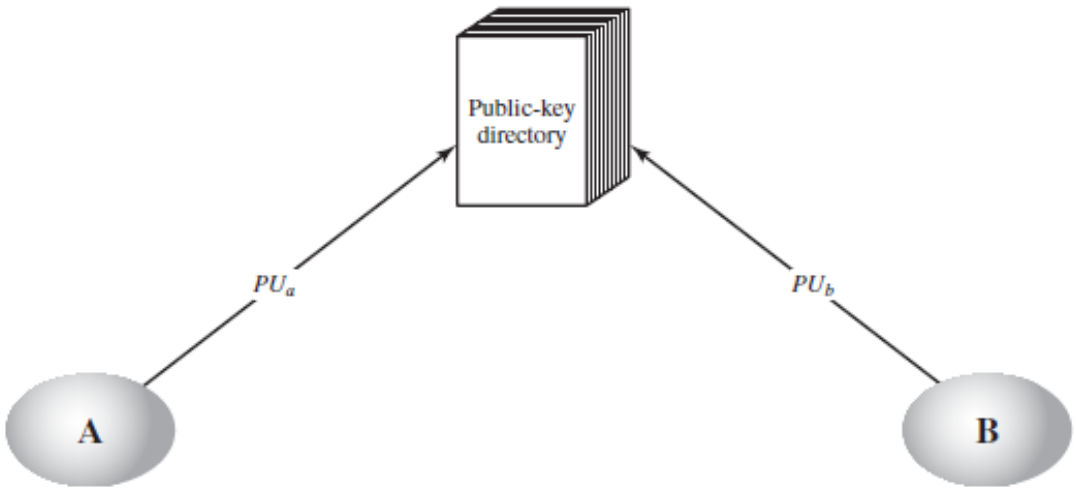
Publikus közlemény (Public Announcement) segítségével

- a publikus kulcsok nyilvánosak, ezért bárki nyilvánosan közzé teheti a saját publikus kulcsát, pl. hozzáfűzve egy adott üzenethez
- könnyű a hamisítás, mert bárki kiadhatja magát másnak

Publikus kulcsok megosztása

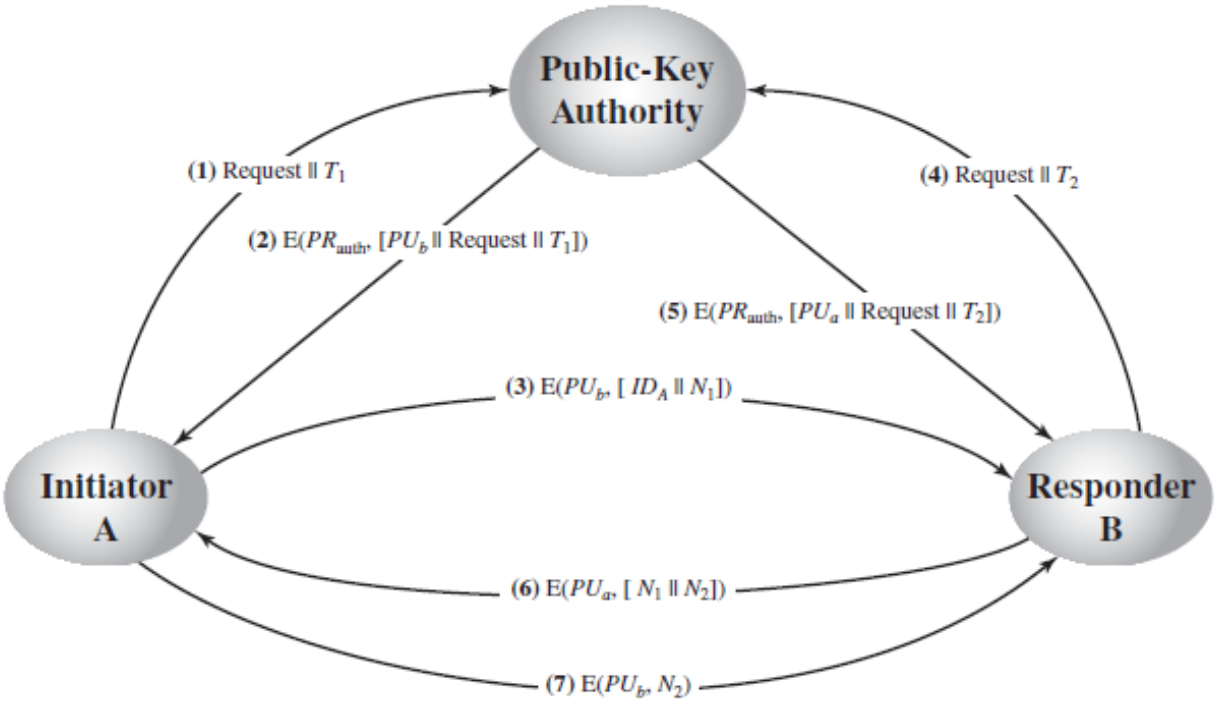
Publikusan elérhető könyvtár (Publicly Available Directory) segítségével:

- egy publikusan elérhető könytárat biztosítunk a felhasználók számára
- feltételezzük, hogy a publikus könyvtár használatát lehetővé tevő szerv megbízható
- a könyvtár használata regisztrációhoz kötött, amely a felhasználó autetifikálását is megoldja
- egy támadó ha hozzáfér a könyvtárhoz tetszőleges számú hamis publikus kulcsot elhelyezhet itt



Publikus kulcsok megosztása

Publikus-kulcsú hatóság (Public-key Authority) segítségével:



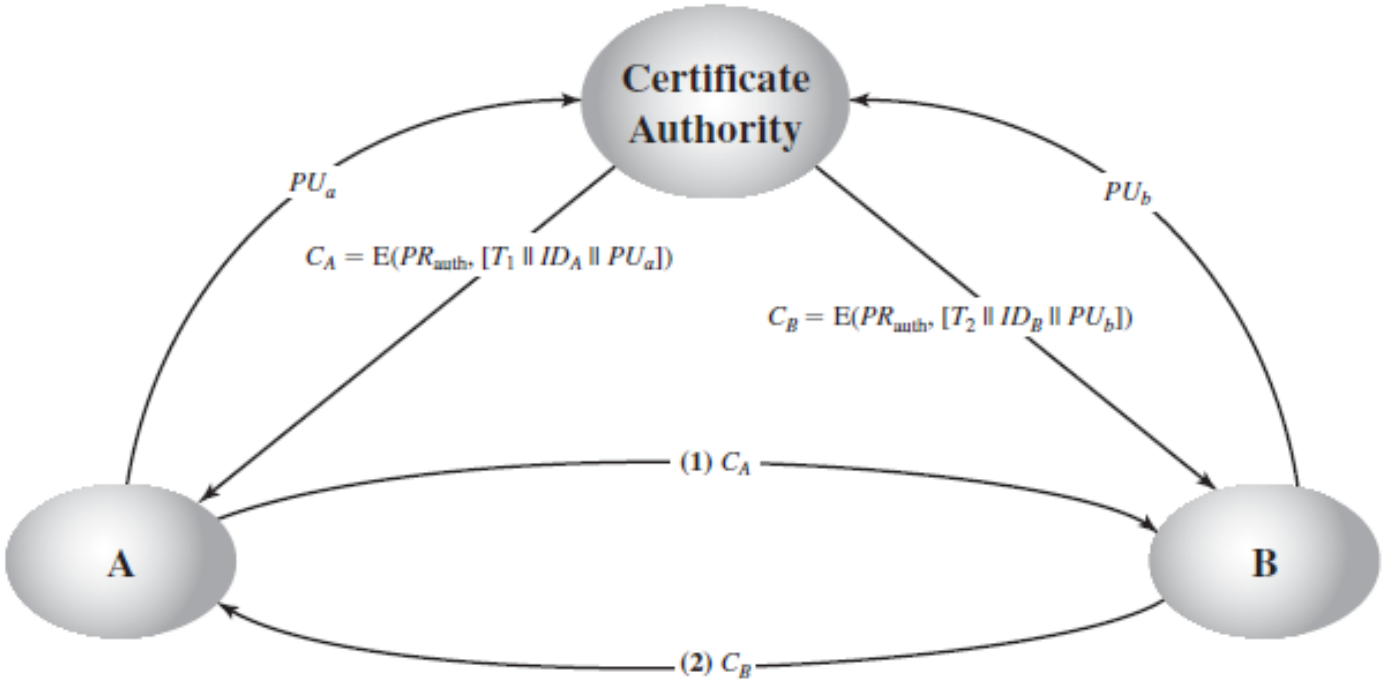
Publikus kulcsok megosztása

Publikus-kulcsú hatóság segítségével:

- minden felhasználó számára kiosztujuk a hatóság egy publikus kulcsát, ahol a megfelelő privát kulcsot csak a hatóság ismeri
- a felek közötti megeggyezés több üzenetcserét igényel
- egy fél annyi publikus kulcsot kell lekérjen, ahány eszközzel (féllel) szeretne kommunikálni, amelyek mindegyikét ismeri a központi hatóság, ez komoly gyengesége a rendszernek

Publikus kulcsok megosztása

Publikus-kulcsú tanusítványok (Public-key Certificates) használata:



Publikus kulcsok megosztása

Publikus-kulcsú tanusítványok használata:

- egy tanusítvány a következőket tartalmazza: a publikus kulcsot, a kulcstulajdonos azonosítóját és ezen adatok digitálisan aláírt értékét
- a tanusítványt egy megbízható intézmény/hatóság kell kiállítsa
- a tanusítvány kiállítása azt jelenti, hogy a felhasználó bemutatja publikus kulcsát a hatóságnak, amit az *aláír.*
- az aláírt tanusítványt, az aláírt értékkel a felhasználó ezután közzé teszi
- ha egy publikus kulcsot használni szeretnénk, akkor le kell ellenőrizni, ami azt jelenti, hogy a hatóság publikus kulcsát használva elvégezzük az aláírás-ellenőrzési folyamatot
- ha az ellenőrzés sikerrel jár az azt jelenti, hogy a publikus kulcs tulajdonosa az akinek mondja magát

Ál-véletlenszám generátorok

- a kriptográfiában alkalmazható ál-véletlenszám generátorok (pseudo-random generators, PRG) biztonsága más kritériumokon alapszik
- PRG: egy hatékony G determinisztikus algoritmus, amely egy kezdeti s értékből (seed) kiindulva egy r kimenetet generál,
- $s \in \mathbb{S}, r \in \mathbb{R}, \mathbb{S} = \{0,1\}^k, \mathbb{R} = \{0,1\}^K$, ahol k sokkal kisebb mint K,
- ekkor azt mondjuk, hogy a G egy PRG, amely $(\mathbb{S}, \mathbb{R})$ felett van értelmezve
- informálisan egy PRG biztonságát az határozza meg, hogy $s \xleftarrow{R} \mathbb{S}$, illetve $r \xleftarrow{R} \mathbb{R}$ esetében egy hatékony támadó ne tudjon különbséget tenni $G(s)$ és r között

Ál-véletlenszám generátorok

A következő bit teszt (The next bit test):

- legyen G az $(\{0,1\}^k, \{0,1\}^K)$ felett értelmezett ál-véletlenszám generátor,
- abban az esetben, amikor egy hatékony támadó a G utolsó $K-1$ bit-értékei alapján megtudja határozni a K-adik, azaz az utolsó bit értkét, a G nem lesz biztonságos,
- a fenti kijelentés fordítottja is igaz,

Kriptográfiában alkalmas PRG-k:

- Blum-Blum-Shub generátor
- Goldreich-Levin generátor

A Blum-Blum-Shub generátor

- alkalmas kriptográfiai rendszerekben, biztonságát a faktorizációs és kvadratikus maradék feltételezés adja
- nagy számításigényű, ezért csak nagy biztonságot követelő rendszereknél alkalmazzák
- az alkalmazott függvény: a moduláris négyzetre emelés
- legyen a generált bitsor:

  $(b_1, b_2, \ldots b_n)$, ahol $b_i = u_i \pmod 2$ és $u_i = u_{i-1}^2 \pmod N$, ahol

  $N = P \cdot Q$ és P, Q prímek, úgy hogy $P = 2 \cdot p + 1$, $Q = 2 \cdot q + 1$ és p, q is prímek
- a generált P, Q értékeket biztonságos prímeknek (safe prime) hívják
- a generált prímekre fennáll: $P \equiv Q \equiv 3 \pmod 4$, ezért ezeket még Blum egészeknek is hívják
- u_0 a generátor egy kezdeti értéke: $u_0 = r^2 \pmod N$, ahol $r \xleftarrow{R} \mathbb{Z}_N^*$

A Blum-Blum-Shub generátor

Példa

- legyen $P = 23 = 2 * 11 + 1, Q = 83 = 2 * 42 + 1$
- $N = 1909$
- $r = 784$
- $u_0 = r^2 \pmod N = 1867$

$u_i = u_{i-1}^2 \pmod N$	$b_i = u_i \pmod 2$
1764	0
26	0
676	0
725	1
650	0
611	1
...	

A Goldreich-Levin generátor

- kriptográfiai biztonságát a faktorizációs és kvadratikus maradék feltételezés adja
- legyen $N = P \cdot Q$, és P, Q Blum egészek: $P, Q \equiv 3 \pmod 4$, és P, Q prímek.
- legyen s, r véletlenszerűen generált értékek, ahol $s, r \leq N$, és $u = r^2 \pmod N$
- a generált bitsort a következőképpen határozzuk meg:

$$(GL_s(u \pmod N), GL_s(u^2 \pmod N), \ldots, GL_s(u^n \pmod N))), \text{ ahol}$$

$$GL_s(u) = \bigoplus_{i=1}^{n} (s_i \otimes u_i),\ s = s_1 s_2 \ldots s_n,\ u = u_1 u_2 \ldots u_n$$

- a $\otimes$ operátor bitenkénti és műveletet jelent, azaz a bementi két bitsort AND művelettel kell "összeadni"
- az $\bigoplus$ operátor modulo 2 szerinti összeadást jelent, azaz a bemeneti bitsor bitjein páronként XOR műveletet kell végezni

A Goldreich-Levin generátor

Példa

- legyen $P = 19 \equiv 3 \pmod 4$, $Q = 23 \equiv 3 = \pmod 4$
- $N = 437$
- $s = 345 = (101011001)_2$
- $r = 329$, $u = r^2 \pmod N = 302$

$u \pmod N$	$bin(u)$	$s \bigotimes u$	$bin(s \bigotimes u)$	$GL_s(u)$
302	100101110	264	100001000	0
308	100110100	272	100010000	0
35	100011	1	1	1
351	101011111	345	101011001	1
404	110010100	272	100010000	0
215	11010111	81	1010001	1
340	101010100	336	101010000	1
...				

Megjegyzés: $GL_s(u)$ kimenete 0, ha a bináris alakban, azaz $bin(s \bigotimes u)$-ban páros számú egyes van, másképp 1.