Katz, Lindell
Introduction to Modern Cryptrography
Slides Chapter 12

Markus Bläser, Saarland University

Digital signature schemes

- Goal: integrity of messages
- Signer "signs" a message using a private key
- Receiver can verify the signature using a public key

Difference to MACs:

- Signature can by verified publicly
- Signature is transferable
- Signature cannot be "taken back"

Definition

Definition (12.1)

A *digital signature scheme* is a triple of ppt algorithms $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ such that

1. Gen on input 1^n outputs a public key pk and a private key sk. Both have length at least n and we can determine n from them.
2. The signing algorithm Sign takes as input sk and a message m (from some message space that might depend on pk). It outputs a signature σ.
3. The deterministic verification algorithm Vrfy takes as input pk, a message m, and a signature σ. It outputs a bit b. $b = 1$ means that *valid* and $b = 0$ means invalid.

For every legal message m, we have $\mathsf{Vrfy}_{pk}(m, \mathsf{Sign}_{sk}(m)) = 1$ except with negligible probability over (pk, sk) generated by $\mathsf{Gen}(1^n)$.

Security

The signature experiment Sig-Forge$_{\mathcal{A},\Pi}(n)$**:**

1. Run Gen(1^n) to generate (pk, sk).
2. $\mathcal{A}$ is given pk and oracle access to Sign$_{sk}(.)$. $\mathcal{A}$ outputs (m, σ). Let $\mathcal{Q}$ denote the set of all queries asked by $\mathcal{A}$ to the oracle.
3. $\mathcal{A}$ wins if (1) Vrfy$_{pk}(m, \sigma) = 1$ and (2) $m \notin \mathcal{Q}$. Otherwise he looses.

Definition (12.2)

A signature scheme π is *secure* (existentially unforgeable under an adaptive chosen-message attack) if for all ppt $\mathcal{A}$,

$$\Pr[\text{Sig-Forge}_{\mathcal{A},\Pi}(n) = 1] \leq \text{negl}(n).$$

Hash and sign

Construction 12.3 Let $\Pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ be a signature scheme for messages of length $\ell(n)$ and let $\Pi_H = (\mathsf{Gen}_H, H)$ be a hash function with output length $\ell(n)$. Construct Π' as follows:

- Gen': On input 1^n run $\mathsf{Gen}(1^n)$ to obtain (pk, sk) and run $\mathsf{Gen}_H(1^n)$ to obtain s. The public key is (pk, s) and the private key is (sk, s).
- Sign': on input $m \in \{0,1\}^*$, output $\sigma \leftarrow \mathsf{Sign}_{sk}(H^s(m))$.
- Vrfy': on input m and signature σ output 1 if and only if $\mathsf{Vrfy}_{pk}(H^s(m), \sigma) = 1$.

Theorem (12.4)

If Π is a secure signature scheme for messages of length ℓ and Π_H is collision resistant, then Construction 12.3 is secure (for arbitrary message lengths).

Plain RSA

Construction 12.5

- Gen: on input 1^n, run GenRSA(1^n) to obtain (N, e, d). The public key is (N, e) and the private key is (N, d).
- Sign: on input (N, d) and $m \in \mathbb{Z}_N^*$, output $\sigma := m^d \mod N$.
- Vrfy: on input (N, e) and message $m \in \mathbb{Z}_N^*$ and signature $\sigma \in \mathbb{Z}_N^*$, output 1 if and only if $m = \sigma^e \mod N$.

Not secure!

RSA-FDH

Construction 12.6

- Gen: on input 1^n, run GenRSA(1^n) to obtain (N, e, d). The public key is (N, e) and the private key is (N, d).
  Choose a function $H : \{0,1\}^* \to \mathbb{Z}_N^*$.
- Sign: on input (N, d) and $m \in \{0,1\}^*$, output $\sigma := H(m)^d \mod N$.
- Vrfy: on input (N, e) and message $m \in \{0,1\}^*$ and signature $\sigma \in \mathbb{Z}_N^*$, output 1 if and only if $H(m) = \sigma^e \mod N$.

Theorem (12.7)

If the RSA problem is hard relative to GenRSA *and* H *is modelled as a random oracle, then Construction 12.6 is secure.*

Proof

$\text{Sig-Forge}_{\mathcal{A},\Pi}(n)$:

1. $\text{GenRSA}(1^n)$ is run to obtain (N, e, d). A random $H : \{0,1\}^* \to \mathbb{Z}_N^*$ is chosen.
2. $\mathcal{A}$ is given $pk = (N, e)$ and may query $H(.)$ as well as $\text{Sign}_{(N,d)}(.)$ (where $\text{Sign}_{(N,d)}(m) = H(m)^d \mod n$).
3. $\mathcal{A}$ outputs (m, σ), where it had to not previously requested a signature on m. The output of the experiment is 1 iff $\sigma^e = H(m) \mod N$.

Proof (2)

W.l.o.g.:

- when $\mathcal{A}$ calls the signature oracle on m or outputs (m, σ), it previously requested $H(m)$.
- $\mathcal{A}$ makes exactly q queries to H. (Let $m_1, \ldots, m_q$ be these queries.)

Sig-Forge$'_{\mathcal{A},\Pi}(n)$:

1. Choose uniform $j \in \{1, \ldots, q\}$.
2. GenRSA(1^n) is run to obtain (N, e, d). A random $H : \{0,1\}^* \to \mathbb{Z}_N^*$ is chosen.
3. $\mathcal{A}$ is given $pk = (N, e)$ and may query $H(.)$ as well as Sign$_{(N,d)}(.)$ (where Sign$_{(N,d)}(m) = H(m)^d \mod N$).
4. $\mathcal{A}$ outputs (m, σ), where it had to not previously requested a signature on m. Let i be such that $m = m_i$. The output of the experiment is 1 iff $\sigma^e = H(m) \mod N$ and $j = i$.

Proof (3)

Algorithm $\mathcal{A}'$: input (N, e, y)

1. Choose uniform $j \in \{1, \ldots, q\}$.
2. Run $\mathcal{A}$ on input $pk = (N, e)$. $\mathcal{A}'$ maintains a table: An entry (m_i, σ_i, y_i) indicates that $\mathcal{A}'$ has set $H(m_i) = y_i$ and $\sigma_i^e = y_i \mod N$.
3. When $\mathcal{A}$ makes its ith query $H(m_i)$, answer it as follows:
   - 3.1 if $i = j$, then return y as the answer.
   - 3.2 else choose $\sigma_i \in \mathbb{Z}_N^*$ uniformly at random, compute $y_i := \sigma_i^e \mod N$, return y_i and store (m_i, σ_i, y_i) in the table.

   When $\mathcal{A}$ requests a signature on message m, let i be such that $m = m_i$, and answer the query as follows:
   - 3.1 if $i = j$, then $\mathcal{A}'$ aborts
   - 3.2 if $i \neq j$, then there is an entry (m_i, σ_i, y_i) in the table. Return σ_i.
4. $\mathcal{A}$ outputs (m, σ) in the end. If $m = m_j$ and $\sigma^e = y \mod N$, then output σ.

Identification schemes

Scenario:

- A prover ("user") tries to identify itself to a verifier ("server").
- Public key setting: Verifier only knows the public key of the prover.

Three-round identification protocol:

1. Prover runs an algorithm $\mathrm{P}_1(sk)$ obtaining an initial message I and some state st and sends I to the verifier.
2. The verifier chooses a challenge $\mathrm{r} \in \Omega_{pk}$ uniformly at random and sends it to the prover.
3. The prover runs a second algorithm P_2, computes $s := \mathrm{P}_2(sk, \mathrm{st}, \mathrm{r})$ and sends it to the verifier.
4. The verifier runs an algorithm V and accepts if $V(pk, \mathrm{r}, s) = \mathrm{I}$

- P_1 is ppt, P_2 and V are deterministic polytime.
- *nondegenerate*: probability that P_1 outputs some specific I should be negligible.

Security

- Let $\text{trans}_{sk}()$ be an oracle that, when called without any input, outputs a transcript $(\mathrm{I}, \mathrm{r}, s)$
- models eavesdropping of multiple (honest) executions of the protocol.

The identification experiment $\text{Ident}_{\mathcal{A},\Pi}(\mathrm{n})$**:**

1. $\text{Gen}(1^{\mathrm{n}})$ is run to obtain (pk, sk).
2. Adversary $\mathcal{A}$ is given pk and access to an oracle $\text{trans}_{sk}()$.
3. Eventually, $\mathcal{A}$ outputs a message I. A challenge $\mathrm{r} \in \Omega_{pk}$ is chosen and given to $\mathcal{A}$. $\mathcal{A}$ outputs some s.
4. The outcome of the experiment is 1 iff $\mathrm{V}(pk, \mathrm{r}, s) = \mathrm{I}$.

Security (2)

Definition (12.8)

An identification scheme $\Pi = (\mathsf{Gen}, \mathsf{P}_1, \mathsf{P}_2, \mathsf{V})$ is *secure* (against passive attacks) if for all ppt $\mathcal{A}$,

$$\Pr[\mathsf{Ident}_{\mathcal{A},\Pi}(n) = 1] \leq \mathsf{negl}(n).$$

Fiat-Shamir transform

Construction 12.9

Let $(\mathsf{Gen}_{\mathsf{id}}, \mathsf{P}_1, \mathsf{P}_2, \mathsf{V})$ be an identification scheme and construct a signature scheme as follows:

- Gen: On input 1^n, run $\mathsf{Gen}_{\mathsf{id}}$ to obtain (pk, sk). The public key pk specifies a set of challenges Ω_{pk}. A function $\mathsf{H} : \{0,1\}^* \times \{0,1\}^* \to \Omega_{pk}$ is chosen.
- Sign: On input sk and $\mathrm{m} \in \{0,1\}^*$ do:
  1. $(\mathrm{I}, \mathrm{st}) \leftarrow \mathsf{P}_1(sk)$
  2. $\mathrm{r} := \mathsf{H}(\mathrm{I}, \mathrm{m})$
  3. $s := \mathsf{P}_2(sk, \mathrm{st}, \mathrm{r})$
  4. Output (r, s) as a signature.
- Vrfy: On input pk, a message m, and signature (r, s), compute $\mathrm{I} := \mathsf{V}(pk, \mathrm{r}, s)$ and output 1 iff $\mathsf{H}(\mathrm{I}, \mathrm{m}) = \mathrm{r}$.

From identification schemes to signatures

Theorem (12.10)

Let Π be an identification scheme and let Π' be the signature scheme that results by applying the Fiat-Shamir transform to it. If Π is secure and H is modelled as a random oracle, then Π' is secure.

Proof

Algorithm $\mathcal{A}$:

1. Choose uniform $\mathrm{j} \in \{1, \ldots, \mathrm{q}\}$
2. Run $\mathcal{A}'(pk)$. When $\mathcal{A}'$ makes a query $\mathrm{H}(\mathrm{I_i}, \mathrm{m_i})$:
   - 2.1 If $\mathrm{i} = \mathrm{j}$, then output $\mathrm{I_j}$ and receive a challenge r. Return r to $\mathcal{A}'$ as the answer to the query.
   - 2.2 If $\mathrm{i} \neq \mathrm{j}$, then choose a uniform $\mathrm{r} \in \Omega_{pk}$ and return r as the answer to the query.

   When $\mathcal{A}'$ request a signature on m:
   - 2.1 Query $\mathrm{trans}_{sk}()$ to obtain a transcript $(\mathrm{I}, \mathrm{r}, \mathrm{s})$.
   - 2.2 Return (r, s)
3. If $\mathcal{A}'$ outputs a forged signature (r, s) on a message m, compute $\mathrm{I} := \mathrm{V}(pk, \mathrm{r}, s)$ and check whether $(\mathrm{I}, \mathrm{m}) = (\mathrm{I_j}, \mathrm{m_j})$. If yes, output s. Otherwise, abort.

Schnorr identification scheme

Setup:

- Discrete-logarithm-based identification scheme
- A group (G, q, g) is generated
- The Prover chooses a uniform $x \in \mathbb{Z}_q$ and sets $y := g^x$. The private key is x and the public key is (G, q, g, y).

Protocol:

1. The prover chooses $k \leftarrow \mathbb{Z}_q$ uniformly at random and sends $I := g^k$.
2. The verifier chooses $r \in \mathbb{Z}_q$ uniformly at random.
3. The prover computes $s := rx + k \mod q$ and sends s.
4. The verifier checks whether $g^s \cdot y^{-r} = I$.

Security

Theorem (12.11)

If the discrete logarithm problem is hard relative to the group generation algorithm, then the Schnorr identification algorithm is secure.

Oracle access to $\text{trans}_{sk}()$ is of no use!

1. In an honest transcript, I and r are uniform and independent and $s = \log_{\mathrm{g}}(\mathrm{I} \cdot \mathrm{y}^{\mathrm{r}})$.
2. $\mathcal{A}$ can generate a transcript by choosing s and r uniformly at random and setting $\mathrm{I} := \mathrm{g}^{\mathrm{s}} \cdot \mathrm{y}^{-\mathrm{r}}$.
3. Since s is uniform and independent of r, I is also uniform.

Proof

Algorithm $\mathcal{A}'$: Input (G, q, g, y)

1. Run $\mathcal{A}(pk)$ answering all its queries to $\text{trans}_{sk}()$ by generating some transcript.
2. When $\mathcal{A}$ outputs I, choose $r_1 \in \mathbb{Z}_q$ uniformly at random. Give r_1 to $\mathcal{A}$ and let s_1 be its response.
3. Run $\mathcal{A}(pk)$ a second time (from scratch) using the same random bits as before except for a new uniform $z_2 \in \mathbb{Z}_q$. Let s_2 be $\mathcal{A}$'s new response.
4. If $g^{s_1} \cdot y^{-r_1} = I$ and $g^{s_2} \cdot y^{-r_2} = I$ and $r_1 \neq r_2$, then output $(s_1 - s_2)(r_1 - r_2)^{-1} \mod q$. Else output nothing.

Schnorr signature scheme

Construction 12.12

- Gen: run the group generation algorithm to generate (G, q, g). Choose a uniform $x \in \mathbb{Z}_q$ and set $y = g^x$. The private key is x and the public key is (G, q, g, y). Choose a function $H : \{0,1\}^* \to \mathbb{Z}_q$.
- Sign: in input x and message $m \in \{0,1\}^m$, choose a uniform $k \in \mathbb{Z}_q$ and set $I := g^k$. Compute $r := H(I, m)$ and $s := rx + k \mod q$. Output (r, s).
- Vrfy: on input (G, q, g, y), a message m, and a signature (r, s), compute $I := g^s \cdot y^{-r}$ and output 1 if $H(I, m) = r$ and 0 otherwise.

DSA and ECDSA

- G cyclic group of prime order q with generator g.
- $sk = x \in \mathbb{Z}_q$ and $pk = (G, q, g, y)$ with $y = g^x$.
- DSA (digital signature algorithm): subgroup of $\mathbb{Z}_p^*$
- ECDSA: subgroup of some elliptic curve $E(\mathbb{Z}_p)$.

Underlying identification scheme:

1. The prover chooses uniform $k \in \mathbb{Z}_q^*$ and sends $I := g^k$.
2. The verifier chooses and sends uniform $\alpha, r \in \mathbb{Z}_q$.
3. The prover sends $k^{-1}(\alpha + xr) \mod q$.
4. The verifier accepts if $s \neq 0$ and $g^{\alpha s^{-1}} \cdot y^{rs^{-1}} = I$

- $s = 0$ only when $\alpha = -xr \mod q \longrightarrow$ negligible.

Signing scheme

- $\alpha := H(m)$ for some cryptographic hash function
- $r := F(I)$ for some "simple" function $F : G \to \mathbb{Z}_q$
- When G is a subgroup of $\mathbb{Z}_p*$, then $F(I) := I \mod q$.
- When G is an elliptic curve, then $F((x, y)) := x \mod q$.

Construction 12.13

- Gen: Generate the underlying group. Choose uniform $x \in \mathbb{Z}_q$ and set $y := g^x$. The public key is (G, q, g, y) and the private key is x. Choose the functions H and F.
- Sign: on input x and message $m \in \{0,1\}^*$, choose a uniform $k \in \mathbb{Z}_q^*$ and set $r := F(g^k)$. Then compute $s := k^{-1} \cdot (H(m) + xr) \mod q$. If $r = 0$ or $s = 0$, then choose a new k. Output (r, s).
- Vrfy: on input a public key, a message m, and a signature (r, s) with $r, s \neq 0 \mod q$ output 1 iff $r = F(g^{H(m)s^{-1}} y^{rs^{-1}})$.

Security

Construction 12.13

- is secure, if discrete logarithm is hard and H and F are random oracles.
- However, F is "far" from begin a random oracle.
- Nevertheless, no attack is known.

One-time security

The one-time signature experiment $\text{Sig-Forge}^{\text{1-time}}_{\mathcal{A},\Pi}(n)$**:**

1. Run $\text{Gen}(1^n)$ to generate (pk, sk).
2. $\mathcal{A}$ is given pk and asks a single query m' to $\text{Sign}_{sk}(.)$. $\mathcal{A}$ outputs (m, σ) with $m \neq m'$.
3. $\mathcal{A}$ wins if $\text{Vrfy}_{pk}(m, \sigma) = 1$. Otherwise he looses.

Definition (12.14)

A signature scheme $\Pi = (\text{Gen}, \text{Sign}, \text{Vrfy})$ is *one-time secure* if for all ppt $\mathcal{A}$,

$$\Pr[\text{Sig-Forge}^{\text{1-time}}_{\mathcal{A},\Pi}(n) = 1] \leq \text{negl}(n)$$

Lamport's signature scheme

Construction 12.15

Let $H : \{0,1\}^* \to \{0,1\}^*$ be a function.

- Gen: On input 1^n, do for $i \in \{1, \dots \ell\}$:
  1. Choose uniform $x_{i,0}, x_{i,1} \in \{0,1\}^n$.
  2. Compute $y_{i,0} := H(x_{i,0})$ and $y_{i,1} := H(x_{i,1})$.

  The public and private key are

$$pk = \begin{pmatrix} y_{1,0} & \cdots & y_{\ell,0} \\ y_{1,1} & \cdots & y_{\ell,1} \end{pmatrix}, \quad sk = \begin{pmatrix} x_{1,0} & \cdots & x_{\ell,0} \\ x_{1,1} & \cdots & x_{\ell,1} \end{pmatrix}.$$

- Sign: on input sk and message $m \in \{0,1\}^\ell$ with $m = m_1 \dots m_\ell$, output $(x_{1,m_1}, \dots, x_{\ell,m_\ell})$.
- Vrfy: in input pk and message $m \in \{0,1\}^\ell$ and signature $\sigma = (x_1, \dots, x_\ell)$, output 1 iff $H(x_i) = y_{i,m_i}$, $1 \le i \le \ell$.

Lamport's signature scheme

Theorem (12.16)

Let ℓ be any polynomial. If H is one-way, then Construction 12.15 is one-time-secure.

Corollary (12.17)

IF one-way functions exist, then there is a one-time secure signature scheme for messages of length ℓ for any polynomial ℓ.

Proof

Algorithm I: Input is 1^n and y

1. Choose uniform $i^* \in \{1, \ldots, \ell\}$ and $b^* \in \{0,1\}$. Set $y_{i^*,b^*} := y$.
2. For all $i \in \{0, \ldots, \ell\}$ and $b \in \{0,1\}$ with $(i,b) \neq (i^*, b^*)$: Choose uniform $x_{i,b} \in \{0,1\}^n$ and set $y_{i,b} := H(x_{i,b})$.
3. Run $\mathcal{A}$ on $pk = \begin{pmatrix} y_{1,0} & \cdots & y_{\ell,0} \\ y_{1,1} & \cdots & y_{\ell,1} \end{pmatrix}$.
4. When $\mathcal{A}$ requests a signature on message m':
   - 4.1 If $m'_{i^*} = b^*$, then I aborts the execution.
   - 4.2 Otherwise, i returns $\sigma = (x_{1,m'_1}, \ldots, x_{\ell,m'_\ell})$.
5. When $\mathcal{A}$ outputs (m, σ) with $\sigma = (x_1, \ldots, x_\ell)$: If $\mathcal{A}$ outputs a forgery at (i^*, b^*), then output x_{i^*}.

Stateful signature schemes

Definition (12.18)

A *stateful signature scheme* is a tuple of ppt algorithms $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$:

1. Gen on input 1^n outputs keys pk and sk and in addition, an *initial state* s_0.
2. The signing algorithm Sign takes sk, a state s_{i-1} and a message $m \in \{0,1\}^*$. It outputs a signature σ and new state s_i.
3. The deterministic verification algorithm Vrfy takes a public key pk, a message m and a signature σ and outputs a bit b.

For every n, (pk, sk, s_0) output by Gen, and any messages $m_1, \ldots, m_t \in \{0,1\}^*$, if we iteratively compute $(\sigma_i, s_i) \leftarrow \mathsf{Sign}_{sk, s_{i-1}}(m_i)$ for $i = 1, \ldots, t$, then for every $1 \leq i \leq t$, $\mathsf{Vrfy}_{pk}(m_i, \sigma_i) = 1$.

Stateful signature schemes (2)

- State is not known to Vrfy. In some schemes, it must even be kept secret.
- Security is defined as for stateless schemes, but the signing oracle does not give the state to the verifier.

Tree-based construction

Lemma (12.19)

If collision resistant hash functions exist, then there exist one-time secure signature schemes (for arbitrary messages length).

Theorem (12.21)

Let Π be a one-time secure signature scheme. Then Construction 12.20 is a secure signature scheme

Tree-based construction

Construction 12.20:

Let $\Pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ be a signature scheme.

- Gen^*: on input 1^n is computes $(pk_\epsilon, pk_\epsilon) \leftarrow \mathsf{Gen}(1^n)$.
- Sign^*: on input $m \in \{0,1\}^n$, do:
  1. for $i = 0$ to $n-1$:
     If $pk_{m|_i 0}, pk_{m|_i 1}$ and $\sigma_{m|_i}$ are not in the state, then compute $(pk_{m|_i 0}, sk_{m|_i 0}) \leftarrow \mathsf{Gen}(1^n)$, $(pk_{m|_i 0}, sk_{m|_i 0}) \leftarrow \mathsf{Gen}(1^n)$, and $\sigma_{m|_i} \leftarrow \mathsf{Sign}_{sk_{m|_i}}(pk_{m|_i 0} \| pk_{m|_i 1})$. Add these values to the state.
  2. If σ_m is not included in the state, then $\sigma_m \leftarrow \mathsf{Sign}_{sk_m}(m)$. Add it to the state.
  3. Output $(\{\sigma_{m|_i}, pk_{m|_i 0}, pk_{m|_i 1}\}_{i=0}^{n-1}, \sigma_m)$.
- Vrfy^*: Given a public key pk_ϵ and a signature as above, output 1 iff
  1. $\mathsf{Vrfy}_{pk_{m|_i}}(pk_{m|_i 0} \| pk_{m|_i 1}, \sigma_{m|_i}) = 1$ for all i and
  2. $\mathsf{Vrfy}_{pk_m}(m, \sigma_m) = 1$.

Proof

Adversary $\mathcal{A}$:

1. Choose $i^* \in \{1, \ldots, \ell\}$. Construct a list $pk^1, \ldots, pk^\ell$ of keys as follows:
   - Set $pk^{i^*} := pk$.
   - For all other i, $(pk^i, sk^i) \leftarrow \mathsf{Gen}(1^n)$.
2. Run $\mathcal{A}^*$ on $pk_\epsilon = pk^1$. When $\mathcal{A}^*$ requests a signature on m:
   - 2.1 For $i := 0$ to $n-1$ do:
     If $pk_{m|_i 0}$, $pk_{m|_i 1}$, and $\sigma_{m|_i}$ have not yet been defined, then set $pk_{m|_i 0}$, $pk_{m|_i 1}$ to the next two unused keys. Compute a signature $\sigma_{m|_i}$ on $pk_{m|_i 0} \| pk_{m|_i 1}$ (when $i \neq i^*$, use sk^i, otherwise use the one query to the signing oracle).
   - 2.2 If σ_m is not defined, compute a signature using pk_m (see above).
   - 2.3 Give $(\{\sigma_{m|_i}, pk_{m|_i 0}, pk_{m|_i 1}\}_{i=0}^{n-1}, \sigma_m)$ to $\mathcal{A}^*$.

Proof (2)

3. When $\mathcal{A}*$ outputs m and a signature $(\{\sigma'_{m|_i}, pk'_{m|_i 0}, pk'_{m|_i 1}\}_{i=0}^{n-1}, \sigma'_m)$, check whether this is valid. If yes:

   Case 1: If there is j for which $pk'_{m|_j 0} \neq pk_{m|_j 0}$ or $pk'_{m|_j 1} \neq pk_{m|_j 1}$ (this includes the case that $pk_{m|_j 0}$ or $pk_{m|_j 1}$ were never defined by $\mathcal{A}$), then take the minimal such j and let i be such that $pk^i = pk_{m|_j} = pk'_{m|_j}$. If $i = i^*$, then output $(pk'_{m|_j 0} || pk'_{m|_j 1}, \sigma'_{m|_j})$.

   Case 2: If case 1 does not hold, then $pk'_m = pk_m$. Let i be such that $pk^i = pk_m$. If $i = i^*$, output (m, σ'_m).

A stateless solution

Theorem (12.22)

If collision-resistant hash functions exist, then there exists a (stateless) secure signature scheme.