

Aszimmetrikus (nyilvános) kulcsú rendszerök



Aszimmetrikus (nyilvános) kulcsú rendszerek

- az E_k kódoló eljárás kulcsa is nyilvános, tehát az egyetlen titkos adat a $D_{k'}$ dekódoló eljárás k' kulcsa
- az asszimmetria azt jelenti, hogy $k \neq k'$, sőt k -ból k' -et nagyon nehezen, csak exponenciális idő alatt futó algoritmussal lehet megkapni

Aszimmetrikus (nyilvános) kulcsú rendszerek

- Kulcsgenerálás:
 - egy $(E_k, D_{k'})$ pár szerkesztése, ahol k nyilvánossá tehető és k' titkos
 - szükség van egy $f = E_k$ ún. csapóajtó-függvényre (trapdoor function)

Csapóajtó-függvények

- A **csapóajtó-függvény** egy olyan $f = E_k$ bijektív függvény, ami (algoritmikusan) könnyen kiértékelhető, viszont az $f^{-1} = D_{k'}$ inverz csak nagyon nehezen számolható, kivéve ha ismerjük a k' adatot, a „kiskaput”, ami hatékonytá teszi f^{-1} kiértékelését.

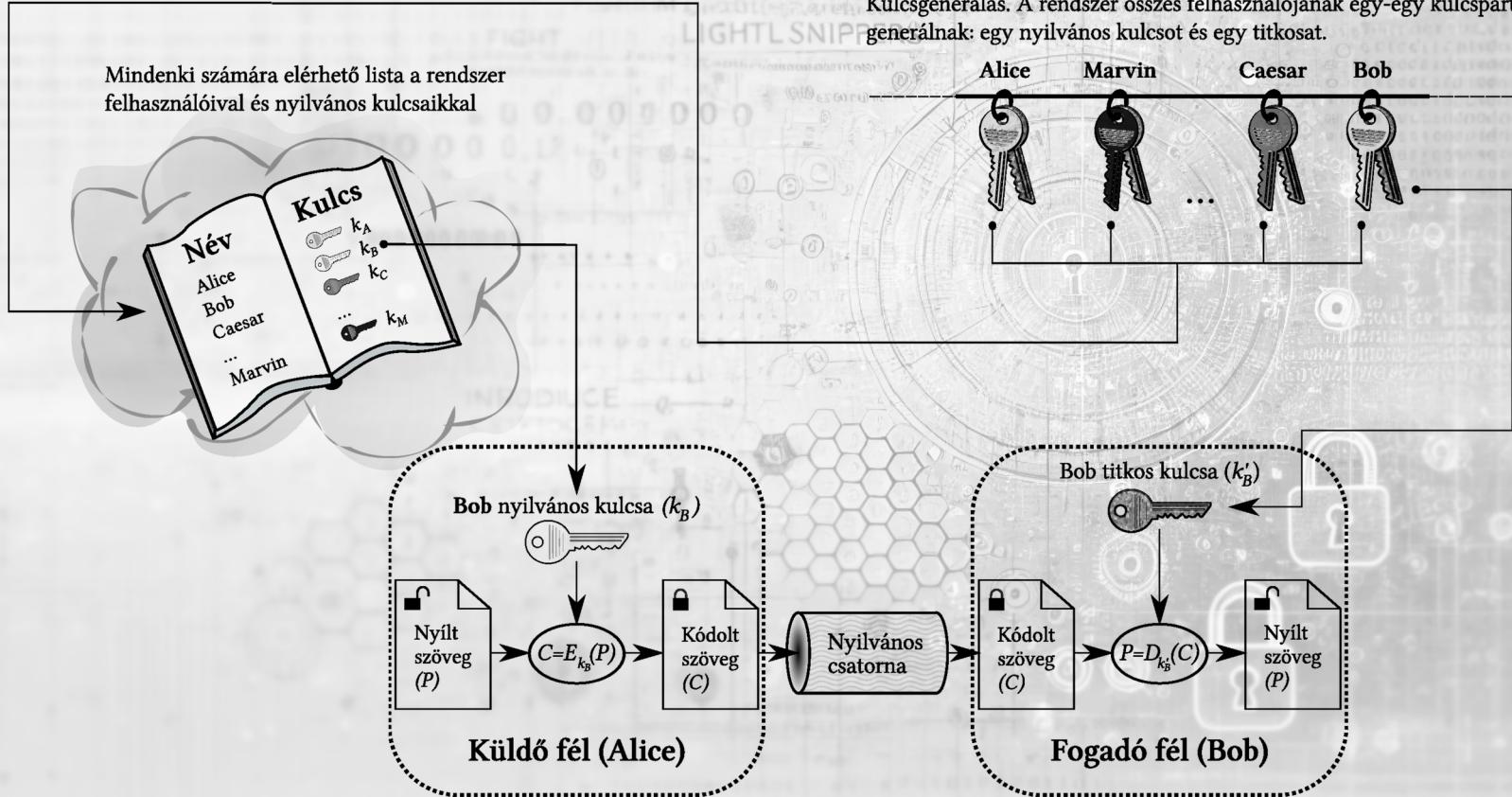


Csapóajtó-függvények

- ha nem ismerjük a kiskaput jelentő plusz információt, akkor f^{-1} számolásához látszólag meg kell oldanunk egy NP-feladatot
- megtörténhet, hogy akár a kiskapu ismerete nélkül is elkerülhetjük az NP-feladat megoldását (ez illető függvényre alapozott kriptorendszer „végét” jelenti)
- szinte lehetetlen szigorú matematikai értelemben bizonyítani egy függvényről, hogy valóban csapóajtó-függvény

Aszimmetrikus (nyilvános) kulcsú rendszerek

A nyilvános kulcsok bekerülnek egy nyilvános adatbázisba.



Aszimmetrikus (nyilvános) kulcsú rendszerek

- Előnyök:
 - sokkal gazdaságosabb és biztonságosabb kulcskezelés
 - letagadhatatlanság (non-repudiation) és hitelesítés (authentication)
 - skálázhatóság
 - biztonságos kommunikáció bármilyen környezetben

Aszimmetrikus (nyilvános) kulcsú rendszerek

- Hátrányok:

- többnyire csak empirikus (tapasztalati) tények támasztják alá az egyes aszimmetrikus kulcsú rendszerek biztonságát (lásd [csapóajtó-függvények](#))
- gyakran lassúbbak a szimmetrikus rendszereknél (például mert sokszor moduláris hatványozást végeznek, ami sokkal több időt igényel, mint néhány XOR)
- nagy kulcsméret

A Merkle–Hellman (knapsack) kriptorendszer

- Ralph C. Merkle és M. Hellman publikálták a módszert 1978-ban
- a módszer csapóajtó-függvénye egy NP-teljes feladatra, a hátizsák (knapsack) feladatra épül
- a rendszer feltörhető (A. Shamir, 1984), iparban már nem használatos

A Merkle–Hellman (knapsack) kriptorendszer

A hátízsák feladat: adott egy $(v_0, v_1, \dots, v_{n-1})$ pozitív egészekből álló sorozat és a V természetes szám. Találjunk egy n -bites p egész számot úgy, hogy

$$\varepsilon_0 v_0 + \varepsilon_1 v_1 + \cdots + \varepsilon_{n-1} v_{n-1} = V,$$

ahol $p = (\varepsilon_{n-1} \varepsilon_{n-2} \dots \varepsilon_1 \varepsilon_0)_2$ és $\varepsilon_i \in \{0, 1\}$ a p bináris felírásában szereplő számjegyek.

Megj.: a feladatnak nulla vagy több megoldása is lehet

A Merkle–Hellman (knapsack) kriptorendszer

- **Példa:** $(v_0, v_1, v_2, v_3, v_4) = (2, 3, 1, 4, 5)$, $V=5$. Ekkor $v_0 + v_1 = v_2 + v_3 = v_5 = V$, tehát $p_1 = (11000)_2 = 40$, $p_2 = (00110)_2 = 6$, illetve $p_3 = (00001)_2 = 1$ mind jó megoldások.
- **Példa:** $(v_0, v_1, v_2) = (10, 20, 30)$, $V=5$. Nincs megoldás.

A Merkle–Hellman (knapsack) kriptorendszer

- az általános hátizsák-feladat NP-teljes
- a csapóajtó-függvényhez szükség van a feladatnak egy könnyen megoldható változatára
- a könnyen megoldható változat az úgynévezett *szupernövekvő hátizsák* feladat

A Merkle–Hellman (knapsack) kriptorendszer

A szupernövekvő hátízsák feladat: adott egy $(v_0, v_1, \dots, v_{n-1})$ pozitív egészekből álló szupernövekvő sorozat (ami azt jelenti, hogy $v_1 > v_0, v_2 > v_0 + v_1, \dots, v_{n-1} > \sum_{i=0}^{n-2} v_i$) és a V természetes szám. Találunk egy n -bites p egész számot úgy, hogy

$$\varepsilon_0 v_0 + \varepsilon_1 v_1 + \cdots + \varepsilon_{n-1} v_{n-1} = V,$$

ahol $p = (\varepsilon_{n-1} \varepsilon_{n-2} \dots \varepsilon_1 \varepsilon_0)_2$ és $\varepsilon_i \in \{0, 1\}$ a p bináris felírásában szereplő számjegyek.

A Merkle–Hellman (knapsack) kriptorendszer

- Példa: a következő sorozat segítségével szupernövekvő hátizsák feladatot lehet megadni:

(3, 5, 9, 20, 40)

Algorithm SuperIncreasingKnapsack:

Input: $V, (v_0, v_1, \dots, v_{n-1})$

Output: p

$s := V$

for $i := n - 1, \dots, 0$ **do**

if $v_i \leq s$ **then**

$\varepsilon_i := 1$

$s := s - v_i$

else

$\varepsilon_i := 0$

end if

end for

if $s = 0$ **then**

$p := (\varepsilon_{n-1} \varepsilon_{n-2} \dots \varepsilon_1 \varepsilon_0)_2$

 return p

else

 return „Nincs megoldás”

end if

end Algorithm

A Merkle–Hellman (knapsack) kriptorendszer

Példa. Adott $a(v_0, v_1, v_2, v_3, v_4) = (3, 5, 9, 20, 40)$, $V = 32$ szupernövekvő háti-zsák feladat. Látható, hogy

$$i_0 = 3 \Rightarrow \varepsilon_3 = 1, \varepsilon_4 = 0 \text{ és } V - v_{i_0} = 32 - 20 = 12 > 0$$

$$i_1 = 2 \Rightarrow \varepsilon_2 = 1 \text{ és } V - v_{i_0} - v_{i_1} = 32 - 20 - 9 = 3 > 0$$

$$i_2 = 0 \Rightarrow \varepsilon_0 = 1, \varepsilon_1 = 0 \text{ és } V - v_{i_0} - v_{i_1} - v_{i_2} = 0,$$

tehát a megoldás $p = (10110)_2 = 42$.

A Merkle–Hellman (knapsack) kriptorendszer

Kulcsgenerálás:

- megválasztjuk a következő értékeket:
 - $v = (v_0, v_1, \dots, v_{n-1})$ szupernövekvő sorozat
 - $m \in \mathbb{N}, m > \sum_{i=0}^{n-1} v_i$
 - $a \in \mathbb{N}, 0 < a < m, \text{lko}(a, m) = 1$
- az előbbiek alapján kiszámoljuk következő értékeket:
 - $b = a^{-1} \pmod{m}$
 - $w = (w_0, w_1, \dots, w_{n-1}), w_i = av_i \pmod{m}$

A Merkle–Hellman (knapsack) kriptorendszer

- **Nyilvános (kódoló) kulcs:** $k = (w_0, w_1, \dots, w_{n-1})$
- **Titkos (dekódoló) kulcs:** $k' = (b, m)$, ami alapján kiszámolható a és v :
 - $a = b^{-1} \text{ mod } m$
 - $bw_i = v_i \text{ mod } m$

A Merkle–Hellman (knapsack) kriptorendszer

Kódolás: A P üzenet most egy n -bites tömb, vagyis $P = (\varepsilon_{n-1}\varepsilon_{n-2}\dots\varepsilon_1\varepsilon_0)_2$. Ha például angol ábécét használó szövegünk van, akkor minden betű az ábécébeli sorszámán alapulva 5 biten ábrázolható:

$$\mathbf{A} \rightarrow 0 = (00000)_2$$

$$\mathbf{B} \rightarrow 1 = (00001)_2$$

$$\vdots$$

$$\mathbf{Z} \rightarrow 25 = (11001)_2$$

Ekkor $C = E_k(P) = \sum_{i=0}^{n-1} \varepsilon_i w_i \in \mathbb{N}^*$.

A Merkle–Hellman (knapsack) kriptorendszer

Dekódolás: Legyen $V = bC \bmod m$, $V < m$. Ekkor

$$\begin{aligned} V &= bC \bmod m = \sum_{i=0}^{n-1} \varepsilon_i b w_i \bmod m \\ &= \sum_{i=0}^{n-1} \varepsilon_i b a v_i \bmod m = \sum_{i=0}^{n-1} \varepsilon_i v_i \bmod m, \end{aligned}$$

ahonnan $V = \sum_{i=0}^{n-1} \varepsilon_i v_i$, hiszen $V < m$ és $\sum_{i=0}^{n-1} \varepsilon_i v_i \leq \sum_{i=0}^{n-1} v_i < m$. V -re és $(v_0, v_1, \dots, v_{n-1})$ -re megoldva a szupernövekvő hátizsák feladatot, megkapjuk $P = (\varepsilon_{n-1} \varepsilon_{n-2} \dots \varepsilon_1 \varepsilon_0)_2$ -t, ami az egyetlen megoldás.

A Merkle–Hellman (knapsack) kriptorendszer

Példa (kulcsgenerálás):

- $v = (3, 5, 9, 20, 40)$ szupernövekvő sorozat
- $m = 79, a = 17 \implies b = 14$
- $w = (3 \cdot 17, 5 \cdot 17, 9 \cdot 17, 20 \cdot 17, 40 \cdot 17) \bmod 79 = (51, 6, 74, 24, 48)$
- **nyilvános kulcs:** $k = (51, 6, 74, 24, 48)$
- **titkos kulcs:** $k' = (14, 79)$

A Merkle–Hellman (knapsack) kriptorendszer

Példa (a SZIA szó kódolása):

$$S \rightarrow 18 = (10010)_2 = P_1 \Rightarrow C_1 = E_k(P_1) = 51 + 24 = 75$$

$$Z \rightarrow 25 = (11001)_2 = P_2 \Rightarrow C_2 = E_k(P_2) = 51 + 6 + 48 = 105$$

$$I \rightarrow 8 = (01000)_2 = P_3 \Rightarrow C_3 = E_k(P_3) = 6$$

$$A \rightarrow 0 = (00000)_2 = P_4 \Rightarrow C_4 = E_k(P_4) = 0.$$

A SZIA titkosított változata tehát $(75, 105, 6, 0)$ lesz.

A Merkle–Hellman (knapsack) kriptorendszer

Példa (dekódolás): w és b alapján meghatározzuk a szupernövekvő $v = (3, 5, 9, 20, 40)$ sorozatot

$$V_1 = bC_1 \bmod m = 23 = 1 \cdot 3 + 1 \cdot 20 \Rightarrow P_1 = (10010)_2 = 18 \rightarrow \mathbf{S}$$

$$V_2 = bC_2 \bmod m = 48 = 1 \cdot 3 + 1 \cdot 5 + 1 \cdot 40 \Rightarrow P_2 = (11001)_2 = 25 \rightarrow \mathbf{Z}$$

$$V_3 = bC_3 \bmod m = 5 = 1 \cdot 5 \Rightarrow P_3 = (01000)_2 = 8 \rightarrow \mathbf{I}$$

$$V_4 = bC_4 \bmod m = 0 \Rightarrow P_4 = (00000)_2 = 0 \rightarrow \mathbf{A}$$

A Merkle–Hellman (knapsack) kriptorendszer

- Kriptoanalízis:
 - a $k = (w_0, \dots, w_{n-1})$ nyilvános kulcs nem szupernövekvő, de speciális, mert egy szupernövekvő sorozatból származik (ez a módszer egyik gyenge pontja)
 - A. Shamir algoritmusa (1982) polinomiális időben generálhat olyan megfelelő (b', m') párokat (nem feltétlenül a titkos kulcsot), amellyel $v' = (v'_0, \dots, v'_{n-1})$ szupernövekvő, ahol $v'_i = b'w_i \text{ mod } m'$
 - a módszer támadható az LLL-algoritmus (Lenstra–Lenstra–Lovász) segítségével is

Az RSA

- Az RSA módszer szerzői Ronald Linn Rivest, Adi Shamir és Leonard Max Adleman (1978).
- Az RSA-ban használt csapóajtó elvi alapja, hogy könnyebb két nagyon nagy prímszámot generálni és összeszorozni, mint szorzatukat faktorizálni.
- Az RSA napjaink egyik leggyakrabban használt titkosítási eljárása.

Az RSA

Kulcsgenerálás:

- véletlenszerűen válasszunk két (nagy, legalább 100 számjegyű) prímet, jelölje ezeket p és q
- számoljuk ki a köv. értékeket: $n = pq$, $\varphi(n) = (p-1)(q-1)$, ahol $\varphi(n)$ az Euler-függvény értéke n -ben
- válasszunk egy e egész számot úgy, hogy teljesüljenek: $1 < e < \varphi(n)$ és $\text{lko}(e, \varphi(n)) = 1$ (jó, ha bináris felírásban e -nek kevés 1-es bitje van)
- számoljuk ki: $d = e^{-1} \bmod \varphi(n)$
- **nyilvános kulcs:** $k = (n, e)$
- **titkos kulcs:** $k' = d$

Az RSA

Kódolás: Tételezzük fel, hogy a szövegünk egy N betűs ábécében íródott. Legyen ℓ olyan, hogy $N^\ell \leq n \leq N^{\ell+1}$, vagyis $\ell = \lfloor \log_N n \rfloor$. Legyen a P nyílt üzenet egy ℓ betűs tömb. Ez azt jelenti, hogy

$$P = (b_{\ell-1} \dots b_0)_N = b_{\ell-1}N^{\ell-1} + \dots + b_1N + b_0 < N^\ell \leq n,$$

tehát $P \in \mathbb{Z}_n$. Ekkor $C = E_k(P) = P^e \pmod{n}$, tehát $C \in \mathbb{Z}_n$, $C < n < N^{\ell+1}$, vagyis C egy $\ell + 1$ betűs tömb.

Dekódolás: $P = D_{k'}(C) = C^d \pmod{n}$.

Tétel. Az előbbi jelölésekkel: $a^{ed} \equiv a \pmod{n}$, $\forall a \in \mathbb{Z}$.

Az RSA

$$(m^e)^d \% n = m$$
$$m^e \% n = x$$
$$x^d \% n = m$$

Kódolás

Dekódolás

Az RSA

Példa:

- $p = 19, q = 41, n = pq = 779, \varphi(n) = 18 \cdot 40 = 720$
- $e = 17$ (igaz, hogy $\text{lko}(e, \varphi(n)) = 1$)
- $d = 17^{-1} \text{ mod } 720 = 593$
- **nyilvános kulcs:** $k = (n, e) = (779, 17)$
- **titkos kulcs:** $k' = d = 593$

Az RSA

*A szöveg, amelyet titkosítani szeretnénk, angol ábécét használ ($N = 26$). Legyen tehát a titkosítandó üzenet a következő: **SZIASZTOKXBU**. Mivel $26^2 < 779 < 26^3$, kétbetűs tömböket kódolunk. A tömbök a következők lesznek:*

$$P_1 = \mathbf{SZ} = 18 \cdot 26 + 25 = 493$$

$$P_2 = \mathbf{IA} = 8 \cdot 26 + 0 = 208$$

$$P_3 = \mathbf{SZ} = 18 \cdot 26 + 25 = 493$$

$$P_4 = \mathbf{TO} = 19 \cdot 26 + 14 = 508$$

$$P_5 = \mathbf{KX} = 10 \cdot 26 + 23 = 283$$

$$P_6 = \mathbf{BU} = 1 \cdot 26 + 20 = 46$$

Az RSA

A kódolt tömbök:

$$C_1 = P_1^e \bmod n = 493^{17} \bmod 779 = 0 \cdot 26^2 + 18 \cdot 26 + 25 = \mathbf{ASZ}$$

$$C_2 = P_2^e \bmod n = 208^{17} \bmod 779 = 0 \cdot 26^2 + 8 \cdot 26 + 0 = \mathbf{AIA}$$

$$C_3 = P_3^e \bmod n = 493^{17} \bmod 779 = 0 \cdot 26^2 + 18 \cdot 26 + 25 = \mathbf{ASZ}$$

$$C_4 = P_4^e \bmod n = 508^{17} \bmod 779 = 0 \cdot 26^2 + 13 \cdot 26 + 0 = \mathbf{ANA}$$

$$C_5 = P_5^e \bmod n = 283^{17} \bmod 779 = 0 \cdot 26^2 + 6 \cdot 26 + 24 = \mathbf{AGY}$$

$$C_6 = P_6^e \bmod n = 46^{17} \bmod 779 = 1 \cdot 26^2 + 0 \cdot 26 + 1 = \mathbf{BAB}$$

Az RSA

Tétel. Ha a következő adatok közül az egyik ismert, a többi is megtalálható rövid (polinomiális) idő alatt: $p, q, \varphi(n), d$.

- Kriptoanalízis:
 - jelenlegi (nyilvános) matematikai ismeretek szerint, jól megválasztott (bizonyos **kritériumoknak** megfelelő) és elég nagy értéket használó RSA feltörése gyakorlatban kivitelezhetetlen
 - P. Shor algoritmusa (1994) egy kvantumszámítógépen futtatva polinomiális időben faktorizál (de ez egyelőre csak elméleti lehetőség)

Az RSA

1. A prímszámokra vonatkozó kritériumok:

- (a) **Legyenek véletlenszerűek.** Lehetőleg ne legyenek „híres” prímek (pl. Mersenne-prímek, Fermat-prímek stb.).
- (b) **p és q ne legyen túl közel egymáshoz.** Ha p és q értékek közel vannak egymáshoz, akkor $n = pq$ könnyen faktorizálható az úgynevezett Fermat-faktorizációval. Ennek lényege a következő: mivel $n = pq \implies \frac{(p+q)^2}{4} - n = \frac{(p-q)^2}{4}$, tehát ahogy p és q közelítenek egymáshoz $p - q \rightarrow 0 \implies \frac{p+q}{2} \rightarrow \sqrt{n}$. Ekkor \sqrt{n} -hez közeli x értékekkel próbálkozunk, megnézzük, hogy $x^2 - n$ teljes négyzet-e. Ha igen, mondjuk $x^2 - n = y^2$, akkor $\frac{p+q}{2} = x$ és $\frac{p-q}{2} = y \implies p = x + y, q = x - y$.

Az RSA

2. $\varphi(n)$ -re vonatkozó kritériumok.

- (a) $\varphi(n)$ -nek a prímtényezői között legyenek nagyobb prímek is (mert ha nem, $\varphi(n)$ próbálgatással meghatározható).
- (b) $[p - 1, q - 1]$ (a legkisebb közös többszörös) legyen lehetőleg nagy, mert ismerete ekvivalens a rendszer feltörésével. Nagyon rossz, ha például $(p - 1) \mid (q - 1)$ vagy fordítva, mert akkor a legkisebb közös többszörös, $[p - 1, q - 1]$, kicsi.

Az RSA

3. e -re vonatkozó kritériumok:

- (a) Jó, ha e kisebb, mert akkor $d = e^{-1} \pmod{\varphi(n)}$ nagyobb lesz.
- (b) Jó, ha e -nek kevés (például csak 1 vagy 2 darab) 1-es bitje van (ezért jó választást jelentenek e értékére a $17 = 2^4 + 1$ vagy $65537 = 2^{16} + 1$ prímek). Ez azért fontos, mert az e -vel való moduláris hatványozás az ismételt négyzetre emeléses módszerrel gyorsabb.
- (c) Lehetőleg $e \neq \frac{\varphi(n)}{2} + 1$, mert ha igen, akkor $e - 1$ többszöröse $p - 1$ -nek és $q - 1$ -nek is. Ebből következik, hogy $a^{e-1} \equiv a \pmod{n}$, $\forall a \in \mathbb{N}$ -re, ahol $(a, n) = 1$, és akkor $a^e \equiv a \pmod{n}$ bármely a -ra, így minden szövegtömb önmagába kódolódna.

Az RSA

4. A szövegtömbökre vonatkozó kritériumok:

- (a) Jó, ha mindegyik tömb végén valamilyen véletlenszerű, értelmetlen szövegrész van. Ezt kitöltésnek nevezzük (angolul padding). Kitöltés alkalmazás nélkül, mivel n és e nyilvánosak, a támadó fél beazonosíthat meghatározott titkosított szövegtömböket a kódolt üzenetünkben.
- (b) Megtörtéhet, hogy bizonyos tömbök önmagukba kódolódnak. Például, ha

$$\begin{cases} a_0 \equiv u \pmod{p} \\ a_0 \equiv v \pmod{q} \end{cases},$$

ahol $(u, v) \in \{(1, 1), (-1, 1), (1, -1), (-1, -1)\}$, akkor $a_0^e \equiv a_0 \pmod{n}$. Tehát jó, ha $P \neq a_0$.

Az RSA

- Jelölés: RSA- s , ahol s az algoritmusban használt $n = pq$ érték bithossza (pl. RSA-512, RSA-1024, RSA-2048, stb.)
- A gyakorlatban a következő változatokat ajánlják. RSA-1024, RSA-2048, RSA-3072, RSA-7680, RSA-15360.
- 2009-ben egy RSA-768-as számot 2 év alatt faktorizáltak (kb. 1000 processzoros számítógépen), ugyanazon a hardveren az RSA-1024-et kb. 7500 év alatt lehetne feltörni

Diszkrét logaritmáláson alapuló rendszerek

- Ezek a rendszerek a diszkrét logaritmálás nehézségére alapoznak: ha \mathbb{F}_q a q -elemű test és $\mathbb{F}^* = \langle g \rangle$, akkor $y = g^x$ hatványozás hatékonyan elvégezhető, de x -et meghatározni y és g ismeretében nehéz (algoritmikusan)
- $\log_g y$ diszkrét logaritmus meghatározása (ha q elég nagy és $q-1$ -nek van legalább egy nagy prímosztója) egy hasonló bithosszúságú RSA-szám faktorizációjával azonos nehézségű feladat
- Példák: Shamir háromlépéses protokollja, Massey–Omura-rendszer, ElGamal titkosítási rendszer

A Diffie–Hellman-hipotézis

- Hipotézis: Legyen \mathbb{F}_q egy véges test (ahol $q=p^\ell$ elég nagy), $\mathbb{F}_q^* = \langle g \rangle$ és legyenek $a, b \in \{1, \dots, q-1\}$ véletlenszerűen kiválasztott (titkos) értékek. Ekkor g, g^a , és g^b ismeretéből g^{ab} csak a diszkrét logaritmus kiszámításával vezethető le, például a következő módon: $a = \log_g g^a$ és $g^{ab} = (g^b)^a$.

Az ElGamal titkosítási rendszer

- Szerzője Taher Elgamal egyiptomi kriptográfus (1985).
- Az ElGamal biztonsága a Diffie–Hellman-sejtésen, illetve a diszkrét logaritmálás nehézségén alapul.
- Legyen \mathbb{F}_q véges test ($q = p^\ell$ elég nagy) és $\mathbb{F}_q^* = \langle g \rangle$. Ezek rögzített, nyilvános adatok.

Az ElGamal titkosítási rendszer

Kulcsgenerálás:

- véletlenszerűen választunk egy $a \in \{1, \dots, q-1\}$ értéket
- **nyilvános kulcs:** $k = g^a \in \mathbb{F}_q^*$
- **titkos kulcs:** $k' = a$

Az ElGamal titkosítási rendszer

Kódolás:

- véletlenszerűen választunk egy $b \in \{1, \dots, q-1\}$ értéket
- $(C, C') = (g^b, Pg^{ab})$, ahol g^a a fogadó fél nyilvános kulcsa

Dekódolás:

- $P = C'(C^a)^{-1}$, mert $C'(C^a)^{-1} = Pg^{ab}((g^b)^a)^{-1} = Pg^{ab}g^{-ab} = P$

Az ElGamal titkosítási rendszer

Példa. Legyen $q = 65537$ prímszám. Ekkor $\mathbb{F}_q = \mathbb{Z}_q$, és ismert, hogy $\mathbb{Z}_q^* = \langle 3 \rangle$. Alice titkos kulcsa $k'_A = 12907$, és nyilvánossá teszi $k_A = 3^{12907} \pmod{q} = 34625$ -öt. Bob kódolni akarja Alice-nek a $P = \text{WE_GO!}$ üzenetet. Tegyük fel, hogy a következő 31 betűs ábécét használja:

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z . ? ! ,

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30

A szöveg számokká alakítását (65537-nél kisebb számokká) az RSA-nál ismertetett módon végzi. $31^3 < 65537 < 31^4$, tehát hárombetűs tömböket kódolunk.

$$P_1 = \text{WE_} = 22 \cdot 31^2 + 4 \cdot 31 + 26 = 21292$$

$$P_2 = \text{GO!} = 6 \cdot 31^2 + 14 \cdot 31 + 29 = 6229$$

Az ElGamal titkosítási rendszer

Véletlenszerűen választ $b_1 = 11618$ -ot és $b_2 = 9017$ -et. Ekkor:

$$(C_1, C'_1) = (3^{b_1}, P_1 \cdot 3^{k_A b_1}) = (14155, 42568) = (\mathbf{AOWT}, \mathbf{BNJF})$$

$$(C_2, C'_2) = (3^{b_2}, P_2 \cdot 3^{k_A b_2}) = (20715, 48875) = (\mathbf{AVRH}, \mathbf{BT_T})$$

Az átalakítások:

$$14155 = 0 \cdot 31^3 + 14 \cdot 31^2 + 22 \cdot 31 + 19 = \mathbf{AOWT}$$

$$42568 = 1 \cdot 31^3 + 13 \cdot 31^2 + 9 \cdot 31 + 5 = \mathbf{BNJF}$$

$$20715 = 0 \cdot 31^3 + 21 \cdot 31^2 + 17 \cdot 31 + 7 = \mathbf{AVRH}$$

$$48875 = 1 \cdot 31^3 + 19 \cdot 31^2 + 26 \cdot 31 + 19 = \mathbf{BT_T}$$

Az ElGamal titkosítási rendszer

Alice a következőképpen dekódolja az üzenetet. Megkapja a párokat:

$$(C_1, C'_1) = (14155, 42568) \quad (C_2, C'_2) = (20715, 48875),$$

elvégzi a

$$P_1 = C'_1 \cdot C_1^{q-1-k'_A} \bmod q = 42568 \cdot 14155^{65536-12907} \bmod 65537 = 21292 = \text{WE_}$$

$$P_2 = C'_2 \cdot C_2^{q-1-k'_A} \bmod q = 48875 \cdot 20715^{65536-12907} \bmod 65537 = 6229 = \text{GO!}$$

műveleteket, és el tudja olvasni a $P = P_1 \parallel P_2 = \text{WE_GO!}$ üzenetet.

A Diffie–Hellman-kulcscsere

- Mivel a nyilvános kulcsú rendszerek nem annyira hatékonyak, mint a szimmetrikusak (pl. DES, AES), a legjobb, ha a tulajdonképpeni titkosítást szimmetrikus rendszerrel végezzük, de a kulcskezelést (kulcscserét) aszimmetrikus kulcsú titkosítási rendszerrel valósítjuk meg.
- A Diffie–Hellman-kulcscsere protokollt kifejezetten ehhez szerkesztették 1976-ban.
- A két kommunikáló fél megegyezhet az általuk használt szimmetrikus titkosítási rendszer közös kulcsában anélkül, hogy ezt a kulcsot valamilyen formában el kellene küldeni.

A Diffie–Hellman-kulcscsere

Legyen \mathbb{F}_q egy véges test (ahol $q=p^\ell$ elég nagy), $\mathbb{F}_q^*=\langle g \rangle$, ahol g egy nyilvános generátor.

Kulcsgenerálás:

- minden felhasználó választ egy $a \in \{1, \dots, q-1\}$ elemet.
- **nyilvános kulcs:** $k = g^a \in \mathbb{F}_q^*$
- **titkos kulcs:** $k' = a$

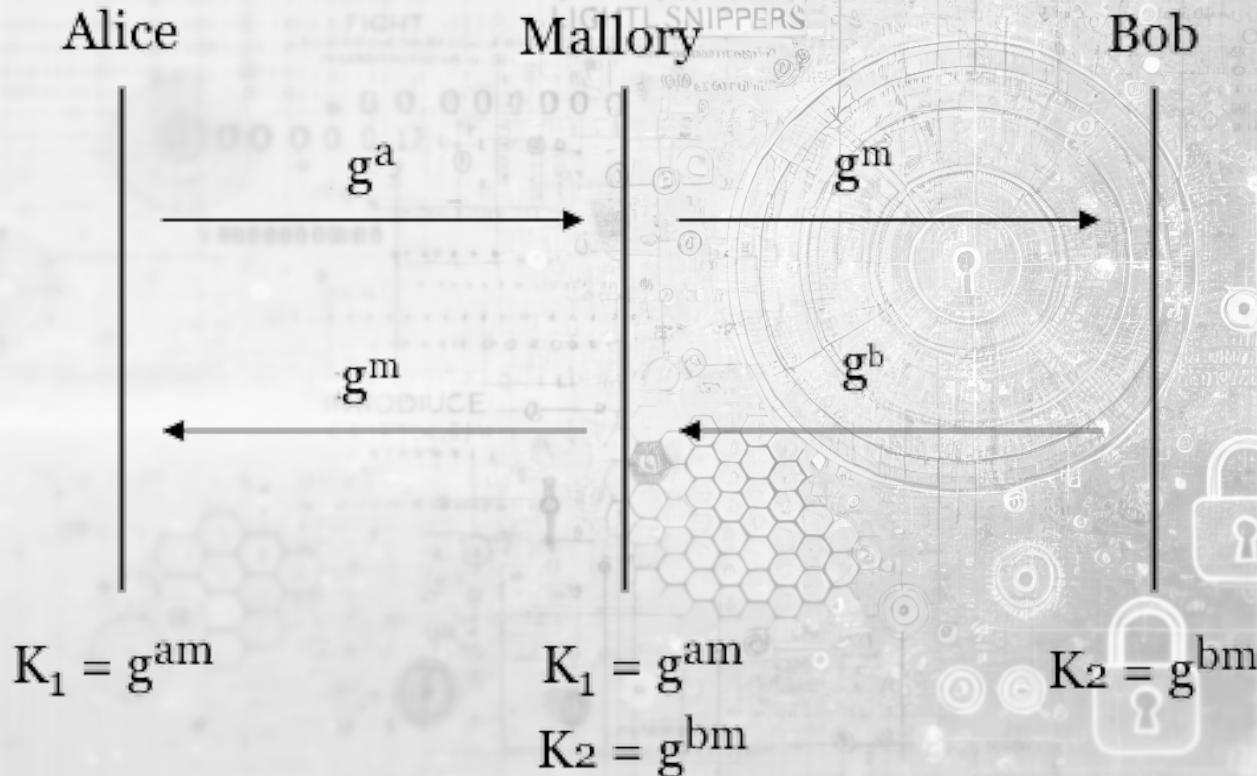
A Diffie–Hellman-kulcscsere

Két felhasználó (Alice és Bob) a következő módon egyeznek meg az (E_k, D_k) szimmetrikus kulcsú rendszer közös k kulcsában:

1. Alice elküldi a saját $k_A = g^a$ nyilvános kulcsát Bobnak.
2. Bob visszaküldi a saját $k_B = g^b$ nyilvános kulcsát Alice-nek.
3. A közös k kulcsot le lehet vezetni a $g^{ab} \in \mathbb{F}_q$ -ból (ezt mindenki tudják számolni a titkos kulcsuk segítségével).

A Diffie–Hellman-kulcscsere

Támadás:



Háromlépéses protokollok

- A háromlépéses protokollok (three-pass protocol) segítségével két fél előzetes kulcsmegosztás nélkül tud kommunikálni (nyilvános kulcsot sem kell közzétenni).
- Ha (e, d) egy megfelelő módon kiválasztott kulcspár és k egy ezektől független kulcs, akkor a következő tulajdonságnak kell teljesülnie: $D_d(E_k(E_e(P))) = E_k(P)$, ahol P az üzenet, E_k és E_e a kódoló eljárások, D_d a dekódoló eljárás.

Háromlépéses protokollok

Ha A el akarja küldeni B -nek a P üzenetet, akkor a következő lépések kell végrehajtani:

1. A generál egy (e_A, d_A) kulcspárt, majd elküldi B -nek az $E_{e_A}(P)$ üzenetet.
2. B generál egy (e_B, d_B) kulcspárt, majd visszaküldi A -nak az $E_{e_B}(E_{e_A}(P))$ üzenetet.
3. A elküldi B -nek az $D_{d_A}(E_{e_B}(E_{e_A}(P)))$ üzenetet.

A fogadó fél dekódolni tudja a 3. lépésben kapott üzenetet, mert $D_{d_A}(E_{e_B}(E_{e_A}(P))) = E_{e_B}(P)$, és $D_{d_A}(E_{e_B}(P)) = P$.

Shamir háromlépéses protokollja

Legyen q egy nagy prímszám. minden X felhasználó választ magának egy titkos $e_X \in \{1, 2, \dots, q-2\}$ kitevőt úgy, hogy $\text{lko}(e_X, q-1) = 1$, és meghatározza $d_X = e_X^{-1} \pmod{q-1}$ -et (kiterjesztett euklideszi algoritmussal).

Alice a következő módon küldi el Bobnak a $P \in \{1, \dots, q-1\}$ üzenetet:

1. lépés: Alice elküldi Bobnak $P^{e_A} \pmod{q}$ -t.
2. lépés: Bob elküldi Alice-nek $P^{e_A e_B} \pmod{q}$ -t.
3. lépés: Alice elküldi Bobnak $P^{e_A e_B d_A} = P^{e_B} \pmod{q}$ -t.

Bob a $P^{e_B} \pmod{q}$ -t dekódolni tudja d_B segítségével, hiszen $P^{e_B d_B} \equiv P \pmod{q}$. Itt felhasználtuk, hogy $P^{e_A d_A} \equiv P^{1+\ell(q-1)} \equiv P \pmod{q}$, hiszen a kis Fermat-tételből $P^{q-1} \equiv 1 \pmod{q} \implies P^{\ell(q-1)} \equiv 1 \pmod{q}$. Hasonlóan $P^{e_B d_B} \equiv P \pmod{q}$.

A Massey–Omura-rendszer

Minden X felhasználó választ magának egy titkos $e_X \in \{1, 2, \dots, 2^n - 2\}$ kitevőt úgy, hogy $\text{lnko}(e_X, 2^n - 1) = 1$, és meghatározza $d_X = e_X^{-1} \pmod{2^n - 1}$ -et.

Alice a következő módon küldi el Bobnak a $P \in \mathbb{F}_{2^n}^*$ üzenetet:

1. lépés: Alice elküldi Bobnak P^{e_A} -t ($\mathbb{F}_{2^n}^*$ -ban).
2. lépés: Bob elküldi Alice-nek $P^{e_A e_B}$ -t.
3. lépés: Alice elküldi Bobnak $P^{e_A e_B d_A} = P^{e_B}$ -t.

Bob a P^{e_B} -t dekódolni tudja d_B segítségével, hiszen $P^{e_B d_B} = P$. Itt felhasználtuk, hogy az $\mathbb{F}_{2^n}^*$ csoportban, mely $2^n - 1$ elemű, $P^{2^n - 1} = 1 \implies P^{e_A d_A} = P^{1 + \ell(2^n - 1)} = P$. Hasonlóan $P^{e_B d_B} = P$.