Nyilvános kulcsú titkosítás és az RSA kriptorendszer

2011. március 22.

1 Nyilvános kulcsú kriptorendszerek

2 Az RSA kriptorendszer

Egyirányú függvények

Definíció.

Egy $f : \mathcal{M} \to \mathcal{C}^*$ függvényt **egyirányú függvény***nek nevezünk, ha $f(m)$ könnyen kiszámítható minden $m \in \mathcal{M}$ esetén, de bármely véletlenszerűen az f képteréből választott c esetén nagyon nehéz (lényegében lehetetlen, computationally infeasible) olyan $m \in \mathcal{M}$ értéket találni, melyre $f(m) = c$.*

- A fenti "könnyen kiszámítható", "lényegében lehetetlen" fogalmak nincsenek matematikailag pontosan definiálva, mégis a gyakorlatban számos kriptorendszer biztonsága alapszik rajtuk.
- A fentiek miatt egyetlen függvényről sincs bizonyítva, hogy egyirányú.
- Valójában egyirányúnak sejtett függvényekről beszélhetünk (conjectured, candidate one-way functions).

Hash függvények

Definíció.

*Hash függvény*nek egy olyan hatékony módon kiszámítható függvényt nevezünk, amely tetszőleges hosszúságú bitsorozatokat adott hosszúságú bitsorozatokra képez. Ezeket az adott hosszúságú bitsorozatokat *hash érték*eknek nevezzük.
*Egyirányú hash függvény*en egy olyan hash függvényt értünk, amely kielégíti az egyirányú függvény definícióját.

- A kriptográfiában csak az egyirányú hash függvényeknek van jelentőségük.
- Az egyirányú hashfüggvény értékét szokták digitális lenyomatnak is nevezni (digital fingerprint, imprint, message digest).

Egyirányú csapóajtó függvények

Definíció.

Egyirányú csapóajtó függvénynek, másnéven nyilvános kulcsú titkosító függvénynek, egy olyan $f : \mathcal{M} \to \mathcal{C}^*$ injektív egyirányú függvényt nevezünk, mely rendelkezik az alábbi tuljdonsággal:

- *létezik egy olyan információ (melyet csapóajtó információnak, vagy csapóajtónak nevezünk), melynek segítségével egy adott $c \in \mathrm{Im}(f)$ értékhez meg tudjuk határozni azt az $m \in \mathcal{M}$ értéket, melyre $f(m) = c$, de ezen információ hiányában ezen m meghatározása lényegében lehetetlen.*

Példa

Legyen $f(x) \equiv x^e \pmod{n}$, ahol $n = pq$, $p \neq q$ prímek, és legyen d olyan, hogy

$$ed \equiv 1 \pmod{(p-1)(q-1)}.$$

1. Ennek a függvénynek az értékét polinomiális időben ki lehet számolni, a moduláris gyorshatványozással.
2. A d ismeretében az $f^{-1}(x^e) \equiv \pmod{n}$ kiszámítása ugyanilyen egyszerű, hiszen

$$(x^e)^d \equiv x^{ed} \equiv x \pmod{n}.$$

3. Megmutatjuk, hogy a d kiszámítása azonos nehézségű feladat, mint n faktorizálása.

Moduláris gyorshatványozás

Cél: Számítsuk ki $b^r \pmod{n}$ értékét, ahol $b, r, n \in \mathbb{N}$.

Probléma: Egyetlen hatványozással nem lehet a problémát megoldani, mert nagy számok esetén nem elég a memória, de az a megoldás is túl lassú, ha r-szer szorzunk b-vel, mindig redukálva modulo n.

Alapötlet: Használjuk r kettes számrendszerbeli előállítását:

$$r = \sum_{j=0}^{k} a_j 2^j.$$

A $b^r \pmod{n}$ értékét az alábbi lépésekben határozzuk meg:

Moduláris gyorshatványozás lépései

- Kezdőlépés: Legyen $b_0 := b$ és $c := \begin{cases} 1 & \text{ha } a_0 = 0 \\ b & \text{ha } a_0 = 1 \end{cases}$

Hajtsuk végre az alábbi lépést $j = 1, 2, \ldots, k$ esetén:

- j-edik lépés:
  - Számítsuk ki $b_j := b_{j-1}^2 \pmod{n}$ értékét.
  - Ha $a_j = 1$ akkor $c := c \cdot b_j \pmod{n}$.
  - Ha $a_j = 0$ akkor c változatlan marad.

A j-edik lépésben kiszámolt $c = c_j$ értékre igaz, hogy

$$c_j \equiv b^{r_j} \pmod{n},$$

ahol c_j a b^{r_j} legkisebb nem-negatív maradéka modulo n, és

$$r_j = \sum_{i=0}^{j} a_i 2^i.$$

Így a k-adik lépésben kiszámítjuk $c \equiv b^r \pmod{n}$ értékét.

Példa

Számítsuk ki $3^{61} \pmod{101}$.

1. Nyilván $61 = 1 + 2^2 + 2^3 + 2^4 + 2^5$, így $k = 5$, $a_1 = 0$ és $a_j = 1$, ha $j \neq 1$. Mivel $a_0 = 1$, így $c = b_0 = b = 3$, és most végigmegyünk a $j = 1, 2, 3, 4, 5$ eseteken.
2. $b_1 \equiv 3^2 \equiv 9 \pmod{101}$, így $b_1 = 9$, és mivel $a_1 = 0 \Rightarrow c = 3$.
3. $b_2 \equiv 9^2 \equiv 81 \pmod{101}$, így $b_2 = 81$, és mivel $a_2 = 1 \Rightarrow c \equiv 3 \cdot 81 \pmod{101}$, azaz $c = 41$.
4. $b_3 \equiv 81^2 \equiv 97 \pmod{101}$, így $b_3 = 97$, és mivel $a_3 = 1 \Rightarrow c \equiv 41 \cdot 97 \pmod{101}$, azaz $c = 38$.
5. $b_4 \equiv 97^2 \equiv 16 \pmod{101}$, így $b_4 = 16$, és mivel $a_4 = 1 \Rightarrow c \equiv 38 \cdot 16 \pmod{101}$, azaz $c = 2$.
6. $b_5 \equiv 16^2 \equiv 54 \pmod{101}$, így $b_5 = 54$, és mivel $a_4 = 1 \Rightarrow c \equiv 54 \cdot 2 \pmod{101}$, azaz $c = 7$.

Így $3^{61} \equiv 7 \pmod{101}$

Példa – Emlékeztető

Legyen $f(x) \equiv x^e \pmod{n}$, ahol $n = pq$, $p \neq q$ prímek, és legyen d olyan, hogy

$$ed \equiv 1 \pmod{(p-1)(q-1)}.$$

1. Ennek a függvénynek az értékét polinomiális időben ki lehet számolni, a moduláris gyorshatványozással.
2. A d ismeretében az $f^{-1}(x^e) \equiv x \pmod{n}$ kiszámítása ugyanilyen egyszerű, hiszen

$$(x^e)^d \equiv x^{ed} \equiv x \pmod{n}.$$

3. Megmutatjuk, hogy a d kiszámítása azonos nehézségű feladat, mint n faktorizálása.

$(p-1)(q-1)$ kiszámítása polinomiálisan ekvivalens n faktorizálásával

- Ha adott $n = pq$ faktorizált alakban, akkor $(p-1)(q-1)$ kiszámítható polinomiális időben.
- Ha adott n és $(p-1)(q-1)$, akkor

$$p + q = n - (p-1)(q-1) + 1$$

$$p - q = \sqrt{(p+q)^2 - 4n}$$

ahonnan

$$p = \frac{1}{2}[(p+q) + (p-q)]$$

$$q = \frac{1}{2}[(p+q) - (p-q)]$$

és mindez polinom időben kiszámítható.

d ismerete polinomiálisan ekvivalens n faktorizálásával

- Ha adott $n = pq$, akkor $(p-1)(q-1)$ és e ismeretében d kiszámítható polinom időben a kiterjesztett Euklideszi algoritmussal.
- Ha adott d, akkor (mivel n és e amúgy is nyilvános)
$$ed - 1 = 2^k s, \qquad k \in \mathbb{N},\ s \text{ páratlan.}$$
  - $ed \equiv 1 \pmod{\varphi(n)}$, így $\exists a \in \mathbb{Z}_n^*$, melyre $a^{2^k s} \equiv 1 \pmod{n}$.
  - legyen $j \in \mathbb{N}$ a legkisebb, melyre $a^{2^j s} \equiv 1 \pmod{n}$.
  - Ha $a^{2^{j-1} s} \not\equiv 1 \pmod{n}$ és $a^{2^j s} \equiv 1 \pmod{n}$ egyszerre igaz, akkor $b = a^{2^{j-1} s}$ nem-triviális négyzetgyöke 1-nek mod n, azaz
$$n \mid (b+1)(b-1), \quad \text{úgy, hogy} \quad n \nmid (b+1), \quad \text{és} \quad n \nmid (b-1)$$
  - Ekkor $\gcd(b+1, n) = p$ vagy $\gcd(b+1, n) = q$
  - a többszöri újraválasztásával d ismerete majdnem biztosan n faktorizációjára váltható

Nyilvános kulcsú kriptorendszerek

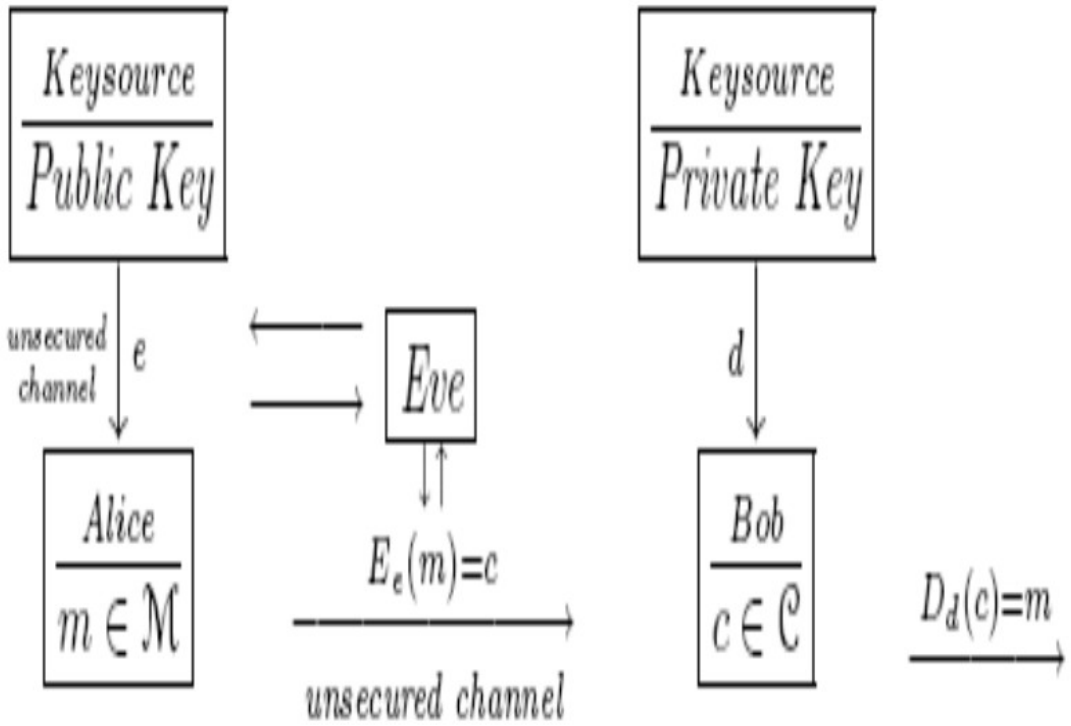
Definíció.

Egy olyan kriptorendszert, mely titkosító függvények egy $\{E_e\}$ és visszafejtő függvények egy $\{D_d\}$ halmazából áll nyilvános kulcsú kriptorendszernek vagy aszimetrikus kriptorendszernek nevezzünk, ha minden (e, d) kulcspár esetén az e titkosító kulcsot nyilvánosságra hozzák, míg a d visszafejtő kulcsot titokban tartják. A kriptorendszernek teljesítenie kell továbbá azt a feltételt is, hogy a d (titkos) visszafejtő kulcs kiszámítása az e (nyilvános) titkosító kulcsból lényegében lehetelen.

A nyilvános kulcsú kriptorendszer működése az alábbi analógiával világítható meg:

- Bobnak van egy széfje, melynek kombinációját csak Ő ismeri, és ezt a széfet nyitva hagyja.
- Ha valaki (pl. Alice) üzenetet akar hagyni Bobnak, akkor azt beteszi a széfbe, és a széfet bezárja.
- Ezután az üzenethez csak Bob férhet hozzá, más nem (sőt, még Alice sem!!!).

Nyilvános kulcsú kriptorendszer diagrammja



Szimmetrikus és aszimmetrikus kriptorendszerek összehasonlítása

	Előny	Hátrány
Szimmetrikus: DES, TDES, AES, ...	Közérthető, Egyszerű programozni, Rövid kulcshossz, Gyors	Legalább két személy a titokgazda, A kulcsot rövid ideig lehet tárolni, Kulcscsere.
Aszimmetrikus: RSA, ElGamal, ...	Matematikai eszközökkel elemezhető, Egy személy a titokgazda! A kulcs tárolható. Nyilvános/titkos kulcs	Lassú, Komplikált, Nehéz programozni.

Az RSA kriptorenszer

- Az RSA-t 1977-ben publikálta Ronald Rivest, Adi Shamir és Leonard Adleman
- Családneveik kezdőbetűiből lett az RSA betűszó.
- Algoritmusuk a fenti példában már elemzett elemi számelméleti ötleten alapszik.
- Ez volt az első olyan (nyilvánosságra hozott) algoritmus, mely megvalósította a nyilvános kulcsú kriptorendszer (Whitfield Diffie és Martin E. Hellman 1976-os dolgozatában megfogalmazott) alapötletét

I. RSA kulcsgenerálás

1. Bob generál két nagyjából azonos méretű $p \neq q$ nagy prímszámot
2. Kiszámítja az $n = pq$ és $\phi(n) = (p-1)(q-1)$ értékeket
3. Rögzít egy véletlen $e \in \mathbb{N}$, $1 < e < \phi(n)$ számot, melyre $\gcd(e, \phi(n)) = 1$
4. A kiterjesztett Euklideszi algoritmus segítségével kiszámítja azt a $d \in \mathbb{N}$ számot, melyre $1 < d < \phi(n)$, és

$$ed \equiv 1 \pmod{\phi(n)}$$

5. Bob nyilvánosságra hozza az (n, e) párt, és titokban tartja a d, p, q és $\phi(n)$ értékeket

Elnevezések

A fent definiált értékekre a következő elnevezéseket használjuk:

- (n, e) Bob nyilvános kulcsa
- d Bob titkos kulcsa (más néven privát kulcsa)
- e Bob (RSA) titkosító kitevője
- d Bob (RSA) visszafejtő kitevője

Az RSA sejtés

Az RSA kriptoanalízise legalább olyan bonyolult feladat, mint n faktorizálása.

Az RSA nyilvános kulcsú kriptorendszer

- **titkosító fázis**
  Az egyszerűség kedvéért tfh. az $m \in \mathcal{M}$ nyílt szöveg egy $m < n$ természetes szám formájában van kódolva, továbbá $\mathcal{M} = \mathcal{C}^* = \mathbb{Z}/n\mathbb{Z}$, és $\gcd(m, n) = 1$.
  1. Alice megkeresi Bob (n, e) nyilvános kulcsát az adatbázisból
  2. Kiszámolja a $c \equiv m^e \pmod{n}$, $1 \leq c < n$ értéket, így titkosítva m-et
  3. Elküldi a $c \in \mathcal{C}^*$ értéket Bobnak
- **visszafejtő fázis**
  Amikor Bob megkapja a c értéket, akkor kiszámítja az $m \equiv c^d$, $1 \leq m < n$ értéket.

Blokkok kialakítása a nyílt szövegben

Hogyan garantáljuk azt, hogy a nyilvános szöveg egy $m < n$ numerikus értékkel legyen ábrázolva (kódolva)?

- Tegyük fel, hogy a nyílt szöveg eredetileg egy N elemű ABC feletti nyelven íródott
- Legyen l az az egyetlen természetes szám melyre $N^l < n < N^{l+1}$
- A szöveget bontsuk fel l-betűs blokkokra
- Minden blokknak megfelel egy l számjegyű szám az N alapú számrendszerben, feltéve, hogy az utolsó blokkot szükség esetén nullákkal egészítjük ki (jobbról)
- Mivel $N^l < n$, így minden nyílt szöveg megfelel $\mathbb{Z}/n\mathbb{Z}$ egy elemének, így a kriptoszöveg is $\mathbb{Z}/n\mathbb{Z}$ egy eleme lesz, azaz $n < N^{l+1}$ miatt egy $l+1$ jegyű számmal ábrázolható az N alapú számrendszerben.

Az RSA biztonsága

Támadási lehetőségek az RSA ellen:

- Az n faktorizációja
- Implementációs hiányosságokat kihasználó támadások
  - Közös modulus protokoll-hiba
  - Időméréses támadás
  - Áram-kriptoanalízis
- Támadások a kitevő ellen
  - A Coppersmith támadás
  - Erős Hastad támadás
  - Coppersmith rövid kitöltés támadása
  - Coppersmith támadása részlegesen ismert kulcs esetén
  - A Weiner támadás kis titkos kulcs esetén

Közös modulus protokoll-hiba

Példa

Tegyük fel, hogy Alice és Bob RSA-t használnak, és ugyanazt az n modulust választották, valamint az e_A és e_B nyilvános kulcsokat használják, melyekre $\gcd(e_A, e_B) = 1$. Tegyük fel, hogy Eve megszerez két kriptoszöveget, melyben valaki ugyanazt az üzenetet küldi Alice és Bob számára. Ekkor Eve képes megfejteni az üzenetet n faktorizációjának ismerete nélkül.

Tehát Eve rendelkezésére áll m^{e_A} és m^{e_B}. Mivel $\gcd(e_A, e_B) = 1$, így a kiterjesztett Euklideszi algoritmussal Eve kiszámol olyan u, v egész számokat, melyekre $ue_A + ve_B = 1$. Végül

$$m \equiv m^{ue_A + ve_B} \equiv (m^{e_A})^u (m^{e_B})^v \pmod{n}.$$

- Ez nem az RSA gyengesége, hanem egyszerű protokoll-hiba, ami elkerülhető, ha nincs két olyan résztvevője a kriptorendszernek, akik ugyanazt a modulust használják.

Moduláris gyorshatványozás – Emlékeztető

Adott $n \in \mathbb{N}$ és $d = \sum_j = 0^k d_j 2^j$, $d_j \in \{0, 1\}$. Célunk meghatározni $x^d \pmod{n}$ értékét.
Legyen $x_0 = 0$, $c_0 = 1$ $j = 0$, és hajtsuk végre az alábbiakat:

1. Ha $d_j = 1$, akkor legyen $c_j \equiv x_j \cdot x \pmod{n}$.
2. Ha $d_j = 0$, akkor legyen $c_j \equiv x_j \pmod{n}$.
3. Számítsuk ki $x_{j+1} \equiv c_j^2 \pmod{n}$ értékét.
4. Legyen $j := j + 1$. Ha j=k+1, akkor fejezzük be az algoritmust

$$c_k \equiv x^d \pmod{n}$$

érték visszaadásával, ellenkező esetben vissza az 1 pontba.

Időméréses támadás

- Feltesszük, hogy Eve ismeri a hardvert, és hogy a támadás előtt Eve lemérte r darab (kellően nagy számú) x_i kriptoszöveg esetén, hogy mennyi időbe telik a hardvernek $x_i^{d_i}$ kiszámítása a gyorshatványozás módszerével: T_i.
- Eve célja, hogy megtudja a d visszafejtő kulcsot. Mivel d páratlan, így a 0-adik bit $d_0 = 1$.
- Tegyük fel, hogy Eve már kitalálta $d_0, d_1, \ldots, d_{l-1}$ értékét.
- Mivel Eve ismeri a hardvert, így tudja, hogy mennyi időbe telik c_l kiszámítása a fenti algoritmusban (azaz c_{j-1} négyzetre emelése, és $d = 1$ esetén megszorzása x-szel): t_i.
- Kocher észrevette, hogy $d_l = 0$ esetén $\{T_i\}$ és $\{t_i\}$ függetlenek, míg $d_l = 1$ esetén $\{T_i\}$ és $\{t_i\}$ korreláltak.
- Eve a $\{T_i\}$ és $\{t_i\}$ korrelációját vizsgálva meg tudja állapítani d_l értékét.

Védekezési lehetőségek

1. Késleltetés beépítésével gondoskodhatunk arról, hogy egy moduláris hatványozás mindig ugyanannyi ideig tartson.
2. A második módszer neve "vakítás", és Rivest javasolta.
   - Mielőtt a c kriptoszöveget visszafejtenénk, választunk egy $r \in \mathbb{Z}/n\mathbb{Z}$ véletlen számot.
   - Kiszámítjuk $c' = c \cdot r^e$ értékét, ahol e a nyilvános (titkosító) kitevő.
   - Visszafejtjük a c' üzenetet: $m' = (c')^d$.
   - Kiszámítjuk $m = m' \cdot r^{-1}$ értékét.

Egyéb gyenge implementáción alapuló támadások

Áram-kriptoanalízis

Mivel multi-precíziós szorzás esetén a számítógép áramfelvétele megnő, így a gép áramfelvételének alapos elemzésével a visszafejtés során, Eve meghatározhatja d értékét.

Szemantikai biztonság

- Szemantikai biztonság alatt azt értjük, hogy az (n, e) nyilvános kulcs és c kriptoszöveg alapján semmilyen információhoz nem juthatunk az m nyílt szövegről.
- Az RSA alap verziója szemantikailag nem biztonságos, mert

$$\left(\frac{c}{n}\right) = \left(\frac{m^e}{n}\right) = \left(\frac{m}{n}\right)^e$$

könnyen kiszámolható.

Kis titkosító hatvány elleni támadások

A hatékonyság érdekében nagyon előnyös lenne az $e = 3$ választás, hiszen így egy üzenetet titkosítani mindössze egy négyzetre emelés és egy szorzás segítségével lehetne.

Példa

Alice m üzenetet akar küldeni 3 barátjának, akik mind az $e_i = 3$ titkosító hatványt használják, páronként relatív prím n_i modulusokkal, melyekre $m < n_i$ minden $i = 1, 2, 3$ esetén. Határozzuk meg m-et.

- A kínai maradék tételt alkalmazva, az $x \equiv c_i \pmod{n_i}$ rendszernek egyértelmű megoldása van modulo $n_1 n_2 n_3$.
- Az $m^3 < n_1 n_2 n_3$ nyilván ilyen megoldás, tehát $x = m^3$
- Az x-ből köbgyököt vonva megkaphatjuk m értékét.

Védekezési lehetőségek

1. Válasszunk nagyobb nyilvános kitevőt
   - Az igen realisztikus eset, hogy 3 résztvevőnek küldjük ugyanezt az üzenetet.
   - A fenti támadás általánosítható $e = r$ esetére és r darab üzenetre.
   - Ha $e = 65537 = 2^{16} + 1$ választással élünk, még mindig elég hatékony a titkosítás (16 négyzetre emelés, és egy szorzás), és nem valószínű, hogy 65537 résztvevőnek ugyanazt az üzenetet akarnánk küldeni.
2. Az üzenet egyik végéhez egy megfelelő hosszúságú véletlen bitsorozatot ragasztunk.
   - Elnevezés: "padding" vagy "salting the message"
   - Fontos, hogy minden résztvevő üzenetéhez más sorozatot használjunk.
   - Ez véd a fenti támadás ellen, de sajnos vannak erősebb támadások a kis méretű nyilvános kulcs ellen, amelyek nem védhetők ki ilyen módon!

A Coppersmith támadás

Tétel. (Coppersmith)

Legyen $n \in \mathbb{N}$ *összetett,*
$f(x) := a_d x^d + a_{d-1} x^{d-1} + \cdots + a_0 \in \mathbb{Z}[x], d \in \mathbb{N}$ *és tegyük fel hogy létezik* x_0*, melyre* $f(x_0) \equiv 0 \pmod{n}$ *és* $|x_0| < n^{1/d}$*. Ekkor* x_0 *a* d *és* $\ln n$ *értékének polinom idejében meghatározható.*

- Ez a tétel hatékony algoritmust szolgáltat egy f polinom modulo n vett mindazon x_0 gyökeinek meghatározására, melyekre $|x_0| < n^{1/d}$
- Így $x^e - c$ kis gyökei is meghatározhatók
- Ez a támadás szükségessé teszi, hogy az e kitevőt nagyra válasszuk.

Példa

Tegyük fel, hogy Alice a $B_1, \ldots, B_r$ résztvevőkkel kommunikál az $e < r$ titkosító kitevővel. Szeretné elküldeni mind az r partnernek az m üzenetet. A fenti támadási lehetőségek miatt minden üzenethez más és más blokkot ragaszt hozzá az

$$f_j(m) = 2^t j + m$$

függvénnyel, ahol t az m bináris hossza. Ezután minden B_j-nek elküldi az

$$f_j^e(m) \equiv (2^t j + m)^e \pmod{n_j}$$

üzenetet. Van-e esélye Eve-nek megfejteni az üzenetet?

Az erős Hastad támadás

Tétel. (Hastad)

Legyenek $n_1, \ldots, n_r \in \mathbb{N}$ páronként relatív prímek, $n_1 \leq n_j$ minden $j = 1, \ldots, r$. Legyenek $f_j(x) \in (\mathbb{Z}/n_j\mathbb{Z})[x]$ polinomok, melyek fokszáma legfeljebb l. Ha létezik olyan egyértelmű $m < n_1$, melyre $f_j(m) \equiv 0 \pmod{n}_j$ minden $j = 1, \ldots, r$ esetén, valamint $r > l$, akkor m hatékony módon meghatározható.

- Ez a tétel hatékony algoritmust szolgáltat egy polinom kongruenciarendszer megoldására, ha elegendő egyenletünk van
- Ha Eve az előző példában több mint e üzenetet elfog, akkor meg tudja határozni $m - et$
- A fenti tétel bármilyen fix f_j függvényekre működik
- Így a blokk hozzáragasztása csak akkor segíthet, ha az véletlen függvényekkel történik.

Példa

- Tegyük fel, hogy Alice Bobnak szeretne egy m üzenetet küldeni, melyet véletlen bitekkel kitölt.
- Mallory elfogja az üzenetet, és megakadályozza azt, hogy Bob megkapja az üzenetet.
- Alice, mikor nem kap választ Bob-tól, újra elküldi az m üzenetet, most egy másik véletlen bitsorozattal kitöltve.
- Mallory ezt is elfogja, és most már van két titkosított példánya m-ből, két különböző kitöltéssel.
- A következő tétel megmutatja, hogy tudja ebből Mallory meghatározni az m üzenetet.

Coppersmith rövid kitöltés támadása

Tétel. (Coppersmith's Short Pad Attack)

Legyenek $n \in \mathbb{N}$ egy N bites RSA modulus, melyhez az e titkosító kitevő tartozik, és legyen $l := [N/e^2]$. Tegyük fel, hogy $m \in (\mathbb{Z}/n\mathbb{Z})^*$ egy nyílt szöveg egység, melynek bináris hossza legfeljebb $N - l$. Legyenek $m_1 = 2^l m + r_1$ és $m_2 = 2^l m + r_2$, $r_1 \neq r_2$, $0 \leq r_1, r_2 < 2^l$. Legyen az m_i-nek megfelelő kriptoszöveg c_i. Ekkor n, e, c_1, c_2 ismeretében m hatékonyan meghatározható.

- Ha $e = 3$, akkor a fenti támadás működik, ha Alice olyan kitöltést használ, melynek hossza rövidebb az üzenet hosszának 1/9-ed részénél.
- Valóban, ha az üzenet hossza M, akkor $r_1, r_2 < 2^{M/9} < 2^{N/9} = 2^l$
- Az is világos, hogy hasonló támadás nem működik akkor, ha $e = 65337$ kitevőt használunk.

Coppersmith támadása részlegesen ismert kulcs esetén

Tétel. (Coppersmith's Key Exposure Attack)

Legyenek $n = pq$ egy l bites RSA modulus. Ekkor megadva d, első vagy utolsó $[l/4] + 1$ bitjét, Eve $e \log_2 e$ lineáris idejében rekonstruálni tudja d értékét.

- Így, ha $e < \sqrt{n}$, akkor d egy szakaszának ismeretéből a teljes d visszanyerhető
- Ez Coppersmith egy tételén múlik:

Tétel. (Coppersmith's Key Exposure Attack)

Legyenek $n = pq$ egy l bites RSA modulus. Ekkor megadva d, első vagy utolsó $l/4$ bitjét, Eve hatékonyan faktorizálni tudja n-et.

Wiener támadása kis titkos kitevő esetén

Tétel. (Wiener's Attack)

Legyenek $n = pq$, ahol p, q prímek, melyekre $q < p < 2q$ és $d < n^{1/4}/3$. Ekkor megadva az e nyilvános kulcsot, melyre $ed \equiv 1 \pmod{\phi(n)}$, a d értéke hatékony módon meghatározható.

- Ezt Wiener a lánctörtek klasszikus elméletét felhasználva igazolta
- Ha $ed = 1 + k\phi(n)$ valamely $k \in \mathbb{N}$ esetén, akkor a tétel feltételei mellett k/d az e/n lánctörtbe fejtésének egyik konvergense
- Így d meghatározásához elegendő a nyilvánosan ismert e/n néhány konvergensét meghatározni
- Tehát kis titkos kitevő használata ugyan hatékonyabbá teszi az alkalmazást, de a biztonság teljes elvesztésével jár
- Megmutatták, hogy az RSA nem biztonságos, ha $d < n^{0,292}$

Helyes RSA paraméter választás

Az előbbiekben láttuk, hogy a rossz implementáció és helytelen paraméter választás, hogyan szolgáltat lehetőséget a támadóknak. Most lássuk, hogyan érdemes a paramétereket választani:

- Válasszunk $p \neq q$ prímeket úgy, hogy:
  - p is és q is legalább 512 bites
  - p és q nincs túl közel egymáshoz abban az értelemben, hogy számtani és mértani közepük nincs közel egymáshoz (hogy n ellenálljon a Fermat négyzetek különbsége módszerével történő faktorizációnak)
  - A $p-1, p+1, q-1, q+1$ mindegyike tartalmazzon nagy prímfaktort (hogy n ellenálljon a $p-1$ és $p+1$ faktorzációs eljárásoknak)
- Válasszuk úgy az (e, d) exponens párt, hogy e és d közül egyik se legyen kicsi, sőt $d > n^{0.292}$