CS 4770: Cryptography

CS 6750: Cryptography and Communication Security

Alina Oprea
Associate Professor, CCIS
Northeastern University

March 26 2017

Outline

- RSA encryption in practice
  - Transform RSA trapdoor into CCA secure encryption
  - PKCS standard and attacks
  - OAEP standard
- ElGamal encryption
  - Based on Diffie-Hellman key exchange
  - Proof of security based on DDH assumption
- Digital signatures
  - Integrity in public-key world
  - Equivalent of MACs
  - Public verifiability

Trapdoor functions

Def: a trapdoor function $X \longrightarrow Y$ is a triple of efficient algorithms (Gen, F, F^{-1})

- Gen(): randomized alg. outputs a key pair (pk, sk)
- $F(pk,\cdot)$: deterministic alg. that defines a function $X \longrightarrow Y$
- $F^{-1}(sk,\cdot)$: defines a function $Y \longrightarrow X$ that inverts $F(pk,\cdot)$

Correctness: $\forall (pk, sk)$ output by G

$$\forall x \in X: \quad F^{-1}(sk, F(pk, x)) = x$$

Trapdoor permutation $F: X \longrightarrow X$, $F^{-1}: X \longrightarrow X$

The RSA trapdoor permutation

Gen(): Choose random primes $p,q \approx 1024$ bits.

Set **N=pq**. **RSA modulus**

Choose integers **e , d** s.t. $e \cdot d = 1 \pmod{\varphi(N)}$

Output pk = (N, e) , sk = (d)

F(pk, x): $\mathbb{Z}_N^* \to \mathbb{Z}_N^*$; $F(pk, x) = x^e \bmod N$

$F^{-1}(sk, y) = y^d \bmod N$

$$y^d = \mathbf{RSA(x)^d} = x^{ed} = x^{k\varphi(N)+1} = \left(x^{\varphi(N)}\right)^k \cdot \mathbf{x} = x$$

The RSA assumption

RSA assumption: RSA is trapdoor permutation

For all PPT algorithms A:

$$\Pr\Big[\ A(N,e,y) = y^{1/e}\ \Big] < \text{negligible}$$

where $p,q \xleftarrow{R}$ n-bit primes, $N \leftarrow pq$, $y \xleftarrow{R} Z_N^*$

RSA public-key encryption

(E, D): authenticated encryption scheme

H: $Z_N \to K$ where K is key space of (E_s, D_s)

- **Gen**(): generate RSA parameters:
  pk = (N,e), sk = (d)
- **Enc**(pk, m): (1) choose random x in Z_N
  (2) $y \leftarrow RSA(x) = x^e$, $k \leftarrow H(x)$
  (3) output (y , E(k,m)) ⟶ Randomized
- **Dec**(sk, (y, c)): output D($H(RSA^{-1}(y))$, c)

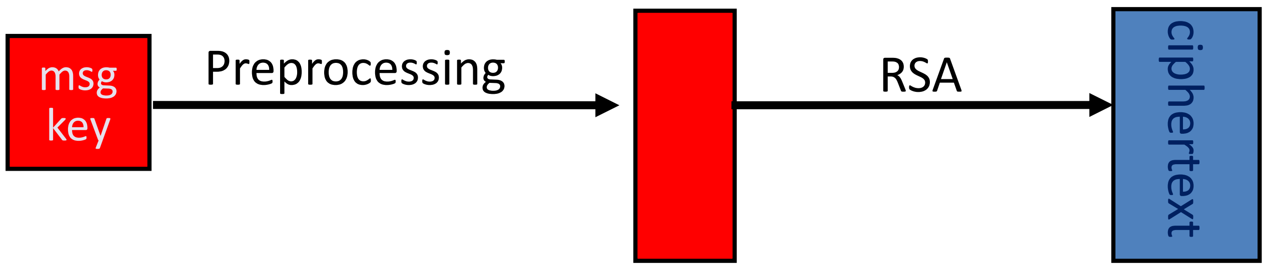
CCA secure

ISO Standard

RSA encryption in practice

Never use textbook RSA.

RSA in practice (since ISO standard is not often used) :

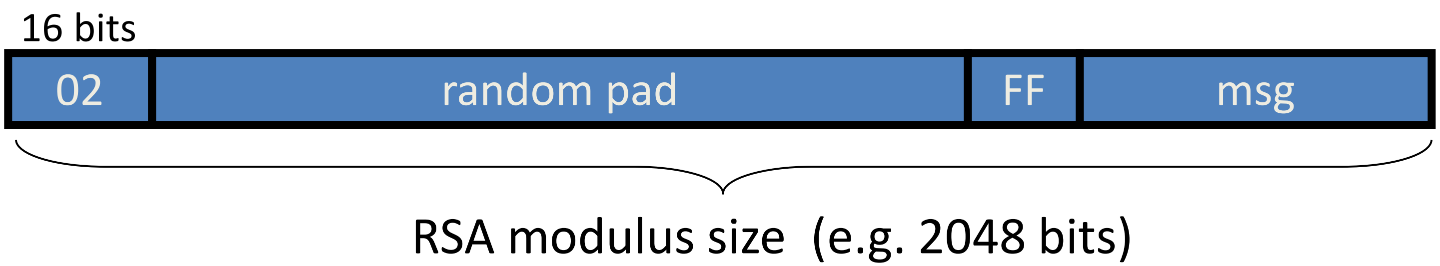


Main questions:

- How should the preprocessing be done?
- Can we argue about security of resulting system?

PKCS1 v1.5

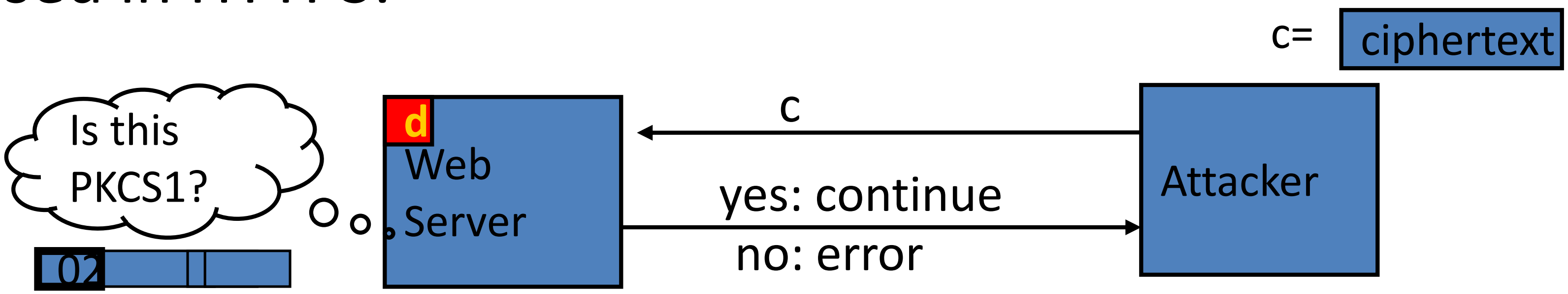
PKCS1 mode 2: (encryption)



- Resulting value is RSA encrypted
- Widely deployed, e.g. in HTTPS

Attack on PKCS1 v1.5 (Bleichenbacher 1998)

PKCS1 used in HTTPS:

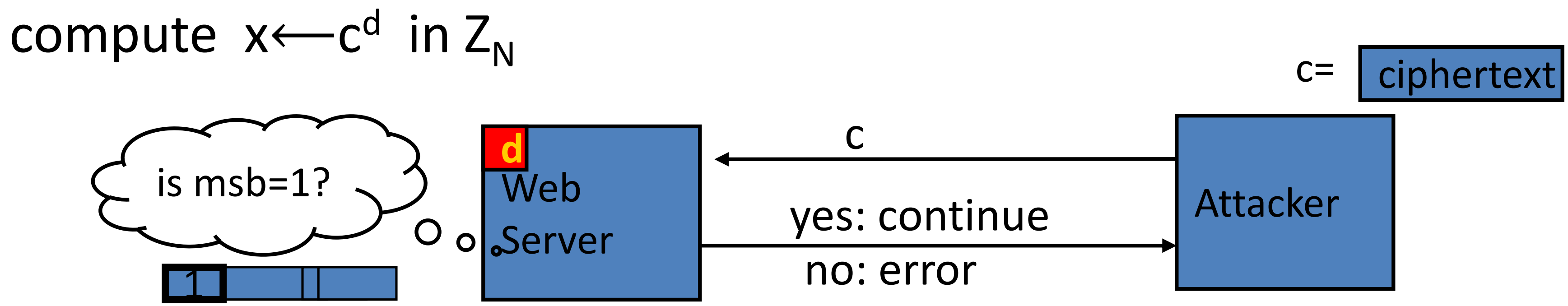


$\Rightarrow$ attacker can test if 16 MSBs of plaintext = '02'

Chosen-ciphertext attack: to decrypt a given ciphertext c do:

- Choose $r \in Z_N$. Compute $c' \leftarrow r^e \cdot c = (r \cdot \text{PKCS1}(m))^e$
- Send c' to web server and use response

Simple example - Bleichenbacher



Suppose N is $N = 2^n$ (an invalid RSA modulus). Then:

- Sending c reveals msb(x)
- Sending $\mathbf{2^e \cdot c = (2x)^e}$ in Z_N reveals msb(2x mod N) = $\mathbf{msb_2(x)}$
- Sending $\mathbf{4^e \cdot c = (4x)^e}$ in Z_N reveals msb(4x mod N) = $\mathbf{msb_3(x)}$
- … and so on to reveal all of x

HTTPS Defense (RFC 5246)

Attacks discovered by Bleichenbacher resulted in the following change:

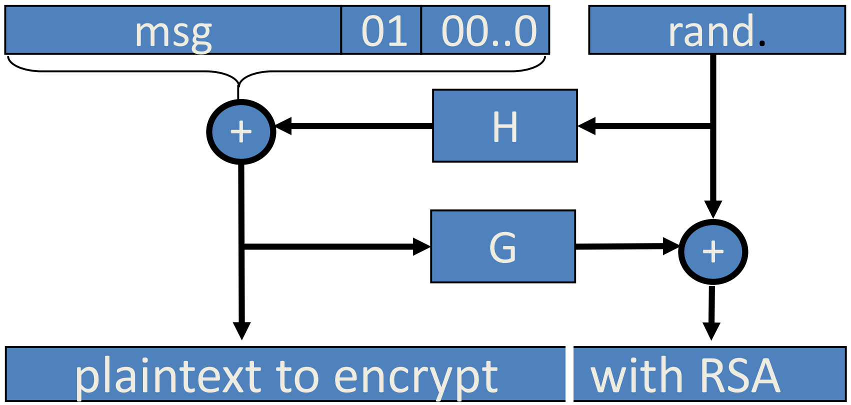
1. *Decrypt the message to recover plaintext m*
2. *If the PKCS#1 padding is not correct*
   3. *Generate a string* ***R*** *of 46 random bytes*
   4. *pre_master_secret =* ***R***

Still no proof of security

PKCS1 v2.0: OAEP

New preprocessing function: OAEP [BR94]

check pad
on decryption.
reject CT if invalid.



$\in \{0,1\}^{n-1}$

Theorem [FOPS'01] : RSA is a trapdoor permutation $\Rightarrow$
RSA-OAEP is CCA secure when H,G are *random functions*

in practice: use SHA-256 for H and G

Review: the Diffie-Hellman protocol (1977)

Fix a finite cyclic group G (e.g $G = (Z_p)^*$) of order q

Fix a generator g in G (i.e. $G = \{1, g, g^2, g^3, \dots, g^{q-1}\}$)

Alice **Bob**

choose random **x** in {1,…,q}

choose random **y** in {1,…,q}

$$A = g^x \longrightarrow$$

$$B = g^y \longleftarrow$$

$$\mathbf{B^x} = (g^y)^x = \boxed{\mathbf{k_{AB} = g^{xy}}} = (g^x)^y = \mathbf{A^y}$$

ElGamal: converting to pub-key enc. (1984)

Fix a finite cyclic group G (e.g $G = (Z_p)^*$) of order q

Fix a generator g in G (i.e. $G = \{1, g, g^2, g^3, \ldots, g^{q-1}\}$)

Alice **Bob**

choose random $\mathbf{x}$ in $\{1,\ldots,q\}$ choose random $\mathbf{y}$ in $\{1,\ldots,q\}$

$h = g^x$ $\longrightarrow$

compute $k=g^{xy} = h^y$

$\mathrm{Enc}(m) = [\ u=g^y, c=k\cdot m\ ]$

$\longleftarrow$

ElGamal: converting to pub-key enc. (1984)

Fix a finite cyclic group G (e.g $G = (Z_p)^*$) of order q

Fix a generator g in G (i.e. $G = \{1, g, g^2, g^3, \ldots, g^{q-1}\}$)

Alice **Bob**

choose random $\mathbf{x}$ in $\{1,\ldots,q\}$

choose random $\mathbf{y}$ in $\{1,\ldots,q\}$

$h = g^x$ $\longrightarrow$

compute $k=g^{xy} = h^y$

$\mathrm{Enc}(m) = [\ u=g^y, c= k\cdot m\ ]$

$\longleftarrow$

To decrypt (u,c):

compute $k = u^x$

and decrypt $m = k^{-1}\cdot c$

The ElGamal system (a modern view)

G: finite cyclic group of order q

We construct a pub-key enc. system (Gen, Enc, Dec):

- Key generation Gen:
  - choose random generator g in G and random x in Z_q
  - output $sk = x$, $pk = (g, h=g^x)$

Enc(pk=(g,h), m) :

$y \longleftarrow Z_q$, $u \longleftarrow g^y$, $k \longleftarrow h^y$

$c \longleftarrow k \cdot m$

output (u, c)

Dec(sk=x, (u,c)) :

$k \longleftarrow u^x$

$m \longleftarrow k^{-1} \cdot c$

output m

ElGamal performance

Enc(pk=(g,h), m) :

$y \longleftarrow Z_q$, $u \longleftarrow g^y$, $v \longleftarrow h^y$

Dec(sk=x, (u,c)) :

$k \longleftarrow u^x$

Encryption: 2 exp. (fixed basis)

- Can pre-compute [$g^{(2\text{^}i)}$, $h^{(2\text{^}i)}$ for $i=1,\ldots,\log_2 n$]
- 3x speed-up (or more)

Decryption: 1 exp. (variable basis)

Decisional Diffie-Hellman

Let **G** be a finite cyclic group and **g** generator of G

$$G = \{ 1 , g , g^2 , g^3 , \ldots , g^{q-1} \}$$

q is the order of G

Definition: We say that **DDH is hard in G** if for all PPT adversaries D:

$$| \Pr[D(g^x ,g^y ,g^{xy}) = 1] - \Pr[D(g^x ,g^y ,g^z) = 1] | < \text{negligible}$$

G, q and g are public and known to D

x, y, z are chosen uniformly at random in {1,…q-1}

Security

Theorem: Let G be a cyclic group of order q. Assuming that the DDH problem is hard, then El-Gamal encryption is CPA secure.

In particular, for every PPT adversary A attacking the CPA security of El-Gamal:

$$\Pr[\mathrm{Exp}_{\Pi,A}^{\mathrm{CPA}}(n) = 1] = 1/2 + \text{negligible}(n)$$

Proof of security - Intuition

Π **Enc(pk=(g,h), m)**

$y \leftarrow Z_q$, $u \leftarrow g^y$

$c \leftarrow h^y \cdot m$ ($= g^{xy} \cdot m$)

output (u, c)

1. Success of adversary to break **Π** and **Π'** in CPA game is similar

Under the assumption that DDH is hard !

Π' **Enc'(pk=(g,h), m)**

$y \leftarrow Z_q$, $u \leftarrow g^y$, $z \leftarrow Z_q$

$c \leftarrow g^z \cdot m$

output (u, c)

2. Success of adversary to break **Π'** in CPA game is negligible

Proof of security – step 1

1. Success of adversary to break **Π** and **Π'** in CPA game is similar

Assume that DDH is hard.
For any PPT adversary A:

$$\left|\mathbf{Pr}\left[\mathrm{Exp}_{\Pi,A}^{\mathrm{CPA}}(n)=\mathbf{1}\right]-\mathbf{Pr}\left[\mathrm{Exp}_{\Pi',A}^{\mathrm{CPA}}(n)=\mathbf{1}\right]\right| \leq \mathrm{negl(n)}$$

- Let A be a PPT adversary in CPA game
- We build D a distinguisher for DDH
- D knows (G, q, g) and is given input (g^x, g^y, w)
- World 1: w = g^{xy}
- World 0: w = g^z

Proof of security – step 1

1. Success of adversary to break **Π** and **Π’** in CPA game is similar

Assume that DDH is hard.
Then for any PPT adversary A:

$$|\mathbf{Pr}[\mathrm{Exp}^{\mathrm{CPA}}_{\Pi,A}(n) = \mathbf{1}] - \mathbf{Pr}[\mathrm{Exp}^{\mathrm{CPA}}_{\Pi',A}(n) = \mathbf{1}]| \leq \mathrm{negl(n)}$$

- D runs A. A chooses two messages m_0 and m_1
- D picks a bit b at random and send $c = w \cdot m_b$
- World 1: $c = g^{xy} \cdot m$; D simulates exactly scheme **Π**
- World 0: $c = g^{z} \cdot m$; D simulates exactly scheme **Π’**
- D outputs what A outputs

Proof of security – step 1

1. Success of adversary to break **Π** and **Π'** in CPA game is similar

Assume that DDH is hard.

Then for any PPT adversary A:

$$|\mathbf{Pr}[\mathrm{Exp}_{\Pi,A}^{\mathrm{CPA}}(n) = \mathbf{1}] - \mathbf{Pr}[\mathrm{Exp}_{\Pi',A}^{\mathrm{CPA}}(n) = \mathbf{1}]| \leq \mathrm{negl(n)}$$

- D runs A.
- D outputs what A outputs
- $|\Pr[\mathrm{Exp}_{\Pi,A}^{\mathrm{CPA}}(n) = 1] - \Pr[\mathrm{Exp}_{\Pi',A}^{\mathrm{CPA}}(n) = 1]| =$

$|\ \Pr[\ D(g^x, g^y, g^{xy}) = 1\ ] - \Pr[\ D(g^x, g^y, g^z) = 1\ ]\ |$, which is negligible(n)

Proof of security – step 2

2. Success of adversary to break **Π’** in CPA game is negligible

For any PPT adversary A:

$$\textbf{Compute Pr}[\mathrm{Exp}^{\mathrm{CPA}}_{\Pi',A}(n) = \mathbf{1}]$$

- Let A be an adversary in CPA game for Π'
- A chooses two messages m_0 and m_1
- A receives $(g^y, g^z \cdot m_b)$
- First part is independent on message
- If z is random, then g^z is random in G
  - For any v in G, $\Pr[g^z \cdot m_b = v] = \Pr[g^z = (m_b)^{-1} \cdot v] = 1/q$
  - $g^z \cdot m_b$ does not reveal any information about m_b

Conclusion

- For any PPT adversary A:

$$\mathbf{Pr}[\mathrm{Exp}_{\Pi,A}^{\mathrm{CPA}}(n) = \mathbf{1}] \leq |\mathbf{Pr}[\mathrm{Exp}_{\Pi,A}^{\mathrm{CPA}}(n) = \mathbf{1}]$$

$$-\mathbf{Pr}[\mathrm{Exp}_{\Pi,'A}^{\mathrm{CPA}}(n) = \mathbf{1}]| + \mathbf{Pr}[\mathrm{Exp}_{\Pi,'A}^{\mathrm{CPA}}(n) = \mathbf{1}]$$

= ½ + negligible(n)

- El-Gamal encryption is CPA secure under DDH assumption

Key insights

- Trapdoor permutations (e.g., RSA) are not a secure encryption method
  - They are deterministic
- Secure public-key encryption can be constructed from trapdoor permutations
  - ISO standard – CCA secure
  - PKCS1 v1.5 (susceptible to padding oracles)
  - OAEP – CCA secure
- Discrete log based schemes
  - El Gamal encryption constructed from Diffie-Hellman
  - CPA security based on hardness of DDH

Acknowledgement

Some of the slides and slide contents are taken from
http://www.crypto.edu.pl/Dziembowski/teaching
and fall under the following:

We have also used slides from Prof. Dan Boneh online cryptography course at Stanford University:

http://crypto.stanford.edu/~dabo/courses/OnlineCrypto/