Provable Security in the Computational Model

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**

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

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

**IND – CPA Security Game**

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

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

---

**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
Signature

Goal: Authentication of the sender

The adversary knows the public key only, whereas signatures are not private!
**H-Query Selection**

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

**Event \( Ev \)**

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

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

Then, clearly

\[
\Pr_{Game_3}[1] = \Pr_{Game_2}[1] \times \Pr[\neg Ev] = 1 - 1/q_H
\]

\[
\Pr_{Game_3}[1] = \Pr_{Game_2}[1] \times \frac{1}{q_H}
\]

**Summary**

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

and the simulator computes one exponentiation per hashing:

\[
\Pr_{Game_4}[1] \leq \text{Succ}_{FDH}^{euf-cma}(A) \leq q_H \times \text{Succ}_{P}^{ow}(t + q_H \tau_f)
\]

- If one wants \( \text{Succ}_{FDH}^{euf-cma}(A) \leq \varepsilon \) with \( t/\varepsilon \approx 2^{80} \)
- If one allows \( q_H \) up to \( 2^{60} \)

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

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

Game$_3$: use of the simulation of the Signing Oracle

Simulation of $S$

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

Fails (with output 0) if $H(m) = M = y \times f(\mu)$:

$$\implies \text{Hop-S-Non-Negl: } \Pr_{\text{Game}_3}[1] = \Pr_{\text{Game}_2}[1] \times p^{q_S}$$

Summary

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

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

$$\Pr_{\text{Game}_3}[1] \leq \text{Succ}^{ow}(t + q_H \tau_f) / (1 - p)$$

$$\Pr_{\text{Game}_3}[1] = \Pr_{\text{Game}_2}[1] \times p^{q_S}$$

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

$$\Pr_{\text{Game}_1}[1] = \Pr_{\text{Game}_0}[1]$$

$$\Pr_{\text{Game}_0}[1] = \text{Succ}^{\text{euf-cma}}(A)$$

$$\text{Succ}^{\text{euf-cma}}_\text{FDH}(A) \leq \frac{1}{(1 - p)p^{q_S}} \times \text{Succ}^{ow}_P(t + q_H \tau_f)$$

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

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

Key Size

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

If one allows $q_S$ up to $2^{30}$

Then one needs $\text{Succ}^{ow}_P(t) \leq \varepsilon$ 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?

(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.

Secure Multiple Proofs of Knowledge: Authentication

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

Schnorr Proofs

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 \( K \sim \mathbb{Z}_q \)
- \( P \) computes and sends \( r = g^K \)
- \( V \) chooses \( h \sim \{0, 1\}^k \)
- \( V \) checks whether \( r \equiv g^{xy} \)

Signature

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 \( m \rightarrow (r, h, s) \)
  - \( K \sim \mathbb{Z}_q \)
  - \( r = g^K \)
  - \( h = H(m, r) \)
  - \( s = K + xh \mod q \)
- Verification of \( (m, r, s) \)
  - compute \( h = H(m, r) \)
  - and check \( r \equiv g^s y^h \)

Generic Zero-Knowledge 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 \( h \sim \{0, 1\}^k \) 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 \( h = H(m, r) \)
  - Answer \( s \)
- Verification of \( (m, r, s) \)
  - compute \( h = H(m, r) \)
  - and check \( (r, h, s) \)
**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 $h = H(m, r)$
  - 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 $A \subseteq X \times Y$ such that $\Pr[(x, y) \in A] \geq \varepsilon$. For any $\alpha < \varepsilon$, define

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

then

(i) $\Pr[B] \geq \alpha$

(ii) $\forall (x, y) \in B, \Pr_{y \in Y}[x, y] \in A] \geq \varepsilon - \alpha$.

(iii) $\Pr[B \mid A] \geq \alpha / \varepsilon$.

**Splitting Lemma – Proof**

(i) we argue by contradiction, using the notation $\bar{B}$ for the complement of $B$ in $X \times Y$. Assume that $\Pr[\bar{B}] < \alpha$. 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 \(T' \leq 16q_HT/\varepsilon\), 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 \(h \neq h'\).

Accordingly, we define \(\text{Ind}_\mathcal{H}(\omega)\) to be the index of this question:

\((m,r) = Q_{\text{Ind}_\mathcal{H}(\omega)} \) \((\text{Ind}_\mathcal{H}(\omega) = \infty\) if the question is never asked).


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 $1 - (1 - \nu)^2 \geq 4/5$, 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,
  \[ \forall (\omega, H') \in \Omega_i, \quad \Pr_{H^0}[(\omega, H') \in S_i | H'|_i = H|_i] \geq \frac{\nu}{4q_H}, \]
  \[ \Pr[\Omega_i | S_i] \geq \frac{1}{2}. \]

Finally, after less than $2/\varepsilon + 14q_H/\varepsilon$ repetitions of the attack, with probability greater than $1/5 \times 3/5 \geq 1/9$, we have obtained two signatures $(m, r, h, s)$ and $(m, r, h', s')$, both valid w.r.t. their specific random oracle $H$ or $H'$:

\[ Q = (m, r) \text{ and } h = H(Q) \neq H'(Q) = h'. \]
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