Securitatea Sistemelor Informatice

- Curs 10.1 -
PKCS

Adela Georgescu

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

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: $(m^e)^d \bmod N = m^{ed \bmod \Phi(N)} \bmod N = m^1 \bmod N = m$

Padded RSA

- Am văzut că Textbook RSA este nesigur;

Padded RSA

- Am văzut că Textbook RSA este nesigur;
- Eliminăm una dintre problemele principale, aceea este că sistemul este determinist;

Padded RSA

- Am văzut că Textbook RSA este nesigur;
- Eliminăm una dintre problemele principale, aceea este că sistemul este determinist;
- Introducem în acest sens Padded RSA;

Padded RSA

- Am văzut că Textbook RSA este nesigur;
- Eliminăm una dintre problemele principale, aceea este că sistemul este determinist;
- Introducem în acest sens Padded RSA;
- Ideea este să se adauge un număr aleator (*pad*) la mesajul clar înainte de criptare;

Padded RSA

- Am văzut că Textbook RSA este nesigur;
- Eliminăm una dintre problemele principale, aceea este că sistemul este determinist;
- Introducem în acest sens Padded RSA;
- Ideea este să se adauge un număr aleator (*pad*) la mesajul clar înainte de criptare;
- Notăm în continuare cu n parametrul de securitate (conform GenRSA).

Padded RSA

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 $m \in \{0,1\}^{l(n)}$, alege $r \xleftarrow{R} \{0,1\}^{|N|-l(n)-1}$, interpretează $r||m$ ca un element în $\mathbb{Z}_N$ și întoarce $c = (r||m)^e \mod N$;

3. **Dec**: dată o cheie secretă (N, d) și un mesaj criptat $c \in \mathbb{Z}_N$, calculează $c^d \mod N$ și întoarce ultimii $l(n)$ biți.

Padded RSA

- Pentru $l(n)$ foarte mare, atunci este posibil un atac prin forță brută care verifică toate valorile posibile pentru r;

Padded RSA

- Pentru $l(n)$ foarte mare, atunci este posibil un atac prin forță brută care verifică toate valorile posibile pentru r;
- Pentru $l(n)$ mic se obține securitate CPA:

Padded RSA

- Pentru $l(n)$ foarte mare, atunci este posibil un atac prin forță brută care verifică toate valorile posibile pentru r;
- Pentru $l(n)$ mic se obține securitate CPA:

Teoremă

Dacă problema RSA este dificilă, atunci Padded RSA cu $l(n) = O(\log n)$ este CPA-sigură.

PKCS #1 v1.5

- martie 1998 - Laboratoarele RSA introduc PKCS #1 v1.5;

PKCS #1 v1.5

- martie 1998 - Laboratoarele RSA introduc PKCS #1 v1.5;
- PKCS = Public-Key Cryptography Standard;

PKCS #1 v1.5

- martie 1998 - Laboratoarele RSA introduc PKCS #1 v1.5;
- PKCS = Public-Key Cryptography Standard;
- Folosește Padded RSA;

PKCS #1 v1.5

- martie 1998 - Laboratoarele RSA introduc PKCS #1 v1.5;
- PKCS = Public-Key Cryptography Standard;
- Folosește Padded RSA;
- Utilizat în HTTPS, SSL/TLS, XML Encryption.

PKCS #1 v1.5

- Notăm k lungimea modulului N în bytes: $2^{8(k-1)} \leq N < 2^{8k}$;

PKCS #1 v1.5

- Notăm k lungimea modulului N în bytes: $2^{8(k-1)} \leq N < 2^{8k}$;
- Mesajele m care se criptează se consideră multiplii de 8 biți, de maxim $k - 11$ bytes;

PKCS #1 v1.5

- Notăm k lungimea modulului N în bytes: $2^{8(k-1)} \leq N < 2^{8k}$;
- Mesajele m care se criptează se consideră multiplii de 8 biți, de maxim $k - 11$ bytes;
- Criptarea se realizează astfel:

$$(00000000||00000010||r||00000000||m)^e \mod N$$

PKCS #1 v1.5

- Notăm k lungimea modulului N în bytes: $2^{8(k-1)} \leq N < 2^{8k}$;
- Mesajele m care se criptează se consideră multiplii de 8 biți, de maxim $k - 11$ bytes;
- Criptarea se realizează astfel:

$$(00000000||00000010||r||00000000||m)^e \mod N$$

- r este ales aleator, pe $k - D - 3$ bytes nenuli, unde D este lungimea lui m în bytes;

Securitatea PKCS #1 v1.5

- Se crede că este CPA-sigur, dar acest lucru nu este demonstrat;

Securitatea PKCS #1 v1.5

- Se crede că este CPA-sigur, dar acest lucru nu este demonstrat;
- Cu siguranță însă nu este CCA-sigur;

Securitatea PKCS #1 v1.5

- Se crede că este CPA-sigur, dar acest lucru nu este demonstrat;
- Cu siguranță însă nu este CCA-sigur;
- În 1998, D.Bleichenbacher publică un atac care bazându-se pe faptul că serverul web (HTTPS) întoarce eroare dacă primii 2 octeți nu sunt **02**;

Securitatea PKCS #1 v1.5

- Se crede că este CPA-sigur, dar acest lucru nu este demonstrat;
- Cu siguranță însă nu este CCA-sigur;
- În 1998, D.Bleichenbacher publică un atac care bazându-se pe faptul că serverul web (HTTPS) întoarce eroare dacă primii 2 octeți nu sunt **02**;
- Scopul adversarului este să decripteze un text c;

Securitatea PKCS #1 v1.5

- Se crede că este CPA-sigur, dar acest lucru nu este demonstrat;
- Cu siguranță însă nu este CCA-sigur;
- În 1998, D.Bleichenbacher publică un atac care bazându-se pe faptul că serverul web (HTTPS) întoarce eroare dacă primii 2 octeți nu sunt **02**;
- Scopul adversarului este să decripteze un text c;
- Adversarul transmite către server $c' = r^e \cdot c \mod N$;

Securitatea PKCS #1 v1.5

- Se crede că este CPA-sigur, dar acest lucru nu este demonstrat;
- Cu siguranță însă nu este CCA-sigur;
- În 1998, D.Bleichenbacher publică un atac care bazându-se pe faptul că serverul web (HTTPS) întoarce eroare dacă primii 2 octeți nu sunt **02**;
- Scopul adversarului este să decripteze un text c;
- Adversarul transmite către server $c' = r^e \cdot c \mod N$;
- Răspunsul serverului indică adversarului dacă c' este valid (i.e. începe cu **02**);

Securitatea PKCS #1 v1.5

- Se crede că este CPA-sigur, dar acest lucru nu este demonstrat;
- Cu siguranță însă nu este CCA-sigur;
- În 1998, D.Bleichenbacher publică un atac care bazându-se pe faptul că serverul web (HTTPS) întoarce eroare dacă primii 2 octeți nu sunt **02**;
- Scopul adversarului este să decripteze un text c;
- Adversarul transmite către server $c' = r^e \cdot c \mod N$;
- Răspunsul serverului indică adversarului dacă c' este valid (i.e. începe cu **02**);
- Adversarul folosește răspunsul primit pentru a determina informații despre m;

Securitatea PKCS #1 v1.5

- Se crede că este CPA-sigur, dar acest lucru nu este demonstrat;
- Cu siguranță însă nu este CCA-sigur;
- În 1998, D.Bleichenbacher publică un atac care bazându-se pe faptul că serverul web (HTTPS) întoarce eroare dacă primii 2 octeți nu sunt **02**;
- Scopul adversarului este să decripteze un text c;
- Adversarul transmite către server $c' = r^e \cdot c \mod N$;
- Răspunsul serverului indică adversarului dacă c' este valid (i.e. începe cu **02**);
- Adversarul folosește răspunsul primit pentru a determina informații despre m;
- Repetă atacul până determină mesajul m.

OAEP

- octombrie 1998 - Laboratoarele RSA introduc un nou standard PKCS demonstrat CCA-sigur...

OAEP

- octombrie 1998 - Laboratoarele RSA introduc un nou standard PKCS demonstrat CCA-sigur...
- ...în modelul $\mathcal{ROM}$ (Random Oracle Model);

OAEP

- octombrie 1998 - Laboratoarele RSA introduc un nou standard PKCS demonstrat CCA-sigur...
- ...în modelul $\mathcal{ROM}$ (Random Oracle Model);
- Este vorba despre PKCS $\#1$ v2.0 sau OAEP $=$ Optimal Asymmetric Encryption Standard;

OAEP

- OAEP este de fapt o modalitate de padding;

OAEP

- OAEP este de fapt o modalitate de padding;
- OAEP este o metodă **nedeterministă** și **inversabilă** care transformă un mesaj m de lungime $n/2$ într-o secvență m' de lungime $2n$;

OAEP

- OAEP este de fapt o modalitate de padding;
- OAEP este o metodă **nedeterministă** și **inversabilă** care transformă un mesaj m de lungime $n/2$ într-o secvență m' de lungime $2n$;
- Notăm $m' = OAEP(m, r)$, unde r este o secvență aleatoare (de lungime n);

OAEP

- OAEP este de fapt o modalitate de padding;
- OAEP este o metodă **nedeterministă** și **inversabilă** care transformă un mesaj m de lungime $n/2$ într-o secvență m' de lungime $2n$;
- Notăm $m' = OAEP(m, r)$, unde r este o secvență aleatoare (de lungime n);
- RSA-OAEP criptează $m \in \{0,1\}^{n/2}$ folosind cheia publică (N, e) ca:
$$(OAEP(m,r))^e \mod N$$

OAEP

- OAEP este de fapt o modalitate de padding;
- OAEP este o metodă **nedeterministă** și **inversabilă** care transformă un mesaj m de lungime $n/2$ într-o secvență m' de lungime $2n$;
- Notăm $m' = OAEP(m, r)$, unde r este o secvență aleatoare (de lungime n);
- RSA-OAEP criptează $m \in \{0, 1\}^{n/2}$ folosind cheia publică (N, e) ca:

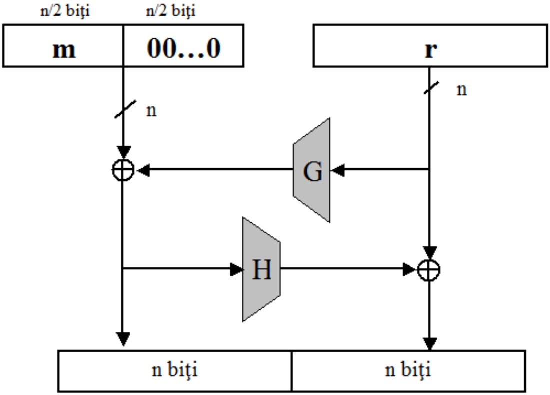
$$(OAEP(m, r))^e \mod N$$

- RSA-OEAP decriptează c folosind cheia secretă (N, d) ca:

$$(m, r) = OAEP^{-1}(c^d \mod N)$$

OAEP

- OAEP este definit astfel:



Notații

- $G, H =$ funcții hash
- $m \in \{0,1\}^{n/2}$ mesajul
- $r \leftarrow^{R} \{0,1\}^{n}$
- se obține $OAEP(m, r)$ pe $2n$ biți

Important de reținut!

- Utilizarea RSA în practică:
  PKCS #1 v1.5 și PKCS #1 v2.0 (OAEP)