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
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**

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

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

- **Goal:** Authentication of the sender

### EUF – NMA

- **Succ\textsuperscript{euf}(A) = Pr[(sk, pk) \leftarrow K(); (m, \sigma) \leftarrow A(pk) : V_{pk}(m, \sigma) = 1]**

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

Signature Scheme

- Key generation: the public key $f \gets P$ 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)

$H : \{0, 1\}^2 \rightarrow Y$

- in order to sign $m$, one computes $M = H(m) \in Y$, and $\sigma = g(M)$
- and the verification consists in checking whether $f(\sigma) = H(m)\text{ or } f(\sigma) = H(m')$
**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:

$$\text{Succ}_{SG}^{\text{euf-cma}}(A) = \Pr \left[ H \overset{R}{\rightarrow} Y^\infty ; (sk, pk) \leftarrow K(); (m, \sigma) \leftarrow A^{S,H}(pk) : \forall i, m \notin m_i \land V_{pk}(m, \sigma) = 1 \right]$$

**Theorem**

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

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

**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 $f \in P$

**Real Attack Game**

- **Game 0**: use of the oracles $K$, $S$ and $H$
- **Game 1**: use of the simulation of the Random Oracle

**Simulation of $H$**

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

$$\Rightarrow \text{Hop-D-Perfect} \quad \text{Pr}_{\text{Game}_0[1]} = \text{Pr}_{\text{Game}_1[1]}$$

**Simulation of $S$**

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

$$\Rightarrow \text{Hop-S-Perfect} \quad \text{Pr}_{\text{Game}_2[1]} = \text{Pr}_{\text{Game}_1[1]}$$

**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**: 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 $\text{Ev}$ happens

$\Rightarrow \text{Hop-S-Non-Negl}$

Then, clearly

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

**Summary**

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

and the simulator computes one exponentiation per hashing:

- $\Pr_{\text{Game}_3}[1] \leq \text{Succ}^\text{cma}_{\text{FDH}}(A) \leq q_H \times \text{Succ}^\text{OW}_P(t + q_H\tau_f)$

- If one wants $\text{Succ}^\text{cma}_{\text{FDH}}(t) \leq \varepsilon$ 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.
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 $H$**

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

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

- **Game$_2$**: use of the *homomorphic property*
  - $P - OW$ instance $(f, y)$ (where $f \xleftarrow{\$} P, x \xleftarrow{\$} X, y = f(x)$)

**Simulation of $H$**

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

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

**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 $H(m) = M = y \times f(\mu)$:
but with probability $p^{qs}$

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

**Summary**

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

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

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

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

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

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

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

The maximal for $p \not\geq (1 - p)p^{qs}$ is reached for

$p = 1 - \frac{1}{q_S + 1} \Rightarrow \frac{1}{q_S + 1} \times 1 - \frac{1}{q_S + 1} \approx e^{-1}$

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

Then one needs $\text{Succ}_P^{OW}(t) \leq \varepsilon$ 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 I 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. 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**

**[Schnorr – Eurocrypt ’89 - Crypto ’89]**

---

**Generic Zero-Knowledge Proofs**

**Zero-Knowledge Proof**

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

**Signature**

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

**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 \( h \subset \{0,1\}^k \) for P
- P computes and sends the answer \( s \)
- V checks \( (r, h, s) \)

**Signature**

- H viewed as a random oracle
- Key Generation \( \to (y, x) \)
  - private: \( x \)
  - public: \( y \)
- Signature of \( m \to (r, h, s) \)
  - 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 $h \notin h^0$, $s$ and $s^0$ 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

$$B_\alpha = \{(x, y) \in X \times Y \mid \Pr_{y' \in Y}[(x, y^0) \in A] \geq \varepsilon - \alpha\},$$

then

$(i)$ $\Pr[B_\alpha] \geq \alpha$
$(ii)$ $\forall (x, y) \in B_\alpha, \Pr_{y' \in Y}[(x, y^0) \in A] \geq \varepsilon - \alpha$.
$(iii)$ $\Pr[B_\alpha \mid A] \geq \alpha/\varepsilon$.

---

**Splitting Lemma – Proof**

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

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

$(ii)$ 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 \(h = H(m, r)\) 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 \(T^0 = 16q_HT/\varepsilon\), and with probability \(\varepsilon^0 \geq 1/9\), a replay of this machine outputs two valid signatures \((m, r, h, s)\) and \((m, r, h^0, s^0)\) such that \(h \neq h^0\).

Proof

We then define the sets

\[
S = (\omega, H) \mid A^H(\omega) \text{ succeeds } \land \text{ Ind}_H(\omega) \equiv \infty,
\]

\[
S_i = (\omega, H) \mid A^H(\omega) \text{ succeeds } \land \text{ Ind}_H(\omega) = i, \quad i \in \{1, \ldots, q_H\}.
\]

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

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

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

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

Lemma

\[
\Pr[\text{Ind}_H(\omega) \in l | S] \geq \frac{1}{2}.
\]

By definition of \(S_i\),

\[
\Pr[\text{Ind}_H(\omega) \in l | S] = \prod_{i \not\in l} \Pr[S_i | S] = 1 - \prod_{i \in l} \Pr[S_i | S].
\]

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

\[
\Pr[S_i | S] \leq q_H \times 1/2q_H \leq 1/2.
\]
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 $\Pr[S] \geq \nu / 2q_H$, there exists a subset $\Omega_i$ such that

$$\forall (\omega, H) \in \Omega_i, \quad Pr_H[(\omega, H^0) \in S \mid H_i^0 = H_i^i] \geq \frac{\nu}{4q_H}$$

$$\Pr[\Omega_i \mid S] \geq \frac{1}{2}.$$

Since all the subsets $S_i$ are disjoint,

$$\Pr[(\exists i \in I) (\omega, H) \in \Omega_i \cap S_i \mid S]$$

$$= \Pr \left[ \bigcap_{i \in I} (\Omega_i \cap S_i) \mid S \right] = \prod_{i \in I} \Pr[\Omega_i \cap S_i \mid S]$$

$$= \prod_{i \in I} \Pr[S_i \mid S] \cdot \Pr[S_i \mid S] \geq \prod_{i \in I} \Pr[S_i \mid S] / 2 \geq \frac{1}{4}.$$

We let $\beta$ denote the index $\text{Ind}_H(\omega)$ of the successful pair.

With prob. at least $1/4$, $\beta \in I$ and $(\omega, H) \in S_\beta \cap \Omega_\beta$.

With prob. greater than $4/5 \times 1/4 = 1/5$, the $2/\varepsilon$ attacks provided a successful pair $(\omega, H)$, with $\beta = \text{Ind}_H(\omega) \in I$ and $(\omega, H) \in S_\beta$.

Forking Lemma - Proof

We know that $\Pr_H[(\omega, H^0) \in S_\beta \mid H^0_\beta = H^0_\beta] \geq \nu / 4q_H$. Then

$$\Pr_H[(\omega, H^0) \in S_\beta \text{ and } h_\beta \notin H^0_\beta \mid H^0_\beta = H^0_\beta]$$

$$\geq \Pr_H[(\omega, H^0) \in S_\beta \mid H^0_\beta = H^0_\beta] - \Pr_H[H^0_\beta = h_\beta] \geq \nu / 4q_H - 1/2^k,$$

where $h_\beta = H(Q_\beta)$ and $h^0_\beta = H^0(Q_\beta)$.

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 $H^0$ such that $H^0_\beta = H^0_\beta$, and get another success with probability greater than

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

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 $(m, r, h^0, s^0)$, both valid w.r.t. their specific random oracle $H$ or $H^0$.

$$Q_\beta = (m, r)$$

and $h = H(Q_\beta) \notin H^0(Q_\beta) = h^0$. 

**Q**
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