Stateful signatures

Definition 2.18 A stateful signature scheme Π is a triple of probabilistic polynomial time algorithms (ppts) $(\text{Gen}, \text{Sign}, \text{Vrfy})$, where

1. $\text{Gen}(1^n)$ outputs a key pair (pk, sk) with $|pk|, |sk| \geq n$ and a state s_0.
2. Sign on input a secret key sk, a state s_{i-1}, and message $m \in \{0,1\}^*$, outputs a signature σ and a state s_i.
3. Vrfy takes as input a public key pk, a message $m \in \{0,1\}^*$, and a signature σ. It ouputs $b \in \{0,1\}$.

For every key pair (pk, sk), state s_0, and message m:

$$\text{Vrfy}_{pk}\left(m, \text{Sign}_{sk, s_{i-1}}(m)\right) = 1.$$

Stateful signatures - remarks

1. If $(\mathrm{Gen}, \mathrm{Sign}, \mathrm{Vrfy})$ is such that for every (pk,sk) output byGen(1^n), algorithm Sign_{sk} is only defined for $m \in \{0,1\}^{l(n)}$, then we say that (Gen, Sign, Vrfy) is a stateful signature scheme for messages of length l(n).

2. The verfication algorithm does not need the state to verify signatures.

From one-time signatures to stateful signatures

$\Pi = (\text{Gen}, \text{Sign}, \text{Vrfy})$ (one-time) signature scheme.

$l = l(n) :=$ number of signatures to be computed (known in advance)

$\Pi' = (\text{Gen}', \text{Sign}', \text{Vrfy}')$

Gen' runs Gen to obtain l pairs (pk_i, sk_i), state s set to 1.
pk is the sequence of public keys pk_i, sk is the sequence of secret keys sk_i.

Sign' on input sk,s and message m, sets $\sigma \leftarrow \text{Sign}_{sk_s}(m)$, $s := s + 1$.

Vrfy' on input (m,σ) outputs 1, iff there is an $i \in \{1,\ldots,l\}$ such that $\text{Vrfy}_{pk_i}(m,\sigma) = 1$.

From one-time signatures to stateful signatures

$\Pi = (\mathrm{Gen}, \mathrm{Sign}, \mathrm{Vrfy})$ (1-time) signature scheme for messages of length 2n and such that $\mathrm{Gen}(1^n)$ outputs public keys of length n.

$\Pi' = (\mathrm{Gen}', \mathrm{Sign}', \mathrm{Vrfy}')$, stateful for messages of length n.

Gen' runs Gen to obtain a pair $(pk, sk) = (pk_1, sk_1)$, state s is the empty string ϵ.

Sign' on input sk, s and message m_i, runs Gen to obtain (pk_{i+1}, sk_{i+1}), $\sigma_i \leftarrow \mathrm{Sign}_{sk_i}(m_i \| pk_{i+1})$ and add $(m_i, pk_{i+1}, sk_{i+1}, \sigma_i)$ to the state.
The signature for m_i is $\{(m_j, pk_{j+1}, \sigma_j)\}_{j=1}^{i-1}$ and (pk_{i+1}, σ_i).

Vrfy' on input $(pk_{i+1}, \sigma_i, \{(m_j, pk_{j+1}, \sigma_j)\}_{j=1}^{i-1})$ outputs 1, iff $\mathrm{Vrfy}_{pk_j}(m_j \| pk_{j+1}, \sigma_j) = 1$ for $j = 1, \ldots, i$.

Tree-based signatures – preliminaries and Gen

$\Pi = (\mathrm{Gen}, \mathrm{Sign}, \mathrm{Vrfy})$ (one-time) signature scheme for messages of length 2n and such that $\mathrm{Gen}(1^n)$ outputs public keys of length n.

For $m \in \{0,1\}^*$ denote by $m|_i$ the prefix of m of length i.

$\Pi^\star = (\mathrm{Gen}^*, \mathrm{Sign}^*, \mathrm{Vrfy}^*)$ is a stateful signature scheme for messages of length n.

Gen^* on input 1^n: compute $(pk_\epsilon, sk_\epsilon)$, output public key pk_ϵ and state $s = sk_\epsilon$.

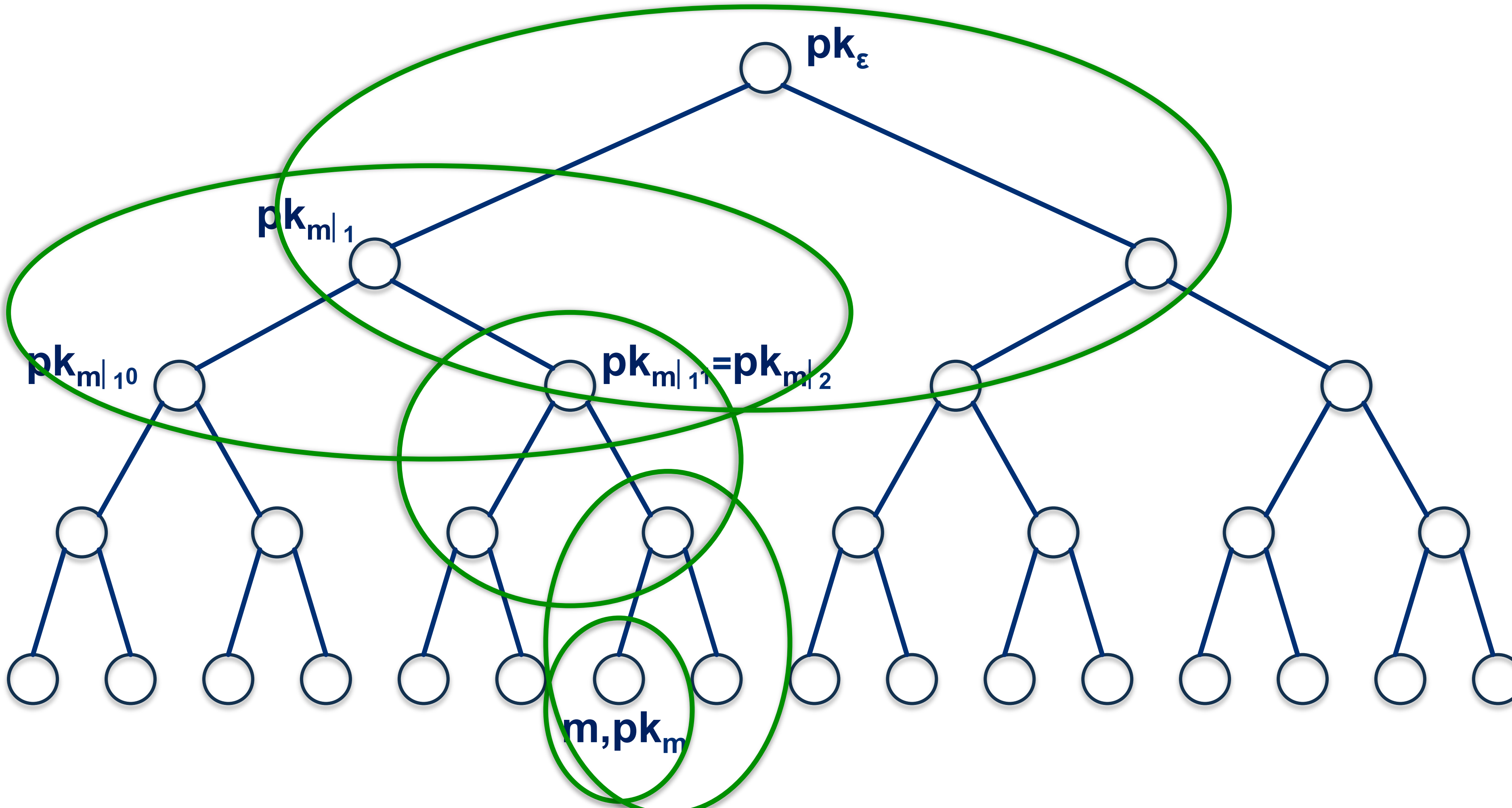
Tree-based signatures - Sign

Sign^* on input $m \in \{0,1\}^n$ and state:

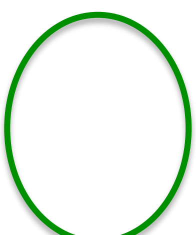
1. for $i = 0$ to $n-1$:
   - if $pk_{m|_i 0}, pk_{m|_i 1}$, and $\sigma_{m|_i}$ are not in the state, compute $(pk_{m|_i 0}, sk_{m|_i 0}) \leftarrow \text{Gen}(1^n)$, $(pk_{m|_i 1}, sk_{m|_i 1}) \leftarrow \text{Gen}(1^n)$, and $\sigma_{m|_i} \leftarrow \text{Sign}_{sk_{m|_i}}(pk_{m|_i 0} \,\|\, pk_{m|_i 1})$. Add these values to state.
2. if σ_m is not in the state, compute $\sigma_m \leftarrow \text{Sign}_{sk_m}(m)$.
3. output the signature $(\{(\sigma_{m|_i}, pk_{m|_i 0}, pk_{m|_i 1})\}_{i=0}^{n-1}, \sigma_m)$.

Remark: Sign^* uses each key on at most one message.

Tree-based signatures



leaves = messages



key in parent node to compute signature of concatenation of public keys in children.

Tree-based signatures - Vrfy

Vrfy^* on input a public key pk_ϵ, message m, and signature $(\{(\sigma_{m|_i}, pk_{m|_i0}, pk_{m|_i1})\}_{i=0}^{n-1}, \sigma_m)$, output 1, iff

1. $\mathrm{Vrfy}_{pk_{m|_i}}(pk_{m|_i0} \| pk_{m|_i1}, \sigma_{m|_i}) = 1$ for $i = 0, \ldots, n-1$
2. $\mathrm{Vrfy}_{pk_m}(m, \sigma_m) = 1$.

Theorem 2.19 If Π is a one-time signature, then Π^* is a secure stateful signature scheme for messages of length n.

From A^* to A (1)

A on input public key pk:

- choose random index $i^* \leftarrow \{1,\ldots,l^*\}$. Construct list $pk^1,\ldots,pk^{l^*}$ of keys as follows:
  - set $pk^{i^*} := pk$
  - for $i \neq i^*$, compute $(pk^i, sk^i) \leftarrow Gen(1^n)$.
- run A^* on input $pk_\epsilon = pk^1$. When A^* requests a signature on m, do:
  1. for $i = 0$ to $n-1$:
     - if the values $pk_{m|_i 0}, pk_{m|_i 1}$, and $\sigma_{m|_i}$ have not been defined, set $pk_{m|_i 0}, pk_{m|_i 1}$ to the next unused keys pk^j, pk^{j+1}, and compute signature $\sigma_{m|_i}$ on $pk_{m|_i 0} \| pk_{m|_i 1}$ with respect to key $pk_{m|_i}$.
  2. if σ_m is not yet defined, compute a signature σ_m on m with key pk_m.
  3. give $(\{(\sigma_{m|_i}, pk_{m|_i 0}, pk_{m|_i 1})\}_{i=0}^{n-1}, \sigma_m)$ to A^*.

From A^* to A (2)

- if A^* outputs a valid signature $(\{(\sigma'_{m|_i}, pk'_{m|_i 0}, pk'_{m|_i 1})\}_{i=0}^{n-1}, \sigma'_m)$ on message m, then

  case 1: if there is a $j \le n-1$ such that $pk'_{m|_j 0} \ne pk_{m|_j 0}$ or $pk'_{m|_j 1} \ne pk_{m|_j 1}$, take minimal j and let i be such that $pk^i = pk'_{m|_j} = pk_{m|_j}$. If $i = i^*$, 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).

Removing statefulness

- in state store key pairs and signatures for internal nodes of tree
- instead of storing these values want to recompute them when needed
- however, Gen and Sign are probabilistic, and recomputation may lead to different values
- need randomness use in computation of key pairs and signatures
- replace randomness by pseudorandomness
- computed using PRFs and based on index of internal node

Existence of secure signatures

Theorem 2.20 (restated) Secure digital signature schemes exist if and only if one-way functions exist.

Proof sektch Let $\Pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ be an existentially unforgeable signature scheme. Then the function f that on input r outputs the public key generated by Gen if started with random bits r is a one-way function.

RSA signatures - prerequisits

$\mathbb{Z}_N \quad := \quad$ ring of integers modulo N

$$\mathbb{Z}_N^* \quad := \quad \{a \in \mathbb{Z}_N : \gcd(a,N) = 1\}$$

$$\phi(N) \quad := \quad |\mathbb{Z}_N^*|$$

$$\gcd(a,m) = 1 \quad \Rightarrow \quad \exists\, u,v \in \mathbb{Z}\ u \cdot a + v \cdot m = 1 \ \text{(EEA)}$$

$$\Rightarrow \quad u \cdot a = 1 \bmod m$$

$$\Rightarrow \quad u = a^{-1} \bmod m$$

$$N = \prod_{i=1}^{K} p_i^{e_i} \quad \Rightarrow \quad \phi(N) = \prod_{i=1}^{K} \left(p_i^{e_i} - p_i^{e_i - 1}\right) = N \cdot \prod_{i=1}^{K} \left(1 - 1/p_i\right).$$

RSA signatures

$\mathrm{Gen}(1^n)$: choose 2 random primes $p, q \in [2^{n-1}, 2^n - 1]$,

$N := p \cdot q, e \leftarrow \mathbb{Z}^*_{\phi(N)}, d := e^{-1} \bmod \phi(N)$,

$pk := (N, e), sk := (N, d)$.

$\mathrm{Sign}_{sk}(m)$ $m \in \{0,1\}^{2n-2}$ interpreted as element in $\mathbb{Z}_N$,

$\sigma := m^d \bmod N$.

$\mathrm{Vrfy}_{pk}(m, \sigma)$ output 1, if and only if $\sigma^e = m \bmod N$.

RSA signatures - forgeries

existential forgeries

- $\mathrm{Sign}_{sk}(0) = 0$
- $\mathrm{Sign}_{sk}(1) = 1$
- $\mathrm{Sign}_{sk}(-1) = -1$

selective forgery of $\mathrm{Sign}_{sk}(m)$

- query signature oracle with input $\hat{m} := 2^e m \bmod N$ and obtain $\hat{\sigma}$.
- compute $\sigma = 2^{-1}\hat{\sigma} \bmod N$.

Random oracle model (ROM)

Goal Construct $H:\{0,1\}^* \rightarrow R, |R| < \infty$, "random" function.

$H(x)?$

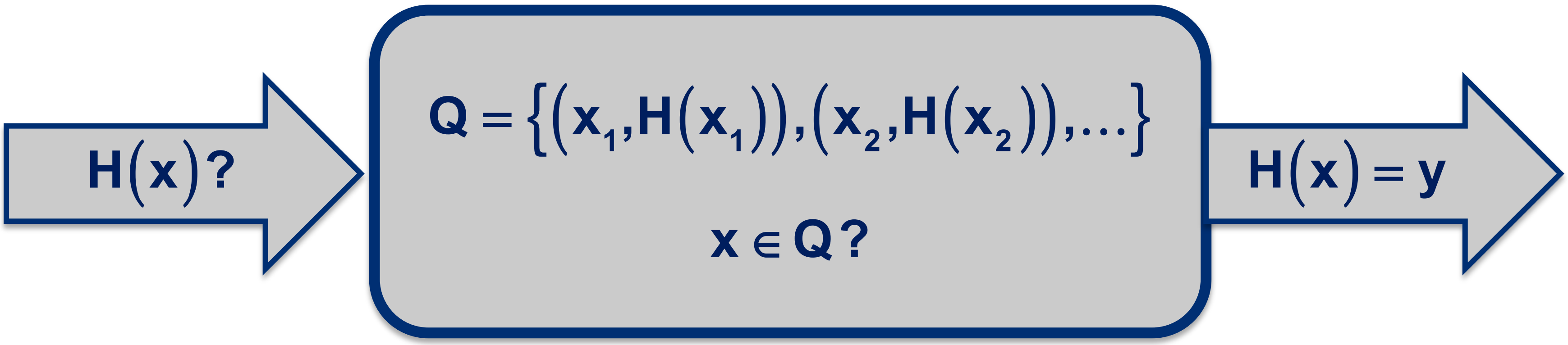
$$Q = \{(x_1, H(x_1)), (x_2, H(x_2)), \ldots\}$$

$$x \in Q?$$

$H(x) = y$

- If $x = x_i$ for $x_i \in Q$, return $H(x_i)$.
- If $x \neq x_i$ for all $x_i \in Q$,
  a) $y \leftarrow R$
  b) return $H(x) = y$
  c) add pair $(x, H(x))$ to Q

Random oracle model (ROM)



- Random oracle model idealization of
  - one-way functions
  - random functions
  - collision-resistant hash functions.
- In practice they can not be implemented in this form.
- Often collision-resistant hash functions used instead.

RSA-Full-Domain-Hash (RSA-FDH)

By Gen denote an algorithm that on input 1^n computes 2 random primes $p,q \in [2^{n-1}, 2^n - 1], p \neq q$, sets $N = p \cdot q$, chooses $e \leftarrow Z^*_{\phi(N)}$, sets $d := e^{-1} \bmod \phi(N)$, and outputs $pk := (N,e), sk := (N,d)$.

Construction 2.21 (RSA-FDH)

- Run $\mathrm{Gen}(1^n)$ to obtain $pk := (N,e)$ and $sk := (N,d)$.

  Let $H : \{0,1\}^* \to \mathbb{Z}_N$ be modeled as a random oracle.
- Sign on input $m \in \{0,1\}^*$ and (N,d) outputs

  $\sigma := (H(m))^d \bmod N$.
- Vrfy on input $m, \sigma, (N,e)$ outputs 1 $\Leftrightarrow \sigma^e = H(m) \bmod N$.

RSA assumption

RSA inverting game $\text{RSA-inv}_{A,\text{Gen}}(n)$

1. Run Gen to obtain (N,e).
2. $y \leftarrow \mathbb{Z}_N$.
3. A is given (N,e) and y. A outputs $x \in \mathbb{Z}_N$.
4. Output of experiment is 1, if and only if $x^e = y \bmod N$.

Write $\text{RSA-inv}_{A,\text{Gen}}(n) = 1$, if output is 1.

Definition 2.22 The RSA problem is hard relative to the generation algorithm Gen if for every ppt adversary A there is a negligible function $\mu : \mathbb{N} \to \mathbb{R}^+$ such that

$$\Pr\left[\text{RSA-inv}_{A,\text{Gen}}(n) = 1\right] \le \mu(n).$$

RSA assumption

Construction 2.21 (RSA-FDH)

- Run $\mathrm{Gen}(1^n)$ to obtain $pk := (N, e)$ and $sk := (N, d)$.

  Let $H : \{0,1\}^* \to \mathbb{Z}_N$ be modeled as a random oracle.
- Sign on input $m \in \{0,1\}^*$ and (N, d) outputs

  $\sigma := (H(m))^d \bmod N.$
- on input $m, \sigma, (N, e)$ output 1 $\Leftrightarrow \sigma^e = H(m) \bmod N.$

Theorem 2.23 If the RSA problem is hard relative to the generation algorithm Gen, then RSA-FDH (Construction 2.21) is existentially unforgeable under an adaptive chosen-message attack.

From forger to inverter

Signature forging game $\text{Sig-forge}_{A,\Pi}(n)$

1. $(pk, sk) \leftarrow \text{Gen}(1^n)$.
2. A is given $1^n, pk$ and oracle access to $\text{Sign}_{sk}(\cdot)$. It outputs pair (m, σ). $\mathcal{Q} :=$ set of queries made by A to $\text{Sign}_{sk}(\cdot)$.
3. Output of experiment is 1, if and only if (1) $\text{Vrfy}_{pk}(m, \sigma) = 1$, and (2) $m \notin \mathcal{Q}$.

Assume:

1. A never queries for the same hash value twice.
2. Before querying $\text{Sign}_{sk}(\cdot)$ on message m, A queries $H(\cdot)$ on m.

Let $q = q(n)$ denote number of hash queries made A, q bounded by polynomial in n.

From forger to inverter

I on input (N, e, y^*)

1. Choose $j \leftarrow \{1, \ldots, q\}$.
2. Simulate A with public key (N, e). Table T stores triples (m_i, σ_i, y_i) with meaning that I has set $H(m_i) = y_i$ and $\sigma_i^e = y_i \mod N$.
3. When A makes i-th random oracle query $H(m_i)$, do
   - if $i = j$, return y^*
   - otherwise, $\sigma_i \leftarrow \mathbb{Z}_N, y_i := [\sigma_i^e \mod N]$, return y_i, add (m_i, σ_i, y_i) to T.

   When A makes signature query $m = m_i$, do
   - if $i \neq j$, then T contains triple (m_i, σ_i, y_i), return σ_i.
   - if $i = j$, then abort experiment.
4. Let (m, σ) be A's output. If $m = m_j$ and $\sigma^e = y^* \mod N$, then output σ.

Certificates and trusted authorities

How can we guarantee that pk_A belongs to A?

- **certificates from trusted authorities (TA)**
- **certificates are signatures**
- **leads to hierarchie of certificates/signatures**
- **must stop at (really) trusted authority**