Part I

Introduction

1. Introduction to Cryptography
   - What Cryptography is about
   - Classic Goals

What Cryptography is about

Cryptography is the discipline that studies systems (schemes, protocols) that preserve their functionality (their goal) even under the presence of an active disrupter.

What Cryptography is about

Cryptography is the discipline that studies systems (schemes, protocols) that preserve their functionality (their goal) even under the presence of an active disrupter.

Classic Problems/Goals

- **Integrity:** Messages have not been altered
- **Authenticity:** Message comes from sender
- **Secrecy:** Message not known to anybody else

Integrity

Alice wants to be sure that a message has not been modified.

Analogy with mail

We want to know that the envelope has not been opened

Authenticity

There are two types:

Case 1: Bob wants to interactively prove his identity to Alice. (eg. talking by phone)

Case 2: Bob wants to prove his identity non-interactively to Alice. If the proof can convice a third party (judge), it's a *signature*.

Secrecy

We want to

1. Store a document
2. Send a message

We want...

... that no unauthorized person can learn any information about the document (or message).

Cryptography: A Brief History

- Until 1918: Ancient history
  - Ciphers based on sustitution and permutations
  - Secrecy = Secrecy of the Mechanism
- 1918-1975: Technical period: Cipher Machines (Enigma)
  - Fast, automated permutations and substitutions.
- 1976: Modern Cryptography,
  - Given a scheme, use assumptions (eg. one-way functions) to show evidence of security (a proof?).

Part II

Provable Security

Provably Security: The Short Story

- Originated in the late 80's
  - Encryption [Goldwasser, Micali 84]
  - Signatures [Goldwasser, Micali, Rivest 88]
- Popular using ideal substitutes
  - Random oracles vs. hash functions [Fiat, Shamir 86, Bellare-Rogaway 93]
  - Generic groups vs. Eliptic curves [Nechaev 94; Shoup 97]
  - Ideal ciphers vs. Block ciphers [Nechaev 94; Shoup 97]
- Proven useful to analyze a complex scheme in terms of the primitives used, in a modular fashion [Bellare-Kohno-Namprempre 04, Paterson et al. 10]
- Now a common requirement to support emerging standards (IEEE P1363, ISO, Cryptrec, NESSIE).

The need for Provable Security

Common approach to evaluate security: Cryptanalysis driven

1. Found an interesting cryptographic goal
2. Propose a solution
3. Search for an attack (ie. bug)
4. If one found, go back to step 2.

After *many* iterations... declare it secure.

Problems:

- When do we stop?
- Results not always trustworthy
  - Chor-Rivest knapsack scheme took 10 years to be totally broken!

Provable Security

The Recipe

1. Define goal of scheme (or adversary)
2. Define attack model
3. Give a protocol
4. Define complexity assumptions (or assumptions on the primitive)
5. Provide a proof by reduction
6. Verify proof
7. Interpret proof

The Need of Computational Assumptions

Consider asymmetric cryptography (Diffie Hellman, 76)
An encryption scheme $\mathcal{AS} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is composed by three algorithms:

- $\mathcal{K}$: Key generation
- $\mathcal{E}$: Encryption
- $\mathcal{D}$: Decryption

$$r' \longrightarrow \boxed{\mathcal{K}} \longrightarrow (k_e, k_d)$$

$$\begin{array}{ccccccc} & & k_e & & & k_d & \\ & & \downarrow & & & \downarrow & \\ m & \longrightarrow & \boxed{\mathcal{E}} & \longrightarrow \quad c \quad \longrightarrow & & \boxed{\mathcal{D}} & \longrightarrow \; m \text{ or } \perp \\ r & \longrightarrow & & & & & \end{array}$$

Unconditional secrecy is not possible

The ciphertext $c = \mathcal{E}_{k_e}(m; r)$ is uniquely determined by

- The public encryption key k_e
- The message m
- The random coins r

So, at least exhaustive search is possible!

Unconditional secrecy is not possible

The ciphertext $c = \mathcal{E}_{k_e}(m; r)$ is uniquely determined by

- The public encryption key k_e
- The message m
- The random coins r

So, at least exhaustive search is possible!
$\Rightarrow$ unconditional secrecy is impossible

We need complexity (algorithmic) assumptions.

Integer Factoring and RSA

Multiplication vs. Factorization

- $p, q \to n = p \cdot q$ is easy (cuadratic)
- $n = p \cdot q \to p, q$ is hard (super-polynomial)

One-way function

RSA Function [Rivest-Shamir-Adleman 78]

The function $f : \mathbb{Z}_n \to \mathbb{Z}_n$, where $n = pq$, for a fixed exponent e:

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

but easy $x = y^d \bmod n$ if trapdoor $d = e^{-1} \bmod \phi(n)$ is known.

We measure the *advantage* of any inverting adversary A by

$$\mathbf{Adv}_{n,e}^{rsa}(A) = \Pr\left[x \stackrel{\$}{\leftarrow} \mathbb{Z}_n^*,\ y = x^e \bmod n :\ A(y) = x\right]$$

The Discrete Logarithm

Let $G = (\langle g \rangle, \times)$ be any finite cyclic group.
For any $y \in G$, we define

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

Exponenciation Function

The function $\mathrm{DExp}_g : \mathbb{Z}_q \to G$, where $q = |G|$:

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

$$\mathbf{Adv}_g^{dl}(A) \quad = \quad \Pr\left[x \stackrel{\$}{\leftarrow} \mathbb{Z}_q,\ y = g^x \ :\ A(y) = x \right]$$

How hard are these problems?

Estimates for integer factorization [Lenstra-Verheul 2000]

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

Reasonable estimates for RSA too, and lower bounds for DL in $\mathbb{Z}_p^*$

Generalization: One-way functions

One-way Function

The function $f : \texttt{Dom}(f) \to \texttt{Rec}(f)$,

- $x \to y = f(x)$ (easy, polynomial-time)
- $y = f(x) \to x$ (difficult for random $x \in \texttt{Dom}(f)$, at least super-polynomial)

The *advantage* of an inverting adversary A is thus

$$\mathbf{Adv}_f^{ow}(A) \quad = \quad \Pr\left[x \xleftarrow{\$} \texttt{Dom}(f),\ y = f(x) :\ A(y) = x\right]$$

Resources of A:

- Running time t (number of operations)
- Number & length of queries (if in random oracle model)

Part III

Reductions

Algorithmic assumptions are necessary

Recall that for RSA

- $n = pq$: public modulus.
- e: public exponent.
- $d = e^{-1} \bmod \phi(n)$: private exponent.
- $\mathcal{E}_{n,e}(m) = m^e \bmod n$ and $\mathcal{D}_{n,d}(c) = c^d \bmod n$

Underlying hard problem:

Computing m from $c = \mathcal{E}_{n,e}(m)$, for $m \xleftarrow{\$} \mathbb{Z}_n^*$.

Easy fact

If the RSA problem is easy, secrecy does not hold: anybody (not only the owner of the trapdoor) can recover m from c.

But are algorithmic assumptions *sufficient*?

We want the guarantee that an assumption is **enough** for security.

But are algorithmic assumptions *sufficient*?

We want the guarantee that an assumption is **enough** for security.

For example, in the case of encryption

IF		Then
an adversary can break the secrecy	$\Rightarrow$	we can break the assumption!

But are algorithmic assumptions *sufficient*?

We want the guarantee that an assumption is **enough** for security.

For example, in the case of encryption

IF		Then
an adversary can break the secrecy	$\Rightarrow$	we can break the assumption!

This is a *reductionist proof*.

Proof by Reduction

Let **P** be a problem.

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

Proof by Reduction

Let **P** be a problem.

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

Instance **I** of **P** $\longrightarrow$

New algorithm for P

Adversary

A

$\longrightarrow$ Solution of **I**

If so, we say solving **P** reduces to breaking the scheme.

Conclusion: *If* **P** *untractable then scheme is unbreakable*

Provable Security?

A misleading name?

Not really *proving* a scheme secure but showing a reduction from security of scheme to the security of the underlying assumption (or primitive).

Provable Security?

A misleading name?

Not really *proving* a scheme secure but showing a reduction from security of scheme to the security of the underlying assumption (or primitive).

$\Rightarrow$ REDUCTIONIST SECURITY

Provably Secure Scheme

Before calling a scheme *provably secure*, we need

1. To make precise the algorithmic assumptions (some given)
2. To define the security notions to be guaranteed (next)
   - Security goal
   - Attack model
3. A reduction!

Complexity-theory vs. Exact Security vs. Practical

The interpretation of the reduction matters!

Given

A within time t, success probability ϵ

$\Rightarrow$

Build

Algorithm against **P** that runs in time $t' = T(t)$ with success probability $\epsilon' = R(\epsilon)$

The reduction requires showing T (for simplicity, suppose R depends only linearly in ϵ).

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

Each gives us a way to interpret reduction results.

Complexity-theory Security

Given

A within time t and success probability ϵ

$\Rightarrow$

Build

Algorithm against **P** that runs in time $t' = T(t, \epsilon)$

- Assumption: **P** is hard $=$ "no polynomial time algorithm"
- Reduction: T is polynomial in t and ϵ
- Security result: There is no polynomial time adversary....

Complexity-theory Security

Given

A within time t and success probability ϵ

$\Rightarrow$

Build

Algorithm against **P** that runs in time $t' = T(t, \epsilon)$

- Assumption: **P** is hard = "no polynomial time algorithm"
- Reduction: T is polynomial in t and ϵ
- Security result: There is no polynomial time adversary.... which really means that there is no attack if the parameters are large enough.

Complexity-theory Security

Given

A within time t and success probability ϵ

$\Rightarrow$

Build

Algorithm against **P** that runs in time $t' = T(t, \epsilon)$

- Assumption: **P** is hard = "no polynomial time algorithm"
- Reduction: T is polynomial in t and ϵ
- Security result: There is no polynomial time adversary.... which really means that there is no attack if the parameters are large enough.

Not always meaningful, as when analyzing block ciphers.

Complexity-theory Security: Results

General Results

Under polynomial reductions, against polynomial-time adversaries

1. Trapdoor one-way permutations are enough for secure encryption
2. One-way functions are enough for secure signatures

If only care about feasibility, these results close the chapter (no more problems left)... but

- the schemes for which these results were originally obtained are rather inefficient,
- looking *into* the complexity of the reduction may gives us some insight

Exact Security

Given		Build
A which on time t breaks scheme with probability ϵ	$\Rightarrow$	Algorithm against **P** that runs in time $t' = T(t, \epsilon)$ and works with probability ϵ'

- Assumption: Solving **P** requires N operations (say, time τ)
- Reduction: exact cost for T as a function of t, ϵ, and other parameters (eg. the key sizes)
- Security result: There is no adversary (for scheme) within time t such that $t' = T(t, \epsilon) \leq \tau$.

Exact Security

Given

A which on time t breaks scheme with probability ϵ

$\Rightarrow$

Build

Algorithm against **P** that runs in time $t' = T(t, \epsilon)$ and works with probability ϵ'

- Assumption: Solving **P** requires N operations (say, time τ)
- Reduction: exact cost for T as a function of t, ϵ, and other parameters (eg. the key sizes)
- Security result: There is no adversary (for scheme) within time t such that $t' = T(t, \epsilon) \leq \tau$.

Why useful

From $T(t) \leq \tau$ we can get bounds on minimal key sizes under which the scheme is secure.

Measuring the Quality of the Reduction

How much is lost in the reduction? How much of the "power" of adversary A breaking the scheme remains in the algorithm breaking the problem **P**

Measuring the Quality of the Reduction

How much is lost in the reduction? How much of the "power" of adversary A breaking the scheme remains in the algorithm breaking the problem **P**

Tightness

A reduction is *tight* if $t' \approx t$ and $\epsilon' \approx \epsilon$. Otherwise, if $t' >> t$ or $\epsilon' << \epsilon$, the reduction is *not tight*.

The *tightness gap* is $(t'\epsilon)/(t\epsilon') = (t'/\epsilon')/(t/\epsilon)$.

We want tight reductions, or at least reductions with small tightness gap.

Part IV

Security Notions

Security Notions: Examples

Problem:

Authentication and no-repudiation (ie. signatures)

How do we come up with a security notion?

Security Notions: Examples

Problem:

Authentication and no-repudiation (ie. signatures)

How do we come up with a security notion?
We need to think and define

1. Security goal of the scheme (= Opposite to Adversary's goal)
    - *Property that needs to be guaranteed*
2. Attack model
    - *Attack venues, what the adversary can and cannot do*
    - *Leaked information, what the adversary can know from honest users* (often modeled by oracles)

Signature Schemes (Authentication)

Goal: Existential Forgery

The adversary wins if it forges a valid message-signature pair without private key

Adversary does a good job (or *the scheme is insecure*) if

- given the verification key k_v,
- outputs a pair m', σ' of message and its signature

such that the following probability is large:

$$\Pr\left[\, Vf(k_v, m', \sigma') = 1 \,\right]$$

Possible Attack Models

- **No-Message Attack (NKA)**: adversary only knows the verification key.
- **Known-Message Attack (KMA)**: adversary also can access list of message/signature pairs.
- **Chosen-Message Attack (CMA)**: adversary can choose the messages for which he can see the message/signature pairs.
Strongest attack

Security Notion for Signature Schemes: EUF-CMA

[Goldwasser, Micali, Rivest 1988]

Given signature scheme $\Sigma = (\mathcal{K}, Sign, Vf)$.

$(k_v, k_s) \xleftarrow{\$} \mathcal{K}(\cdot)$

$k_v \downarrow$

$k_s \downarrow$

Adversary

$\xrightarrow{m}$

$\xleftarrow{\sigma}$

$\xrightarrow{\dots}$

$\longleftarrow$

Signing Oracle

$\sigma \leftarrow Sign(k_s, m)$

$\downarrow (m', \sigma')$

$$\mathbf{Adv}_{\Sigma}^{\text{euf-cma}}(A) = \Pr[\, Vf(k_v, m', \sigma') = 1, \text{ for new } m' \,]$$

(Existential unforgeability under chosen-message attacks)

Security Models

Sometimes it is helpful to consider models where some tools (primitives) used by cryptographic schemes such as,

- Hash functions
- Block ciphers
- Finite groups

are considered to be ideal, that is, the adversary can only use (attack) them in a certain way.

Security Models

Sometimes it is helpful to consider models where some tools (primitives) used by cryptographic schemes such as,

- Hash functions
- Block ciphers
- Finite groups

are considered to be ideal, that is, the adversary can only use (attack) them in a certain way.

$\Rightarrow$ Idealized Security Models:

- Hash function $\rightarrow$ Random oracle
- Block ciphers $\rightarrow$ Ideal cipher
- Finite groups $\rightarrow$ Generic group

Security Models

Sometimes it is helpful to consider models where some tools (primitives) used by cryptographic schemes such as,

- Hash functions
- Block ciphers
- Finite groups

are considered to be ideal, that is, the adversary can only use (attack) them in a certain way.

$\Rightarrow$ Idealized Security Models:

- Hash function $\rightarrow$ Random oracle
- Block ciphers $\rightarrow$ Ideal cipher
- Finite groups $\rightarrow$ Generic group

Standard model: no idealized primitives (sort of)

Security Model: Random Oracle

Arguably the most used idealized model to prove security of practical schemes. [Bellare-Rogaway 93]

Hash function $H\colon \{0,1\}^* \to \mathtt{Rec}(H)$ is analized as it were a perfectly random function

- Each new query receives a random answer in $\mathtt{Rec}(H)$
- The same query asked twice receives the same answer twice

But for actual scheme, H is replaced by cryptographic hash function (SHA-1,RIPEMD-160, etc.)

Security Model: Random Oracle

Arguably the most used idealized model to prove security of practical schemes. [Bellare-Rogaway 93]

Hash function $H\colon \{0,1\}^* \to \mathtt{Rec}(H)$ is analized as it were a perfectly random function

- Each new query receives a random answer in $\mathtt{Rec}(H)$
- The same query asked twice receives the same answer twice

But for actual scheme, H is replaced by cryptographic hash function (SHA-1,RIPEMD-160, etc.)

Examples of use:

1. Signature schemes: Full-Domain Hash [Bellare-Rogaway 96], Schnorr [Schnorr 89]
2. Encryption schemes: OAEP-based constructions [Bellare-Rogaway 94]

Somehow controversial: not really proof, only heuristic [Canetti 98, 04]

An Example of Exact Security

Full-Domain Hash Signatures

Full-Domain Hash Signature [Bellare-Rogaway 1993]

Scheme FDH is $(\mathcal{K}, \mathcal{S}, \mathcal{V})$ as follows

- $\mathcal{K}$: Key Generation returns (f, f^{-1}) where
  - Public key $f : X \to X$, a trapdoor one-way permutation onto X
  - Private key f^{-1}
- $\mathcal{S}$: Signature of m, returns $\sigma \leftarrow f^{-1}(H(m))$
- $\mathcal{V}$: Verification of (m, σ), returns true if $f(\sigma) = H(m)$.

Exact Security: Full-Domain Hash Signatures

Theorem (FDH is EUF-CMA in the RO model)

Let FDH be the FDH signature scheme using one-way permutation f (for example, f =RSA)
For each adversary A there exist an adversary B such that

$$\mathbf{Adv}_{\mathrm{FDH}}^{euf\text{-}cma}(A) \quad \leq \quad (q_h + q_s + 1) \cdot \mathbf{Adv}_f^{ow}(B)$$

where

- *A runs in time t, makes q_h queries to hash function (RO), and q_s signature queries.*
- *T_f is the time to compute f (in the forward direction)*
- *B runs in time $t' = t + (q_h + q_s) \cdot T_f$*

[Bellare-Rogaway 1993, 1996]

Exact Security: Full-Domain Hash Signatures

Theorem (FDH is EUF-CMA in the RO model)

Let FDH *be the FDH signature scheme using one-way permutation f (for example, f =RSA)*
For each adversary A there exist an adversary B such that

$$\mathbf{Adv}_{\mathrm{FDH}}^{euf\text{-}cma}(A) \quad \leq \quad (q_h + q_s + 1) \cdot \mathbf{Adv}_f^{ow}(B)$$

where

- *A runs in time t, makes q_h queries to hash function (RO), and q_s signature queries.*
- *T_f is the time to compute f (in the forward direction)*
- *B runs in time $t' = t + (q_h + q_s) \cdot T_f$*

[Bellare-Rogaway 1993, 1996]

Proof (reduction)?

Exact Security: FDH Signatures & Game-based proofs

We use a *game*-based proofs technique:

[Shoup 2004, Bellare-Rogaway 2004]

1. Define sequence of games $G_0, G_1, \ldots, G_5$ of *games* or *experiments*.
2. All games in the same probability space.
3. Rules on how the *view* of the game is computed differs.
4. Successive games are very similar, typically with slightly different distribution probabilities.
5. G_0 is the actual security game (EUF-CMA)
6. G_5 is the game for the underlying assumption (OW).
7. We relate the probabilities of the events that define the advantages in G_0, and G_5, via all the intermediate games.

Exact Security: FDH Sigs & Game-based proofs (0/5)

(courtesy of [Pointcheval 2005])

Game G_0: the real euf-cma game with signing oracle and a random oracle, but we also provide a *verification oracle* Vf.

Verification oracle $Vf(m, \sigma)$

Return true if $H(m) = f(\sigma)$. The game ends when adversary sends (m, σ) here.

Let S_0 be the event:

"A outputs a pair (m, σ) for which Vf returns true".

Clearly

$$\mathbf{Adv}^{\text{euf-cma}}_{\text{FDH}}(A) = \Pr[\, S_0 \,]$$

Exact Security: FDH Sigs & Game-based proofs (1/5)

Game G_1: as G_0 but oracles are simulated as below.

Hashing oracle $H(q)$

Create an initially empty list called H-List.

- If $(q, \star, r) \in H$-List, return r.
- Otherwise reply using
  Rule $\mathcal{H}^{(1)}$: $r \xleftarrow{\$} X$, and add record $(q, \star, r)$ to H-List.

Signing oracle $S(m)$

- $r \leftarrow H(m)$.
  Reply using
  Rule $\mathcal{S}^{(1)}$: $\sigma \leftarrow f^{-1}(r)$.

Verification oracle $Vf(m, \sigma)$

- $r \leftarrow H(m)$.
  Return true if $r = f(\sigma)$.

Game ends when oracle called.

Let S_1 be the event: "Vf returns true in G_1".
Clearly $\Pr[S_1] = \Pr[S_0]$.

Exact Security: FDH Sigs & Game-based proofs (2/5)

Game G_2: as G_1 but where

- $c \xleftarrow{\$} \{1, \ldots, q_H + q_S + 1\}$
- Let $c' =$ index of first query where message m' (the one for which A outputs a forgery) was sent to the hashing oracle by A.
- If $c \neq c'$, then abort.

Sucess verification is within the game $\Rightarrow$ the adversary must query his output message m.

$$\begin{aligned} \Pr[\, S_2 \,] &= \Pr[\, S_1 \;\wedge\; GoodGuess \,] \\ &= \Pr[\, S_1 \,|\, GoodGuess \,] \times \Pr[\, GoodGuess \,] \\ &\geq \Pr[\, S_1 \,] \times \frac{1}{q_H + q_S + 1} \end{aligned}$$

Exact Security: FDH Sigs & Game-based proofs (3/5)

Game G_3: as G_2 but now use the following rule in the hashing oracle:

- Let y be the challenge from which we want to extract a preimage x by f.
- Rule $\mathcal{H}^{(3)}$:
  - If this is the c-th query, set $r \leftarrow y$.
  - Otherwise, choose random. Add record $(q, \perp, r)$ to H-List.

Exact Security: FDH Sigs & Game-based proofs (3/5)

Game G_3: as G_2 but now use the following rule in the hashing oracle:

- Let y be the challenge from which we want to extract a preimage x by f.
- Rule $\mathcal{H}^{(3)}$:
  - If this is the c-th query, set $r \leftarrow y$.
  - Otherwise, choose random. Add record $(q, \perp, r)$ to H-List.

Since position y is chosen uniformly at random: $\Pr[S_3] = \Pr[S_2]$.

Exact Security: FDH Sigs & Game-based proofs (4/5)

Game G_4: as G_3 but modify simulation of hashing oracle (which may be used in signing queries)

- Rule $\mathcal{H}^{(4)}$:
    - If this is the c-th query, set $r \leftarrow y$ and $s \leftarrow \perp$.
    - Otherwise, choose random $s \xleftarrow{\$} X$, compute $r \leftarrow f(s)$.
    - Add record (q, s, r) to H-List.

Exact Security: FDH Sigs & Game-based proofs (4/5)

Game G_4: as G_3 but modify simulation of hashing oracle (which may be used in signing queries)

- Rule $\mathcal{H}^{(4)}$:
  - If this is the c-th query, set $r \leftarrow y$ and $s \leftarrow \perp$.
  - Otherwise, choose random $s \stackrel{\$}{\leftarrow} X$, compute $r \leftarrow f(s)$.
  - Add record (q, s, r) to H-List.

Since position y is random, f is permutation, and s is random: $\Pr[\,S_4\,] = \Pr[\,S_3\,]$.

Exact Security: FDH Sigs & Game-based proofs (5/5)

Game G_5: except for the c-th query, all preimages are known. Then, we can simulate signing oracle without f^{-1}.

- Rule $\mathcal{S}^{(5)}$:
  - Lookup (m, s, r) in H-List, and set $\sigma \leftarrow s$.

Exact Security: FDH Sigs & Game-based proofs (5/5)

Game G_5: except for the c-th query, all preimages are known. Then, we can simulate signing oracle without f^{-1}.

- Rule $\mathcal{S}^{(5)}$:
  - Lookup (m, s, r) in H-List, and set $\sigma \leftarrow s$.

Since c-th query cannot be asked to hash oracle, then $\Pr[\,S_5\,] = \Pr[\,S_4\,]$.

Exact Security: FDH Sigs & Game-based proofs (5/5)

Game G_5: except for the c-th query, all preimages are known. Then, we can simulate signing oracle without f^{-1}.

- Rule $\mathcal{S}^{(5)}$:
  - Lookup (m, s, r) in H-List, and set $\sigma \leftarrow s$.

Since c-th query cannot be asked to hash oracle, then $\Pr[S_5] = \Pr[S_4]$.
Moreover,

- simulation can be done computing $(q_S + q_H)$ evaluations of f,
- signature forgery for y gives preimage for y:

$$\Pr[S_5] = \mathbf{Adv}_f^{\mathrm{ow}}(B)$$

where $B = G_5$ runs in time $t + (q_S + q_H)T_f$.

Exact Security: FDH Sigs & Game-based proofs, conclusion

Combining the relations from previous games:

$$
\begin{aligned}
\mathbf{Adv}_f^{\mathrm{ow}}(B) &= \Pr[S_5] = \Pr[S_4] = \Pr[S_3] = \Pr[S_2] \\
&\geq \frac{1}{q_H + q_S + 1} \times \Pr[S_1] \\
&\geq \frac{1}{q_H + q_S + 1} \times \Pr[S_0] \\
&= \frac{1}{q_H + q_S + 1} \times \mathbf{Adv}_{\mathrm{FDH}}^{\mathrm{euf\text{-}cma}}(A)
\end{aligned}
$$

$\square$

Game-playing proofs: In general, games can have different distributions, and this gaps are included in the concrete security relation. See [Bellare-Rogaway 2004].

Interpreting Exact Security: FDH Signatures

Let's go back to our first result:

Theorem (FDH is EUF-CMA)

Let FDH *be the FDH signature scheme using one-way permutation f (for example, f =RSA)*
For each adversary A there exist an adversary B such that

$$\mathbf{Adv}_{\mathrm{FDH}}^{\textit{euf-cma}}(A) \quad \leq \quad (q_h + q_s + 1) \cdot \mathbf{Adv}_f^{\textit{ow}}(B)$$

where

- *A runs in time t, makes q_h queries to hash function (RO), and q_s signature queries.*
- *T_f is the time to compute f (in the forward direction)*
- *B runs in time $t' = t + (q_h + q_s) \cdot T_f$*

Interpreting Exact Security: FDH Signatures

Let's go back to our first result:

Theorem (FDH is EUF-CMA)

Let FDH *be the FDH signature scheme using one-way permutation f (for example, $f =$RSA)*
For each adversary A there exist an adversary B such that

$$\mathbf{Adv}^{euf\text{-}cma}_{\mathrm{FDH}}(A) \quad \leq \quad (q_h + q_s + 1) \cdot \mathbf{Adv}^{ow}_{f}(B)$$

where

- *A runs in time t, makes q_h queries to hash function (RO), and q_s signature queries.*
- *T_f is the time to compute f (in the forward direction)*
- *B runs in time $t' = t + (q_h + q_s) \cdot T_f$*

How should we interpret this result?

Full-Domain Hash: Interpreting the Result

Suppose feasible security bounds for *any* adversary are:

- at most 2^{75} operations (t),
- at most 2^{55} hash queries (q_h), and
- at most 2^{30} signing queries (q_s)

$$\begin{aligned} \mathbf{Adv}^{\text{euf-cma}}_{\text{FDH}}(A) &\leq (q_h + q_s + 1) \cdot \mathbf{Adv}^{\text{ow}}_{f}(B) \\ B \quad \text{runs in time} &\quad t' = t + (q_h + q_s) \cdot T_f \end{aligned}$$

The result now says

Interpreting the Result

If one can break the scheme with time t then one can invert f within time $t' \leq (q_h + q_s + 1)(t + (q_h + q_s) \cdot T_f) \leq 2^{130} + 2^{110} \cdot T_f$.

Full-Domain Hash: Interpreting the Result (cont.)

Thus, inverting f can be done in time

$$t' \leq 2^{130} + 2^{110} \cdot T_f \, .$$

Recall that $T_f = \mathcal{O}(k^3)$ operations, if $k = |n|$ and e small.

We compare it with known bounds on inverting RSA (namely, factoring using the best known inverting algorithm, the *Number Field Sieve* (NFS) for f=RSA.

- 1024 bits $\rightarrow t' \leq 2^{140}$... but NFS takes 2^{80}.
- 2048 bits $\rightarrow t' \leq 2^{143}$... but NFS takes 2^{111}.
- 4096 bits $\rightarrow t' \leq 2^{146}$... but NFS takes 2^{149}, ok!

$\Rightarrow$ RSA-FDH is secure for keys at least 4096.

Full-Domain Hash: Improved Reduction

There is a better reduction: [Coron 2000]

$$\mathbf{Adv}^{\text{euf-cma}}_{\text{FDH}}(A) \quad \leq \quad q_s \cdot e \cdot \mathbf{Adv}^{\text{ow}}_{f}(B)$$

where B runs in time $t' = t + (q_h + q_s + 1) \cdot T_f$ if A runs in time t and makes q_h, q_s queries.

Solving, inverting f can be done in time $t' \leq 2^{30} \cdot t + 2^{85} \cdot T_f$ and

- 1024 bits $\rightarrow t' \leq 2^{105}$... but NFS takes 2^{80}.
- 2048 bits $\rightarrow t' \leq 2^{107}$... but NFS takes 2^{111}, ok!
- 4096 bits $\rightarrow t' \leq 2^{109}$... but NFS takes 2^{149}, ok!

$\Rightarrow$ RSA-FDH is secure for keys at least 2048.

Security Notions: Encryption Schemes

Problem:

Secrecy (ie. encryption)

Goal cannot be too strong...

- Perfect Secrecy: not possible, ciphertext (info-theoretically) reveals information about the plaintext.

Goal: Indistinguishability (Semantic Security), Informal

Given the ciphertext and the encryption key, the adversary cannot tell apart two same-length but different messages encrypted under the scheme, even if chose the messages himself.

Attack model

- **Chosen-Plaintext Attack (CPA)**: adversary can get the encryption of any plaintext of his choice.
- **Chosen-Ciphertext Attack (CCA or CCA2)**: adversary also has access to a decryption oracle which (adaptively) decrypts any ciphertext of his choice except one specific ciphertext (called the *challenge*).

  Strongest attack

Security Notion for (Asymmetric) Encryption: IND-CCA

Given (asymmetric) encryption scheme $\mathcal{AS} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$.

$b \stackrel{\$}{\leftarrow} \{0,1\}$, $(k_e, k_d) \stackrel{\$}{\leftarrow} \mathcal{K}(\cdot)$

$k_e \downarrow$

$m_0 \leftarrow$

$m_1 \leftarrow$

Adversary

Challenger

$c^* \leftarrow \mathcal{E}_{k_e}(m_b)$

$\xrightarrow{c^*}$

$b' \leftarrow$

$\xrightarrow{c}$

$\xleftarrow{m \text{ or } \perp}$

$\cdots$

$\rightleftarrows$

CCA1

$m \leftarrow \mathcal{D}_{k_d}(c)$

$\xrightarrow{c \neq c^*}$

$\xleftarrow{m \text{ or } \perp}$

$\rightleftarrows$

CCA2

$m \leftarrow \mathcal{D}_{k_d}(c)$

$$\mathbf{Adv}^{\text{ind-cca}}_{\mathcal{AS}}(A) = \Pr\left[(m_0, m_1) \leftarrow A^{\mathcal{D}}(k_e), c^* \leftarrow \mathcal{E}_{k_e}(m_b) \;:\; b' = b\right]$$

(Indistinguishability against chosen-ciphertext attacks)

A Weaker Security Notion: OW-CPA

It may be helpful to consider a weaker security goal too.

Consider the game:

- Let m be a random message chosen from message space $\mathcal{M}$.
- From ciphertext $c = \mathcal{E}_{k_e}(m)$, adversary A must recover m.

A scheme $\mathcal{AS}$ is *One-Way under chosen-plaintext attack* if no feasible adversary A can win the above game with reasonable probability.

Accordingly, we measure the advantage of A as

$$\mathbf{Adv}_{\mathcal{AS}}^{\text{ow-cpa}}(A) \quad = \quad \Pr\left[m \xleftarrow{\$} \mathcal{M}, c \leftarrow \mathcal{E}_{k_e}(m) \,|\, A(k_e, c) = m \right]$$

Goals Achieved by Practical Encryption Schemes

- Integer Factoring-based: RSA [Rivest-Shamir-Adleman 78]
  - OW-CPA = RSA (modular e-th roots)
  - It's not IND-CPA nor IND-CCA since it's deterministic
- Discrete-Log-based: ElGamal [ElGamal 78]
  - OW-CPA = CDH (*Computational Diffie-Hellman*)
  - IND-CPA = DDH (*Decisional Diffie-Hellman*)
  - It's not IND-CCA because of multiplicativity.

Obs: CDH and DDH are *weaker* problems that DLog (DDH reduces to CDH which reduces to DLog).

Achieving Stronger Goals

We would like to obtain IND-CCA.

What we know at this point:

- Any trapdoor one-way function may yield a OW-CPA encryption scheme
- OW-CPA not enough to IND-CPA nor IND-CCA

So, how do we obtain IND-CCA?

Achieving Stronger Goals

We would like to obtain IND-CCA.

What we know at this point:

- Any trapdoor one-way function may yield a OW-CPA encryption scheme
- OW-CPA not enough to IND-CPA nor IND-CCA

So, how do we obtain IND-CCA?

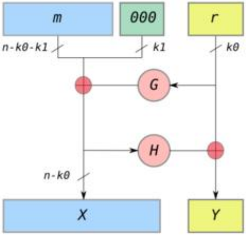
Generic conversion from weakly secure to strongly secure schemes

f-OAEP [Bellare-Rogaway 1994]

Let f be a trapdoor one-way permutation, n, k_0, k_1 integers such that $n > k_0 + k_1$, with

$G : \{0,1\}^{k_0} \to \{0,1\}^{n-k_0}$

$H : \{0,1\}^{n-k_0} \to \{0,1\}^{k_0}$



- $\mathcal{E}(m; r)$: Compute x, y then return $c = f(x||y)$
- $\mathcal{D}(c)$: Compute $x||y = f^{-1}(c)$, invert OAEP, then check redundancy

RSA-OAEP

A (good) reduction from a variant of OW-CPA (called *partial-domain OW*) was given for RSA-OAEP in the random oracle model. [Fujisaki-OPS 00]

The result is

$$\mathbf{Adv}^{\text{ind-cca}}_{RSA-OAEP}(A) \quad \leq \quad 2 \cdot \sqrt{\mathbf{Adv}^{\text{rsa}}_{n,e}(B))}$$

where B runs in time $t' = 2 \cdot t + q_H(2 \cdot q_G + q_H) \cdot k^2$ if A runs in time t and makes q_H, q_G queries to oracles H y G respectively, k is the modulus size and e small.

RSA-OAEP

A (good) reduction from a variant of OW-CPA (called *partial-domain OW*) was given for RSA-OAEP in the random oracle model. [Fujisaki-OPS 00]

The result is

$$\mathbf{Adv}^{\text{ind-cca}}_{RSA-OAEP}(A) \quad \leq \quad 2 \cdot \sqrt{\mathbf{Adv}^{\text{rsa}}_{n,e}(B))}$$

where B runs in time $t' = 2 \cdot t + q_H(2 \cdot q_G + q_H) \cdot k^2$ if A runs in time t and makes q_H, q_G queries to oracles H y G respectively, k is the modulus size and e small.

Solving, inverting f can be done in time $t' \leq 2^{76} + 6 \cdot 2^{110} k^2 \leq 2^{113} \cdot k^2$ and

- 1024 bits $\rightarrow t' \leq 2^{133}$... but NFS takes 2^{80}, no!
- 2048 bits $\rightarrow t' \leq 2^{135}$... but NFS takes 2^{111}, no!
- 4096 bits $\rightarrow t' \leq 2^{137}$... but NFS takes 2^{149}, ok!

$\Rightarrow$ RSA-OAEP is secure for keys at least 4096. ... not tight.

Improving the reduction: f-OAEP++

A new padding scheme OAEP++ was proposed by Jonsson (2002). The one-time pad on the OAEP (xor between random r and output of H) is replaced by a strong block cipher (ideal cipher model).

Ideal Cipher Model

Consider block cipher E as a family of *perfectly random* and *independent* permutations.

Improving the reduction: f-OAEP++ (cont.)

Advantage Bound

The relation (bound) between the IND-CCA-advantage of f-OAEP++ and the OW-CPA advantage of f=RSA is more involved... but esentially linear.

As before, suppose feasible security bounds for *any* adversary attacking f=RSA are:

- at most 2^{75} operations (t)
- at most 2^{55} hash (q_H, q_G) and ideal cipher queries (q_E),

Result: if one can break RSA-OAEP++ on time t, one can invert k-bit-modulus RSA in time $t' \leq t + q_E \cdot k^2 \leq 2^{75} + 2^{55} \cdot k^2$ and

- 1024 bits $\rightarrow t' \leq 2^{76}$... but NFS takes 2^{80}, ok!
- 2048 bits $\rightarrow t' \leq 2^{78}$... but NFS takes 2^{111}, ok!
- 4096 bits $\rightarrow t' \leq 2^{80}$... but NFS takes 2^{149}, ok!

$\Rightarrow$ RSA-OAEP++ is secure for keys 1024 or more.

Revisiting the Assumptions

Classical Assumptions

- Integer Factoring
- Discrete Logarithm (in Finite Fields and in Elliptic Curves)
- Modular Roots (Square roots and e-th roots)

Advantages: Easy to implement, widely used
Drawbacks: Require large keys if in Finite Fields. They are all subject to quantum attacks!

Alternatives: Post-Quantum Cryptography

- Error-Correcting Codes
- Hash-based schemes
- Systems of Multi-Variate Equations
- Lattices

Part V

Concluding Remarks

Limits and Benefits of Provable Security

Provably security does not yield proofs

- Proofs are relative (to computational assumptions) *and* to the definition of the scheme's goal
- Proofs often done in ideal models (Random Oracle Model, Ideal Cipher Model, Generic Group Model) with debatable meaning. [Canetti 98, 04], [Coron 08, Holenstein et al. 11]
- Definitions (models) need time for review and acceptance.
  - Example: proofs for several modes for SSH authenticated encryption [Bellare-Kohno-Namprempre 04], then (one mode) attacked [Albrecht 09], then proofs (for the other mode) in a better model. [Paterson et al. 10]
  - Are we back in time, now with model, attacks, remodel? Crypto as physics! [Nguyen 12, Degabriele et al. 11]

Limits and Benefits of Provable Security

Still, provable security

- provides *some* form of guarantee that the scheme is not flawed
- Motivates us to spell out (clarify) definitions and models formally, *a process that, in itself, may help us to better understand the problem!*
- Gives well-defined reductions from which we can (and must) distill practical implications of the result (exact security)
- is fun! :-)

Acknowledgements and References

Thanks to ASCrypto organizers for the opportunity to give this short tutorial.

Further information:

- *Contemporary Cryptology, Provable Security for Public Key Schemes*, David Pointcheval, Advanced Course on Contemporary Cryptology. Advanced Courses CRM Barcelona, Pages 133-189. Birkhuser Publishers, 2005.
- *On the Role of Definitions in and Beyond Cryptography*, Phillip Rogaway. Manuscript, available from his web page.
- *Practice-Oriented Provable-Security*, Mihir Bellare, In proceedings of First International Workshop on Information Security (ISW'97), LNCS vol. 1396, Springer-Verlag, 1999.

Some slides courtesy of David Pointcheval (thanks!).

Part VI

References

M. R. Albrecht, K. G. Paterson, and G. J. Watson.
Plaintext recovery attacks against ssh.
In *Security and Privacy, 2009 30th IEEE Symposium on*, pages 16–26. IEEE, 2009.

M. Bellare, T. Kohno, and C. Namprempre.
Breaking and provably repairing the SSH authenticated encryption scheme: A case study of the encode-then-encrypt-and-MAC paradigm.
ACMTISS: ACM Transactions on Information and System Security, 7, 2004.

M. Bellare and P. Rogaway.
Random oracles are practical: A paradigm for designing efficient protocols.
In ACM, editor, *Proceedings of the 1st ACM conference on Computer and communications security*. ACM, Nov. 1993.

M. Bellare and P. Rogaway.
Optimal asymmetric encryption: How to encrypt with RSA.
In A. D. Santis, editor, *Advances in Cryptology – EUROCRYPT ' 94*, volume 950 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin Germany, May 1994.
http://www-cse.ucsd.edu/users/mihir.

M. Bellare and P. Rogaway.
The exact security of digital signatures: How to sign with RSA and Rabin.
In U. Maurer, editor, *Advances in Cryptology – EUROCRYPT ' 96*, volume 1070 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin Germany, May 1996.

M. Bellare and P. Rogaway.
The security of triple encryption and a framework for code-based game-playing proofs.
In S. Vaudenay, editor, *Advances in Cryptology – EUROCRYPT ' 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 409–426. Springer, 2006.

R. Canetti, O. Goldreich, and S. Halevi.
The random oracle methodology, revisited.
Journal of the ACM (JACM), 51(4):557–594, 2004.

J.-S. Coron, J. Patarin, and Y. Seurin.
The random oracle model and the ideal cipher model are equivalent.
In *Advances in Cryptology–CRYPTO 2008*, pages 1–20. Springer, 2008.

J. P. Degabriele, K. Paterson, and G. Watson.
Provable security in the real world.
Security & Privacy, IEEE, 9(3):33–41, 2011.

W. Diffie and M. Hellman.
New directions in cryptography.
IEEE Transactions on Information Theory, 22:644–654, 1978.

T. ElGamal.
A public key cryptosystem and signature scheme based on discrete logarithms.
IEEE Transactions on Information Theory, 31:469–472, 1985.

A. Fiat and A. Shamir.
How to prove yourself: Practical solutions to identification and signature problems.
In A. M. Odlyzko, editor, *Advances in Cryptology—CRYPTO '86*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer-Verlag, 1987, 11–15 Aug. 1986.

Fujisaki, Okamoto, Pointcheval, and Stern.
RSA-OAEP is secure under the RSA assumption.
Journal of Cryptology, 17, 2004.

E. Fujisaki, T. Okamoto, D. Pointcheval, and J. Stern.
RSA-OAEP is still alive!
Report 2000/061, Cryptology ePrint Archive, Nov. 2000.

S. Goldwasser and S. Micali.
Probabilistic encryption.
Journal of Computer and System Science, 28:270–299, 1984.

S. Goldwasser, S. Micali, and R. Rivest.
A digital signature scheme secure against adaptive chosen-message attacks.
Siam Journal of Computing, 17(2):281–308, Apr. 1988.

T. Holenstein, R. Künzler, and S. Tessaro.
The equivalence of the random oracle model and the ideal cipher model, revisited.
In *Proceedings of the 43rd annual ACM symposium on Theory of computing*, pages 89–98. ACM, 2011.

J. Jonsson.
An OAEP variant with a tight security proof, 2002.
This paper has not been published elsewhere.
jjonsson@rsasecurity.com 11764 received 18 Mar 2002.

A. K. Lenstra and E. R. Verheul.
Selecting cryptographic key sizes.
J. Cryptology, 14(4):255–293, 2001.

V. I. Nechaev.
Complexity of a determinate algorithm for the discrete logarithm.
Mathematical Notes, 55(2):165–172, 1994.
Translated from Matematicheskie Zametki, 55(2):91–101, 1994.

P. Q. Nguyen.
Cryptanalysis vs. provable security.
In *Information Security and Cryptology*, pages 22–23. Springer, 2012.

K. G. Paterson and G. J. Watson.
Plaintext-dependent decryption: A formal security treatment of ssh-ctr.
In *Advances in Cryptology–EUROCRYPT 2010*, pages 345–361. Springer, 2010.

D. Pointcheval.
Provable security for public key schemes.
In *Catalano & Cramer & Damgard & Di Crescenzo & Pointcheval & Takagi, Contemporary Cryptology*. Birkhauser, 2005.

R. L. Rivest, A. Shamir, and L. Adleman.
A method for obtaining digital signature and public-key cryptosystems.
Communications of the ACM, 21(2):120–126, 1978.

C. P. Schnorr.
Efficient identification and signatures for smart cards.
In *Advances in Cryptology (CRYPTO '89)*, pages 239–252, Berlin - Heidelberg - New York, Aug. 1990. Springer.

V. Shoup.
Lower bounds for discrete logarithms and related problems.
In *Proc. International Advances in Cryptology Conference – EUROCRYPT '97*, pages 256–266, 1997.

V. Shoup.
Sequences of games: a tool for taming complexity in security proofs.
Cryptology ePrint Archive, Report 2004/332, 2004.
http://www.shoup.net/papers/games.pdf".

S. Vaudenay.
Cryptanalysis of the chor - rivest cryptosystem.
J. Cryptology, 14(2):87–100, 2001.