Criptografie și Securitate

- Prelegerea 20.1 -
RSA

Adela Georgescu, Ruxandra F. Olimid

Facultatea de Matematică și Informatică
Universitatea din București

Cuprins

RSA

- 1976 - Diffie și Hellman definesc conceptul de criptografie asimetrică;

RSA

- 1976 - Diffie și Hellman definesc conceptul de criptografie asimetrică;
- 1977 - R.Rivest, A.Shamir și Leonard Adleman introduc sistemul RSA;

RSA

- 1976 - Diffie și Hellman definesc conceptul de criptografie asimetrică;
- 1977 - R.Rivest, A.Shamir și Leonard Adleman introduc sistemul RSA;
- RSA este cel mai cunoscut și mai utilizat algoritm cu cheie publică.

RSA

The era of electronic mail may soon be upon us; we must ensure that two important properties of the current paper mail system are preserved: (a) messages are private, and (b) messages can be signed. We demonstrate in this paper how to build these capabilities into an electronic mail system.

At the heart of our proposal is a new encryption method. This method provides an implementation of a public-key cryptosystem, an elegant concept invented by Diffie and Hellman. Their article motivated our research, since they presented the concept but not any practical implementation of such a system...”

(R.Rivest, A.Shamir, L.Adleman: A Method for Obtaining Digital Signatures and Public-Key Cryptosystems - introduction)

RSA

[http://people.csail.mit.edu/rivest/]

RSA = **R**ivest + **S**hamir + **A**dleman

Problema RSA

- Problema RSA se bazează pe dificultatea factorizării numerelor mari: $N = p \cdot q$, p și q prime;

Problema RSA

- Problema RSA se bazează pe dificultatea factorizării numerelor mari: $N = p \cdot q$, p și q prime;
- Fie $\mathbb{Z}_N^*$ un grup de ordin $\phi(N) = (p-1)(q-1)$;

Problema RSA

- Problema RSA se bazează pe dificultatea factorizării numerelor mari: $N = p \cdot q$, p și q prime;
- Fie $\mathbb{Z}_N^*$ un grup de ordin $\phi(N) = (p-1)(q-1)$;

Problema RSA

Fiind dat N, un întreg $e > 0$ care satisface $(e, \phi(N)) = 1$, și un element $y \in \mathbb{Z}_N^*$, se cere să se găsească x a.î. $x^e = y \bmod N$.

Problema RSA

- Problema RSA se bazează pe dificultatea factorizării numerelor mari: $N = p \cdot q$, p și q prime;
- Fie $\mathbb{Z}_N^*$ un grup de ordin $\phi(N) = (p-1)(q-1)$;

Problema RSA

Fiind dat N, un întreg $e > 0$ care satisface $(e, \phi(N)) = 1$, și un element $y \in \mathbb{Z}_N^*$, se cere să se găsească x a.î. $x^e = y \bmod N$.

- Dacă se cunoaște factorizarea lui N, atunci $\phi(N)$ este ușor de calculat și problema RSA este ușor de rezolvat;

Problema RSA

- Problema RSA se bazează pe dificultatea factorizării numerelor mari: $N = p \cdot q$, p și q prime;
- Fie $\mathbb{Z}_N^*$ un grup de ordin $\phi(N) = (p-1)(q-1)$;

Problema RSA

Fiind dat N, un întreg $e > 0$ care satisface $(e, \phi(N)) = 1$, și un element $y \in \mathbb{Z}_N^*$, se cere să se găsească x a.î. $x^e = y \bmod N$.

- Dacă se cunoaște factorizarea lui N, atunci $\phi(N)$ este ușor de calculat și problema RSA este ușor de rezolvat;
- Dacă nu se cunoacște $\phi(N)$, problema RSA este dificilă.

Experimentul RSA $RSA - inv_{\mathcal{A},GenRSA(n)}$

- Considerăm experimentul RSA pentru un algoritm $\mathcal{A}$ și un parametru n.
  1. Execută GenRSA și obține (N, e, d);

Experimentul RSA $RSA - inv_{\mathcal{A},GenRSA(n)}$

- Considerăm experimentul RSA pentru un algoritm $\mathcal{A}$ și un parametru n.
  1. Execută GenRSA și obține (N, e, d);
  2. Alege $y \leftarrow \mathbb{Z}_N^*$;

Experimentul RSA $RSA - inv_{\mathcal{A},GenRSA(n)}$

- Considerăm experimentul RSA pentru un algoritm $\mathcal{A}$ și un parametru n.
  1. Execută GenRSA și obține (N, e, d);
  2. Alege $y \leftarrow \mathbb{Z}_N^*$;
  3. $\mathcal{A}$ primește N, e, y și întoarce $x \in \mathbb{Z}_N^*$;

Experimentul RSA $RSA - inv_{\mathcal{A},GenRSA(n)}$

- Considerăm experimentul RSA pentru un algoritm $\mathcal{A}$ și un parametru n.
  1. Execută GenRSA și obține (N, e, d);
  2. Alege $y \leftarrow \mathbb{Z}_N^*$;
  3. $\mathcal{A}$ primește N, e, y și întoarce $x \in \mathbb{Z}_N^*$;
  4. Output-ul experimentului este 1 dacă $x^e = y \bmod N$ și 0 altfel.

Experimentul RSA $RSA - inv_{\mathcal{A},GenRSA(n)}$

- Considerăm experimentul RSA pentru un algoritm $\mathcal{A}$ și un parametru n.
  1. Execută GenRSA și obține (N, e, d);
  2. Alege $y \leftarrow \mathbb{Z}_N^*$;
  3. $\mathcal{A}$ primește N, e, y și întoarce $x \in \mathbb{Z}_N^*$;
  4. Output-ul experimentului este 1 dacă $x^e = y \bmod N$ și 0 altfel.

Definiție

Spunem că problema RSA este dificilă cu privire la GenRSA dacă pentru orice algoritm PPT $\mathcal{A}$ există o funcție neglijabilă negl *așa încât*

$$Pr[RSA - inv_{\mathcal{A},GenRSA(n)} = 1] \leq negl(n)$$

GenRSA

- Prezumpția RSA este că există un algoritm GenRSA pentru care problema RSA este dificilă;

GenRSA

- Prezumpția RSA este că există un algoritm GenRSA pentru care problema RSA este dificilă;
- Un algoritm GenRSA poate fi construit pe baza unui număr compus împreună cu factorizarea lui;

GenRSA

- Prezumpția RSA este că există un algoritm GenRSA pentru care problema RSA este dificilă;
- Un algoritm GenRSA poate fi construit pe baza unui număr compus împreună cu factorizarea lui;

Algorithm 3 GenRSA

Input: n

Output: N, e, d

1: **print** N cu factorii p și q
2: $\phi(N) = (p-1)(q-1)$
3: **găsește** e a.î. $gcd(e, \phi(N)) = 1$
4: **calculează** $d := e^{-1} \bmod \phi(N)$
5: **return** N, e, d

GenRSA

- Pentru ca problema RSA să fie dificilă, trebuie ca N-ul ales în GenRSA să fie dificil de factorizat în produs de două numere prime;

GenRSA

- Pentru ca problema RSA să fie dificilă, trebuie ca N-ul ales în GenRSA să fie dificil de factorizat în produs de două numere prime;
- Deci problema RSA nu este mai dificilă decât problema factorizării;

GenRSA

- Pentru ca problema RSA să fie dificilă, trebuie ca N-ul ales în GenRSA să fie dificil de factorizat în produs de două numere prime;
- Deci problema RSA nu este mai dificilă decât problema factorizării;
- Dacă se cunosc N, e și d cu $ed = 1 \bmod \phi(N)$, se poate calcula factorizarea lui N în timp probabilist polinomial;

GenRSA

- Pentru ca problema RSA să fie dificilă, trebuie ca N-ul ales în GenRSA să fie dificil de factorizat în produs de două numere prime;
- Deci problema RSA nu este mai dificilă decât problema factorizării;
- Dacă se cunosc N, e și d cu $ed = 1 \bmod \phi(N)$, se poate calcula factorizarea lui N în timp probabilist polinomial;
- Nu se cunoaște nici o dovadă că nu există o altă metodă de a rezolva problema RSA care să nu implice calculul lui $\phi(N)$ sau al lui d.

GenRSA

- Definim sistemul de criptare *Textbook RSA* pe baza problemei prezentată anterior;
  1. Se rulează GenRSA pentru a determina N, e, d.
     - Cheia publică este: (N, e);
     - Cheia privată este (N, d);
  2. **Enc**: dată o cheie publică (N, e) și un mesaj $m \in \mathbb{Z}_N$, întoarce $c = m^e \mod N$;
  3. **Dec**: dată o cheie secretă (N, d) și un mesaj criptat $c \in \mathbb{Z}_N$, întoarce $m = c^d \mod N$.

Implementarea RSA

- Pentru ca GenRSA să fie dificilă trebuie ca N să fie produs de 2 numere prime mari;

Implementarea RSA

- Pentru ca GenRSA să fie dificilă trebuie ca N să fie produs de 2 numere prime mari;
- Sistemul necesită exponențieri modulare de tip $x^c \mod N$;

Implementarea RSA

- Pentru ca GenRSA să fie dificilă trebuie ca N să fie produs de 2 numere prime mari;
- Sistemul necesită exponențieri modulare de tip $x^c \mod N$;
- Efectuarea a $c - 1$ înmulțiri modulare este foarte ineficientă;

Implementarea RSA

- Pentru ca GenRSA să fie dificilă trebuie ca N să fie produs de 2 numere prime mari;
- Sistemul necesită exponențieri modulare de tip $x^c \mod N$;
- Efectuarea a $c - 1$ înmulțiri modulare este foarte ineficientă;
- Se utilizează algoritmul de exponențiere rapidă.

Implementarea RSA

Algorithm 4 Exponentiere rapidă

Input: N, x, c

Output: $x^c \mod N$

1: **descompune** c în binar: $c = \sum_{i=0}^{n-1} c_i 2^i$
2: $z \leftarrow 1$
3: **for** $i = n - 1$ to 0 **do**
4: $\quad z \leftarrow z^2 \mod N$
5: $\quad$ **if** $c_i = 1$ **then**
6: $\quad\quad z \leftarrow z \cdot x \mod N$
7: $\quad$ **end if**
8: **end for**
9: **return** z

Securitate - Problema 1

Problema 1: Determinismul

- Întrebare: Este Textbook RSA CPA-sigur sau CCA-sigur?

Securitate - Problema 1

Problema 1: Determinismul

- Întrebare: Este Textbook RSA CPA-sigur sau CCA-sigur?
- Răspuns: NU! Sistemul este determinist, deci nu poate rezista definițiilor de securitate!

Securitate - Problema 2

Problema 2: Coeficient de criptare mic

- Întrebare: Este o valoare mică o alegere corectă pentru coeficientul de criptare e?

Securitate - Problema 2

Problema 2: Coeficient de criptare mic

- Întrebare: Este o valoare mică o alegere corectă pentru coeficientul de criptare e?
- Răspuns: NU! Orice mesaj $m < N^{1/e}$ nu folosește reducerea modulară la criptare:

$$c = m^e \mod N = m^e$$

Securitate - Problema 2

Problema 2: Coeficient de criptare mic

- Întrebare: Este o valoare mică o alegere corectă pentru coeficientul de criptare e?
- Răspuns: NU! Orice mesaj $m < N^{1/e}$ nu folosește reducerea modulară la criptare:

$$c = m^e \mod N = m^e$$

- Fiind dat c, se determină imediat m ca:

$$m = c^{1/e} \mod N$$

Securitate - Problema 3

Problema 3: Atac cu text criptat ales

- Fie m_1, m_2 2 texte clare și c_1, c_2 textele criptate corespunzătoare;

Securitate - Problema 3

Problema 3: Atac cu text criptat ales

- Fie m_1, m_2 2 texte clare și c_1, c_2 textele criptate corespunzătoare;
- Atunci:
  $c_1 \cdot c_2 \mod N = m_1^e \cdot m_2^e \mod N = (m_1 \cdot m_2)^e \mod N$

Securitate - Problema 3

Problema 3: Atac cu text criptat ales

- Fie m_1, m_2 2 texte clare și c_1, c_2 textele criptate corespunzătoare;
- Atunci:
  $c_1 \cdot c_2 \mod N = m_1^e \cdot m_2^e \mod N = (m_1 \cdot m_2)^e \mod N$
- Întrebare: Cum poate un adversar să determine mesajul clar m_1, cunoscând textul criptat corespunzător c_1 printr-un atac cu text criptat ales?

Securitate - Problema 3

Problema 3: Atac cu text criptat ales

- Fie m_1, m_2 2 texte clare și c_1, c_2 textele criptate corespunzătoare;
- Atunci:
  $c_1 \cdot c_2 \mod N = m_1^e \cdot m_2^e \mod N = (m_1 \cdot m_2)^e \mod N$
- Întrebare: Cum poate un adversar să determine mesajul clar m_1, cunoscând textul criptat corespunzător c_1 printr-un atac cu text criptat ales?
- Răspuns: Adversarul alege un $m_2 \in \mathbb{Z}_n$ oarecare și cere decriptarea textului criptat $m_2^e \cdot c_1 \mod N$;

Securitate - Problema 3

Problema 3: Atac cu text criptat ales

- Fie m_1, m_2 2 texte clare și c_1, c_2 textele criptate corespunzătoare;
- Atunci:
  $c_1 \cdot c_2 \mod N = m_1^e \cdot m_2^e \mod N = (m_1 \cdot m_2)^e \mod N$
- Întrebare: Cum poate un adversar să determine mesajul clar m_1, cunoscând textul criptat corespunzător c_1 printr-un atac cu text criptat ales?
- Răspuns: Adversarul alege un $m_2 \in \mathbb{Z}_n$ oarecare și cere decriptarea textului criptat $m_2^e \cdot c_1 \mod N$;
- Adversarul primește un messaj clar $m_3 = m_1 \cdot m_2 \mod N$, apoi determină $m_1 = m_3 \cdot m_2^{-1} \mod N$.

Securitate - Problema 4

Problema 4: Utilizarea multiplă a modulului

- Cunoscând e, d, N cu $(e, \phi(N)) = 1$ se poate determina eficient factorizarea lui N;

Securitate - Problema 4

Problema 4: Utilizarea multiplă a modulului

- Cunoscând e, d, N cu $(e, \phi(N)) = 1$ se poate determina eficient factorizarea lui N;
- Întrebare: Este corect să se utilizeze mai multe perechi de chei care folosesc același modul?

Securitate - Problema 4

Problema 4: Utilizarea multiplă a modulului

- Cunoscând e, d, N cu $(e, \phi(N)) = 1$ se poate determina eficient factorizarea lui N;
- Întrebare: Este corect să se utilizeze mai multe perechi de chei care folosesc același modul?
- Răspuns: NU! Fie 2 perechi de chei:

$$pk_1 = (N, e_1);\ sk_1 = (N, d_1)$$
$$pk_2 = (N, e_2);\ sk_2 = (N, d_2)$$

Securitate - Problema 4

Problema 4: Utilizarea multiplă a modulului

- Cunoscând e, d, N cu $(e, \phi(N)) = 1$ se poate determina eficient factorizarea lui N;
- Întrebare: Este corect să se utilizeze mai multe perechi de chei care folosesc același modul?
- Răspuns: NU! Fie 2 perechi de chei:

$$pk_1 = (N, e_1);\ sk_1 = (N, d_1)$$
$$pk_2 = (N, e_2);\ sk_2 = (N, d_2)$$

- Posesorul perechii (pk_1, sk_1) factorizează N, apoi determină $d_2 = e_2^{-1} \mod \phi(N)$.

Sistemul de criptare RSA

- Folosim construcția prezentată în prelegerea anterioară, utilizând RSA ca TDP:

Construcție

Fie (Enc, Dec) *un sistem de criptare simetric sigur cu autentificarea mesajelor definit peste* $\mathbb{Z}_N$ *și* $H : \mathbb{Z}_N \to \mathcal{K}$ *o funcție hash. Definim un sistem de criptare asimetrică peste în felul următor:*

- $\mathrm{Enc_{pk}(m)} = (y, c) = (RSA_{pk}(x), Enc_k(m))$, *unde* $k = H(x)$ *și* $x \leftarrow^R \mathbb{Z}_N$;
- $\mathrm{Dec_{sk}(y, c)} = Dec_k(c)$, *unde* $k = H(x)$ *și* $x = RSA_{sk}(y)$;

Important de reținut!

- RSA este cel mai cunoscut și mai utilizat algoritm cu cheie publică;
- Textbook RSA NU trebuie utilizat!