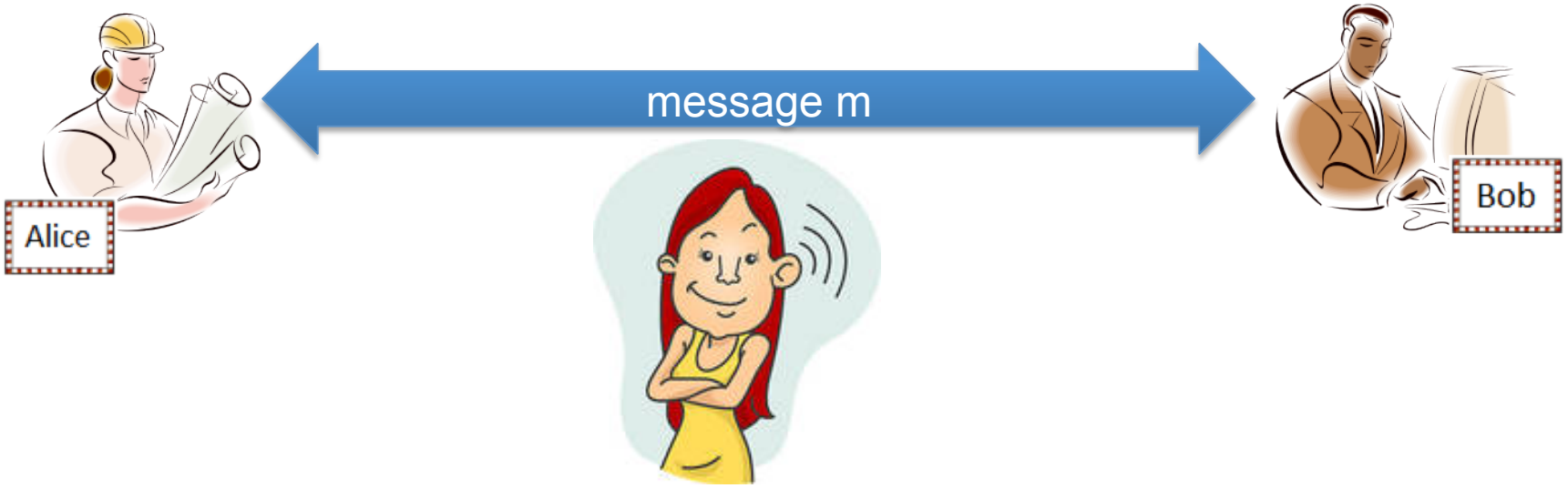
II. Digital signatures



1. Did Bob send message m, or was it Eve?

2. Did Eve modify the message m, that was sent by Bob?

Digital signatures

Digital signatures

- are equivalents of handwritten signatures
- guarantee authenticity and integrity of documents
- also guarantee non-repudiation

Digital signatures

Definition 2.1 A digital signature scheme Π is a triple of probabilistic polynomial time algorithms (ppts) $(\mathrm{Gen}, \mathrm{Sign}, \mathrm{Vrfy})$, where

1. $\mathrm{Gen}(1^n)$ outputs a key pair $(\mathrm{pk}, \mathrm{sk})$ with $|\mathrm{pk}|, |\mathrm{sk}| \geq n$.
2. Sign takes as input a secret key sk and a message $m \in \{0,1\}^*$ and outputs a signature σ, $\sigma \leftarrow \mathrm{Sign}_{\mathrm{sk}}(m)$.
3. Vrfy takes as input a public key pk, a message $m \in \{0,1\}^*$, and a signature σ. It ouputs $b \in \{0,1\}$, $1 \triangleq$ valid, $0 \triangleq$ invalid. Vrfy deterministic, $b := \mathrm{Vrfy}_{\mathrm{pk}}(m, \sigma)$.

For every key pair $(\mathrm{pk}, \mathrm{sk})$ and message m:
$\mathrm{Vrfy}_{\mathrm{pk}}(m, \mathrm{Sign}_{\mathrm{sk}}(m)) = 1$.

Digital signatures

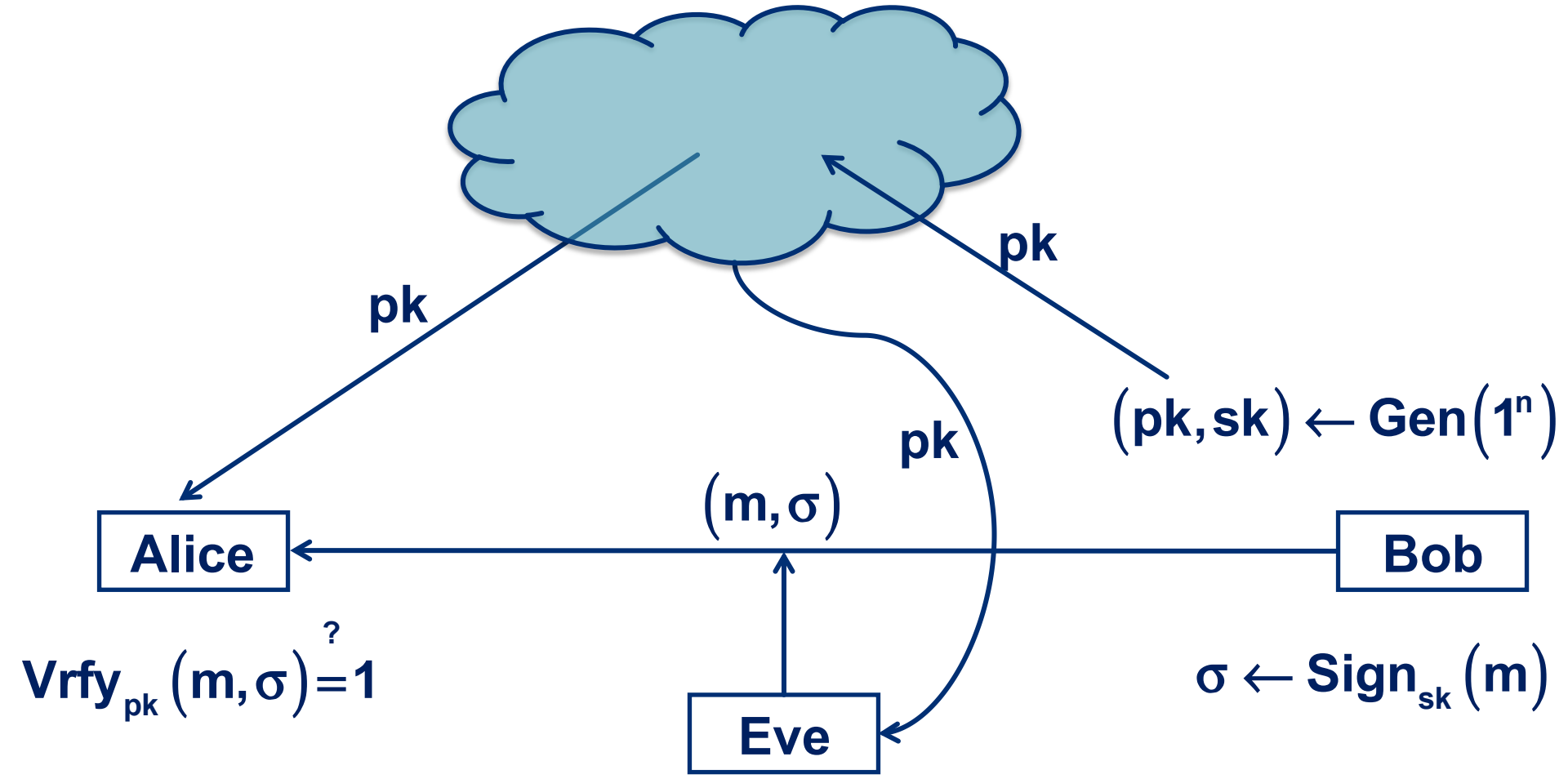
Definition 2.1 A digital signatur scheme Π is a triple of probabilistic polynomial time algorithms (ppts) $(Gen, Sign, Vrfy)$, where

1. $Gen(1^n)$ outputs a key pair (pk, sk) with $|pk| = |sk| = n$.
2. Sign takes as input a secret key sk and a message $m \in \{0,1\}^*$ and outputs a signature σ, $\sigma \leftarrow Sign_{sk}(m)$.
3. Vrfy takes as input a public key pk, a message $m \in \{0,1\}^*$, and a signature σ. It ouputs $b \in \{0,1\}$, $1 \triangleq$ valid, $0 \triangleq$ invalid. Vrfy deterministic, $b := Vrfy_{pk}(m, \sigma)$.

For every key pair (pk, sk) and message m: $Vrfy_{pk}(m, Sign_{sk}(m)) = 1$.

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

Digital signatures



Security of digital signatures

- An adversary should not be able to compute the signature for an arbitrary message even though he knows the public key of correct signee.
- This should remain true, even if the adversary can get signatures for messages of his choice.
- But the adversary must compute the signature for a new message to be successful.
- Restrict adversaries to efficient ones.
- But adversaries should succeed only with tiny probability.

The forging game

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

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

Definition 2.2 Π is called existentially unforgeable under an adaptive chosen-message attack, or secure, if for every ppt adversary A there is a negligible function $\mu : \mathbb{N} \to \mathbb{R}^+$ such that

$$Pr\left[\text{Sig-forge}_{A,\Pi}(n) = 1\right] \leq \mu(n).$$

Oracle access

Algorithm D has oracle access to function $f : U \to R$, if

1. D can write elements $x \in U$ into special memory cells,
2. in one step receives function value $f(x)$.

Notation Often write $D^{f(\cdot)}$ to denote that algorithm D has oracle access to $f(\cdot)$.

Negligible functions

Definition 2.3 A function $\mu:\mathbb{N} \to \mathbb{R}^+$ is called negligible, if

$$\forall c \in \mathbb{N}\ \exists n_0 \in \mathbb{N}\ \forall n \geq n_0\ \mu(n) \leq 1/n^c .$$

RSA signatures - prerequisits

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

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

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

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

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

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

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

RSA signatures

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

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

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

$\mathrm{Sign}_{sk}(m)$: $m \in \{0,1\}^{2n-2}$ 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 - correctness

For special case $m \in \mathbb{Z}_N^*$ based on

Lemma 2.4 Let $N \in \mathbb{N}$ and $m \in \mathbb{Z}_N^*$, then $m^{\phi(N)} = 1 \bmod N$.

RSA signatures - efficiency

Prime generation

1. choose $p \leftarrow \left[2^{n-1}, 2^n - 1\right]$.
2. Test whether p is prime, if so output p, otherwise go back to 1.

Efficiency based on

1. In $\left[2^{n-1}, 2^n - 1\right]$ many primes exist (prime number theorem).
2. Efficient primality test exist (Miller-Rabin, AKS)

RSA signatures - efficiency

Exponent generation

1. choose $e \leftarrow \mathbb{Z}_{\phi(N)}$.
2. Test whether $\gcd(e, \phi(N)) = 1$, if so compute d with $e \cdot d = 1 \bmod \phi(N)$, otherwise go back to 1.

Efficiency based on

1. In $\mathbb{Z}_M$ many elements relatively prime to M exist.
2. Can check efficiently whether $a,b \in \mathbb{Z}$ are relatively prime using Eucledean algorithm.

RSA signatures - efficiency

Efficiency of Sign and Vrfy based on

1. **arithmetic in $\mathbb{Z}_N$ can be done efficiently.**
2. **Exponentiation requires few arithmetic operations using Square-and-Multiply.**

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 $\sigma = 2^{-1}\hat{\sigma} \bmod N$.

General problem of public-key cryptography

Secret key sk must not be efficiently computable from public key pk!

General problem for RSA

Given (N,e) element $d \in \mathbb{Z}^*_{\phi(N)}$ with $e \cdot d = 1 \bmod \phi(N)$ must not be efficiently computable.

Theorem 2.5 **Given e,d,N, $N = p \cdot q$ for primes p,q, and with $e \cdot d = 1 \bmod \varphi(N)$, then the primes p,q can be computed in time polynomial in $\log(N)$.**

Status of factoring problem

Two factoring algorithms

- Number field sieve

  running time $\exp\left(\log(N)^{1/3} \cdot \log\log(N)^{2/3}\right)$

- Elliptic curve method

  running time $\exp\left(\log(p)^{1/2} \cdot \log\log(p)^{1/2}\right)$,

  where p smallest prime factor

Existence of secure signatures

Theorem 2.6 **Secure digital signature schemes exist if and only if one-way functions exist.**

Inverting game

$f : \{0,1\}^* \to \{0,1\}^*$, A a probabilistic polynomial time algorithm

Inverting game $\text{Invert}_{A,f}(n)$

1. $x \leftarrow \{0,1\}^n, y := f(x)$.
2. A given input 1^n and y, outputs x'.
3. Output of game is 1, if $f(x') = y$, otherwise output is 0.

Write $\text{Invert}_{A,f}(n) = 1$, if output is 1. Say A has succeded or A has won.

Definition of one-way function

Definition 2.7 $f : \{0,1\}^* \to \{0,1\}^*$ called one-way, if

1. there is a ppt M_f with $M_f(x) = f(x)$ for all $x \in \{0,1\}^*$
2. for every probabilistic polynomial time algorithm A there is a negligible function $\mu : \mathbb{N} \to \mathbb{R}^+$ such that $\Pr\left[\text{Invert}_{A,f}(n) = 1\right] \le \mu(n)$.

Notation $\Pr_{x \leftarrow \{0,1\}^n}\left[A(f(x)) \in f^{-1}(f(x))\right] \le \mu(n)$

Candidate

1. $f_{mult} : \{0,1\}^* \to \{0,1\}^*$

   $x \mapsto (x_1 \cdot x_2, |x_1|, |x_2|)$,

   where $|x_1| = \lfloor |x|/2 \rfloor, |x_2| = \lceil |x|/2 \rceil$, and identify bit strings and integers via binary representations.

Idea Multiplication easy, factoring hard

Definition of one-way permutation

$f : \{0,1\}^* \to \{0,1\}^*$ length preserving, if for all x $|f(x)| = |x|$.

$f_n := f_{|\{0,1\}^n}$, restriction of f to $\{0,1\}^n$.

Definition 2.8 A one-way function $f : \{0,1\}^* \to \{0,1\}^*$ is called one-way permutation, if

1. f is length-preserving,
2. for every $n \in \mathbb{N}$ the function f_n is a bijection.

One-time signatures

One-time signature forging game $\text{Sig-forge}^{\text{one}}_{A,\Pi}(n)$

1. $(pk, sk) \leftarrow \text{Gen}(1^n)$.
2. A is given $1^n, pk$ and may ask single query m' to $\text{Sign}_{sk}(\cdot)$. It outputs pair (m, σ), where $m \neq m'$.
3. Output of experiment is 1, if and only if (1) $\text{Vrfy}_{pk}(m, \sigma) = 1$.

Definition 2.8 Π is called existentially unforgeable under a single message attack or 1-time signature, if for every ppt adversary A there is a negligible function $\mu : \mathbb{N} \to \mathbb{R}^+$ such that

$$\Pr\left[\text{Sig-forge}^{\text{one}}_{A,\Pi}(n) = 1\right] \leq \mu(n).$$

Lamport's one-time signature

Construction 2.9 $f:\{0,1\}^* \to \{0,1\}^*$, signature scheme

$\Pi_f = (\mathrm{Gen}, \mathrm{Sign}, \mathrm{Vrfy})$ for messages of length $l(n)$ defined as:

$\mathrm{Gen}(1^n)$: $x_{i,b} \leftarrow \{0,1\}^n, y_{i,b} = f(x_{i,b}), i = 1,\ldots,n, b \in \{0,1\}.$

$$pk := \begin{pmatrix} y_{1,0} & y_{2,0} & \cdots & y_{n,0} \\ y_{1,1} & y_{2,1} & \cdots & y_{n,1} \end{pmatrix},$$

$$sk := \begin{pmatrix} x_{1,0} & x_{2,0} & \cdots & x_{n,0} \\ x_{1,1} & x_{2,1} & \cdots & x_{n,1} \end{pmatrix},$$

$\mathrm{Sign}_{sk}(m)$: output $\sigma := (x_{1,m_1},\ldots,x_{n,m_n}), m = m_1 \cdots m_n.$

$\mathrm{Vrfy}_{pk}(m,\sigma)$: output $= 1 \Leftrightarrow y_{i,m_i} = f(x_{i,m_i})$ for $i = 1,\ldots,n.$

Lamport's one-time signature

Theorem 2.10 If f is a one-way function, then Π_f from Construction 2.9 is a 1-time signature.

m' := message whose signature is requested by A

(m, σ) := A's final output

Adverary A outputs forgery at (i, b), if

- $Vrfy_{pk}(m, \sigma) = 1$
- $m_i = b$ and $m_i \neq m'_i$

From forger to inverter

I on input y^*

1. Choose $i^* \leftarrow \{1,\ldots,n\}, b^* \leftarrow \{0,1\}$.
2. For all $i \in \{1,\ldots,n\}, b \in \{0,1\}$ with $(i,b) \neq (i^*,b^*)$ do

   choose $x_{i,b} \leftarrow \{0,1\}^n$, set $y_{i,b} := f(x_{i,b}), y_{i^*,b^*} := y^*$
3. Simulate A on input pk: $= \begin{pmatrix} y_{1,0} & y_{2,0} & \cdots & y_{n,0} \\ y_{1,1} & y_{2,1} & \cdots & y_{n,1} \end{pmatrix}$
4. When A requests a signature on message m':
   - if $m'_{i^*} = b^*$, stop
   - otherwise return the correct signature $\sigma = (x_{1,m'_1},\ldots,x_{n,m'_n})$
5. When A outputs (m,σ) with $\sigma = (x_1,\ldots,x_n)$
   - if A outputs a forgery at (i^*,b^*), output x_{i^*}.

What have we achieved, what's missing?

- just a one-time signature, where
- keys are longer than messages
- need to decouple key and message length
- leads to simple, but inefficient and cumbersome signature scheme for messages of fixed, but arbitrary length
- these show that secure signatures schemes can be constructed from 1-way functions
- constructions on other simpler ingredients also known
- key ingredient will be collision-resistant hash functions

Hash functions

Definition 2.11 A hash function is a pair $\Pi = (\mathrm{Gen}, H)$ of ppts, where

1. $\mathrm{Gen}(1^n)$ takes as input 1^n and outputs a key s.
2. H is deterministic, it takes as input 1^n, a key s, and $x \in \{0,1\}^*$. There is a polynomial $l: \mathbb{N} \to \mathbb{N}$ such that if s was generated with input 1^n, then $H(s,x) \in \{0,1\}^{l(n)}$.

Write $H^s(x)$ for $H(s,x)$.

If H^s is defined only for inputs $x \in \{0,1\}^{l'(n)}$ for some polynomial l', then Π is a fixed-length hash function for inputs of length $l'(n)$.

The collision-finding game

Collision-finding game $\text{Hash-coll}_{A,\Pi}(n)$

1. $s \leftarrow \text{Gen}(1^n)$.
2. A is given 1^n and s. It outputs x,x′ (with length $l'(n)$ if Π is fixed-length).
3. Output of experiment is 1, if and only if $x \neq x'$ and $H^s(x) = H^s(x')$. Say A has found collision.

Definition 2.12 $\Pi = (\text{Gen}, H)$ called collision-resistant, if for every probabilistic polynomial time adversary A there is a negligible function $\mu : \mathbb{N} \to \mathbb{R}^+$ such that

$$\Pr\left[\text{Hash} - \text{coll}_{A,\Pi}(n) = 1\right] \leq \mu(n).$$

Weaker notions

1. coll.-res. …

2. 2^{nd}-preimage res. given s,x, find $x' \neq x$ with $H^s(x) = H^s(x')$

3. pre-image res. given s,$y = H^s(x)$, find x' with $H^s(x') = y$

Fact Under appropriate assumptions

coll.res. $\Rightarrow$ 2^{nd}-preimage res. $\Rightarrow$ pre-image res.

A generic attack & birthday paradoxon

$H^s : \{0,1\}^* \to \{0,1\}^n$ for $s \in \{0,1\}^n$

On input $s \in \{0,1\}^n$

1. Choose $q \in \mathbb{N}$
2. $x_1,\ldots,x_q \leftarrow \{0,1\}^n, y_i := H^s(x_i)$
3. if there exist $i,j, i \neq j,$ such that $y_i = y_j,$ output (x_i, x_j), otherwise output $\perp$.

Fact Assume that for all $x_1,\ldots,x_q \in \{0,1\}^*$ pairwise distinct and all $y_1,\ldots,y_q \in \{0,1\}^n$ we have $\Pr[\forall i : H^s(x_i) = y_i] = 2^{-qn}$, then

$$\frac{q(q-1)}{2^{n+2}} \leq \Pr[\exists i,j \in \{1,\ldots,q\}, i \neq j : y_i = y_j] \leq \frac{q(q-1)}{2^{n+1}}.$$

Hash-and-Sign

$\Upsilon' = (\mathrm{Gen}', \mathrm{Mac}', \mathrm{Vrfy}')$ sig. scheme with message length $l(n)$,

$\Pi = (\mathrm{Gen}_H, H)$ hash function with hash length $l(n)$.

Construction 2.13 Sig. scheme $\Upsilon = (\mathrm{Gen}, \mathrm{Sign}, \mathrm{Vrfy})$ defined as:

$\mathrm{Gen}(1^n)$: $(pk', sk') \leftarrow \mathrm{Gen}'(1^n), s \leftarrow \mathrm{Gen}_H(1^n)$,
$pk = (pk', s), sk = sk'$

$\mathrm{Sign}_{sk}(m)$: $\sigma := \mathrm{Sign}'_{sk}(H^s(m))$.

$\mathrm{Vrfy}_{pk}(m, \sigma)$ output $= 1 \Leftrightarrow 1 = \mathrm{Vrfy}'_{pk'}(H^s(m), \sigma)$.

Theorem 2.14 If Υ' is secure and Π is collision-resistant, then Υ is secure.

Hash-and-Sign

$A :=$ adversary against Υ

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

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) $m \notin \mathcal{Q}$.

$\text{Coll} := \exists m' \in \mathcal{Q} : H^s(m') = H^s(m)$

$$\Pr\left[\text{Sign-forge}_{A,\Upsilon}(n) = 1\right] \leq \Pr\left[\text{Sign-forge}_{A,\Upsilon}(n) = 1 \mid \neg\text{Coll}\right] + \Pr\left[\text{Coll}\right]$$

Collision-finder A_1

A_1 on input 1^n and $s \leftarrow Gen_H$

1. Run Gen′ to obtain key (pk′, sk′).
2. Simulate A. Whenever A queries its Sign-oracle $Sign_{sk}(\cdot)$ on a message m′, do:
   a) Compute $h := H^s(m')$.
   b) Compute $\sigma' := Sign_{sk'}(h)$ and return σ' to A.
3. Let Q be the set of queries made by A and let (m, σ) be A's answer. If there is an $m' \in Q$ with $H^s(m') = H^s(m)$, return the pair (m, m'), otherwise return "failure".

Sign-forger A_2

A_2 on input 1^n and and oracle access to $\mathrm{Sign}'_{sk'}(\cdot)$

1. Run Gen_H to obtain key s.
2. Simulate A. Whenever A queries its Sign-oracle $\mathrm{Sign}_{sk}(\cdot)$ on a message m′, do:
   a) Compute $h := H^s(m')$.
   b) Query $\mathrm{Sign}'_{sk'}(\cdot)$ on input h to obtain $\sigma' := \mathrm{Sign}'_{sk'}(h)$, return σ' to A.
3. Let Q be the set of queries made by A. If A returns a pair (m,t) such that $H^s(m) \neq H^s(m')$ for all $m' \in Q$, then return pair $(H^s(m),t)$, otherwise return "failure".

Stateful signatures

Definition 2.15 A stateful signature scheme Π is a triple of probabilistic polynomial time algorithms (ppts) $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$, where

1. $\mathsf{Gen}(1^n)$ outputs a key pair $(\mathsf{pk}, \mathsf{sk})$ with $|\mathsf{pk}|, |\mathsf{sk}| \geq n$ 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 $(\mathsf{pk}, \mathsf{sk})$, state s_0, and message m:

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

Stateful signatures - remarks

1. If $(Gen, Sign, Vrfy)$ is such that for every (pk,sk) output by$Gen(1^n)$, 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 1-time signatures to stateful signatures

$\Pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ (1-time) signature scheme.

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

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

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 s_i.

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

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

From 1-time signatures to stateful signatures

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

$\Pi' = (\mathsf{Gen}', \mathsf{Sign}', \mathsf{Vrfy}')$, 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}), $\sigma_i \leftarrow \mathsf{Sign}_{sk_i}(m_i \| pk_{i+1})$ 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 $\mathsf{Vrfy}_{pk_j}(m_j \| pk_{j+1}, \sigma_j) = 1$ for $j = 1, \ldots, i$.

Tree-based signatures

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

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

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

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

Tree-based signatures - Sign

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

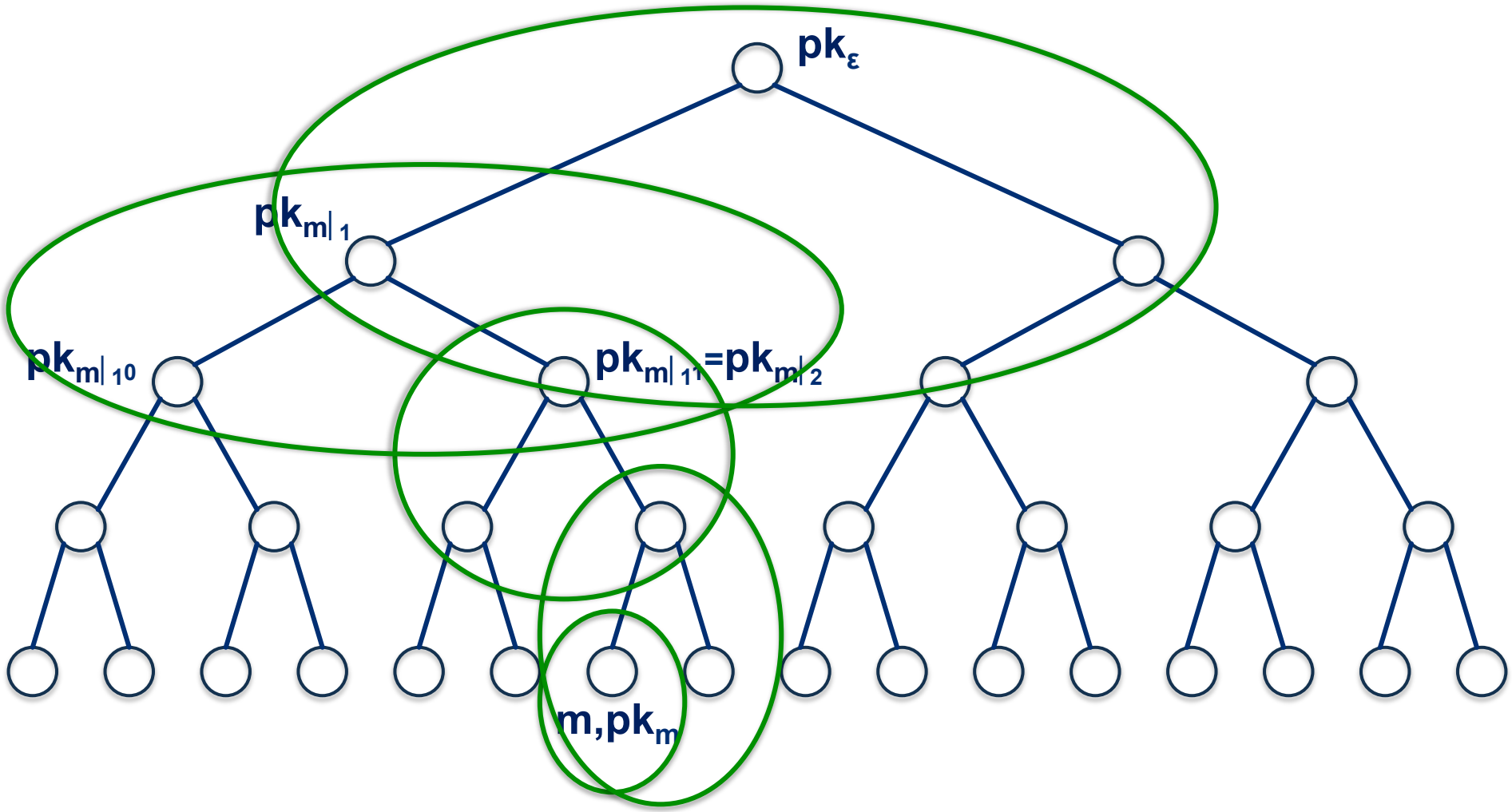
1. for $i = 0$ to $n-1$:
   - if $pk_{m|_i 0}, pk_{m|_i 1}$, and $\sigma_{m|_i}$ are not in the state, compute

     $(pk_{m|_i 0}, sk_{m|_i 0}) \leftarrow \text{Gen}(1^n), (pk_{m|_i 1}, sk_{m|_i 1}) \leftarrow \text{Gen}(1^n)$, 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 $(\{(\sigma_{m|_i}, \text{pk}_{m|_i 0}, \text{pk}_{m|_i 1})\}_{i=0}^{n-1}, \sigma_m)$, output 1, iff

1. $\text{Vrfy}_{\text{pk}_{m|_i}}(\text{pk}_{m|_i 0} \,||\, \text{pk}_{m|_i 1}, \sigma_{m|_i}) = 1$ for $i = 0, \ldots, n-1$
2. $\text{Vrfy}_{\text{pk}_m}(m, \sigma_m) = 1$.

Theorem 2.16 If Π is a one-time signature, then Π^* is a secure 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 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 $j \leq n-1$ such that $pk'_{m|_j 0} \neq pk_{m|_j 0}$ or $pk'_{m|_j 1} \neq pk_{m|_j 1}$, 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).

RSA signatures - prerequisits

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

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

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

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

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

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

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

RSA signatures

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

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

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

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

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

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

RSA signatures - forgeries

existential forgeries

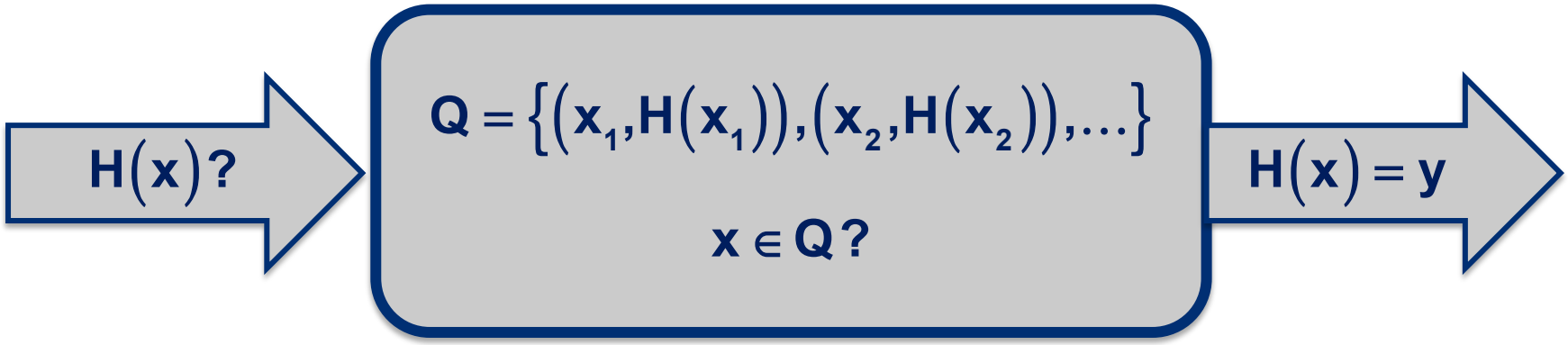
- $\mathrm{Sign}_{sk}(0) = 0$
- $\mathrm{Sign}_{sk}(1) = 1$
- $\mathrm{Sign}_{sk}(-1) = -1$

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

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

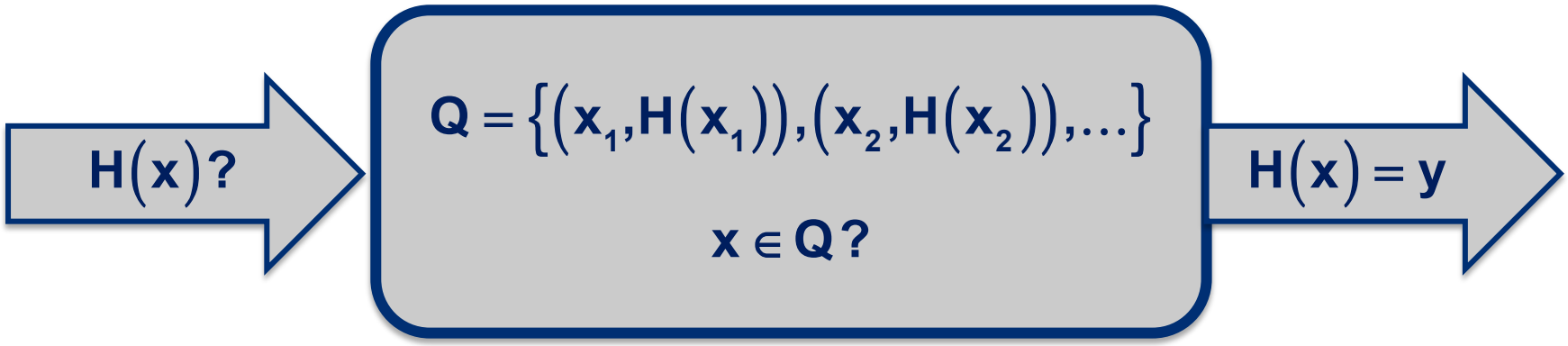
Random oracle model (ROM)

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



- 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 $p,q \in \left[2^{n-1}, 2^n - 1\right], p \neq q$, 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 2.17 (RSA-FDH)

- Run $\text{Gen}\left(1^n\right)$ 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

  $\sigma := \left(H(m)\right)^d \bmod N$.
- 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 2.18 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

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

RSA assumption

Construction 2.19 (RSA-FDH)

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

  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 2.20 If the RSA problem is hard relative to the generation algorithm Gen, then RSA-FDH (Construction 2.19) 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