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

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

### IND – CPA Security Game

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

\[
b \xleftarrow{\text{R}} \{0, 1\}; c = E_{pk}(m_b); b^* \leftarrow A(\text{state}, c)
\]

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

### 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

**Signature**

Goal: Authentication of the sender

The adversary knows the public key only, whereas signatures are not private!
The adversary has access to any signature of its choice:

Chosen-Message Attacks (oracle access):

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

The notion is even stronger (in case of probabilistic signature):

also known as non-malleability:

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

Outline

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

Full-Domain Hash Signature

[Bellare-Rogaway – Eurocrypt ’94]

Signature Scheme

- Key generation: the public key \(f \leftarrow P\) 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)

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

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

Random Oracle

- H is modelled as a truly random function, from \(\{0, 1\}^*\) into \(Y\).
- Formally, \(H\) 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:

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

Real Attack Game

Game 0

- \(K\), \(S\), \(H\) are oracles.
- Adversary queries oracles.
- Challenger checks \((m, \sigma)\).
- If new and valid: 1.
- Else 0.

Simulations

- **Game\(_0\)**: use of the oracles \(K\), \(S\) and \(H\).
- **Game\(_1\)**: use of the simulation of the Random Oracle.
- **Game\(_2\)**: use of the simulation of the Signing Oracle.

Simulation of \(H\)

\(H(m): \mu \overset{R}{\leftarrow} X\), output \(M = f(\mu)\)

\(\Rightarrow\) Hop-D-Perfect: \(\Pr_{\text{Game}_1}[1] = \Pr_{\text{Game}_0}[1]\)

Simulation of \(S\)

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

\(\Rightarrow\) Hop-S-Perfect: \(\Pr_{\text{Game}_2}[1] = \Pr_{\text{Game}_1}[1]\)

Security of the FDH Signature

Theorem

The FDH signature achieves EUF-CMA security, under the One-Wayness of \(P\), in the Random Oracle Model:

\[
\text{Succ}_{\text{Seuf-cma}}^\text{FDH} (t) \leq q_H \times \text{Succ}_{\text{ow}}^{P}(t + q_H^\tau_f)
\]

Assumptions:

- any signing query has been first asked to \(H\)
- the forgery has been asked to \(H\)
- \(\tau_f\) is the maximal time to evaluate \(f \in P\)

Simulations

- **Game\(_0\)**: use of the oracles \(K\), \(S\) and \(H\).
- **Game\(_1\)**: use of the simulation of the Random Oracle.
- **Game\(_2\)**: use of the simulation of the Signing Oracle.
H-Query Selection

- **Game**$_3$: random index $t \overset{R}{\leftarrow} \{1, \ldots, q_H\}$

**Event Ev**

If the $t$-th query to $H$ is not the output forgery

We terminate the game and output 0 if $\text{Ev}$ happens

$\Rightarrow$ **Hop-S-Non-Negl**

Then, clearly

$$\Pr[1] = \frac{\Pr[1] \times \Pr[\neg \text{Ev}]}{\Pr[\text{Ev}]} = 1 - 1/q_H$$

**Summary**

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

and the simulator computes one exponentiation per hashing:

$$\Pr[1] \leq \text{Succ}^{\text{OW}}_P(t + q_H\tau f)$$

$$\Pr[1] = \frac{\Pr[1]}{\Pr[\text{Ev}] = 1 - 1/q_H}$$

**Key Size**

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

If one allows $q_H$ up to $2^{60}$

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

If one uses FDH-RSA: at least 3072 bit keys are needed.
In the case that $f$ is homomorphic (as RSA): $f(ab) = f(a)f(b)$
- Game$_0$: use of the oracles $K$, $S$ and $H$
- Game$_1$: use of the simulation of the Random Oracle

### Simulation of $H$

$H(m) = \mu \xrightarrow{R} X$, output $M = f(\mu)$

- Hop-D-Perfect: $\Pr_{\text{Game}_1}[1] = \Pr_{\text{Game}_0}[1]$
- Game$_2$: use of the homomorphic property
  $P - \text{OW}$ instance $(f, y)$ (where $f \xrightarrow{R} P, x \xrightarrow{R} X, y = f(x)$)

### Simulation of $H$

$H(m)$: flip a biased coin $b$ (with $P[b = 0] = p$), $\mu \xrightarrow{R} X$.
If $b = 0$, output $M = f(\mu)$, otherwise output $M = y \times f(\mu)$

- Hop-D-Perfect: $\Pr_{\text{Game}_2}[1] = \Pr_{\text{Game}_1}[1]

### Summary

In Game$_3$, when the output is 1, with probability $1 - p$:

$$\sigma = g(M) = g(y \times f(\mu)) = g(y) \times g(f(\mu)) = g(f(x)) \times \mu = x \times \mu$$

$$\Pr_{\text{Game}_3}[1] \leq \frac{\Pr_{\text{Game}_2}[1]}{\Pr_{\text{Game}_1}[1]} \times p^{qs}$$

### Key Size

$$\Pr_{\text{Game}_3}[1] = \Pr_{\text{Game}_2}[1] \times \Pr_{\text{Game}_1}[1]$$

$$\Pr_{\text{Game}_0}[1] = \Pr_{\text{Game}_1}[1] = \Pr_{\text{Game}_2}[1] = \Pr_{\text{Game}_3}[1] = \text{Succ}_{\text{FDH}}^{\text{cma}}(A)$$

$$\text{Succ}_{\text{FDH}}^{\text{cma}}(A) \leq \frac{1}{(1 - p)p^{qs}} \times \text{Succ}_{\text{P}}^{\text{cma}}(t + q_{Hf})$$

The maximal for $p \rightarrow (1 - p)p^{qs}$ is reached for

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

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

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

If one uses FDH-RSA: 2048 bit keys are enough.
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?

(a) 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. The verifier checks the validity: 2 different colors.

Schnorr Proofs

Zero-Knowledge Proof

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

Signature

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

Secure Multiple Proofs of Knowledge: Authentication

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

Schnorr Proofs

Zero-Knowledge Proof

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

Signature

- \(H\) viewed as a random oracle
- Key Generation \(\rightarrow (y, x)\)
  - private: \(x\) public: \(y\)
- Signature of \(m \rightarrow (r, h, s)\)
  - Commitment \(r\)
  - Challenge
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 $h = H(m, r)$
  - Answer $s$
- Verification of $(m, r, s)$
  - compute $h = H(m, r)$
  - 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) $\Pr[B_\alpha | A] \geq \alpha / \varepsilon$.

(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,

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

(ii) straightforward.

(iii) using Bayes’ law:

$$\Pr[B | A] = 1 - \Pr[\bar{B} | A]$$

$$= 1 - \Pr[A | \bar{B}] \cdot \Pr[\bar{B}] / \Pr[A] \geq 1 - (\varepsilon - \alpha) / \varepsilon = \alpha / \varepsilon.$$
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 \(h = H(m, r)\) 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 \(T' \leq 16q_HT/\varepsilon\), 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 \(h \neq h'\).

Forking Lemma – Proof

A is a PPTM with random tape \(\omega\).

During the attack, \(A\) asks a polynomial number of queries to \(H\).

We may assume that these questions are distinct:
- \(Q_1, \ldots, Q_{q_H}\) are the \(q_H\) distinct questions
- and let \(H = (h_1, \ldots, h_{q_H})\) be the list of the \(q_H\) answers of \(H\).

Note: a random choice of \(H\) = a random choice of \(H\).

For a random choice of \((\omega, H)\), with probability \(\varepsilon\), \(A\) outputs a valid signature \((m, r, h, s)\).

Since \(H\) is a random oracle, the probability for \(h\) to be equal to \(H(m, r)\) is less than \(1/2^k\), unless it has been asked during the attack.

Accordingly, we define \(I_{H}(\omega)\) to be the index of this question: \((m, r) = Q_{I_{H}(\omega)}\) \((I_{H}(\omega) = \infty\) if the question is never asked).

Let \(I\) be the set consisting of the most likely indices \(i\),
\[
I = \{i \mid \Pr[S_i \mid S] \geq 1/2q_H\}.
\]

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

\[
\nu = \Pr[S] \geq \frac{1}{2}.
\]

By definition of \(S_i\),
\[
\Pr[I \subseteq S_i] = \sum_{i \in I} \Pr[S_i \mid S] = 1 - \sum_{i \notin I} \Pr[S_i \mid S].
\]

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 $H_{\beta}$ the restriction of $H$ to queries of index $\leq i$.

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

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

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 \(H(m, r) \leftarrow h\). 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

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