Securitatea Sistemelor Informatice

- Curs 9.2 -
RSA

Adela Georgescu

Facultatea de Matematică și Informatică
Universitatea din București
Anul universitar 2022-2023, semestrul I

Funcții one-way

- Reprezinta o primitiva criptografica minima, necesara si suficienta pentru criptarea cu cheie secreta dar si pentru codurile de autentificare a mesajelor
- O functie f one-way este usor de calculat si dificil de inversat

Definiție

O funcție $f : \{0,1\}^* \rightarrow \{0,1\}^*$ este one-way dacă următoarele două condiții sunt îndeplinite:

1. $\mathbb{U}$*șor de calculat: Exista un algoritm polinomial pentru calculul lui f*
2. $\mathbb{D}$*ificil de inversat: Pentru orice algoritm polinomial $\mathcal{A}$, exista o functie neglijabila negl asa incat*

$$Pr[Invert_{\mathcal{A},f}(n) = 1] \leq negl(n)$$

Functii one-way

- Problema factorizarii numerelor mari in produs de doua numere prime de aceeasi lungime este one-way ...

Functii one-way

- Problema factorizarii numerelor mari in produs de doua numere prime de aceeasi lungime este one-way ...
- ... însa nu poate fi folosita direct pentru criptografie

Functii one-way

- Problema factorizarii numerelor mari in produs de doua numere prime de aceeasi lungime este one-way ...
- ... însa nu poate fi folosita direct pentru criptografie
- In schimb, introducem o problema apropiata de problema factorizarii pe baza careia putem construi sisteme de criptare

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 $\phi(N) = (p-1)(q-1)$;
- Dacă se cunoaște factorizarea lui N, atunci $\phi(N)$ este ușor de calculat

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 $\phi(N) = (p-1)(q-1)$;
- Dacă se cunoaște factorizarea lui N, atunci $\phi(N)$ este ușor de calculat
- Fixăm e cu $gcd(e, \phi(N)) = 1$. Atunci $(x^e)^d = x^{ed \bmod \phi(N)} = x \bmod N = \bmod N = (x^d)^e$

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 $\phi(N) = (p-1)(q-1)$;
- Dacă se cunoaște factorizarea lui N, atunci $\phi(N)$ este ușor de calculat
- Fixăm e cu $gcd(e, \phi(N)) = 1$. Atunci $(x^e)^d = x^{ed \bmod \phi(N)} = x \bmod N = \bmod N = (x^d)^e$
- x^d se numeste radacina de ordin e a lui x modulo 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 $\phi(N) = (p-1)(q-1)$;
- Dacă se cunoaște factorizarea lui N, atunci $\phi(N)$ este ușor de calculat
- Fixăm e cu $gcd(e, \phi(N)) = 1$. Atunci $(x^e)^d = x^{ed \text{ mod } \phi(N)} = x \text{ mod } N = \bmod N = (x^d)^e$
- x^d se numeste radacina de ordin e a lui x modulo N
- Dacă p și q se cunosc, atunci putem calcula $\phi(N)$ și $d = e^{-1}$ mod $\phi(N)$

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 $\phi(N) = (p-1)(q-1)$;
- Dacă se cunoaște factorizarea lui N, atunci $\phi(N)$ este ușor de calculat
- Fixăm e cu $gcd(e, \phi(N)) = 1$. Atunci $(x^e)^d = x^{ed \bmod \phi(N)} = x \bmod N = \bmod N = (x^d)^e$
- x^d se numeste radacina de ordin e a lui x modulo N
- Dacă p și q se cunosc, atunci putem calcula $\phi(N)$ și $d = e^{-1}$ mod $\phi(N)$
- Daca p și q nu se cunosc

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 $\phi(N) = (p-1)(q-1)$;
- Dacă se cunoaște factorizarea lui N, atunci $\phi(N)$ este ușor de calculat
- Fixăm e cu $gcd(e, \phi(N)) = 1$. Atunci $(x^e)^d = x^{ed \bmod \phi(N)} = x \bmod N = \bmod N = (x^d)^e$
- x^d se numeste radacina de ordin e a lui x modulo N
- Dacă p și q se cunosc, atunci putem calcula $\phi(N)$ și $d = e^{-1} \bmod \phi(N)$
- Daca p și q nu se cunosc
  - calculul lui $\phi(N)$ este la fel de dificil precum factorizarea lui 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 $\phi(N) = (p-1)(q-1)$;
- Dacă se cunoaște factorizarea lui N, atunci $\phi(N)$ este ușor de calculat
- Fixăm e cu $gcd(e, \phi(N)) = 1$. Atunci $(x^e)^d = x^{ed \bmod \phi(N)} = x \bmod N = \bmod N = (x^d)^e$
- x^d se numeste radacina de ordin e a lui x modulo N
- Dacă p și q se cunosc, atunci putem calcula $\phi(N)$ și $d = e^{-1}$ mod $\phi(N)$
- Daca p și q nu se cunosc
  - calculul lui $\phi(N)$ este la fel de dificil precum factorizarea lui N
  - calculul lui d este la fel de dificil precum factorizarea lui N

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

- Consideram algoritmul $\mathrm{GenRSA}(1^n)$ care are ca output (N, e, d) unde $ed = 1 \bmod \phi(N)$

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

- Consideram algoritmul $\mathrm{GenRSA}(1^n)$ care are ca output (N, e, d) unde $ed = 1 \bmod \phi(N)$
- Considerăm experimentul RSA pentru un algoritm $\mathcal{A}$ și un parametru n.
  1. Execută GenRSA și obține (N, e, d);

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

- Consideram algoritmul $\mathrm{GenRSA}(1^n)$ care are ca output (N, e, d) unde $ed = 1 \bmod \phi(N)$
- 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 $RSA - inv_{\mathcal{A},GenRSA(n)}$

- Consideram algoritmul $\mathrm{GenRSA}(1^n)$ care are ca output (N, e, d) unde $ed = 1 \bmod \phi(N)$
- 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 $RSA - inv_{\mathcal{A},GenRSA(n)}$

- Consideram algoritmul $\mathrm{GenRSA}(1^n)$ care are ca output (N, e, d) unde $ed = 1 \bmod \phi(N)$
- 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;
- Un algoritm GenRSA poate fi construit pe baza unui număr compus împreună cu factorizarea lui;

Algorithm 1 GenRSA

Input: n

Output: N, e, d

1: **genereaza** p și q prime pe n-biti; $N = p \cdot q$
2: $\phi(N) = (p-1)(q-1)$
3: **gasește** e a.î. $gcd(e, \phi(N)) = 1$
4: **calculează** $d := e^{-1} \bmod \phi(N)$
5: **return** N, e, d

GenRSA

- Valoarea lui e aleasa pare ca nu afecteaza dificultatea problemei RSA

GenRSA

- Valoarea lui e aleasa pare ca nu afecteaza dificultatea problemei RSA
- în practică se folosește $e = 3$ sau $e = 16$ pentru exponențiere eficientă
- Daca N este usor de factorizat, atunci problema RSA este usoara
- Pentru ca problema RSA să poată fie dificilă, trebuie ca N-ul ales în GenRSA să fie dificil de factorizat în produs de două numere prime;
- 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.

Exemplu

- Presupunem $(N, p, q) = (143, 11, 13)$. Atunci $\phi(N) = 120$.
- Alegem e asa incat $gcd(e, \phi(N)) = 1$, fie $e = 7$.
- Calculăm $d = e^{-1} \bmod \phi(N)$ si obtinem $d = 103$. Deci output-ul algoritmului GenRSA este (143,7,103).

Textbook RSA

- Definim sistemul de criptare *Textbook RSA* pe baza problemei prezentată anterior;
  1. Se rulează GenRSA pentru a determina N, e, d.
     - Cheia publică este: $pk = (N, e)$;
     - Cheia privată este $sk = d$;
  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 $m = c^d \mod N$.
- Sistemul de criptare este corect pentru ca $\mathbf{Dec}_{sk}(\mathbf{Enc}_{pk}(m)) = m$ astfel: $(m^e)^d \bmod N = m^{ed \bmod \Phi(N)} \bmod N = m^1 \bmod N = m$

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: **Utilizarea multiplă a modulului**

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

Securitate - Problema 2

Problema 2: **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 2

Problema 2: **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 $d_2 = e_2^{-1} \mod \phi(N)$.

Important de reținut!

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