Naptári tétel

Naptárak megtekintése

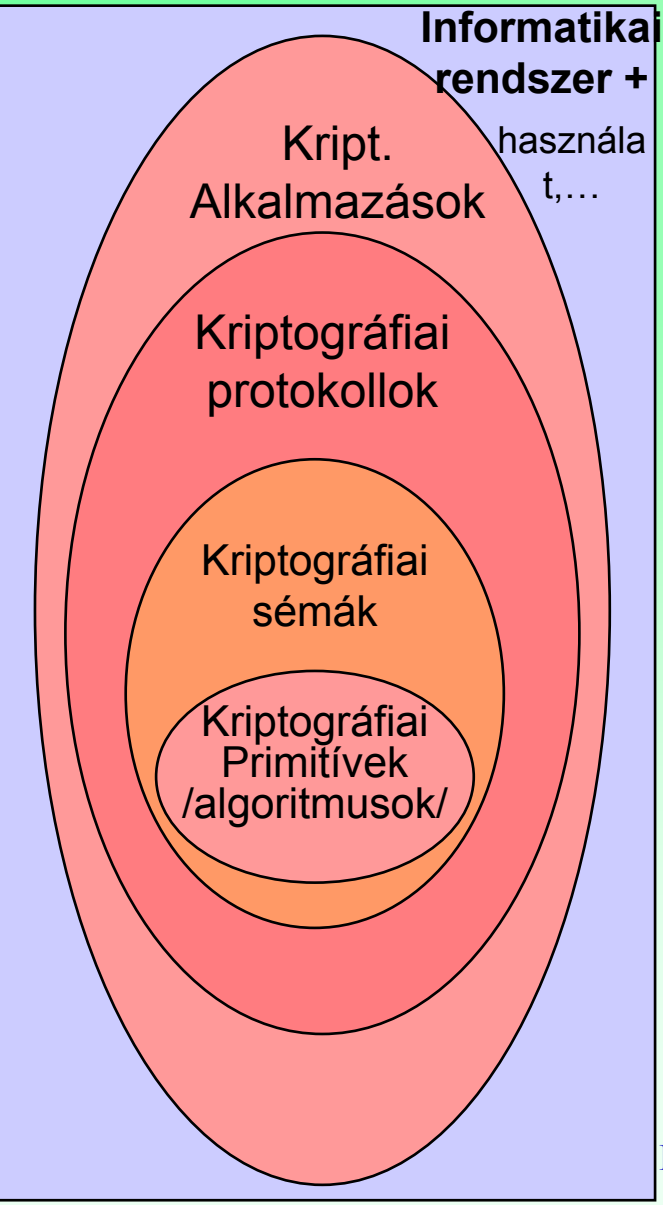
Naptár	hspbp events
Cím	cryptonite: RSA algoritmus: a jó paraméterek jellemzői
Recurrence	This event is not recurrent
Kezdés	Péntek 21 Február, 2014 19:00:00 CET
Befejezés	Péntek 21 Február, 2014 20:55:00 CET
Leírás	*előadó:* *Szabó István* *ELTE, egyetemi docens*

szaboi@cs.elte.hu
ELTE TTK Valószínűségelméleti és Statisztika Tanszék

A kriptográfiát fejlesztői, tesztelői, kódelemzői szemmel nézve fontos ismerni azokat az ökölszabályokat, amelyek alapján a függvényeket biztonságosan meg lehet hívni, a jó prímeket ki lehet választani. Pontosan ezen ökölszabályok kerülnek bemutatásra az RSA algoritmus esetén Szabó István előadásában:

- *mik a jó prímek tulajdonságai, mennyire legyenek egymástól távol, milyen legyen az arányuk (pl. max. p/q arány esetén a 2 kellően magas-e),*
- *prímtesztnél hány ciklusig kell azt futtatni (pl. 10.000 elegendő-e),*
- *kell-e aggódni a beégetett kicsi "e" értékek (pl. 3) miatt chipkártyás kulcsgenerálásoknál,*
- *a kulcsokat milyen méretű adatra kell legalább alkalmazni (pl. 160 bites SHA-1 vagy 128 bites MD5 lenyomatok aláírása, 128 bites AES kulcsok rejtjelezése most is megy a gyakorlatban)*

Kriptográfiai rendszer

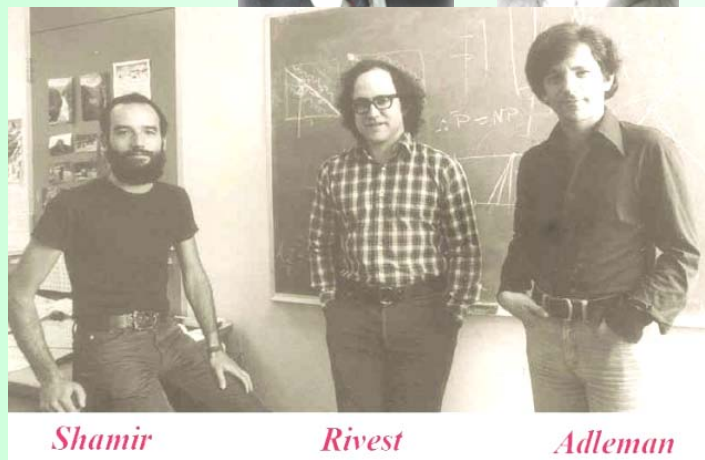


W. Diffie- M.E. Hellman:

New Direction in Cryptography, IEEE Transactions on Information Theory, Vol IT-22, No6, nov.1976.

1977

Rivest, Shamir, Adleman



Shamir *Rivest* *Adleman*

RSA matematikai alapja:

Tétel (Fermat: 1601-1665)

Bármely ***p*** prímszámra és bármely ***a*** poz. egész számra, melyre lnko(a,p)=1

$$a^{p-1} \equiv 1 \pmod{p}$$

Tétel (Euler:1707-1783):

Bármely ***n*** és ***a*** poz. egész számpárra, melyekre lnko(a,n)=1

$$a^{\varphi(n)} \equiv 1 \pmod{n}$$

Ha $n=p*q \rightarrow \varphi(n)=\varphi(p)*\varphi(q)$

Pl. ha p=3 q=5 n=15 φ(n)=8

Hatvány (x) →	1	2	3	4	5	6	7	8
7^x mod(15) →	7	4	13	1	7	4	13	1

RSA titkosítás

Tétel (Euler): **lnko(n,a)=1** → $a^{\varphi(n)} \equiv 1 \pmod n$ bármely poz. **a** egész számra

- Kulcsgenerálás /**prekondícionálás** A oldalon/:
  - ***A* választ** két „nagy” prímszámot: *p és q*,
  - ***A* számol**: n = *pq*, $\varphi(n) = (p-1)(q-1)$
  - ***A választ***: *e*, $1 < e < \varphi(n)$, melyre lnko($\varphi(n)$, *e*) = 1
  - ***A* számol** d számot, melyre $ed \equiv 1 \bmod \varphi(n)$

    Ezért kell lnko($\varphi(n)$, *e*) = 1

    d számolható a kiterjesztett euklideszi algoritmussal, vagy: *pl.* ha $d \equiv e^{\varphi(n)-1} \bmod n \rightarrow e*d \equiv e^{\varphi(n)} \equiv 1 \bmod n$
  - *A* nyilvános kulcs (***n, e***); *A* titkos /magán/ kulcsa: ***d***
- **Titkosítás** /A-nak m(=$m_1, m_2, \ldots, m_k$, $m_i < n$) küldött üzenetre/: $c_i = m_i^e \bmod n$
- **Megoldás** /A megoldja a titkosított üzenetet/: $m_i = c_i^d \bmod n$

Aláírásnál: $s=(h(m))^d \bmod n$

Ellenőrzés: $h(m)=?s^e \bmod n$

$$D_d(E_e(m_i)) = (m_i{}^e)^d = m_i^{ed} = m_i{}^{t\varphi(n)+1} = (m_i{}^{\varphi(n)})^t \times m_i = 1 \times m_i = m_i \bmod n$$

Miért „hatékony”:

Pl. ha $e=11_{DEC}=1011_2$: ki kell számolni: $a_2=m^2$, $a_4=a_2{}^2=(m^2)^2$, $a_8=a_4{}^2=(m^4)^2$, és $m^{11}=m*a_2*a_8$

Hatványozás „gyors”: k bites (***n***,***e***,***d*** esetén) $\mathbf{\log_2 k}$ db. négyzetre emelés, $\mathbf{\log_2(k/2)}$ db. szorzás /+mod. számítások/

Támadás „lassú” (a titkos ***d*** kipróbálása): $c^d \equiv m \pmod n$ $\mathbf{2^k}$ próba – azaz a támadáshoz képest gyors titkosítás, megoldás

Léteznek megfelelő **„gyors” algoritmusok** a paraméterek: nagy p, nagy q **generálására** (n=p*q) számolására
Nem ismertek kellően **gyors algoritmusok** a nyilvános n értékből p,q meghatározására /**faktorizációra**/

RSA-768 bit/232 =
123018668453011775513049495838496272077285356959533479219732245215172640050726365751874520219978646938995647494277406384592519255732630345373154826850791702612214291346167042921431160222124047927473779408066535141959745985690214341 3

RSA-768 =
p=33478071698956898786044169848212690817704794983713768568912431388982883793878002287614711652531743087737814467999489 ×

q=367460436667995904282446337996279526322791581643430876426760322838157396665112792333734171433968102700927987363089 17

2009.12.12. /50.000$

RSA-2048 bit=
25195908475657893494027183240048398571429282126204032027777137836043662020707595556264018525880784406918290641249515082189298559149176184502808489120072844992687392807287776735971418347270261896375014971824691165077613379859095700097330459748808428401797429100642458691817195118746121515172654632282216869987549182422433637259085141865462043576798423387184774447920739934236584823824281198163815010674810451660377306056201619676256133844143603833904414952634432190114657544454178424020924616515723350778707749817125772467962926386356373289912154831438167899885040445364023527381951378636564391212010397122822 120720357

RSA

- **Előnyei**:
  Könnyen megérthető algoritmus;
  Biztonsága klasszikus (faktorizációs) problémára vezethető vissza: IFP; RSA IFP
- **Hátrányai:**
  Ha n(=p*q) nagy,
  nagyon számításigényes (lassú – a szimmetrikus kulcsú titkosításokhoz képest)

•Gyorsítási ötletek

1. „kis” e

2. „kis” d

3. Gyors szorzási algoritmusok

Karatsuba Suppose we have two positive integers a and b, each of length at most ℓ, such that $a = a_1 2^k + a_0$ and $b = b_1 2^k + b_0$, where $0 \le a_0 < 2^k$ and $0 \le b_0 < 2^k$. Then

$$ab = a_1 b_1 2^{2k} + (a_0 b_1 + a_1 b_0) 2^k + a_0 b_0.$$

4. Gyorsabb hatványozás: Sliding windows

Azonos (prekondícionált) bitsorozatok (ablakok) keresése
Pl: $e=45 \rightarrow g^{45}=(g^5)^8 g^5 =(g^{101\ 2})^{10002} * g^{1012}$

$T(l) \le c*l^{\log 2(3)} \approx c* l^{1,585} \rightarrow c_1*l^2$ helyett,
ha l=2.048 c*177.197 c_1*4.194.304

$e=11749=(101101111100101)_2$ 3db. 101, … van
Az OpenSSL 1024 bithez 5 bites ablakot használ

5. Montgomery Modular reduction

6. Gyorsabb megoldási algoritmus:

Chinese Remainder Theorem

Kb. 4-szeres gyorsítás

RSA CRT Algoritmus

1. Megoldónak ismert **C**, **d**, **p**, **q**, **N**=p*q (és **e**)
2. Megoldáskor ki kell számítani: $C^d \pmod N$
3. Előszámítás: $\mathbf{d_p} = d \pmod{(p-1)}$ és
   $\mathbf{d_q} = d \pmod{(q-1)}$
4. Előszámítás **a** és **b** melyekrre
   $\mathbf{a} = 1 \pmod p$ és $a = 0 \pmod q$
   $\mathbf{b} = 0 \pmod p$ és $b = 1 \pmod q$

Megjegyzés: $a=k*p+1=l*q$: $\mathbf{l*q-k*p=1}\rightarrow$
k,*l* számolható a kiterjesztett euklideszi algoritmussal

Példa: N = 33, p = 11, q = 3, d = 7 (e=3)

Előszámítás: $d_p = 7 \pmod{10} = 7$ és
$d_q = 7 \pmod 2 = 1$

→ a = **12** és b = **22**

Tegyük fel: C = 5

Ki kell számolni: $C^d = 5^7 \pmod{33}$

$5^7 = 5*5^2*(5^2)^2 \equiv 125*(-8)^2 \equiv (125-99=)26*(64-33=)31 \equiv (-7)*(-2) = 14$

Adott C-re számoljuk:

$$C_p = C \pmod p \quad \text{and} \quad C_q = C \pmod q$$

$$x_p = C_p^{d_p} \pmod p \quad \text{and} \quad x_q = C_q^{d_q} \pmod q.$$

$C_p = 5 \pmod{11} = 5$ és $C_q = 5 \pmod 3 = 2$

$x_p = 5^7 = \mathbf{3} \pmod{11}$, $x_q = 2^1 = \mathbf{2} \pmod 3$

Végeredmény:

$$C^d \pmod N = (ax_p + bx_q) \pmod N$$

Kapjuk: $5^7 = \mathbf{3} \cdot \mathbf{12} + \mathbf{22} \cdot \mathbf{2} = 14 \pmod{33}$

prekondícionálás után, kapott üzenetenként kell számolni: **modp** és **modq** és két szorzást **mod N**
modN hatványozás helyett

Az RSA algoritmus analízise

a) **Felhasználói oldalon**

Hogyan lehet megfelelő (?) paramétereket, pl. kellően nagy prímeket generálni;

b) **Az RSA kripto-analízise /támadása/**

Matematikai módszerek

Egyéb módszerek

n=p*q faktorizálása

Lehetséges-e $\varphi(n)$, vagy d faktorizációnál gyorsabb meghatározása

Lehetséges-e faktorizáció, $\varphi(n)$ vagy d meghatározása nélkül az üzenetek megismerése vagy a digitális aláírás hamisítása

Hogyan találhatunk „véletlen” nagy prímeket?

Determinisztikus módszerek

Biztosan megállapítják a számról, hogy prímszám-e, lassúak

Legegyszerűbb eljárás, ha a számot sorban elosztjuk a gyökénél nem nagyobb egész számokkal vagy prímekkel:

Pl. 1024 bites számra $\sqrt{2^{1024}} = 2^{512}$ -ig .

A prímszámtételből ($\pi(x) \approx \frac{x}{\ln x}$ ha $x \to \infty$)

Kapjuk, hogy

kb. $\frac{2^{512}}{\ln 2^{512}} \approx (3{,}788 * 10^{151}) \approx 2^{503}$ osztás

1 sec	1 nap = 3600*24	ha 1 utasítás kb. 4 órajel:
2 GHz ~ $2*2^{30}$	$2*2^{30}*8{,}6*10^4$ ~ 2^{46}	2^{44}

1000 PC-vel ≈ 2^{54} elemi művelet/nap

Valószínűségi tesztek

nem döntik el teljes biztonsággal, hogy a kérdéses szám prím-e, de a tévedés valószínűsége a teszt többszöri végrehajtásával, tetszőleges küszöbérték alá csökkenthető,

gyorsak

Tétel (Fermat): (p,a)=1, p prím → $a^{p-1} \equiv 1 \pmod p$

bármely ***a*** poz. egész számra

Fermat teszt: **ha létezik a: $a^{n-1} \not\equiv 1 \pmod n$ → n összetett szám.** /Ekkor **a** az n összetettségének Fermat tanúja./

Tétel: Ha van ***a***: (***a***,n)=1 és ***a*** Fermat tanú, akkor [1,n) legalább fele Fermat tanú

Bizonyítás: legyen ***a*** tanú, c_i-k különböző nem tanúk

$$a^{n-1} \not\equiv 1(n),\ c_1, c_2 .. c_s \text{ nemtanúk} \Rightarrow c_i^{n-1} \equiv 1(n)$$

$$\Rightarrow (ac_i)^{n-1} \equiv a^{n-1} c_i^{n-1} \equiv a^{n-1} \not\equiv 1(n) \Rightarrow ac_i \text{ tanú.}$$

Mivel c_i-k mind különbözők és (a,n)=1 →

→ ac_i-k is mind különbözőek.

Azaz: ha van tanú, jó az esély (legalább 50%) találni egyet.

Fermat sejtés: $F_n = 2^{2^n}+1$ mindig prím: (n,F) = (0,3), (1,5), (2,17),(3,257),(4,65537) Mai sejtés: nincs több F prím

G. A. PAXESON 1961-ben ezzel biz: $F_{13} = 2^{2^{13}}+1$ nem prím, mivel : $3^{F_{13}} = 3^{2^{2^{13}}+1} \equiv / \equiv 3$

Megfelel-e ez prímtesztnek (ha m-nek nincs Fermat- tanúja) prímtesztnek?

Carmichael számok

Definíció: Carmichael-szám az olyan összetett szám, amelynek nincs tanúja. (***n*** összetett szám Carmichael szám iff $a^n \equiv a \pmod n$ bármely ***a*** egészre)

- Példa: 561 = 3 x 11 x 17

561 C szám bizonyítása:

$561 = (3)(11)(17)$. But for any a,

$$a^{561} = (a^2)^{280}(a) \equiv a \pmod 3$$

$$a^{561} = (a^{10})^{56}(a) \equiv a \pmod{11}$$

$$a^{561} = (a^{16})^{35}(a) \equiv a \pmod{17}.$$

Megjegyzés:
Nem kell minden a-ra végigpróbálni
/ha n-et tudjuk faktorizálni/:

Következmény: A Fermat- teszt nem jó prímszámok kereséséhez

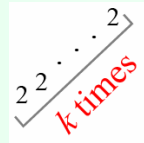
Rabin-Miller teszt

Fermat teszt alapja:

$a^{p-1} \equiv 1 \pmod p$ /lnko(a,p)=1/

Miller-Rabin teszt alapja:

ha $a^2 \equiv 1 \pmod p$, akkor $a \equiv \pm 1 \pmod p$,



Legyen a tesztelendő n számra : $n-1 = 2^k * r$, ahol r ptl. $a^{n-1} = a^{r*2^k} = \left[a^r\right]^{2^k} = \left[a^r\right]$

$$n / a^{n-1} - 1 = (a^{2^k * r} - 1) = (a^{2^{k-1} * r} + 1)(a^{2^{k-1} * r} - 1) = (a^{2^{k-1} * r} + 1)(a^{2^{k-2} * r} + 1)\ldots(a^r + 1)(a^r - 1)$$

Ha n prím, osztja valamelyik tagot.

Miller-Rabin teszt: Egy véletlenül választott n páratlan szám prímszám-e:

Legyen $n-1 = 2^k * r, r$ ptl.

Válasszunk egy $1 < a < n$, $\text{lnko}(a,n) = 1$ számot.

Ha $a^{n-1} = a^{2^k * r} \neq 1 \bmod n$, akkor n összetett szám, és *a* az n összetettségének tanúja.

Ha $a^{2^i * r} = 1 \bmod n$, de $a^{2^{i-1} * r} \neq \pm 1 \bmod n$, akkor az n összetett szám, és *a* az n összetettségének tanúja.

Ha i - t csökkentjük i = k, k - 1,...,1 - ig, és egyik i - re sem lesz *a* n tanú,

ekkor a teszt valószínűsíti, hogy **n** prímszám!

Ha minden ***a***-ra ***n*** átmegy az MR teszten, akkor ***n*** prímszám /ez a tulajdonsága már csak a prímeknek van/

Bizonyítás: Ha $n=n_1*n_2$, $n_1,n_2>1$ ptl. és $\text{lnko}(n_1,n_2)=1$ →

→ az $x*n_1+1=y*n_2-1$ diofantoszi egyenlet megoldható x,y-ra

Ekkor legyen $b=x*n_1+1=y*n_2-1 \rightarrow (b-1)*(b+1)=b^2 \equiv 1 \pmod n$ de $b \equiv \pm 1 \pmod n$ mégsem teljesül:

$n_1/(b-1)$ és $n_2/b+1$ → n nem osztja b-t!, azaz

ha n összetett szám, akkor létezik olyan b: $b^2 \equiv 1 \pmod n$, hogy b-re nem teljesül a $b \equiv \pm 1 \pmod n$

Nyilván, ha p prím, nincs MR tanúja.

A Miller-Rabin teszt műveletigénye: $o(\log_2^3 n)$

Rabin-Miller teszt

Ha n összetett szám, könnyű bizonyítani (a Fermat teszthez hasonló technikával), hogy az [1,n) intervallumnak legalább a fele MR tanú (ld. pl. T.H.Cormen, C.E.Leiserson, R.L.Rivest: Algoritmusok c. könyv: 33.38 Tétel)

Ha n összetett szám, **nagy (legalább ¾) valószínűséggel** találunk **a** tanút az összetettségére:

Bizonyítás, pl.: http://math.mit.edu/classes/18.783/LectureNotes13.pdf

Theorem 13.8 (Monier–Rabin, 1980). *Let N be an odd composite integer. Then, the probability that a random integer $a \in [1, N-1]$ is a witness for N is at least $3/4$.*

A fentinél sokkal erősebb állítás is igaz:

Theorem 13.11 (Damgård-Landrock-Pomerance). *Let N be a random odd integer in $[2^{k-1}, 2^k]$. Let a be a random integer in $[1, N-1]$. Then, if a is not a witness for N, then*

$$\Pr[N \text{ is prime}] \geq 1 - k^2 \cdot 4^{2-\sqrt{k}}.$$

Bizonyítás: Damgárd, P. Landrock, and C. Pomerance, Average case error estimates for the strong probable prime test, 1993: http://www.math.dartmouth.edu/~carlp/PDF/paper88.pdf

MR Teszt 561-re (Carmichael szám): **$561 - 1 = 35 \times 2^4$, a=2-re**

Initialization:	$T = 2^{35} \bmod 561 = 263 \bmod 561$		
$k = 1$:	$T = 263^2 \bmod 561 = 166 \bmod 561$		
$k = 2$:	$T = 166^2 \bmod 561 = 67 \bmod 561$		
$k = 3$:	$T = 67^2 \bmod 561 = +1 \bmod 561$	↑	→ **a composite**

Prímkeresés:

- Választunk nagy véletlen ptl. Számot: ***n***,
- leosszuk az első x prímmel, ha nem bukik el
- MR teszt
  - választunk ***a***-t (1,n) között, megnézzük: (a,n)=1 teljesül-e /euklideszi alg./
    - ha nem: n összetett szám;
    - ha igen, elvégezzük az MR tesztet
  - Új ***a***-t választunk, amíg a kellő hibaküszöböt elérjük. L

Annak valószínűsége, hogy egy véletlenül választott x bites szám prím legyen:

$$\frac{\pi(2^x)-\pi(2^{x-1})}{2^x-2^{x-1}} \approx \frac{\frac{2^x}{\ln 2^x}-\frac{2^{x-1}}{\ln 2^{x-1}}}{2^{x-1}} = \frac{2}{x*\ln 2}-\frac{1}{(x-1)*\ln 2} = \frac{2x-2-x}{x*(x-1)*\ln 2} \approx \frac{1}{x*\ln 2}$$

n bitben:	512	1024	2048
prím valószínűsége:	0,002818	0,001409	0,000704
minden x-edik véletlen:	355	710	1 420

	Time / # Tested		
	256 bits	512 bits	1024 bits
MillerRabin(20)	1.75 / 94	36.5 / 371	210 / 504

http://teal.gmu.edu/courses/ECE543/project/slides_1999/dong.pdf

Az RSA algoritmus analízise

a) Felhasználói oldalon

Hogyan lehet megfelelő (?) paramétereket, pl. kellően nagy prímeket generálni;

b) Az RSA kripto-analízise /támadása/

- Matematikai módszerek
  - n=p*q faktorizálása
    - *Rossz paraméter-választás esetén*
      - I p-q I „kicsi” $n^{1/2}$-től próba
      - Pl. programhiba, vagy csapda /ld. később/
      - Pl. p-1, q-1-nek nincs „nagy” prímosztója,…
      - „Kicsi” d
    - *Jó paraméter-választás esetén*
      - Mekkora n-et tudunk faktorizálni? (n=p*q)
  - Lehetséges-e $\varphi(n)$, vagy d faktorizációnál gyorsabb meghatározása
    - **Állítás**: $\varphi(n)$ /és (e,n)/ ismeretében n hatékonyan faktorizálható.
    - **Állítás:** d /és (e,n)/ ismertében n nagy valószínűséggel hatékonyan faktorizálható
  - Lehetséges-e faktorizáció, $\varphi(n)$ vagy d meghatározása nélkül az üzenetek megismerése vagy a digitális aláírás hamisítása
    - *Pl.*:
      „kicsi” e;
      „kicsi” a lehetséges üzenetek tere;
      stb.
- Egyéb módszerek

Hibás paraméterválasztások esetén működő támadások

John Pollard-féle „p-1” faktorizációs módszer:1974

p-1 vagy q-1-nek nincs „nagy” prímosztója

1. **Állítás**: Ha ismerjük p-1 egy ν többszörösét, RSA IFP megoldható (p visszaállítható)

$\forall a,(a,p)=1 \Rightarrow a^{\nu} \equiv 1\,(p) \Rightarrow r = \mathrm{lnko}(a^{\nu}-1, n)$ *osztója* n-nek, ezért r egy jelölt p vagy q-ra.

($\nu = k*p*q$, $p*q$)

2. **Állítás**: Ha p-1 –nek csak „kis” prímosztói vannak, akkor meghatározható p ismerete nélkül is p-1 egy *ν* többszöröse: *ν=k*(p-1).*

ν "kitalálása": Válasszunk egy B korlátot, erre számoljunk egy

lehetséges *ν* értéket: $\nu = \prod_{\substack{q\ prím \\ q^{k(q)} \le B < q^{k(q)+1}}} q^{k(q)}$ /minden q < B prímre a max k(q)-ra/

Ha $p-1 = \prod_i q_i^{k_i}$ és $q_i^{k_i} < B$ minden i-re $\Rightarrow$ ν többszöröse p-1-nek.

Algoritmus: *Válasszunk B-t, ehhez számoljuk ν –t $q^k<B$; $k=\lfloor \log_q B \rfloor$*

Választott a-ra számoljuk $r=\mathrm{lnko}(a^{\nu}-1,n)$ –t, *megnézzük: ha r nem osztja n-et, új B-t választunk.*

Hogyan generáljunk?

A Sophie Germain prime is a prime p such that $2p+1$ is also prime.

Az első néhány Sophie Germain-prím: 2,3,5,11,…, 1439, 1451, 1481, 1499, 1511, 1559...

Algorithm	Running time
Pollard's $p-1$ algorithm	$\mathcal{O}(B \log B (\log n)^2 + (\log n)^3)$

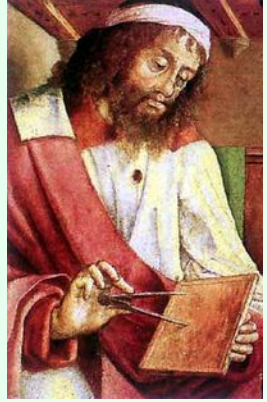
Let $\pi^*(x)$ denote the number of Sophie Germain primes up to x.

Conjecture 5.24. *We have*

$$\pi^*(x) \sim C\frac{x}{(\log x)^2}, \qquad C \approx 1{,}32$$

*John Pollard-féle „p-1” faktorizációs módszer **javítása**: Ha két határérték van: $B_1 << B_2$, csak egy nagy prímosztója van n-nek B_1 és B_2 között, a többi B1-nél kisebb. (pl. $B1=10^6$, $B2=10^{10}$ /Atkin, Rickert, 1984/)*

*Williams **p+1 módszer** /1982/: a p-1 módszer egy variánsa ún. Lucas sorozatokkal*



Euklidészi algoritmus /i.e. IV. század/:
két természetes szám legnagyobb közös osztójának meghatározására.
Ha a és b pozitív egész számok:

$a = b * q + r \rightarrow lnko(a, b) = lnko(b, r)$,

így a problémát visszavezeti két kisebb szám legnagyobb közös osztójának meghatározására.
Folytatva az eljárást, az utolsó, 0-tól különböző maradék a legnagyobb közös osztó

- $a = b \cdot q_1 + r_1$
- $b = r_1 \cdot q_2 + r_2$
- $r_1 = r_2 \cdot q_3 + r_3$
- $\dots$

ahol

$|b| > r_1 > r_2 > r_3 > \dots \geq 0$

A maradék véges sok lépés után nulla lesz,

$r_{n-1} = r_n \cdot q_{n+1}\ (+0)$

Példa:
lnko(2205,252):

	q_i	r_i	
2205=252*8+189,			és 63/2205 és egyben 63/2525 lnko is
252=189*1 + 63,			és 63/252 is (egyben lnko)
189= 63*3 + 0, így			63/189 (és egyben lnko)

Általánosított euklideszi algoritmus:

adott a,b>0 egészek, határozzuk meg x,y-t: a***x**+b***y**=z=lnko(a,b)

A fenti példában: a=2205, b=252, így 189=3*63 → 252=(3+1)*63 →
2205=(8*4+3)*63: **x=35; y=4; z=63** (=lnko(2205,252))

Euklidészi algoritmus sebessége

Fibonacci (≈1170-1250) számok:

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, …

$$F_n = \begin{cases} 0, & \text{ha } n = 0 \\ 1, & \text{ha } n = 1 \\ F_{n-1} + F_{n-2}, & \text{ha } n > 1 \end{cases}$$

A szomszédos Fibonacci-számok aránya (F_{n+1}/F_n) Φ-hez,
az aranymetszés értékéhez tart: $\Phi = \frac{1+\sqrt{5}}{2} \approx 1,\ 618\ 0$

Binet formula: $F_n = \frac{(1+\sqrt{5})^n - (1-\sqrt{5})^n}{\sqrt{5} \cdot 2^n} \approx \frac{1}{\sqrt{5}}\left[(\frac{3,24}{2})^n - (\frac{-1,76}{2})^n\right]$

$$x = \lim_{n\to\infty} \frac{F_{n+1}}{F_n} = \lim_{n\to\infty} \frac{F_n + F_{n-1}}{F_n} = 1 + \lim_{n\to\infty} \frac{F_{n-1}}{F_n} = 1 + \frac{1}{\lim_{n\to\infty} \frac{F_n}{F_{n-1}}} = 1 + \frac{1}{x} \rightarrow x^2 = x + 1$$

Tétel: Lamé tétele

Ha az euklideszi algoritmusban $a > b \geq 0$ és $b < F_{k+1}$ valamely $k > 0$-ra, akkor a rekurziós hívások száma kevesebb, mint k.

$$F_k \approx \frac{1}{\sqrt{5}}\Phi^k,\ \Phi \approx 1,618,\ \log_{10}\Phi \approx 0,2089,\ \frac{1}{\log_{10}\Phi} \approx 4,78 \approx 5$$

$$\log_{10} F_k \approx k \cdot \log_{10}\Phi - \log_{10}\sqrt{5}$$

$$k \approx \frac{1}{\log_{10}\Phi} \cdot \log_{10} F_k + \frac{\log_{10}\sqrt{5}}{\log_{10}\Phi} \approx 5 \cdot \log_{10} F_k \approx 5 \cdot \log_{10} b$$

Lamé tételének következménye: Ha $F_k \leq b < F_{k+1}$, akkor

a rekurziós hívások száma < k, és k ≈ b 10-es számrendszerbeli jegyeinek 5-szöröse.

Pl. ha $b \approx 10^{100}$, akkor a rekurziós hívások száma kb. 500.

Hibás paraméterválasztások esetén működő támadások

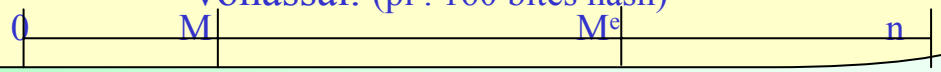
Kicsi e kódoló kulcs választása

Előny: (kisebb hatványra kell emelni) Ettől az RSA alapját képező *nehéz probléma megoldása nem könnyebb*, azonban a titkosított (x^e mod n) üzenetből x kiszámítása könnyebb lehet:
e=3 választás kockázatai:

Biztonságosnak tekinthető az
$e = 65537 = 2^{16} + 1$ választás, ami már elegendően nagy ahhoz, hogy ezek a támadások ne működjenek;

2-es számrendszerbeli alakjában mindössze 2 darab 1-es található, így x^e (mod n) számítása hatékony.

a) **„kicsi M”**-re egyszerű (nem mod) köbgyök-vonással. (pl . 160 bites hash)



b) „Hastad üzenetszórásos támadása”
Pl. ha e=3 és **3 helyre küldik el ugyanazt az üzenetet**: $C_1=M^3$ (n_1), $C_2=M^3$ (n_2), $C_3=M^3$ (n_3), ha n_i-k páronként relatív prímek, a CRT (kínai maradéktétel) alapján az üzenethez kiszámítható $C= M^3$ ($n_1*n_2*n_3$) és mivel $M< n_i$, $M^3< n_1n_2n_3$, így M köbgyök-vonással megkapható.

Biztonságosnak tekinthető a
$d>n^{1/2}$ választás

„Kicsi” d választása

Tétel (Wiener, 1990): *Ha $n=pq$ és $q<p<2q$ valamint $d<1/3*n^{1/4}$, akkor < n, e> ismeretében n hatékonyan faktorizálható*

http://www.iacr.org/archive/pkc2006/39580001/39580001.pdf

Attack for $d_p, d_q < \min\left\{\frac{1}{4}\left(\frac{N}{e}\right)^{\frac{2}{5}}, \frac{1}{3}N^{\frac{1}{4}}\right\}$

$d_p = d \bmod p-1$ and $d_q = d \bmod q-1$ both are small.

c) Coppersmith,Franklin,Reiter **related message attack**
Pl. ha e=3 és m, valamint m+1 is kódolásra kerül:
$c_1=m^3$ (n), $c_2=(m+1)^3$ (n), bár az egyenlet (köb gyök) megoldása nehéz, számoljuk:

$$c_2 = (m+1)^3 = m^3 + 3m^2 + 3m + 1 = c_1 + 3m^2 + 3m + 1$$

$$\frac{c_2 + 2c_1 - 1}{c_2 - c_1 + 2} = \frac{(m+1)^3 + 2m^3 - 1}{(m+1)^3 - m^3 + 2} = \frac{3m^3 + 3m^2 + 3m}{3m^2 + 3m + 3} = m$$

Általánosítás: $m_2=\alpha m_1+\beta$, (α,β) ismert , valamint e>3 esetére is

Hibás paraméterválasztások esetén működő támadások

Az ***m*** üzenet ún. fix pont:

$$c = m^e = m \pmod{n}$$

Az **üzenetek megismerése** faktorizáció, $\varphi(n)$ vagy d meghatározása nélkül

Backgrounds

```
prime numbers and φ(n):
                    p = 13                          (priva
                    q = 11                          (priva

                    n     = p * q = 143             (publi
                    φ(n) = (p - 1) * (q - 1) = 120

coprime number to φ(n):
                    e = 7                           (publi
                    e * d   ≡ 1 (mod φ(n))
                    7 * 103 = 721 = 1 + 6 * 120
                    d = 103                         (priva

public key:
                    n = 143
                    e = 7

private key:
                    n = 143
                    d = 103

operations:
                    data      = 3
```

p=13	q=11	
N=143, fi(N)=120		
fix pontok száma:	e-k száma:	Összes üzenet %-ban
9	6	6,3
15	6	10,5
21	6	14,7
33	2	23,1
39	6	27,3
55	2	**38,5**
77	2	**53,8**
143	1	**100,0**

Blakley-Borosh fix pont tétel /1979/: #(fix pont) = [1+lnko(e-1,p-1)]*[1+lnko(e-1,q-1)]; ha $n>n_0$ küszöbértéknél

Bizonyítást ld. pl.: http://www.inf.elte.hu/karunkrol/digitkonyv/Jegyzetek2010/A_rejtjelezes_nehany_kerdese.pdf

Extrém eset: $e=\varphi(n)+1$, ekkor az összes üzenet fix pont*!!!*

Pl. ha $e=2^k+1$, és p-1=2*P, q-1=2*Q, P, Q ptl, akkor a fix pontok száma minimális!

Hibás paraméterválasztások esetén működő támadások

SIMMONS iterációs támadás:

$c = m^e$ *(modn), n, e, c ismert*

Számoljuk ki:

$$c^e, c^{e^2}, c^{e^3}, \ldots, c^{e^j}, \text{ amíg}$$

$$c^{e^j} = c, \quad \text{ekkor} \quad c^{e^{j-1}} = m$$

Az **üzenetek megismerése** faktorizáció, $\varphi(n)$ vagy d meghatározása nélkül

Backgrounds

RSA

prime numbers and φ( ):

```
p = 13                          (private)
q = 11                          (private)

n    = p * q = 143              (public)
φ(n) = (p - 1) * (q - 1) = 120
```

coprime number to φ(n):

```
e = 7                           (public)
e * d   ≡ 1 (mod φ(n))
7 * 103 = 721 = 1 + 6 * 120
d = 103                         (private)
```

public key:

```
n = 143
e = 7
```

p=13	q=11
N=143, fi(N)=120	

e=7	message / Iterációszám:	e=11	message / Iterációszám:	e=13	message / Iterációszám:
	6: 4 db		23: 2 db		34: 1 db
	11: 2 db		54: 2 db		35: 4 db
	20: 4 db		64: 1 db		49: 4 db
	63: 4 db		86: 2 db		59: 4 db
	64: 4 db		125: 2 db		62: 4 db
	67: 2 db		126: 2 db		75: 4 db
	69: 4 db		127: 2 db		78: 1 db
	106: 4 db		133: 2 db		120: 1 db
	113: 4 db		138: 2 db		123: 4 db
	127: 4 db		142: 1 db		137: 4 db

Superencryption attack

Simmons, Norris, 1977

Typically, number of iterations very large if p and q chosen at random

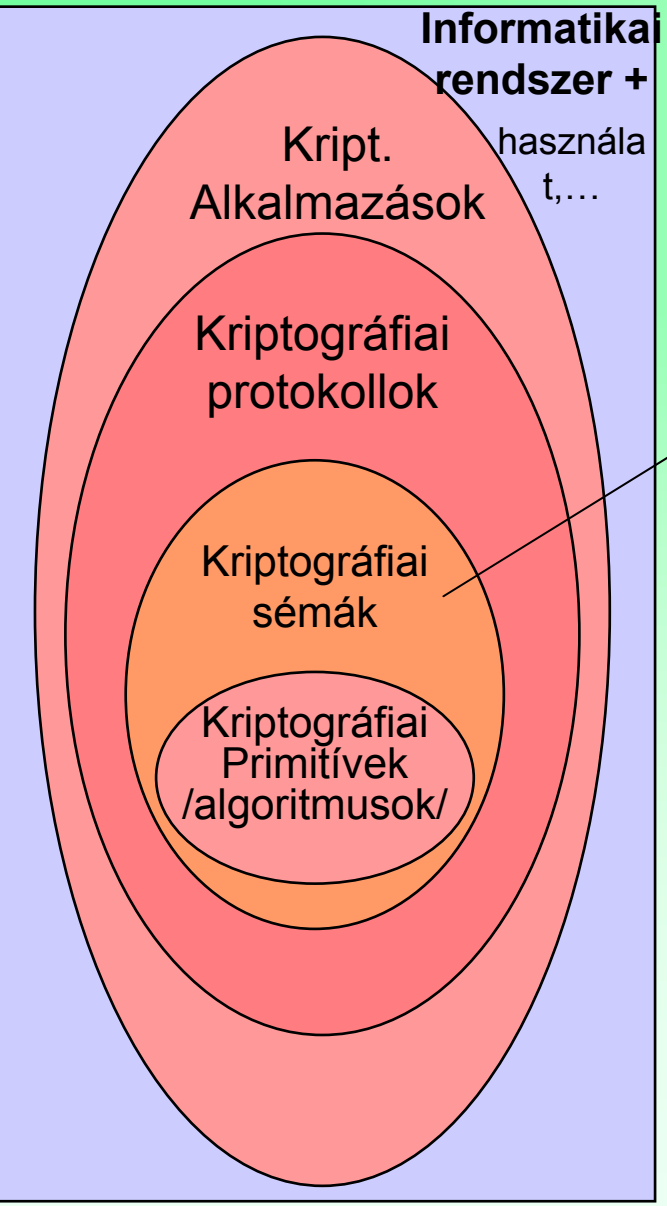
Additional protection may be achieved if:

p-1 has a large prime factor r_p
q-1 has a large prime factor r_q
r_p-1 has a large prime factor t_p
r_q-1 has a large prime factor t_q

$$e^{(rp-1)/tp} \bmod r_p \neq 1$$
$$e^{(rq-1)/tq} \bmod r_q \neq 1$$

For these conditions

$$\#\text{ of iterations, } k \geq t_p \cdot t_q$$



Az **üzenetek megismerése**
faktorizáció, $\varphi(n)$ vagy d
meghatározása nélkül

„Kicsi” üzenet vagy
Üzenet „kicsi” darabokra bontása

a) 2010-es hackerversenyen az m-et túl kicsi (byte-nyi) darabokra bontották, így egyszerű helyettesítéssel fejthető volt az üzenet.

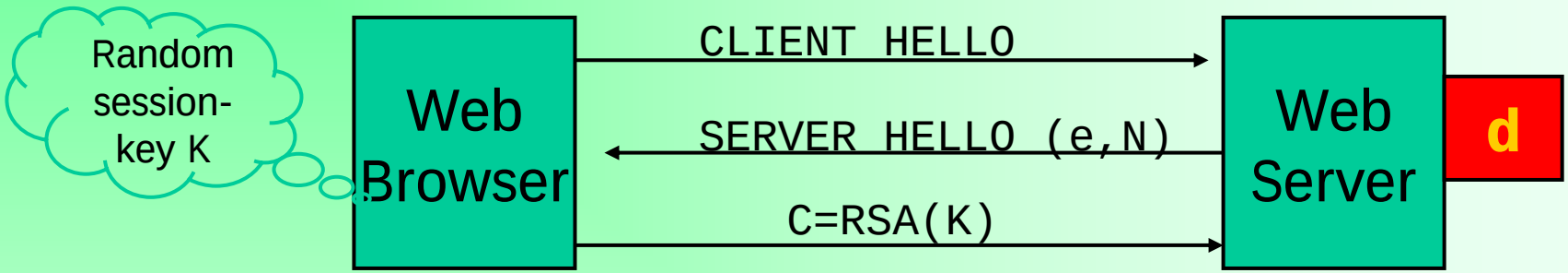
b) „kicsi” üzenetre homomorf struktúra kihasználása:
$E(m_1)*E(m_2)=E(m_1*m_2)$:

$$c_1 \equiv m_1^e, c_2 \equiv m_2^e, \quad ekkor\, c_1 * c_2 \equiv (m_1 * m_2)^e$$

Textbook attack az RSA ellen

„Rövid” üzenet kódolásának veszélyei:

Textbook attack az RSA ellen



- Ha a Session-key K 64 bites: $K \in \{0,\ldots,2^{64}\}$ /pl. DES kulcs/
  - A támadó látja: $\mathbf{C = K^e \pmod n}$
- Tegyük fel, hogy $\mathbf{K = K_1 \cdot K_2}$ ahol $K_1, K_2 < 2^{34}$

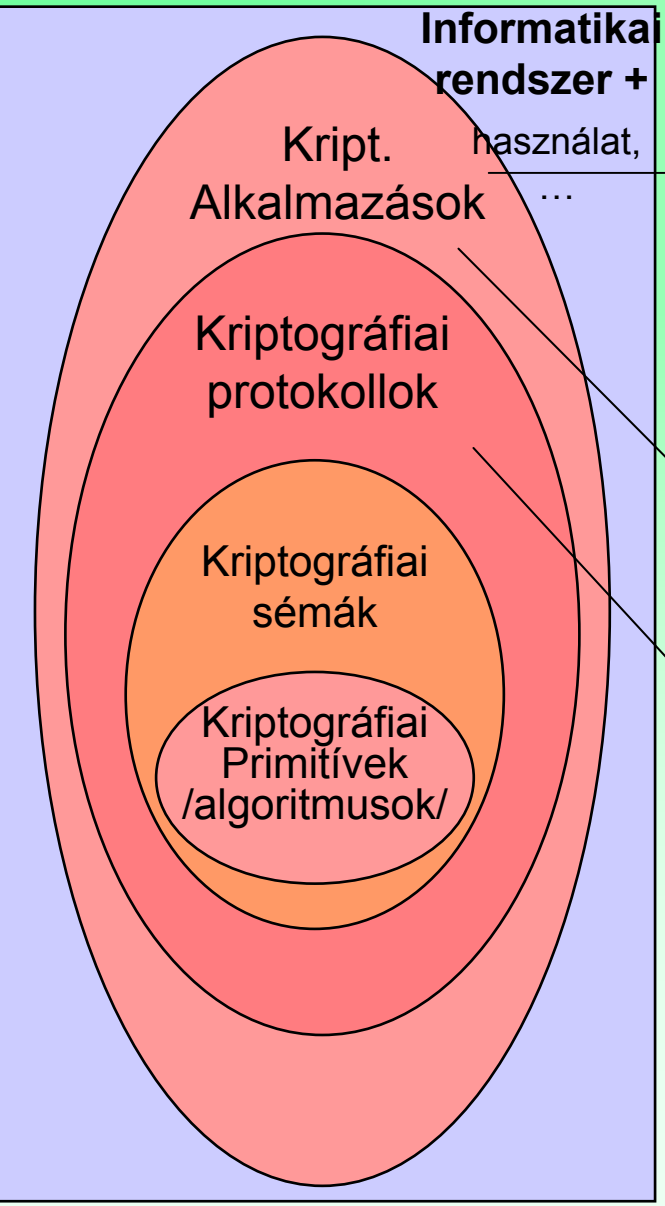
  Ekkor: $\mathbf{C/K_1^e = K_2^e} \pmod N$
- Építsünk adatbázist: $\mathbf{C/1^e, C/2^e, C/3^e, \ldots, C/(2^{34})^e}$. műveletigény: 2^{34}*c

  Teszteljük K_2 –t: $\mathbf{K_2 = 0,\ldots, 2^{34}}$, amíg $\mathbf{K_2^e}$ –t megtaláljuk az adatbázisban.

  time: $2 \cdot 2^{34} \cdot 34$*c
- Támadás műveletigénye : $\mathbf{\approx 2^{40} << 2^{64}}$

Következtetés:
Célszerű a „rövid” kódolandó üzenetet kiegészíteni: „padding”

Kriptográfiai rendszer



Az **üzenetek megismerése** faktorizáció, $\varphi(n)$ vagy d meghatározása nélkül

Közös modulus választása

Pl. központi kulcsgenerálási hiba

a) Számítható e*d-1: φ(n) egy többszöröse (belső támadás)

b) Ha ugyanazt az üzenetet küldi két feladó (külső támadás)

$C_1=m^{e1}$ (n); $C_2=m^{e2}$ (n)

Támadó ismeri: n,e_1,e_2,c_1,c_2,

Euklideszi algoritmussal számolja r,s-t: $r*e_1+s*e_2=1$

Ahonnan: $(c_1^{-1})^{-r}*c_2^{s} = m^{re1+se2} = m$

Rendszeren belül p közös, különböző q-kat generálnak

lnko($n_1=p*q_1$, $n_2=p*q_2$)

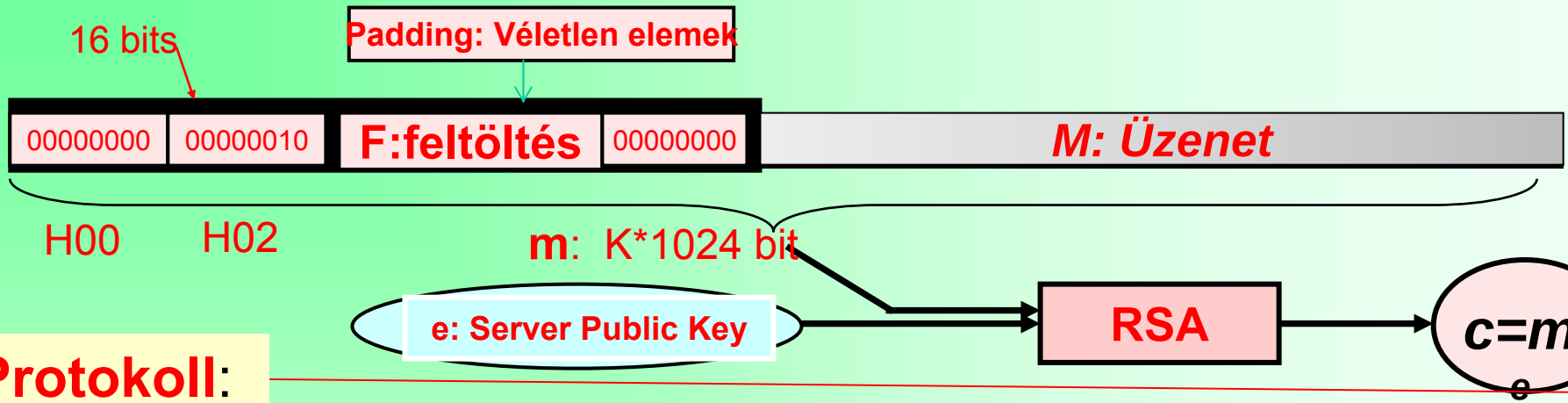
RSA-PKCS #1 v1.5 Encryption

Homomorf struktúra kihasználása: $E(m_1)*E(m_2)=E(m_1*m_2)$

CCA: „Choosen Ciphertext Attack”

RSA-PKCS #1 v1.5 Encryption

Széleskörűen elterjedt a web szerverek és browserek használatában, pl. SSL/TLS



Protokoll:

Kliens Ismeri a szerver nyilvános kulcsait: (n,e);

Titkosítja az M üzenetet /m hatványaként/

c

yes/no

Szerver: Megoldja a titkosított üzenetet, és visszajelzi, hogy helyes-e az ellenőrzés

Bleichenbacher (1998): "Million Message Attack"

http://archiv.infsec.ethz.ch/education/fs08/secsem/Bleichenbacher98.pdf

A támadónak M ismeretlen, C és e ismert: $C=(m)^e=(00,02,F,00,M)^e$

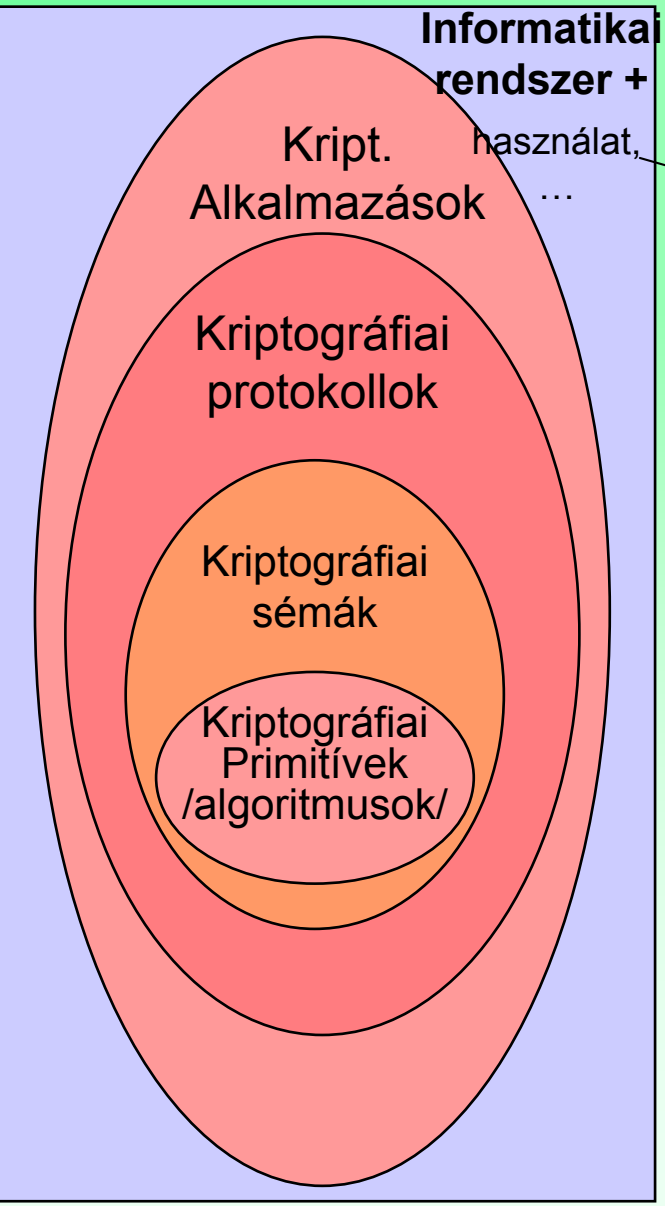
A támadó választ egy véletlen r (r <1024 bites) számot, kiszámolja:
$C' = r^e \cdot c = (r*m)^e$, Elküldi a szervernek C' -t

⇒ A támadó tesztelni tudja, melyik C'-re lesz megoldás után az első két byte: „00,02".

⇒ Kb. 1.000.000 „üzenet" -re adott szerver válaszból M meghatározható

New Attacks on PKCS#1v1.5 Encryption: http://www.iacr.org/archive/eurocrypt2000/1807/18070374-new.pdf

PKCS#1 v2 OAEP



Az **üzenetek megismerése**
faktorizáció, $\varphi(n)$ vagy d
meghatározása nélkül

Megtévesztéses (Social Attack)

Tf. **A** küld üzenetet **B**-nek (**B** nyilvános kulcsával): $c=E(m)=m^e$

A **T** támadó választ egy s „üzenetet” (amelynek kiszámolja az s^{-1} inverzét), lerejtjelzi s-t, majd elküldi **B**-nek a $c'=c*s^e$ üzenetet.

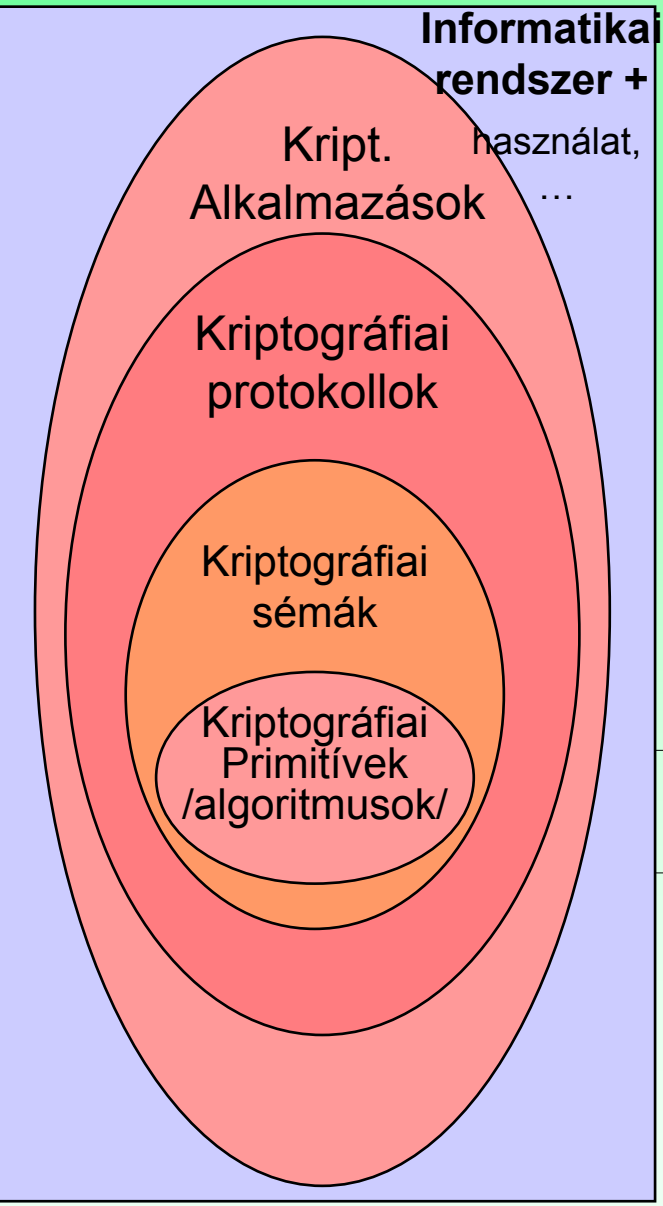
B megoldja a kapott c’ üzenetet: $(c*s^e)^d=m*s$ és látja nem értelmes, visszakérdez **T**-től.

T megírja: küldje vissza a kapott üzenetet, ellenőrzi mi lehet a baj.

T megfejti az eredeti üzenetet: $s*m*s^{-1}=m$.

A **digitális aláírás hamisítása** faktorizáció, $\varphi(n)$ vagy d meghatározása nélkül

homomorf struktúra kihasználása



Tegyük fel, hogy egy **T** támadó hamisítani akarja **XY** aláírását /saját dokumentumot akar aláírni XY nevében/.
Ehhez ismeri XY sok üzenetét (m_i) , és a $H(m_i)$ hash kép aláírását:
$s(m_i)= (H(m_i))^d \bmod n$
Valamint **T** ismeri XY nyilvános kulcsait .

T kiválaszt **XY**-nak t db. olyan m_i üzenetét, amelyek hash képe: $H(m_i)$ B-smooth /$t>\pi(B)$, i=1,2,..,t/:

*Egy poz. egész számot **B-smooth**-nak nevezünk, ha az összes prím-osztója nem nagyobb B-nél:* $H(m_i) = \prod_{j=1}^{\pi(B)} p_j^{v_{ij}}$, $1 \le i \le t$, $p_j \le B$

Pl: **MD5**(message 30854339) = 955dd317dd4715d26465081e4bfac00=
$=2^{14}*3*5^3*13*227*1149*1789*2441*4673*4691*9109*8377619$

A támadás alapja:

Ha $S_1 = M_1^d \bmod n$, $S_2 = M_2^d \bmod n$,

akkor $S = S_1*S_2 = M_1^d M_2^d = (M_1*M_2)^d \bmod n$.

Az RSA alapú aláírás támadása

$$H(m_i) = \prod_{j=1}^{\pi(B)} p_j^{v_{ij}} \ , \ 1 \le \mathrm{i} \le \mathrm{t}, \ \ p_j \le B$$

A támadó az m' hamis üzenetet akarja XY nevében aláírni, ehhez tud generálni olyan hamis m üzenetet, amelynek hash képe szintén B-smooth:

$$H(m) = \prod_{j=1}^{\pi(B)} p_j^{w_j} \ , \ 1 \le \mathrm{i} \le \mathrm{t}, \ \ p_j \le B$$

A támadó megoldja x_1 , x_2 ,…, $x_{\pi(B)}$ -re a következő lineáris kongruencia-rendszert /ahol a sorok már lin. ftl-ek/:

$$v_{11}x_1 + v_{12}x_2 + \ldots + v_{1t}x_t \equiv w_1 \pmod{e}$$

$$v_{21}x_1 + v_{22}x_2 + \ldots + v_{2t}x_t \equiv w_2 \pmod{e}$$

$$v_{t1}x_1 + v_{t2}x_2 + \ldots + v_{tt}x_t \equiv w_t \pmod{e}$$

A kongruenciákból következik, hogy: $w_i = v_{i1}x_1 + v_{i2}x_2 + \ldots + v_{it}x_t + k_i * e$

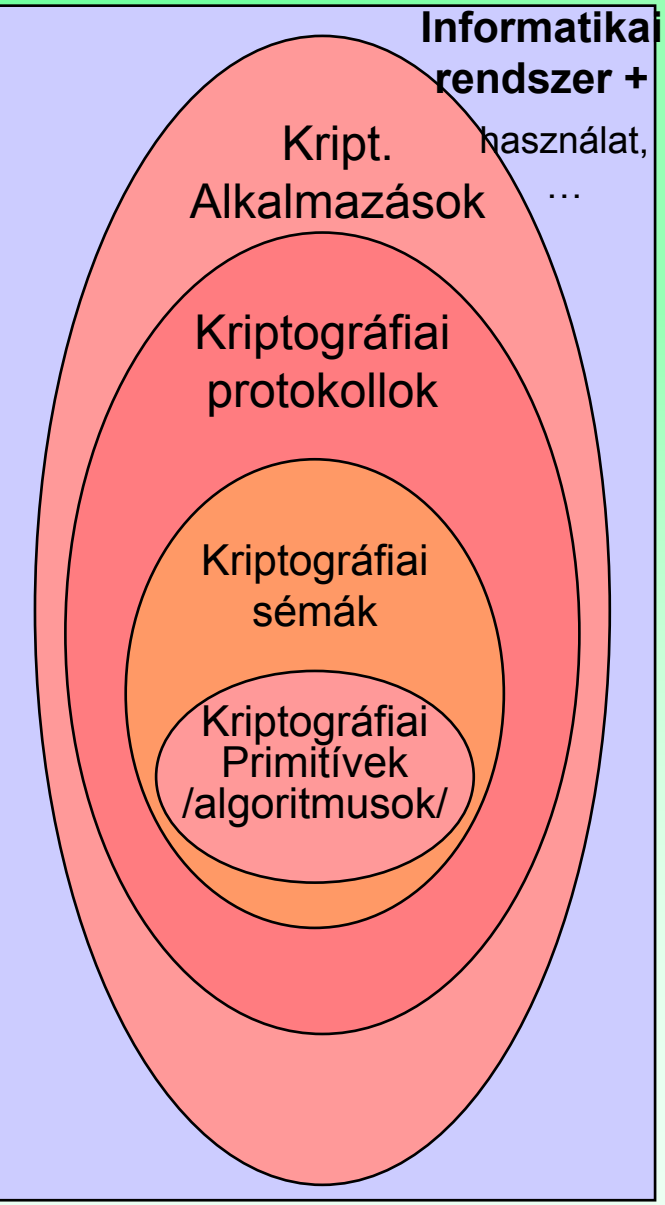
Ekkor a hamis m üzenet hash képe:

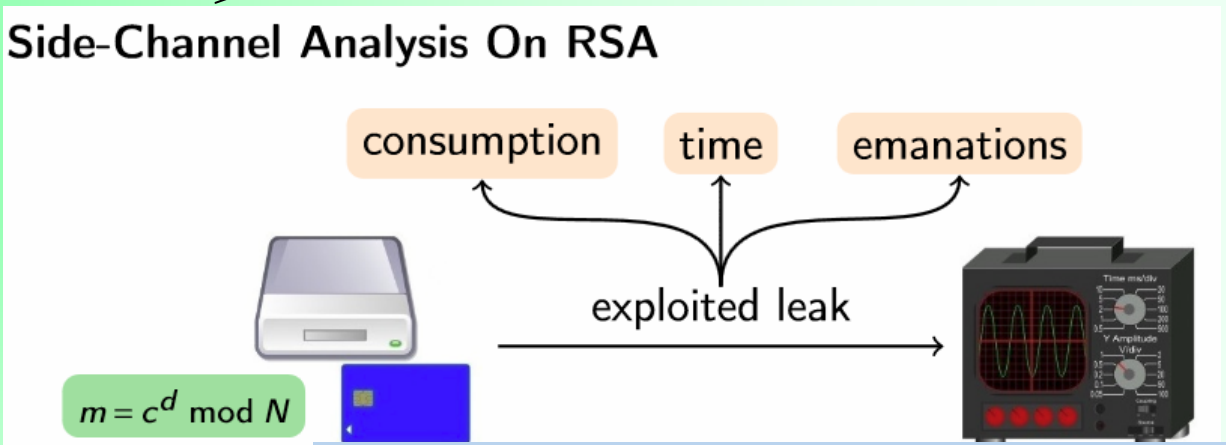
$$H(m) = H(m_1)^{x_1} * H(m_2)^{x_2} * \ldots * H(m_t)^{x_t} * \left(\prod_{j=1}^{t} p_j^{k_j} \right)^e$$

A támadó által (e és sok m_i aláírásának ismeretében) számolható a hamis aláírás:

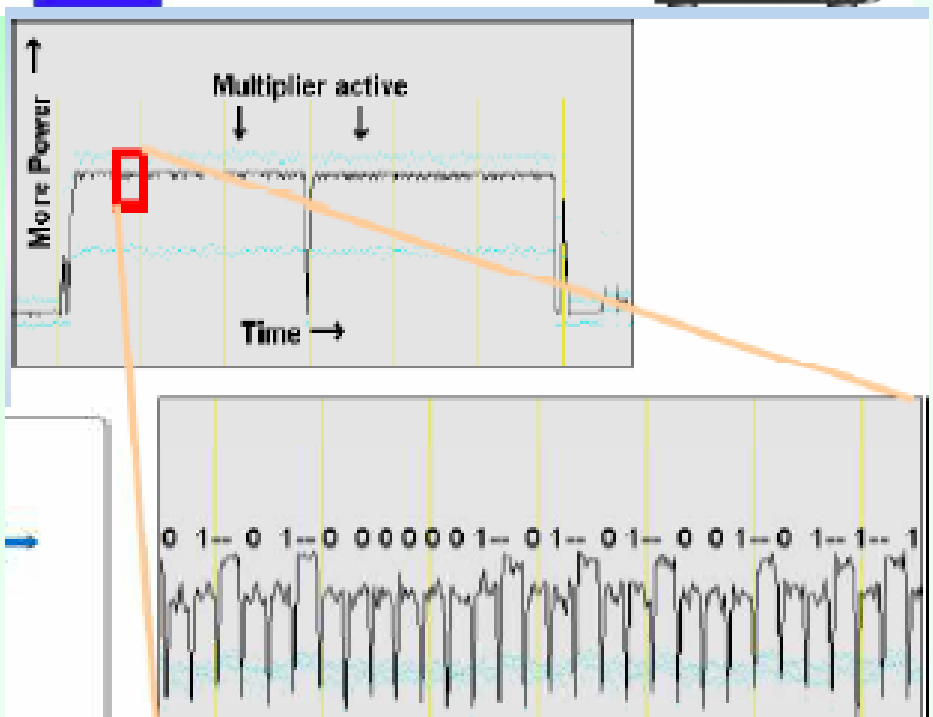
$$s = s_1^{x_1} * s_2^{x_2} * \ldots * s_t^{x_t} * \left(\prod_{j=1}^{t} p_j^{k_j} \right) \pmod{n}$$

Egyéb támadások





POWER ATTACK



http://itcafe.hu/hir/rsa-tamadas_openssl_fault-based_sparc.html

RSA-támadás: árulkodó hibák

Írta: Barna József | 2010-04-28 09:49 | Forrás: IT café

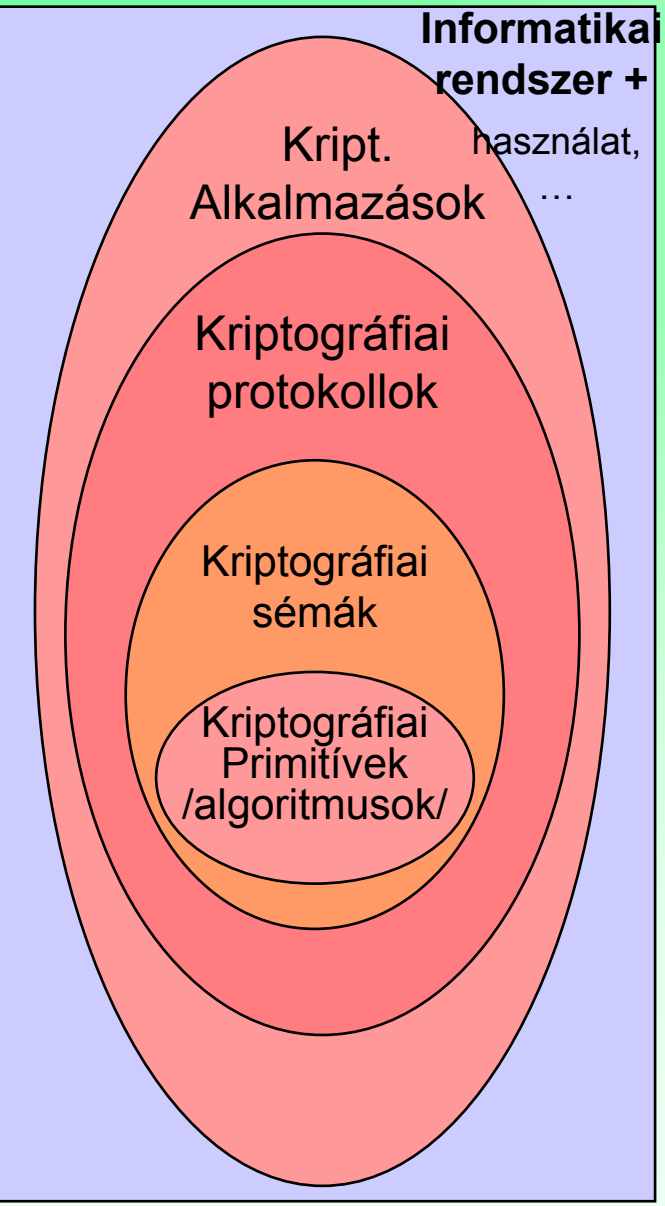
Kutatók egy újfajta módszerrel, irányítottan előidézett számítási hibákra és az OpenSSL egyik hiányosságára támaszkodva vissza tudtak fejteni egy 1024 bites RSA-kulcsot.

Egy, a kriptográfiai algoritmusok elleni újszerű támadást mutattak be nemrég amerikai kutatók.

Az eljárás lényege az, hogy a hitelesítést végző szerver mikroprocesszorát az üzenetek dekódolásához szükséges algoritmusok, konkrétan a moduláris hatványozás végrehajtása során hibás műveletvégzésre „kényszerítik", majd az így generált hibás aláírásokból visszafejtik a titkos kulcsot. A kérdés az, hogy hogyan lehet számítási hibák elkövetésére kényszeríteni a processzor aritmetikai egységét? A kutatók szerint ez meglehetősen egyszerű, mivel a mai modern mikroprocesszorok szorzója (szorzást végző egysége) különösen érzékeny az időzítésre, ezért ha valamilyen külső behatással sikerül a jelterjedést lassítani, ez fog elsőként hibázni. A támadás konkrét megvalósítása során egy SPARC architektúrájú beágyazott chipen a tápfeszültség csökkentésével érték el a kívánt hatást.

Azt, hogy a cél eléréséhez melyik nem gyári feszültségérték a legideálisabb, a processzorral szorzásokat végeztetve kísérletezték ki. Úgy találták, hogy a gyári 1,3 voltos tápfeszültséget 1,25 voltra csökkentve ideális a hibaarány: így az aláírások 88 százaléka volt helytelen, és ezeknek 12 százaléka tartalmazott olyan egybites hibát, amely a visszafejtésnél használható. A rossz kulcsokat aztán a kutatók egy, az OpenSSL-ben lévő hibát kihasználva tudták megszerezni: az SSL-titkosításra és hitelesítésre széles körben használt csomag rögzített ablakos hatványozási algoritmusa nem ellenőrzi az eredményül kapott aláírás helyességét, az aláírást ellenőrzés nélkül elküldi az azt kérő kliensnek. A hibás aláírások és az üzenetek ismeretében aztán egy 81 darab Pentium 4 processzoros gépből álló klaszteren 104 óra alatt visszafejtették az 1024 bit hosszúságú titkos (privát) RSA-kulcsot.

Egyéb támadások



A számítógép feletti rendelkelkezés jogosulatlan átvételével

pl. a titkos kulcs ellopásával

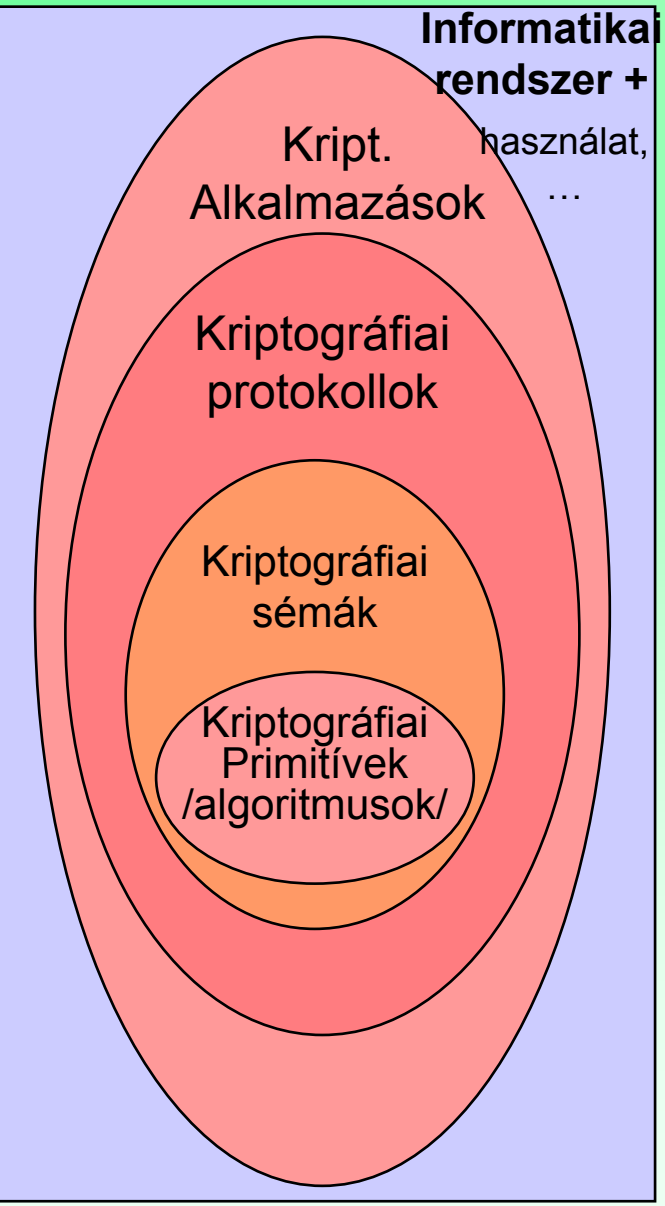
Védelem (pl. elektronikus aláírásnál):

SSCD /BALE/ használatát írja elő az elektronikus aláírásról szóló 2001. évi XXXV. Törvény:

SSCD PP szerint,

Common Criteria szerint értékelt termékek használatával

Egyéb támadások



Csapdák elhelyezése

Csapda lehetőség:

A program választ p 1024 bites prímet, a programozó választ a,b konstanst úgy, hogy q=a*p+b teljesítse a p-től eltérés követelményét.

A program teszteli, q prím-e, ha nem keres új p-t.

Ekkor $n=p*q=a*p^2+b*p$.

Ha (n,a,b) ismert a támadónak, p-re megoldható.

/Pl. a=1 és b egy tetszőleges 1025 jegyű konstans./

Védelem:

-Vagy mi írjuk meg a prímtesztet, vagy,

-Csak akkor fogadjunk el prímteszt algoritmust, ha az megfelelően ellenőrzött (pl. nyílt forráskódú és átnéztük: csak a szükséges programok vannak benne)

Az RSA és a faktorizáció fejlődése

n faktorizálásához keresünk x, y poz. egész számokat: $x^2 = y^2 \bmod n$

Ha n osztja $(x - y)(x + y)$ $\rightarrow$ $lnko(x- y, n)$, vagy lnko$(x + y, n)$ lehet nem triviális faktora n-nek

Az RSA_IFP kutatások ösztönzésére, valamint a gyakorlatban, belátható időn belül faktorizálható számok nagyságára vonatkozó korlát megismerése érdekében hozták létre az *RSA Factoring Challenge-et.* Ezért – pénzdíj ellenében – különböző nehézségű faktorizációs problémákat tűztek ki:

Algorithm	Running time
Pollard's $p-1$ algorithm	$\mathcal{O}(B \log B (\log n)^2 + (\log n)^3)$
Quadratic sieve	$\mathcal{O}\left(e^{(1+o(1))\sqrt{\ln n \ln\ln n}}\right)$
Number field sieve	$\mathcal{O}\left(e^{(1.92+o(1))\sqrt[3]{\ln n}\sqrt[3]{(\ln\ln n)^2}}\right)$
Elliptic curve method	$\mathcal{O}\left(e^{(1+o(1))\sqrt{2\ln p \ln\ln p}}\right)$

Faktorizálási fordulópontok

Number of Decimal Digits	Approximate Number of Bits	Date Achieved	MIPS-years	Algorithm	
100	332	April 1991	7	quadratic sieve	
110	365	April 1992	75	quadratic sieve	
120	398	June 1993	830	quadratic sieve	
129	428	April 1994	5000	quadratic sieve	
130	431	April 1996	1000	generalized number field sieve	
140	465	February 1999	2000	generalized number field sieve	
155	512	August 1999	8000	generalized number field sieve	
160	530	April 2003	—	Lattice sieve	
174	576	December 2003	—	Lattice sieve	10.000 $
200	663	May 2005	—	Lattice sieve	20.000 $
	704				30.000 $
	768				50.000 $
	1024				100.000 $
	2048				200.000 $

RSA paraméter-követelmények

INTERNATIONAL TELECOMMUNICATION UNION

1988:

CCITT **X.509**
(11/1988)

THE INTERNATIONAL
TELEGRAPH AND TELEPHONE
CONSULTATIVE COMMITTEE

C.6 *Security requirements*

C.6.1 *Key lengths*

It is recognized that the acceptable key length is likely to change with time, subject to the cost and availability of hardware, the time taken, advances in techniques and the level of security required. It is recommended that a value for the length of n of 512 bits be adopted initially, but subject to further study.

C.6.2 *Key generation*

The security of RSA relies on the difficulty of factorizing n. There are many algorithms for performing this operation, and in order to thwart the use of any currently known technique, the values p and q must be chosen carefully, according to the following rules [e.g. see Reference 2), Section C.2]:

a) they should be chosen randomly;

b) they should be large;

c) they should be prime;

d) |p-q| should be large;

e) (p+1) must possess a large prime factor;

f) (q+1) must possess a large prime factor;

g) (p–1) must possess a large prime factor, say r;

h) (q–1) must possess a large prime factor, say s;

i) (r–1) must possess a large prime factor;

j) (s–1) must possess a large prime factor.

Strong prím generálására Gordon algoritmusa: pl. Handbook of Applied Cryptography, http://cacr.uwaterloo.ca/hac/about/chap4.pdf 4.53.

Vizsgálták egy véletlen szám legnagyobb, második legnagyobb prímosztóinak eloszlását.

x-nél nem nagyobb véletlen szám első, ill. második legnagyobb prímosztója **F(α)**, ill. **G(β)** valószínűséggel lesz $\mathbf{x^{\alpha}}$, $\mathbf{x^{\beta}}$ –nál kisebb, ahol:

$$Dickman, Ramashwami: F(\alpha) = \int_0^t (F(\frac{t}{1-t})\frac{dt}{t}, \text{ ha } 0 \le \alpha \le 1; G(\beta) = \int_0^t \left[G(\frac{t}{1-t}) - F(\frac{t}{1-t}) \right] \frac{dt}{t} \text{ , ha } 0 \le \beta \le 1/2$$

F(α), ill. G(β):	0,9	0,8	0,5	0,35	0,2	0,1	0,01
1.legn. α :	0,9048	0,8187	0,6065	0,5220	0,4430	0,3785	**0,2697**
2. legn. β :	0,3590	0,3104	0,2117	0,1611	0,1003	0,0531	0,0056

Azaz az 1., ill. 2. legnagyobb prímosztó 1% valószínűséggel kisebb (99% val-gel nagyobb), mint $x^{0,2697}$, ill. $x^{0,0056}$

Bundesamt
für Sicherheit in der
Informationstechnik

BSI - Technische Richtlinie

Bezeichnung:	**Kryptographische Verfahren: Empfehlungen und Schlüssellängen**
Kürzel:	BSI TR-02102
Version:	1.0
Stand:	20.06.2008

Schlüsselgenerierung

1. Wähle zwei Primzahlen p und q zufällig und unabhängig voneinander unter der Nebenbedingung
$$0.1 < |\log_2 p - \log_2 q| < 30.$$
2. Wähle den öffentlichen Exponenten $e \in \mathbb{N}$ unter den Nebenbedingungen
$$\mathrm{ggT}(e, (p-1)\cdot(q-1)) = 1 \text{ und } 2^{16} + 1 \leq e \leq 2^{1824} - 1.$$
3. Berechne den geheimen Exponenten $d \in \mathbb{N}$ in Abhängigkeit von e unter der Nebenbedingung
$$e \cdot d = 1 \bmod \mathrm{kgV}(p-1, q-1).$$

Léteznek speciális támadások „unbalenced” (nagyon eltérő nagyságrendű) prímek esetére

Schlüsselvereinbarung

1. A wählt gleichverteilt einen Zufallswert $x \in \{1, \ldots, p-1\}$ und sendet g^x an B.
2. B wählt gleichverteilt einen Zufallswert $y \in \{1, \ldots, p-1\}$ und sendet g^y an A.
3. A berechnet $(g^y)^x = g^{xy}$.
4. B berechnet $(g^x)^y = g^{xy}$.

Das ausgehandelte Geheimnis ist dann g^{xy}.

Schlüssellänge Die Länge von p sollte mindestens 2048 Bit betragen.

An RSA key pair, in its basic form, consists of an RSA public key (n, e) and an RSA private key (n, d), where:

1. n, the modulus, **shall** be the product of exactly two distinct odd positive prime factors, p and q where *nBits* is the length in bits of n as specified for the desired security strength s (see Table 1) and *nLen* is the corresponding length in bytes.

2. The public exponent e **shall** be selected with the following constraints:

   a. The public exponent e **shall** be selected prior to generating the prime factors p and q and the private exponent d.

   b. The exponent e **shall** be an odd positive integer such that:

   $$65{,}537 \le e < 2^{256}$$

   Note that the value of e may be the same for different key pairs.

3. Two secret and randomly generated positive primes p and q **shall** be selected with the following constraints:

   a. The prime factors of the modulus **shall** be generated independently at random for different key pairs.

   b. LCM$((p-1), (q-1))$ **shall** be greater than e and relatively prime to e.

   c. The private prime factor p **shall** be selected randomly from the primes that satisfy $(\sqrt{2})(2^{(nBits/2)-1}) \le p \le (2^{nBits/2}-1)$.

   d. The private prime factor q **shall** be selected randomly from the primes that satisfy $(\sqrt{2})(2^{(nBits/2)-1}) \le q \le (2^{nBits/2}-1)$.

   e. The difference between p and q **shall** be $> 2^{(nBits/2)-100}$.

   f. The prime factors p and q **shall** be generated using an **approved** method meeting the above constraints. Such methods are provided in Appendix B.3 of FIPS 186-3 [3]. The additional constraints placed on $p \pm 1$ and $q \pm 1$ in that document (in the case of 1024-bit RSA moduli) are not required for compliance with this Recommendation.

4. The private exponent d **shall** be selected with the following constraints after the selection of e and the generation of p and q:

   a. The exponent d **shall** be a positive integer value such that $2^{nBits/2} < d < \text{LCM}((p-1), (q-1))$, and

   b. $1 = ed \bmod \text{LCM}((p-1), (q-1))$. (That is, $d = e^{-1} \bmod (\text{LCM}((p-1), (q-1)))$.

NIST SP 800-56B: Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography

August 2009

NIST: National Institute of Standards and Technology

Target Security Strength:	80 bit	112 bit
RSA modulus length:	1024 bit	2048 bit

Következmény:

- ha $e \approx N$; $p \approx q \rightarrow$ nem biztonságos;
- ha $e < 2^{256}$, $N \approx 2^{2048} \rightarrow$ $\frac{1}{4}(N/e)^{2/5} \approx 2^{714} \rightarrow$ elhanyagolható eséllyel lesz d_p vagy d_q kisebb

Ha n 2048 bites, akkor p és q elejének legfeljebb a 99 bitje egyezhet meg.

Table 7: Recommended parameters for RSA and rsagen1 for a resistance during X years

Parameter	1 year	3 years	6 years	10 years (speculative)
MinModLen	1 536	2 048	2 048	?
ErrProb	2^{-80}	2^{-100}	2^{-100}	2^{-100}
SeedEntropy/EntropyBits	80	100	100	?

NMHH Határozat
(ETSI TS 102 176-1 v 2.1.1 (2011-07))

Egy aláírás-készlet a következő elemekből áll:

- aláíró algoritmus a paramétereivel,
- kulcsgeneráló algoritmus,
- feltöltési eljárás,
- kriptográfiai hash-függvény.

Annex C (informative): Generation of RSA modulus

An RSA modulus is obtained by multiplying two prime numbers of roughly the same size. Furthermore, the two factors must not be too close in order to be far enough from the square root of the modulus.

If we let p and q be the two prime factors of the modulus n, we can require that, for example:

$$0,1 < |\log_2(p) - \log_2(q)| < 30 \quad (1)$$

which means that none of the factors is small or close to the square root of the modulus. This condition implies that:

$$\log_2(n)/2 - 15 < \log_2(p), \log_2(q) < \log_2(n) / 2 + 15 \quad (2)$$

The generation of an RSA modulus of exactly k bits could be done with the following algorithm:

- Choose a random prime number p in the range $]2^{k/2-15}, 2^{k/2+15}[$.
- Choose a random prime number q in the range $[2^{k-1}/p, 2^k/p[$.
- If the condition $0,1 < |\log_2(p)-\log_2(q)| < 30$ is not satisfied, go back to the first step.
- Let n be the product of p and q.

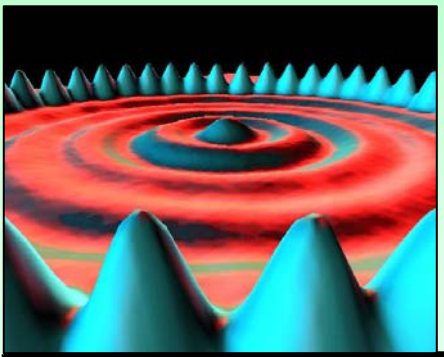
$$0,1 < \log\frac{p}{q} < 30 \quad ha \quad p > q$$

$$2^{0,1} < \frac{p}{q} < 2^{30}, \quad azaz \quad \frac{p}{q} = 2 \quad OK!$$

Ha $n \approx 2^{2048}$, akkor $\log(n)/2 = 1024$

akkor $2^{1019} < q < p < 2^{1039}$ megfelelő

- *mik a jó prímek tulajdonságai, mennyire legyenek egymástól távol, milyen legyen az arányuk (pl. max. p/q arány esetén a 2 kellően magas-e),*
- *prímtesztnél hány ciklusig kell azt futtatni (pl. 10.000 elegendő-e),*
- *kell-e aggódni a beégetett kicsi "e" értékek (pl. 3) miatt chipkártyás kulcsgenerálásoknál,*
- *a kulcsokat milyen méretű adatra kell legalább alkalmazni (pl. 160 bites SHA-1 vagy 128 bites MD5 lenyomatok aláírása, 128 bites AES kulcsok rejtjelezése most is megy a gyakorlatban)*



	Hagyományos PC	Kvantum PC
Faktorizáció	$e^{O\left(n^{1/3}\log^{2/3} n\right)}$	$O(n) \in e^{O(\log n)}$

Köszönöm a figyelmet!

KÉRDÉSEK?