Provable Security
Asymmetric Encryption

DEA – January 29th 2004

David Pointcheval
CNRS-ENS, Paris, France

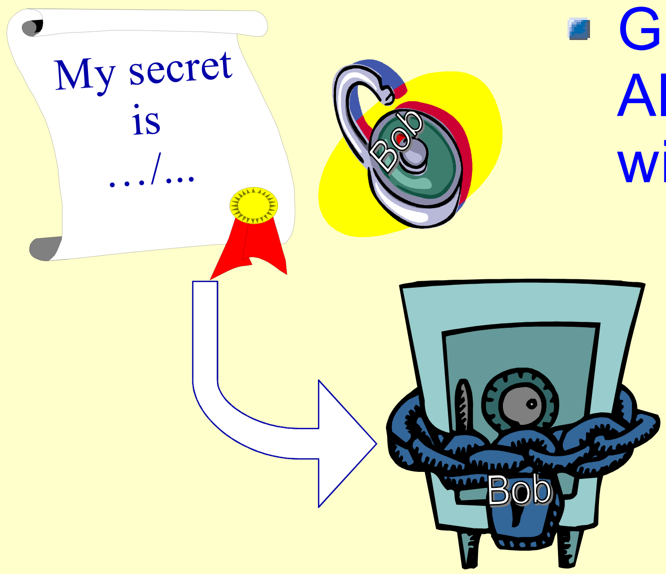
Summary

- Introduction
- Computational Assumptions
- Provable Security
- Asymmetric Encryption
- Example

Summary

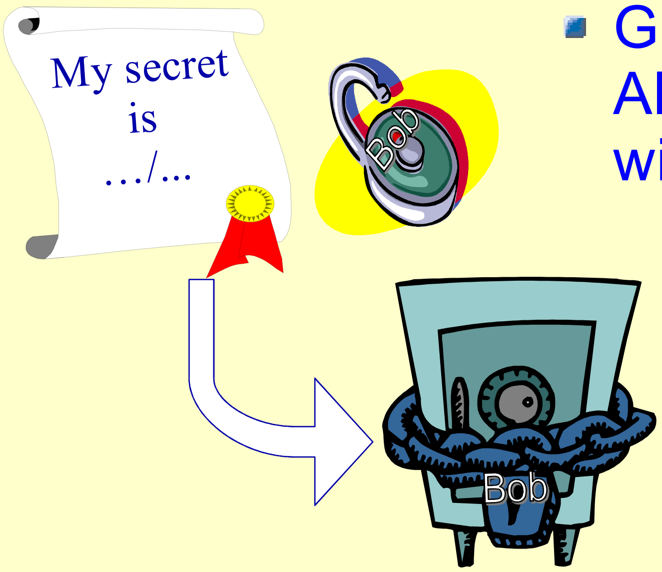
- **Introduction**
- Computational Assumptions
- Provable Security
- Asymmetric Encryption
- Example

Encryption / decryption attack



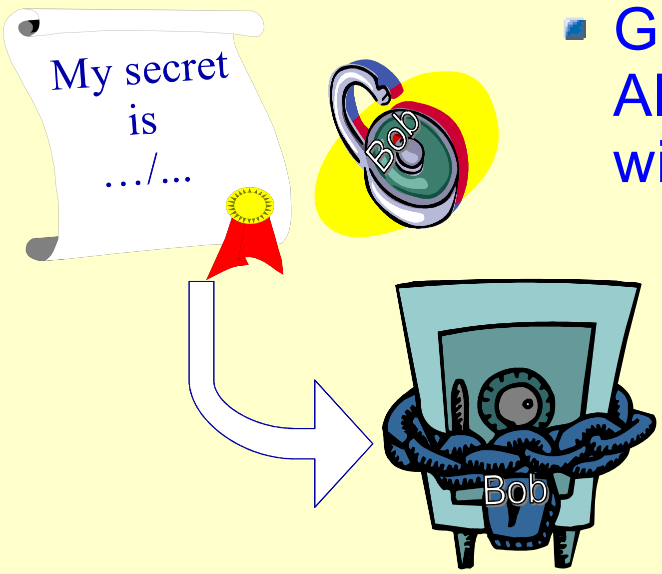
- Granted Bob's public key, Alice can lock the safe, with the message inside
  (*encrypt the message*)

Encryption / decryption attack



- Granted Bob's public key, Alice can lock the safe, with the message inside (*encrypt the message*)
- Alice sends the safe to Bob no one can unlock it *(impossible to break)*

Encryption / decryption attack

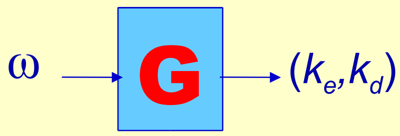


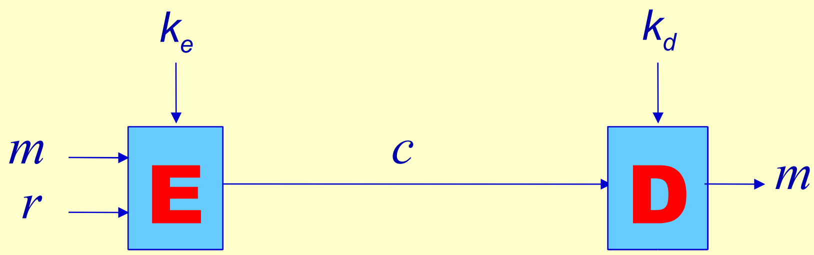
- Granted Bob's public key, Alice can lock the safe, with the message inside (*encrypt the message*)
  - Excepted Bob, granted his private key *(Bob can decrypt)*
- Alice sends the safe to Bob no one can unlock it *(impossible to break)*

Encryption Scheme

3 algorithms:

- **G** - key generation
- **E** - encryption
- **D** - decryption





Conditional Secrecy

The ciphertext comes from $c = \mathbf{E}_{k_e}(m; r)$

- The encryption key k_e is public
- A unique m satisfies the relation (with possibly several r)

At least exhaustive search on m and r can lead to m, maybe a better attack!

⇒ unconditional secrecy impossible

Algorithmic assumptions

Summary

Integer Factoring and RSA

One-Way Function

- Multiplication/Factorization:
  - $p, q \mapsto n = p.q$ easy (quadratic)
  - $n = p.q \mapsto p, q$ difficult (super-polynomial)

Integer Factoring and RSA

- Multiplication/Factorization:
  - $p, q \mapsto n = p.q$ easy (quadratic)
  - $n = p.q \mapsto p, q$ difficult (super-polynomial)

One-Way Function

- RSA Function, from $\mathbf{Z}_n$ in $\mathbf{Z}_n$ (with $n=pq$)

  for a fixed exponent e

  Rivest-Shamir-Adleman 1978

  - $x \mapsto x^e \bmod n$ easy (cubic)
  - $y=x^e \bmod n \mapsto x$ difficult (without p or q)

    $x = y^d \bmod n$ where $d = e^{-1} \bmod \varphi(n)$

RSA Problem

Integer Factoring and RSA

- Multiplication/Factorization:
  - $p, q \mapsto n = p.q$ easy (quadratic)
  - $n = p.q \mapsto p, q$ difficult (super-polynomial)

One-Way Function

- RSA Function, from $\mathbf{Z}_n$ in $\mathbf{Z}_n$ (with $n=pq$)

  for a fixed exponent e

  Rivest-Shamir-Adleman 1978

  - $x \mapsto x^e \bmod n$ easy (cubic)
  - $y=x^e \bmod n \mapsto x$ difficult (without p or q)

    $x = y^d \bmod n$ where $d = e^{-1} \bmod \varphi(n)$

encryption

Integer Factoring and RSA

- Multiplication/Factorization:
  - $p, q \mapsto n = p.q$ easy (quadratic)
  - $n = p.q \mapsto p, q$ difficult (super-polynomial)
- RSA Function, from $\mathbb{Z}_n$ in $\mathbb{Z}_n$ (with $n=pq$)

  for a fixed exponent e

  Rivest-Shamir-Adleman 1978
  - $x \mapsto x^e \bmod n$ easy (cubic)
  - $y=x^e \bmod n \mapsto x$ difficult (without p or q)

    $x = y^d \bmod n$ where $d = e^{-1} \bmod \varphi(n)$

One-Way Function

difficult to break

Integer Factoring and RSA

- Multiplication/Factorization:
  - $p, q \mapsto n = p.q$ easy (quadratic)
  - $n = p.q \mapsto p, q$ difficult (super-polynomial)
- RSA Function, from $\mathbb{Z}_n$ in $\mathbb{Z}_n$ (with $n=pq$)

  for a fixed exponent e

  Rivest-Shamir-Adleman 1978
  - $x \mapsto x^e \bmod n$ easy (cubic)
  - $y=x^e \bmod n \mapsto x$ difficult (without p or q)

    $x = y^d \bmod n$ where $d = e^{-1} \bmod \varphi(n)$

One-Way Function

trapdoor

key

decryption

The RSA Problem

Let $n=pq$ where p and q are large primes

The RSA problem: for a fixed exponent e

$$\mathrm{Succ}_{n,e}^{\mathrm{rsa}}(\mathbf{A}) = \Pr_{y \in Z_n^*}\left[y = x^e \bmod n \,|\, \mathbf{A}(y) = x\right]$$

The Discrete Logarithm

- Let **G** = $(\langle g \rangle, \times)$ be any finite cyclic group
- For any $y \in$ **G**, one defines

  $\mathrm{Log}_g(y) = \min\{x \geq 0 \mid y = g^x\}$

One-way function

- $x \rightarrow y = g^x$ easy (cubic)
- $y = g^x \rightarrow x$ difficult (super-polynomial)

$$\mathrm{Succ}_g^{\mathrm{dl}}(\mathbf{A}) = \Pr_{x \in Z_q}\left[\mathbf{A}(y) = x \,|\, y = g^x\right]$$

Any Trapdoor …?

- The Discrete Logarithm is difficult and no information could help!
- The Diffie-Hellman Problem (1976):
  - Given $A=g^a$ and $B=g^b$
  - Compute DH(A,B) = $C=g^{ab}$

Clearly CDH ≤ DL: with $a = \mathrm{Log}_g A$, $C = B^a$

$$\mathrm{Succ}_g^{\mathrm{cdh}}(\mathbf{A}) = \Pr_{a,b\in Z_q}\left[\mathbf{A}(A,B)=C \,|\, A=g^a, B=g^b, C=g^{ab}\right]$$

Another DL-based Problem

- The **Decisional Diffie-Hellman Problem**:
  - Given A, B and C in $\langle g \rangle$
  - Decide whether C = DH(A,B)

Clearly DDH ≤ CDH ≤ DL

$$\mathrm{Adv}_g^{\mathrm{ddh}}(\mathbf{A}) = \left| \Pr_{a,b,c\in Z_q}\left[\mathbf{A}(A,B,C)=1 \,|\, A=g^a, B=g^b, C=g^c\right] - \Pr_{a,b\in Z_q}\left[\mathbf{A}(A,B,C)=1 \,|\, A=g^a, B=g^b, C=g^{ab}\right] \right|$$

Complexity Estimates

Estimates for integer factoring Lenstra-Verheul 2000

Modulus (bits)	Mips-Year ($\log_2$)	Operations (en $\log_2$)
512	13	58
1024	35	80
2048	66	111
4096	104	149
8192	156	201

Can be used for RSA too
Lower-bounds for DL in $\mathbb{Z}_p^*$

Summary

- Introduction
- Computational Assumptions
- **Provable Security**
- Asymmetric Encryption
- Example

Algorithmic Assumptions *necessary*

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

RSA Encryption

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

If the RSA problem is easy,
secrecy is not satisfied:
anybody may recover m from c

Algorithmic Assumptions *sufficient?*

Security proofs give the guarantee that the assumption is **enough** for secrecy:

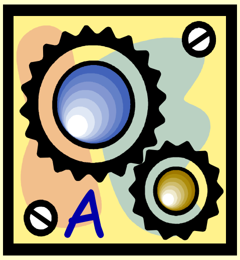
- if an adversary can break the secrecy
- one can break the assumption

⇒ “reductionist” proof

Proof by Reduction

Reduction of a problem **P** to an attack *Atk*:

- Let *A* be an adversary that breaks the scheme
- Then *A* can be used to solve **P**

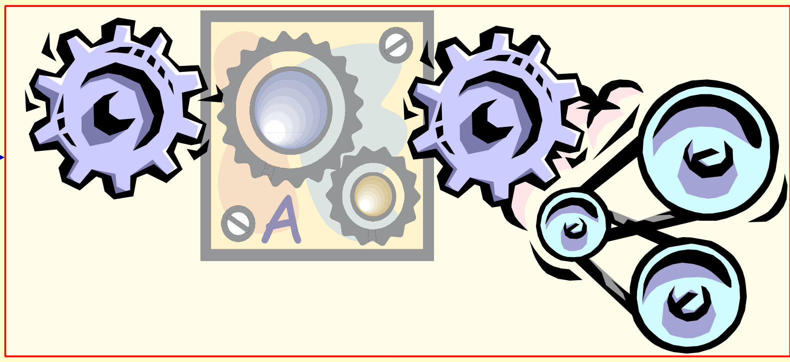


Proof by Reduction

Reduction of a problem **P** to an attack *Atk*:

- Let *A* be an adversary that breaks the scheme
- Then *A* can be used to solve **P**

Instance **I** of **P**



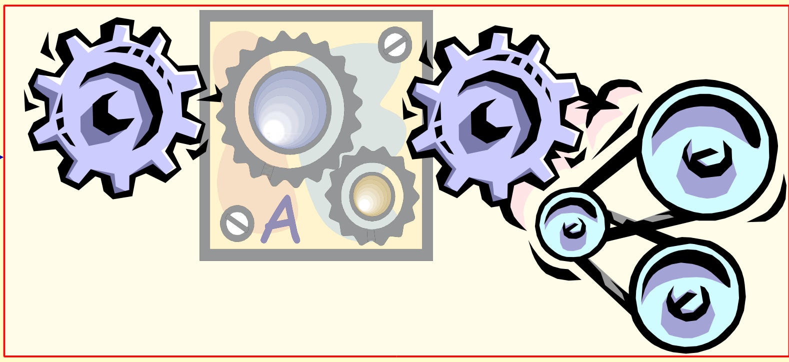
Solution of **I**

Proof by Reduction

Reduction of a problem **P** to an attack *Atk*:

- Let *A* be an adversary that breaks the scheme
- Then *A* can be used to solve **P**

Instance **I** of **P** →



→ Solution of **I**

P intractable ⇒ scheme unbreakable

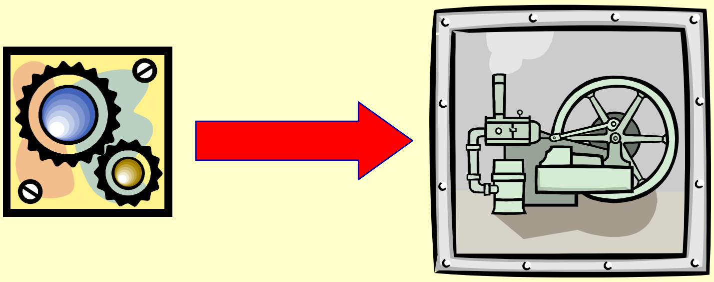
Provably Secure Scheme

To prove the security of a cryptographic scheme, one has to make precise

- the algorithmic assumptions
  - some have been presented
- the security notions to be guaranteed
  - depends on the scheme (see later)
- a reduction:
  an adversary can help
  to break the assumption

Practical Security

Adversary within t



Algorithm against **P** within $t' = T(t)$

- Complexity theory: T polynomial
- Exact Security: T explicit
- Practical Security: T small (linear)

Practical Security: Encryption

- Security bound: 2^{75}
  - and 2^{55} hash queries
- RSA-OAEP
  - 1024 bits $\rightarrow 2^{143}$ (NFS: 2^{80}) ✘
  - 2048 bits $\rightarrow 2^{146}$ (NFS: 2^{111}) ✘
  - 4096 bits $\rightarrow 2^{149}$ (NFS: 2^{149}) ✔
- RSA-BR/REACT: $t' \approx 2t$
  - 1024 bits $\rightarrow 2^{75}$ (NFS: 2^{80}) ✔

$\Rightarrow$ Practical security

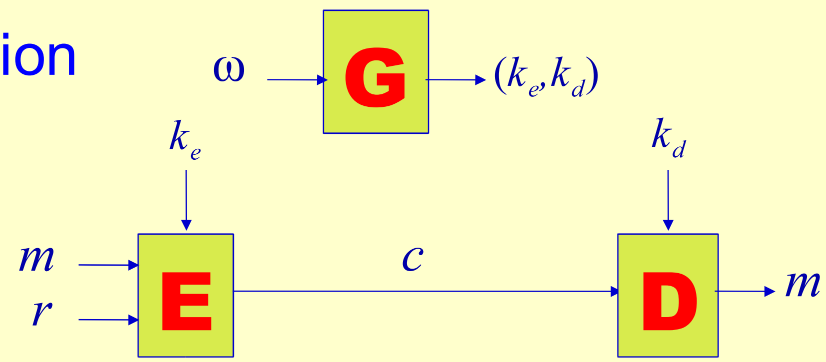
Summary

- Introduction
- Computational Assumptions
- Provable Security
- **Asymmetric Encryption**
- Example

Asymmetric Encryption

An asymmetric encryption scheme π = (**G**,**E**,**D**)
is defined by 3 algorithms:

- **G** – key generation
- **E** – encryption
- **D** – decryption



Security = secrecy : impossible
to recover m from public information
(i.e from c, but without k_d)

Security Notions

According to the needs, one defines

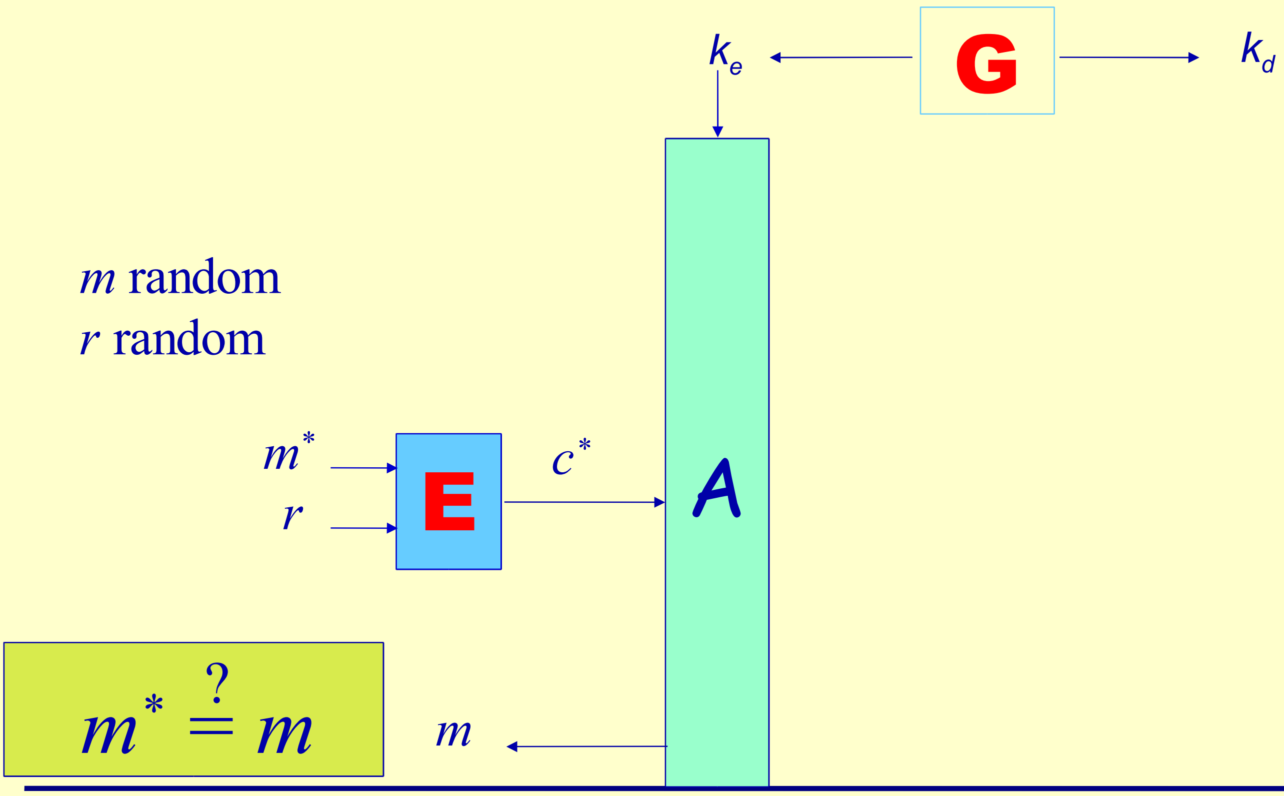
- the goals of an adversary
- the means of an adversary,
  i.e. the available information

Basic Secrecy

- One-Wayness (OW) :
  without the private key, it is computationally impossible to recover the plaintext

$$\mathrm{Succ}^{ow}(\mathbf{A}) = \Pr_{m,r}\left[\mathbf{A}(k_e, c) = m \,\middle|\, c = \mathbf{E}(m;r)\right]$$

One-Wayness



Not Enough

- One-Wayness (OW) :
  - without the private key, it is computationally impossible to recover the plaintext
  - but it does not exclude the possibility of recovering half of the plaintext!
- It is not enough if one already has some information about m:
  - “Subject: XXXXX”
  - “My answer is XXX” (XXX = Yes/No)

Strong Secrecy

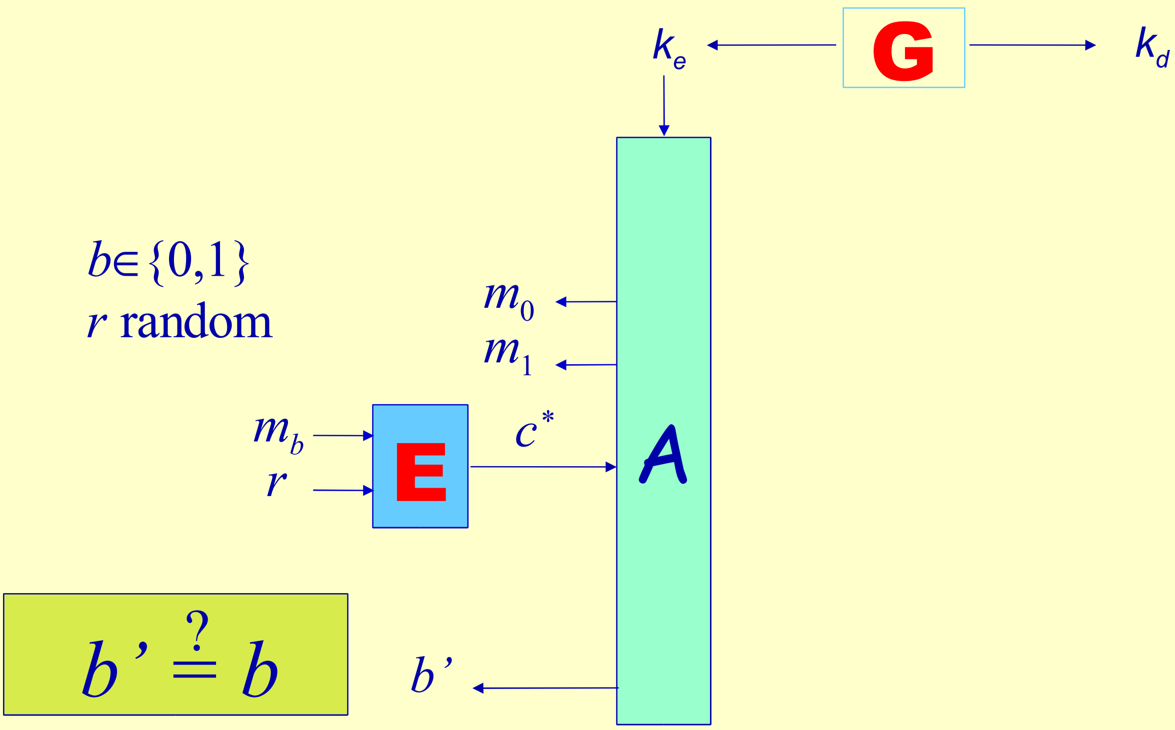
- Semantic Security (IND - Indistinguishability):

  GM 1984

  the ciphertext reveals *no more* information about the plaintext to a **polynomial adversary**

$$\mathrm{Adv}^{ind}(\mathbf{A}) = 2\Pr_{r,b}\left[\mathbf{A_2}(m_0, m_1, c, s) = b \,\middle|\, \begin{array}{c}(m_0, m_1, s) \leftarrow \mathbf{A_1}(k_e) \\ c \leftarrow \mathbf{E}(m_b, r)\end{array}\right] - 1$$

Semantic Security



Non-Malleability

- Non-Malleability (NM):

DDN 1991

No polynomial adversary can derive,
from a ciphertext $c = \mathbf{E}(m;r)$, a second one $c' = \mathbf{E}(m';r')$
so that the plaintexts m and m' are meaningfully related

non-malleability
⇩
semantic security
⇩
one-wayness

Basic Attacks

- Chosen-Plaintext Attacks (CPA)

In public-key cryptography setting,
the adversary can encrypt any message
of its choice, granted the public key

⇒ the basic attack

Improved Attacks

- More information: oracle access
  - reaction attacks
    - oracle which answers, on c,
      whether the ciphertext c is valid or not
  - plaintext-checking attacks
    - oracle which answers, on a pair (m,c),
      whether the plaintext m is really
      encrypted in c or not (whether $m = \mathbf{D}(c)$)

Strong Attacks

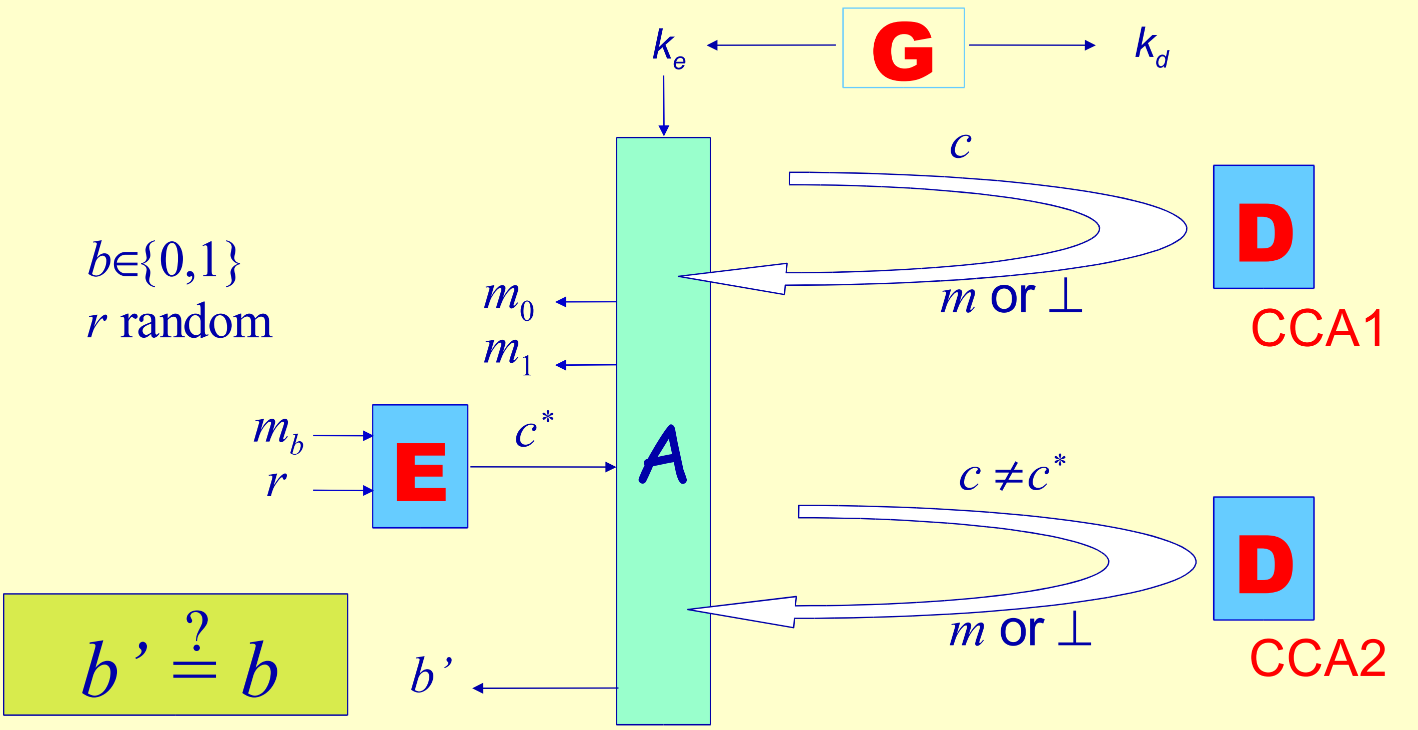
- Chosen-Ciphertext Attacks (CCA)

  The adversary has access to the strongest oracle: the decryption oracle
  (with the natural restriction not to use it on the challenge ciphertext)

The adversary can obtain the plaintext of any ciphertext of its choice (except the challenge)

  - non-adaptive (CCA1) NY 1990
    - only before receiving the challenge
  - adaptive (CCA2) RS 1991
    - unlimited oracle access

IND-CCA2



Relations

BDPR C-1998

Implications and separations

NM-CPA ⇐ NM-CCA1 ⇐ NM-CCA2

⇓ ⇓ ⇕

IND-CPA ⇐ IND-CCA1 ⇐ IND-CCA2

⇓

OW-CPA

minimal security (IND-CPA)

weak security (OW-CPA)

strong security: CCA (NM-CCA2 ⇔ IND-CCA2)

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

Nothing to prove = definition

El Gamal Encryption

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

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

OW-CPA = CDH Assumption

IND-CPA = DDH Assumption

To be proven to see the restrictions

El Gamal: OW-CPA

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

$$\mathrm{Succ}^{ow}(\mathbf{A})=\Pr_{m,r}\left[\mathbf{A}(y,(c,d))=m \,\middle|\, (c,d)=\mathbf{E}(m;a)\right]$$

B is given as input **G** = (<g>, ×) and (A,B)

- $y \leftarrow A$ and $c \leftarrow B$
- choose a random value d : A(y,(c,d)) → m
- output d/m

If m is correct, DH(A,B) = d/m

Succ^{cdh} (B) = Succ^{ow}(A)

El Gamal: IND-CPA

$$\mathrm{Adv}^{ind}(\mathbf{A})=2\Pr_{a,b}\left[\mathbf{A}_2(m_0,m_1,(c,d),s)=b \,\middle|\, \begin{array}{c}(m_0,m_1,s)\leftarrow\mathbf{A}_1(y)\\ (c,d)\leftarrow\mathbf{E}(m_b;a)\end{array}\right]-1$$

B is given as input **G** = (<g>, ×) and (A, B, C)

- $y \leftarrow A$ and $c \leftarrow B$: A$_1$(y) → (m_0, m_1)
- $b \in \{0,1\}$ and $d \leftarrow C\, m_b$: A$_2$(c,d) → b'
- output $\beta = (b = b')$

- Let us assume that m_0, $m_1 \in$ **G**:
  - If C = DH(A,B), Pr[b=b'] = Pr[A(c,d) = b]
  - If $C \neq$ DH(A,B), Pr[b=b'] = 1/2

El Gamal: IND-CPA (Cnt'd)

- If the messages are encoded into **G**:
  - If $C = \mathrm{DH}(A,B)$, $\Pr[b=b'] = \Pr[\mathcal{A}(c,d) = b]$
  - If $C \neq \mathrm{DH}(A,B)$, $\Pr[b=b'] = 1/2$

$$\mathrm{Adv}^{\mathrm{ddh}}(\mathbf{B}) = \Pr[\beta=1 \mid C=\mathrm{CDH}(A,B)] - \Pr[\beta=1 \mid C \neq \mathrm{CDH}(A,B)]$$
$$= \Pr[b'=b] - \frac{1}{2} = \frac{1}{2}\mathrm{Adv}^{ind}(A)$$

$$\mathrm{Adv}(\mathbf{D}) = 2\Pr[b'=b] - 1$$
$$= \Pr[b'=b \mid b=1] + \Pr[b'=b \mid b=0] - 1$$
$$= \Pr[b'=b \mid b=1] - \Pr[b' \neq b \mid b=0]$$
$$= \Pr[b'=1 \mid b=1] - \Pr[b'=1 \mid b=0]$$

Thus,
$\mathrm{Adv}^{ind}(t) \leq 2\,\mathrm{Adv}^{\mathrm{ddh}}(t')$

Strong Security Notions

- Signature: difficult to obtain security against existential forgeries
- Encryption: difficult to reach CCA security
- Maybe possible, but with inefficient schemes
- Inefficient schemes are unuseful in practice:

Everybody wants security,
but only if it is transparent

→ one makes some ideal assumptions

The Random-Oracle Model

Introduced by Bellare-Rogaway ACM-CCS '93

- The most admitted model
- It consists in considering some functions
  as perfectly random functions,
  or replacing them by random oracles:
  - each new query is returned a random answer
  - a same query asked twice receives twice
    the same answer

Modeling a Random Oracle

A usual way to model a random oracle H is to maintain a list Λ_H which contains all the query-answers (x,ρ):

- Λ_H is initially set to an empty list
- A query x to H is answered the following way
  - if for some ρ, $(x,\rho) \in \Lambda_H$, ρ is returned
  - Otherwise,
    - a random ρ is drawn from the appropriate range
    - (x,ρ) is appended to Λ_H
    - ρ is returned

Summary

- Introduction
- Computational Assumptions
- Provable Security
- Asymmetric Encryption
- **Example**

Generic Construction Bellare-Rogaway '93

- Let f be a trapdoor one-way permutation then (with $G \to \{0,1\}^n$ and $H \to \{0,1\}^k$)
- $\mathbf{E}(m;r) = f(r) \parallel m \oplus G(r) \parallel H(m,r)$
- $\mathbf{D}(a,b,c)$:
  - $r = f^{-1}(a)$
  - $m = b \oplus G(r)$
  - $c = H(m,r)$?

IND-CCA2: Game 0

- Adversary $A=(A_1,A_2)$
  - $A_1(f) \rightarrow (m_0,m_1)$
  - One randomly chooses $\beta \in \{0,1\}$ and r^*, and computes $C^* = \mathbf{E}(m_\beta ; r^*) = (a^*,b^*,c^*)$:
    $$a^* = f(r^*),\ b^* = m_\beta \oplus G(r^*),\ c^* = H(m_\beta ,r^*)$$
  - $A_2(C^*) \rightarrow \beta'$

  both with permanent access to
  - the decryption oracle **D** — $q_{\mathbf{D}}$ queries
  - the random oracles G and H — q_G, q_H queries

IND-CCA2: Game 0

- On this probability space, we consider event S: $\beta' = \beta$
- In Game i: S_i
- Note that

$$\Pr[S_0] = 1/2 + \mathrm{Adv}^{ind}(A)/2$$

Indeed, by definition (in the attack game):

$$\mathrm{Adv}^{ind}(A) = 2\Pr[\beta' = \beta] - 1$$

IND-CCA2: Game 1

- Classical simulation of the random oracles

One does not change the distribution:

$$\Pr[\mathtt{S}_1] = \Pr[\mathtt{S}_0]$$

IND-CCA2: Game 2

- We choose h^+, and then set $H(m_\beta,r^*) \leftarrow h^+$
  - For $C^* = \mathbf{E}(m_\beta;r^*)$: $H(m_\beta,r^*) \leftarrow h^+$
  - H simulation: $H(m_\beta,r^*)$ independent

One introduces inconsistencies,
if the adversary asks $H(m_\beta,r^*)$

We consider event $\mathtt{AskR}$: r^* asked to G or H

In Game i: $\mathtt{AskR}_i$

$$|\Pr[\mathtt{S}_2] - \Pr[\mathtt{S}_1]| \leq \Pr[\mathtt{AskR}_2]$$

IND-CCA2: Game 3

- We now start modifying the simulation of the decryption oracle **D**:
  - For a query $(a', b', c') = \mathbf{E}(m'; r')$
    - If $H(m', r')$ has not been asked: rejection

Bad simulation BadS: $c' = H(m', r')$, whereas $H(m', r')$ has not been asked:

$$\Pr[\texttt{BadS}] \leq q_{\mathbf{D}} / 2^k$$

IND-CCA2: Game 4

- We choose r^+, g^+ and h^+, and then set $r^* \leftarrow r^+$, and $G(r^*) \leftarrow g^+$ and $H(m_\beta, r^*) \leftarrow h^+$
  - For $C^* = \mathbf{E}(m_\beta; r^+)$: $G(r^+) \leftarrow g^+$
    $H(m_\beta, r^+) \leftarrow h^+$
  - G simulation: $G(r^+) \leftarrow$ random
    H simulation: $H(m_\beta, r^+) \leftarrow$ random

Event AskR already cancelled: no modification:

$$\Pr[\texttt{S}_4] = \Pr[\texttt{S}_3] \qquad \Pr[\texttt{AskR}_4] = \Pr[\texttt{AskR}_3]$$

IND-CCA2: Game 4

- One randomly chooses r^+, g^+ and h^+
  - $A_1(f) \rightarrow (m_0, m_1)$
  - One randomly chooses $\beta \in \{0,1\}$,
    - $C^* = \mathbf{E}(m_\beta; r^+) = (a^* = f(r^+), b^* = m_\beta \oplus g^+, c^* = h^+)$
  - $A_2(C^*) \rightarrow \beta'$

  with permanent access to
    - the decryption oracle **D**
    - the random oracles G and H: Λ_G and Λ_H
  - and $G(r^+)$ or $H(m_\beta, r^+)$ never asked

$\Pr[\mathtt{S}_4] = 1/2$

IND-CCA2: Game 5

- We now manufacture the challenge ciphertext: we are given $y = f(x)$

$$C^* = (a^* = y, b^* = m_\beta \oplus g^+, c^* = h^+)$$

- This simply defines $r^+ = x$

This does not modidify the probability space:

$$\Pr[\mathtt{AskR}_5] = \Pr[\mathtt{AskR}_4]$$

IND-CCA2: Game 6

- We complete the simulation of the decryption oracle **D**:
  - For a query $(a', b', c') = \mathbf{E}(m'; r')$
    - One looks for $G(r')$ such that $a' = f(r')$
    - Not found: rejection
    - Otherwise: easy decryption

Modification if $H(m', r')$ queried while $G(r')$ is unpredicable, and m' is so too:

$$\Pr[\mathtt{BadS'}] \leq q_H / 2^n$$

IND-CCA2: Game 6

- One is given $y = f(x)$
- One randomly chooses g^+ and h^+
  - $A_1(f) \rightarrow (m_0, m_1)$
  - One randomly chooses $\beta \in \{0,1\}$,
    $C^* = (a^* = y, b^* = m_\beta \oplus g^+, c^* = h^+)$
  - $A_2(C^*) \rightarrow \beta'$
  - with permanent access to
    - the decryption oracle **D**: simulation
    - the random oracles G and H: Λ_G and Λ_H

$$\Pr[\mathtt{AskR}_6] \leq \mathrm{Succ}^{ow}(t')$$
$$t' = t_6 + (q_G + q_H)\, T_f$$

IND-CCA2: Sum up 1

- $\Pr[S_0] = 1/2 + \mathrm{Adv}^{ind}(A)/2$

$$\Pr[S_1] = \Pr[S_0]$$

- $|\Pr[S_2] - \Pr[S_1]| \leq \Pr[\mathsf{AskR}_2]$
- $|\Pr[S_3] - \Pr[S_2]| \leq q_D / 2^k$

$$\Pr[S_4] = \Pr[S_3]$$

- $\Pr[S_4] = 1/2$

$$|\Pr[S_0] - \Pr[S_4]| = \mathrm{Adv}^{ind}(A)/2 \leq \Pr[\mathsf{AskR}_2] + q_D / 2^k$$

IND-CCA2: Sum up 2

- $|\Pr[\mathsf{AskR}_3] - \Pr[\mathsf{AskR}_2]| \leq q_D / 2^k$

$$\Pr[\mathsf{AskR}_5] = \Pr[\mathsf{AskR}_4] = \Pr[\mathsf{AskR}_3]$$

- $|\Pr[\mathsf{AskR}_6] - \Pr[\mathsf{AskR}_5]| \leq q_H / 2^n$
- $\Pr[\mathsf{AskR}_6] \leq \mathrm{Succ}^{ow}(t + (q_G + q_H)\, T_f)$

$$\Pr[\mathsf{AskR}_2] \leq q_D/2^k + q_H/2^n + \mathrm{Succ}^{ow}(t+(q_G+q_H)T_f)$$

IND-CCA2: End

- Simple bookkeeping:
  - one avoids the factor q_D
- An additional variable in Λ_G and Λ_H:
  - $(x,\rho,y) \in \Lambda_G$ means $G(x)=\rho$ and $f(x)=y$
  - $(m,x,\rho,y) \in \Lambda_H$ means $H(m,x)=\rho$ and $f(x)=y$

$$\mathrm{Adv}^{ind}(A)/2 \leq q_D/2^k + 2q_H/2^n + \mathrm{Succ}^{ow}(t+(q_G+q_H)T_f)$$

Practical Security

$$\mathrm{Adv}^{ind}(A)/2 \leq q_D/2^k + 2q_H/2^n + \mathrm{Succ}^{ow}(t+(q_G+q_H)T_f)$$

- Security bound: 2^{75}
  - and 2^{55} hash queries and 2^{30} decryption queries
- Break the scheme within t, invert f within time $t' \leq t + (q_G + q_H)\ T_f \leq t + 2^{55}\ T_f$
  - RSA: 1024 bits → 2^{75} (NFS: 2^{80}) ✔
    2048 bits → 2^{77} (NFS: 2^{111}) ✔
    4096 bits → 2^{79} (NFS: 2^{149}) ✔