III – Signatures

David Pointcheval
Ecole normale supérieure, CNRS & INRIA

MPRI – Paris

Outline

1 Basic Security Notions
   - Public-Key Encryption
   - Signatures

2 Advanced Security for Signature
   - Advanced Security Notions
   - Hash-then-Invert Paradigm

3 Forking Lemma
   - Zero-Knowledge Proofs
   - The Forking Lemma

4 Conclusion

Public-Key Encryption

Goal: Privacy/Secrecy of the plaintext
**OW – CPA Security Game**

\[ m^* \overset{?}{=} m \]

\[ \text{Succ}_{OW}^S(A) = \Pr[(sk, pk) \leftarrow \mathcal{K}(); m \overset{R}{\leftarrow} \mathcal{M}; c = \epsilon_{pk}(m) : A(pk, c) \rightarrow m] \]

**IND – CPA Security Game**

\[ b \in \{0,1\} \]

\[ (sk, pk) \leftarrow \mathcal{K}(); (m_0, m_1, \text{state}) \leftarrow A(pk); \]

\[ b' \overset{?}{=} b \]

\[ \text{Adv}_{S}^{\text{ind-\text{cpa}}} (A) = \Pr[b' = 1|b = 1] - \Pr[b' = 1|b = 0] = 2 \times \Pr[b' = b] - 1 \]

---

**Outline**

1. **Basic Security Notions**
   - Public-Key Encryption
   - Signatures
2. **Advanced Security for Signature**
3. **Forking Lemma**
4. **Conclusion**

**Signature**

Goal: Authentication of the sender
EUF − NMA

The adversary knows the public key only, whereas signatures are not private!

The goal is authentication of the sender.
The adversary has access to any signature of its choice:

**Chosen-Message Attacks (oracle access):**

\[
\text{Succ}_{\text{euf-cma}}(A) = \Pr \left[ (sk, pk) \leftarrow K(); (m, \sigma) \leftarrow A^S(pk) : \forall i, m \neq m_i \land V_{pk}(m, \sigma) = 1 \right] 
\]

The notion is even stronger (in case of probabilistic signature):
also known as **non-malleability**:

\[
\text{Succ}_{\text{suf-cma}}(A) = \Pr \left[ (sk, pk) \leftarrow K(); (m, \sigma) \leftarrow A^S(pk) : \forall i, (m, \sigma) \neq (m_i, \sigma_i) \land V_{pk}(m, \sigma) = 1 \right] 
\]

---

**Outline**

1. **Basic Security Notions**
2. **Advanced Security for Signature**
   - Advanced Security Notions
   - Hash-then-Invert Paradigm
3. **Forking Lemma**
4. **Conclusion**

**Signature Scheme**

- Key generation: the public key \( f \xleftarrow{R} \mathcal{P} \) is a trapdoor one-way bijection from \( X \) onto \( Y \); the private key is the inverse \( g : Y \rightarrow X \);
- Signature of \( M \in Y \): \( \sigma = g(M) \);
- Verification of \( (M, \sigma) \): check \( f(\sigma) = M \)

**Full-Domain Hash (Hash-and-Invert)**

\( \mathcal{H} : \{0, 1\}^* \rightarrow Y \)

- in order to sign \( m \), one computes \( M = \mathcal{H}(m) \in Y \), and \( \sigma = g(M) \)
- and the verification consists in checking whether \( f(\sigma) = H(m) \)
Random Oracle Model

Random Oracle

- $\mathcal{H}$ is modelled as a truly random function, from $\{0, 1\}^* \rightarrow Y$.
- Formally, $\mathcal{H}$ is chosen at random at the beginning of the game.
- More concretely, for any new query, a random element in $Y$ is uniformly and independently drawn.

Any security game becomes:

$$\text{Succ}_{\text{SG}}^{\text{euf-cma}}(A) = \Pr \left[ \begin{array}{l}
\mathcal{H} \xleftarrow{\$} Y^\infty; (sk, pk) \xleftarrow{\$} K(); (m, \sigma) \xleftarrow{\$} A^S, H(pk) : \\
\forall i, m \neq m_i \land \forall_{pk}(m, \sigma) = 1
\end{array} \right]$$

Security of the FDH Signature

Theorem

The FDH signature achieves $\text{EUF} - \text{CMA}$ security, under the One-Wayness of $\mathcal{P}$, in the Random Oracle Model:

$$\text{Succ}_{\text{FDH}}^{\text{euf-cma}}(t) \leq q_H \times \text{Succ}_{\text{ow}}^\mathcal{P}(t + q_H \tau_f)$$

Assumptions:

- any signing query has been first asked to $\mathcal{H}$
- the forgery has been asked to $\mathcal{H}$
- $\tau_f$ is the maximal time to evaluate $f \in \mathcal{P}$

Real Attack Game

Simulations

- **Game**$_0$: use of the oracles $\mathcal{K}$, $\mathcal{S}$ and $\mathcal{H}$
- **Game**$_1$: use of the simulation of the Random Oracle
- **Game**$_2$: use of the simulation of the Signing Oracle

Simulation of $\mathcal{H}$

$\mathcal{H}(m): \mu \xleftarrow{\$} X$, output $M = f(\mu)$

$\implies \text{Hop-D-Perfect: } \Pr_{\text{Game}_1}[1] = \Pr_{\text{Game}_0}[1]$

Simulation of $\mathcal{S}$

$\mathcal{S}(m):$ find $\mu$ such that $M = \mathcal{H}(m) = f(\mu)$, output $\sigma = \mu$

$\implies \text{Hop-S-Perfect: } \Pr_{\text{Game}_2}[1] = \Pr_{\text{Game}_1}[1]$
**H-Query Selection**

- **Game**\(_3\): random index \( t \leftarrow \{1, \ldots, q_H\} \)

**Event Ev**

If the \( t \)-th query to \( H \) is not the output forgery

We terminate the game and output 0 if \( Ev \) happens

\[ \Rightarrow \text{Hop-S-Non-Negl} \]

Then, clearly

\[
\Pr[1]_{\text{Game}_3} = \Pr[1]_{\text{Game}_2} \times \Pr[\neg Ev] \quad \Pr[Ev] = 1 - 1/q_H
\]

**Summary**

In **Game**\(_4\), when the output is 1, \( \sigma = g(y) = g(f(x)) = x \)

and the simulator computes one exponentiation per hashing:

\[
\Pr[1]_{\text{Game}_4} \leq \text{Succ}^{\text{ow}}_P(t + q_H\tau_f) \\
\Pr[1]_{\text{Game}_4} = \Pr[1]_{\text{Game}_3} \\
\Pr[1]_{\text{Game}_3} = \Pr[1]_{\text{Game}_2} \times \frac{1}{q_H} \\
\Pr[1]_{\text{Game}_2} = \Pr[1]_{\text{Game}_1} \\
\Pr[1]_{\text{Game}_1} = \Pr[1]_{\text{Game}_0} \\
\text{Succ}^{\text{euf-cma}}_{\text{FDH}}(A) \leq q_H \times \text{Succ}^{\text{ow}}_P(t + q_H\tau_f)
\]

**Key Size**

- **Game**\(_4\): \( \mathcal{P} - \text{OW} \) instance \((f, y)\) (where \( f \leftarrow \mathcal{P}, x \leftarrow X, y = f(x)\))

Use of the simulation of the Key Generation Oracle

**Simulation of \( \mathcal{K} \)**

\( \mathcal{K}() : \text{set } pk \leftarrow f \)

**Modification of the simulation of the Random Oracle**

**Simulation of \( H \)**

If this is the \( t \)-th query, \( H(m) \):

\( M \leftarrow y \), output \( M \)

The unique difference is for the \( t \)-th simulation of the random oracle, for which we cannot compute a signature.

But since it corresponds to the forgery output, it cannot be queried to the signing oracle:

\[ \Rightarrow \text{Hop-S-Perfect: } \Pr[1]_{\text{Game}_4} = \Pr[1]_{\text{Game}_3} \]

If one wants \( \text{Succ}^{\text{euf-cma}}_{\text{FDH}}(A) \leq \varepsilon \) with \( t/\varepsilon \approx 2^{80} \)

If one allows \( q_H \) up to \( 2^{60} \)

Then one needs \( \text{Succ}^{\text{ow}}_P(t) \leq \varepsilon \) with \( t/\varepsilon \geq 2^{140} \).

If one uses FDH-RSA: at least 3072 bit keys are needed.
Summary

In Game3, when the output is 1, with probability $1 - p$: 

$$
\begin{align*}
\sigma &= g(M) = g(y \times f(\mu)) = g(y) \times g(f(\mu)) = g(f(x)) \times \mu = x \times \mu \\
Pr_{\text{Game}_3}[1] &\leq \text{Succ}^{\text{ow}}_{\text{FDH}}(t + q_{Hf})/(1 - p) \\
Pr_{\text{Game}_0}[1] &= \text{Succ}^{\text{cma}}_{\text{FDH}}(A) \\
Pr_{\text{Game}_0}[1] &= \text{Succ}^{\text{cma}}_{\text{FDH}}(A) \\
\end{align*}
$$

Key Size

The maximal for $p \mapsto (1 - p)p^{q_S}$ is reached for

$$
p = 1 - \frac{1}{q_S + 1} \rightarrow \frac{1}{q_S + 1} \times \left(1 - \frac{1}{q_S + 1}\right)^{q_S} \approx \frac{e^{-1}}{q_S}
$$

- If one wants $\text{Succ}^{\text{cma}}_{\text{FDH}}(t) \leq \varepsilon$ with $t/\varepsilon \approx 2^{80}$
- If one allows $q_S$ up to $2^{30}$

Then one needs $\text{Succ}^{\text{ow}}_{\text{P}}(t) \leq \varepsilon$ with $t/\varepsilon \geq 2^{110}$.

If one uses FDH-RSA: 2048 bit keys are enough.
Proof of Knowledge

How do I prove that I know a solution $s$ to a problem $P$?

Proof of Knowledge: Soundness

If I can be accepted, I really know a solution: extractor

Proof of Knowledge: Zero-Knowledge

How do I prove that I know a solution $s$ to a problem $P$?
I reveal the solution...
How can do it without revealing any information?
Zero-knowledge: simulator
**Proof of Knowledge**

How do I prove that I know a 3-color covering, without revealing any information?

I choose a random permutation on the colors and I apply it to the vertices I mask the vertices and send it to the verifier. The verifier chooses an edge, I open it, and the verifier checks the validity: 2 different colors.

---

**Schnorr Proofs**

[Schnorr – Eurocrypt '89 - Crypto '89]

**Zero-Knowledge Proof**
- Setting: $(G = \langle g \rangle)$ of order $q$
- $P$ knows $x$, such that $y = g^{-x}$ and wants to prove it to $V$
- $P$ chooses $K \leftarrow \mathbb{Z}_q^*$ and sends $r = g^K$
- $V$ chooses $h \leftarrow \{0, 1\}^k$ and sends it to $P$
- $P$ computes and sends $s = K + xh \mod q$
- $V$ checks whether $r = g^s y^h$

**Signature**
- $(G = \langle g \rangle)$ of order $q$
- $\mathcal{H}: \{0, 1\}^* \rightarrow \mathbb{Z}_q$
- Key Generation $\rightarrow (y, x)$
  - private key $x \in \mathbb{Z}_q^*$
  - public key $y = g^{-x}$
- Signature of $m \rightarrow (r, h, s)$
  - $K \leftarrow \mathbb{Z}_q^*$, $r = g^K$
  - $h = \mathcal{H}(m, r)$ and $s = K + xh \mod q$
- Verification of $(m, r, s)$
  - compute $h = \mathcal{H}(m, r)$ and check $r = g^s y^h$

---

**Secure Multiple Proofs of Knowledge: Authentication**

If there exists an efficient adversary, then one can solve the underlying problem:

---

**Generic Zero-Knowledge Proofs**

- Proof of knowledge of $x$, such that $R(x, y)$
- $P$ builds a commitment $r$ and sends it to $V$
- $V$ chooses a challenge $h \leftarrow \{0, 1\}^k$ for $P$
- $P$ computes and sends the answer $s$
- $V$ checks $(r, h, s)$

**Signature**
- $\mathcal{H}$ viewed as a random oracle
- Key Generation $\rightarrow (y, x)$
  - private: $x$ public: $y$
- Signature of $m \rightarrow (r, h, s)$
  - Commitment $r$
  - Challenge $h = \mathcal{H}(m, r)$
  - Answer $s$
- Verification of $(m, r, s)$
  - compute $h = \mathcal{H}(m, r)$ and check $(r, h, s)$
Protocols

Zero-Knowledge Proof
- Proof of knowledge of $x$
- $P$ sends a commitment $r$
- $V$ sends a challenge $h$
- $P$ sends the answer $s$
- $V$ checks $(r, h, s)$

Signature
- Key Generation $\rightarrow (y, x)$
- Signature of $m \rightarrow (r, h, s)$
  - Commitment $r$
  - Challenge $h = \mathcal{H}(m, r)$
  - Answer $s$
- Verification of $(m, r, s)$
  - compute $h = \mathcal{H}(m, r)$
  - and check $(r, h, s)$

Special soundness
If one can answer to two different challenges $h \neq h'$: $s$ and $s'$ for a unique commitment $r$, one can extract $x$

Splitting Lemma

Idea
When a subset $A$ is “large” in a product space $X \times Y$, it has many “large” sections.

The Splitting Lemma
Let $A \subset X \times Y$ such that $\Pr[(x, y) \in A] \geq \varepsilon$. For any $\alpha < \varepsilon$, define

$$B_\alpha = \left\{ (x, y) \in X \times Y \mid \Pr_y[(x, y') \in A] \geq \varepsilon - \alpha \right\},$$

then

(i) $\Pr[B_\alpha] \geq \alpha$
(ii) $\forall (x, y) \in B_\alpha, \Pr_y[(x, y') \in A] \geq \varepsilon - \alpha$.
(iii) $\Pr[B_\alpha \mid A] \geq \alpha / \varepsilon$.

Outline

1 Basic Security Notions
2 Advanced Security for Signature
3 Forking Lemma
  - Zero-Knowledge Proofs
  - The Forking Lemma
4 Conclusion

Splitting Lemma – Proof

(i) we argue by contradiction, using the notation $\bar{B}$ for the complement of $B$ in $X \times Y$. Assume that $\Pr[B_\alpha] < \alpha$. Then,

$$\varepsilon \leq \Pr[B] \cdot \Pr[A \mid B] + \Pr[\bar{B}] \cdot \Pr[A \mid \bar{B}] < \alpha \cdot 1 + 1 \cdot (\varepsilon - \alpha) = \varepsilon.$$

(ii) straightforward.

(iii) using Bayes’ law:

$$\Pr[B \mid A] = 1 - \Pr[\bar{B} \mid A] = 1 - \Pr[A \mid \bar{B}] \cdot \Pr[\bar{B}] / \Pr[A] \geq 1 - (\varepsilon - \alpha) / \varepsilon = \alpha / \varepsilon.$$
Theorem (The Forking Lemma)

Let \((K, S, V)\) be a digital signature scheme with security parameter \(k\), with a signature as above, of the form \((m, r, h, s)\), where \(h = \mathcal{H}(m, r)\) and \(s\) depends on \(r\) and \(h\) only.

Let \(A\) be a probabilistic polynomial time Turing machine whose input only consists of public data and which can ask \(q_H\) queries to the random oracle, with \(q_H > 0\).

We assume that, within the time bound \(T\), \(A\) produces, with probability \(\varepsilon \geq 7q_H/2^k\), a valid signature \((m, r, h, s)\).

Then, within time \(T' \leq 16q_H T/\varepsilon\), and with probability \(\varepsilon' \geq 1/9\), a replay of this machine outputs two valid signatures \((m, r, h, s)\) and \((m, r, h', s')\) such that \(h \neq h'\).

\[
\begin{align*}
S &= \{(\omega, \mathcal{H}) \mid A^\mathcal{H}(\omega) \text{ succeeds} \land \text{Ind}_{\mathcal{H}}(\omega) \neq \infty\}, \\
S_i &= \{(\omega, \mathcal{H}) \mid A^\mathcal{H}(\omega) \text{ succeeds} \land \text{Ind}_{\mathcal{H}}(\omega) = i\} \quad i \in \{1, \ldots, q_H\}.
\end{align*}
\]

Note: the set \(\{S_i\}\) is a partition of \(S\).

\[
\nu = \Pr[S] \geq \varepsilon - 1/2^k.
\]

Since \(\varepsilon \geq 7q_H/2^k \geq 7/2^k\), then

\[
\nu \geq 6\varepsilon/7.
\]

\[
\Pr[\text{Ind}_{\mathcal{H}}(\omega) \in I \mid S] \geq \frac{1}{2}.
\]

Let \(I\) be the set consisting of the most likely indices \(i\),

\[
I = \{i \mid \Pr[S_i \mid S] \geq 1/2q_H\}.
\]

By definition of \(S_i\),

\[
\Pr[\text{Ind}_{\mathcal{H}}(\omega) \in I \mid S] = \sum_{i \in I} \Pr[S_i \mid S] = 1 - \sum_{i \notin I} \Pr[S_i \mid S].
\]

Since the complement of \(I\) contains fewer than \(q_H\) elements,

\[
\sum_{i \notin I} \Pr[S_i \mid S] \leq q_H \times 1/2q_H \leq 1/2.
\]
Forking Lemma – Proof

- We run $2/\varepsilon$ times $A$, with independent random $\omega$ and random $H$. Since $\nu = \Pr[S] \geq 6\varepsilon/7$, with probability greater than $1 - (1 - \nu)^2/\varepsilon \geq 4/5$, we get at least one pair $(\omega, H)$ in $S$.
- We apply the Splitting Lemma, with $\varepsilon = \nu/2q_H$ and $\alpha = \varepsilon/2$, for $i \in I$. We denote by $H_i$ the restriction of $H$ to queries of index $< i$.

Since $\Pr[S_i] \geq \nu/2q_H$, there exists a subset $\Omega_i$ such that,

$$\forall (\omega, H) \in \Omega_i, \quad \Pr[H_i(\omega, H') \in S_i | H_i = H_i] \geq \frac{\nu}{4q_H}$$

$$\Pr[\Omega_i | S_i] \geq \frac{1}{2}.$$ 

We know that $\Pr_{H'}[(\omega, H') \in S_\beta | H'_\beta = H_\beta] \geq \nu/4q_H$. Then

$$\Pr[(\omega, H') \in S_\beta \text{ and } h_\beta \neq h'_\beta | H'_\beta = H_\beta]$$

$$\geq \Pr[(\omega, H') \in S_\beta | H'_\beta = H_\beta] - \Pr[H'_\beta = h_\beta] \geq \nu/4q_H - 1/2^k,$$

where $h_\beta = H(Q_\beta)$ and $h'_\beta = H'(Q_\beta)$.

Using the assumption that $\varepsilon \geq 7q_H/2^k$, the above prob. is $\geq \varepsilon/14q_H$.

We replay the attack $14q_H/\varepsilon$ times with a new random oracle $H'$ such that $H'_\beta = H_\beta$, and get another success with probability greater than

$$1 - (1 - \varepsilon/14q_H)^{14q_H/\varepsilon} \geq 3/5.$$
Chosen-Message Attacks

In order to answer signing queries, one simply uses the simulator of the zero-knowledge proof: \((r, h, s)\), and we set \(\mathcal{H}(m, r) \leftarrow h\).

The random oracle programming may fail, but with negligible probability.

Conclusion

Two generic methodologies for signatures

- hash and invert
- the Forking Lemma

Both in the random-oracle model

- Cramer-Shoup: based on the flexible RSA problem
- Based on Pairings
- etc