Practical Security in Public-Key Cryptography

4th International Conference on Information Security and Cryptography Seoul - Korea

December 6th 2001

David Pointcheval
Département d'Informatique
ENS - CNRS

David.Pointcheval@ens.fr http://www.di.ens.fr/~pointche

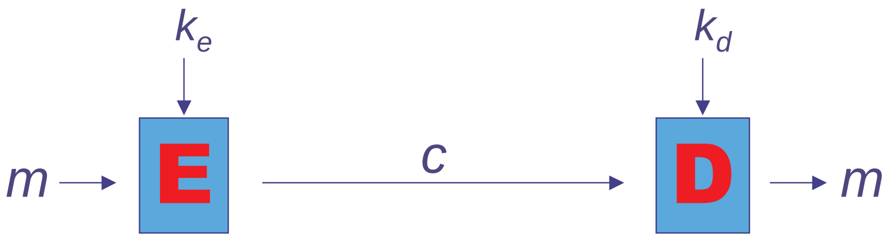
Overview

- Provable Security
- Computational Assumptions
- Exact/Practical Security
- Signature
- Encryption
- Conclusion

Asymmetric Encryption

Encryption Algorithm **E**

Decryption Algorithm **D**

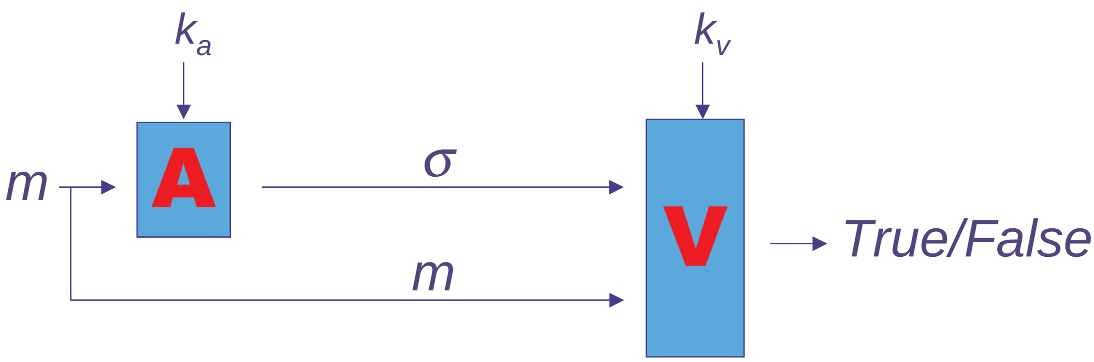


Security: it is impossible to get back m just from c, k_e, **E** and **D** (without k_d)

Signature

Authentication Algorithm **A**

Verification Algorithm **V**



Security: it is impossible to produce a new valid pair (m,σ) (without k_a)

Provable Security

For a provably secure protocol,

- one formally defines
  the security notions to achieve
- one makes precise
  the computational assumptions
- one designs a protocol
- one exhibits a "reduction"

Security Notions

Depending on the security concerns,
one defines

- the goals that an adversary
  may would like to reach
- the means/information
  available to the adversary

Computational Assumptions

To build such an asymmetric primitive, one needs *(trapdoor)* ***one-way functions***:

$x \to y = f(x)$ is easy (Encryption, Verification)

$y = f(x) \to x$ is difficult (Decryption, Signature)

The assumptions are thus

- a specific function is one-way
- a specific problem is intractable

Integer Factoring - RSA

- One-way function
  - $p, q \to n = pq$ easy
  - $n = pq \to p, q$ seems difficult (FACT)
- The **RSA Problem** (1978):
  - given $n=pq$, e and y
  - compute x such that $x^e = y \bmod n$

The DL Problems

- Let **G** = (<g>, ×) be any finite cyclic group
- One-way function
  - $x \rightarrow y = g^x$ easy
  - $y = g^x \rightarrow x$ seems difficult (DL Problem)
- The **Diffie-Hellman Problem** (1976):
  - given $A = g^a$ and $B = g^b$
  - compute DH(A,B) = $C = g^{ab}$
- The **Decisional Diffie-Hellman Problem**:
  - given A, B and C in <g>
  - decide whether C = DH(A,B)

"Reductionist" Security

One provides a reduction from a "difficult" problem **P** to an attack *Atk*:

the adversary *A* reaches the "prohibited" goals
⇒ *A* can be used to break **P**

P intractable ⇒ scheme secure

Cost of the reduction:

- complexity theory: polynomial reduction
  ⇒ asymptotic security (for huge parameters)
- exact security: exact/efficient reduction
  ⇒ helps to find the good parameters

Ideal Assumptions

Efficient reductions are very rare

$\Rightarrow$ one makes some ideal assumptions:

- ideal random hash function:
  random oracle model
- ideal symmetric encryption:
  ideal cipher model
- ideal group:
  generic (group) model

= generic adversary w.r.t. to some objects:
resp. hash function, encryption, group

Practical Security

- "Reductionist" Security:
  - if the adversary can break the security notion with probability ε within time t (expected time T)
  - the underlying problem can be solved with probability ε' within time t' (expected time T')
- Exact Security:

  ε' and t' are explicitly given from ε and t
- Practical Security:

  the relations are BOTH very tight $\Rightarrow T' \approx T$

Signature Schemes

- Goals:
  - Total Break: to recover the secret key
  - Universal Forgery: to sign any message
  - Existential Forgery: new valid pair (m,σ)
- Attacks:
  - No-message Attacks: with the public key only
  - Known-message Attacks: with some pairs
  - Adaptive Chosen-message Attacks:
    access to a signing oracle

Secure Signature

A Signature Scheme is said SECURE
if it prevents existential forgeries
under adaptive chosen-message attacks

$$\boxed{\Pr\left[V_{k_v}(m,\sigma)=1 \middle| (m,\sigma)\leftarrow A^{\Sigma}(k_v)\right]}$$ succ negligible

Then, the signature guarantees:

- the identity of the sender
- the non-repudiation:
  the sender won't be able to deny it later

DL-based Signatures

G $= \langle g \rangle$, q and g : **common data**
x : **private** key $y = g^x$: **public** key

$\sigma = (e,s)$

Schnorr's signature of the message m :
$k \in \mathbf{Z}_q$, $r = g^k$, $e = h(m,r)$, $s = k - xe \bmod q$

Verification of (m,σ) : $u = g^s y^e$ $(= g^{k-xe} g^{xe})$
test whether $e = h(m,u)$?

Existential Forgery
under chosen-message attacks
= computation of $x = \log_g y$

Exact Security

Idea: *Forking Lemma*

(Pointcheval-Stern EC '96)

$\mathcal{A}$ succeeds in expected time $T \Rightarrow$ one solves the DL problem in expected time $T' = 207\, q_h\, T$

For a security level in T, $q_h = 2^k$: $T' \geq 2^{2k+7}$ $(=2^{167})$

Nothing better for any DL-based signature

$$\mathcal{A} \xrightarrow{h(m,r) \quad e} (e,s)$$
$$\phantom{\mathcal{A}} \xrightarrow{\quad e' \quad} (e',s')$$

$$g^s y^e = r = g^{s'} y^{e'}$$
$$\Rightarrow g^{s-s'} = y^{e'-e}$$

RSA-based Signatures

$n=pq$, e : **public** key $d = e^{-1} \bmod \varphi(n)$: **private** key

Signature of the message $m \in \mathbf{Z}_n$: $\sigma = m^d \bmod n$

Verification of (m,σ): test whether $\sigma^e = m \bmod n$

Weak security, unless one signs $h(m)$

FDH-RSA (Bellare-Rogaway EC '96)

Attack in time T $\Rightarrow$ RSA in time $T' = q_s\, T$

… better, but still bad.

PSS-RSA: attack in time $T \Rightarrow$ RSA in time $T' \approx T$

… practical security!

Encryption Schemes

- Security (impossibility to):
  - One-wayness: recover the whole plaintext
  - Semantic Security: learn any information
- Attacks:
  - Chosen-Plaintext: with the public-key only
  - Chosen-Ciphertext (adaptively): access to a decryption oracle

Main Security Levels

- OW-CPA: *(the weakest)*

$$\Pr_{m,r}\left[\mathcal{A}(c)=m \,\middle|\, c=\mathbf{E}(m;r)\right]$$

= Succ negligible

- IND-CCA: *(the strongest* - BDPR C '98*)*

$$2\Pr_{r,b}\left[\mathcal{A}_2^{\mathbf{D}}(m_0,m_1,c,s)=b \,\middle|\, \begin{array}{c}(m_0,m_1,s)\leftarrow\mathcal{A}_1^{\mathbf{D}}(k_e)\\ c\leftarrow\mathbf{E}(m_b,r)\end{array}\right]-1$$

= Adv negligible

Example I: RSA Encryption

- $n = pq$, product of large primes
- e, relatively prime to $\varphi(n) = (p\text{-}1)(q\text{-}1)$
- n, e : **public** key
- $d = e^{-1} \bmod \varphi(n)$: **private** key

$$\mathbf{E}(m) = m^e \bmod n \qquad \mathbf{D}(c) = c^d \bmod n$$

OW-CPA = RSA problem

$\text{Succ}^{\text{ow-cpa}}(t) = \text{Succ}^{\text{rsa}}(t)$

Example II: El Gamal Encryption

- **G** = (<g>, ×) group of prime order q
- x : **private** key
- $y=g^x$: **public** key

$$\mathbf{E}(m) = (g^a, y^a m) \rightarrow (c,d) \qquad \mathbf{D}(c,d) = d / c^x$$

OW-CPA = CDH Assumption
$\text{Succ}^{\text{ow-cpa}}(t) \leq \text{Succ}^{\text{cdh}}(t)$

IND-CPA = DDH Assumption
$\text{Adv}^{\text{ind-cpa}}(t) \leq 2\ \text{Adv}^{\text{ddh}}(t)$

Chosen-Ciphertext Attacks

We have efficient encryption schemes
with practical security ($T' \approx c\ T$)
but for OW-CPA, or best IND-CPA, only.

- Cramer-Shoup, in 1998,
  proposed the first efficient example
  - not as efficient as El Gamal (twice as slow)
  - IND-CCA = DDH: weak problem

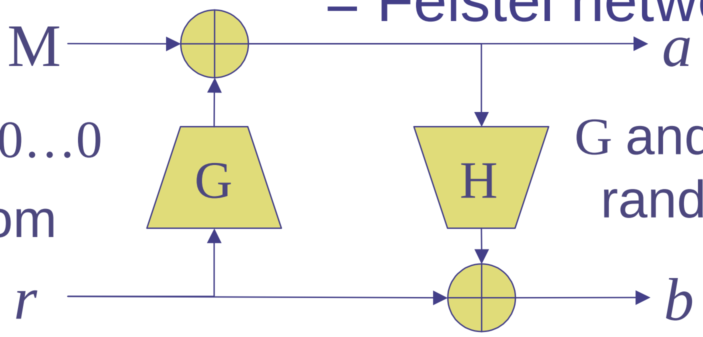
But many practical schemes in the ROM
what about their practical security?

Conversion: OAEP

Bellare-Rogaway EC '94

Optimal Asymmetric Encryption Padding:

= Feistel network:



$M = m||0\ldots0$

r random

G and H random functions

E(m): Compute a,b and output $f\,(a||b)$

D(c): Compute $a||b = f^{-1}(c)$

invert the Feistel network $\rightarrow$ (M,r)

and output m (*if the redundancy holds*)

OAEP: Security

It provides an optimal conversion of any *trapdoor partial one-way* ***permutation***

(Fujisaki-Okamoto-Pointcheval-Stern C '01)

into an IND-CCA cryptosystem

Optimal:

- Efficiency: just 2 more hashing
- Ciphertext: the shortest as possible

OAEP: Reduction

$$\mathbf{E}(\mathrm{M}, e) = f\left(a = \mathrm{M} \oplus \mathrm{G}(r) \,\|\, b = r \oplus \mathrm{H}(a)\right) \rightarrow c$$

1 bit of M $\Leftrightarrow$ guess r $\Leftrightarrow$ guess a $\Leftrightarrow$ guess (a,b)

$\mathrm{Adv}^{\text{ind-cpa}}(t) \approx \mathrm{Succ}^{f}(t)$

$\mathbf{D}(c) = f^{-1}(c) \rightarrow (a, b)$
$r = \mathrm{H}(a) \oplus b$ and $\mathrm{M} = a \oplus \mathrm{G}(r)$
if $\mathrm{M} = m \| 0\ldots0$ then $m = x$ else "reject"

Valid ciphertext $\Leftrightarrow$ (r,a) asked to G and H
$\Leftrightarrow$ known plaintext: **Plaintext Awareness**

Simulation of the decryption: try any (r,a) pair

OAEP: Practical Security

$$T' \geq T + q_{\mathrm{G}} \times q_{\mathrm{H}}\, T_f$$

Integer factoring:

- 512-bit modulus: time $\approx 2^{56}$
- 1024-bit modulus: time $\approx 2^{72}$

Security-level of RSA-OAEP:

- 512-bit modulus: time $\approx 2^{28}$
- 1024-bit modulus: time $\approx 2^{36}$

For a provably secure level in 2^{64}:
more than 4000 bits!

Other Conversions

Trapdoor one-way permutation is a strong restriction (only one candidate!)

- Fujisaki-Okamoto (PKC '99):
  any IND-CPA into IND-CCA
- Fujisaki-Okamoto (Crypto '99)
- Pointcheval (PKC '00):
  any OW-CPA into IND-CCA

But in all of them,
the decryption algorithm is not optimal

New Conversion: REACT

Okamoto-Pointcheval RSA '01

Rapid Enhanced-security
Asymmetric Cryptosystem Transform

E$(m,r||s)$ = $a = f(x, r)$ with $x \in \mathcal{X}$, $r \in \mathcal{R}$
$b = k \oplus m$ where $k = \mathrm{G}(x)$
$c = \mathrm{H}(m,x,a,b)$

D(a,b,c): Compute $x = f^{-1}(a)$ and $k = \mathrm{G}(x)$
extract $m = k \oplus b$
if $c = \mathrm{H}(m,x,a,b)$ and $x \in \mathcal{X}$ then output m

Practical Security

$$G : \mathcal{X} \to \{0,1\}^{\ell_G} \qquad H : \{0,1\}^* \to \{0,1\}^{\ell_H}$$

If an adversary $\mathcal{A}$ against IND-CCA reaches an advantage $\mathrm{Adv}^{\mathcal{A}}$ after q_G, q_H and $q_{\mathcal{D}}$ queries to G, H and $\mathcal{D}$ resp. in time t one can invert f after q_G+q_H tests $x=f^{-1}(y)$ within time $t' \le t + (q_G+q_H)\, T_{\text{test}}$ with probability greater than $\dfrac{\mathrm{Adv}^{\mathcal{A}}}{2} - \dfrac{q_{\mathcal{D}}}{2^{\ell_H}}$

Therefore $T' \approx 2\, T$

Applications

Security relies on the Gap-Problems

Okamoto-Pointcheval PKC '2001

- RSA-REACT: IND-CCA = RSA
  1024-bit modulus: security-level $\approx 2^{72}$
  (To be compared with 2^{36} for RSA-OAEP!)
- EG-REACT: IND-CCA = Gap DH $\approx$ CDH

Efficiency: with any symmetric encryption which is just semantically secure

Example: EG-REACT

$\mathbb{G}$ is any group, and g of order q

G and H: two hash functions

$\mathbf{E}$, $\mathbf{D}$: symmetric encryption scheme

x : **private** key

$y = g^x$: **public** key

$\mathbf{E}(m)$: $a \leftarrow_R \mathbb{Z}_q$, $R \leftarrow_R \mathbb{G}$

$A \leftarrow g^a$, $A' \leftarrow R\, y^a$

$k \leftarrow \mathrm{G}(R)$, $B \leftarrow \mathbf{E}_k(m)$,

$C \leftarrow \mathrm{H}(R, m, A, A', B)$

$\longrightarrow (A, A', B, C)$

$\mathbf{D}(A, A', B, C)$: $R \leftarrow A'/A^x$,

$k \leftarrow \mathrm{G}(R)$, $m \leftarrow \mathbf{D}_k(B)$,

check whether $C = \mathrm{H}(R, m, A, A', B)$

Conclusion

Provable security requires

1. formal security notions
2. well-defined computational assumptions
3. reductions between the assumptions break and the security notions break

For practical impact

1. reduction : VERY efficient
2. computational problem: VERY strong