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 one-time signature, if for every ppt adversary A there is a negligible function $\mu : \mathbb{N} \to \mathbb{R}^+$ such that

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

Lamport's one-time signature

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

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

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

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

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

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

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

Lamport‘s one-time signature

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

$m' :=$ message whose signature is requested by A

$(m,\sigma) :=$ A's final output

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

- $\mathrm{Vrfy}_{pk}(m,\sigma) = 1$
- $m_i = b$ 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
- key ingredient to achieve this are collision-resistant hash functions
- constructions works for one-time signatures and general signatures
- constructions based on simpler ingredients i.e. universal one-way hash functions also known
- these can be constructed from one-way functions
- to go from one-time signatures to general signatures first construct stateful signatures
- use PRFs to remove statefulness

Hash functions

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

1. $\text{Gen}(1^n)$ 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 $\text{H}(s,x) \in \{0,1\}^{l(n)}$.

Write $\text{H}^s(x)$ for $\text{H}(s,x)$.

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 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 $\mu : \mathbb{N} \rightarrow \mathbb{R}^+$ such that

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

Weaker notions

1. coll.-res. $\ldots$

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 $\Pr\left[\forall i : H^s(x_i) = y_i\right] = 2^{-qn}$, then

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

Arbitrary length hash functions

Construction 2.13 (Merkle-Damgård) $\Pi' = (\mathrm{Gen}', h)$ fixed-length hash-function with input length $2l(n)$, output length $l(n)$.

$\Pi = (\mathrm{Gen}, H)$ defined as:

Gen: same as Gen$'$.

H: on input key s and $x \in \{0,1\}^*$, $|x| = L < 2^{l(n)}$ do:

1. $B := \lceil L/l \rceil$ and pad x with 0's so its length is multiple of l, $x := x_1 \ldots x_B$, $x_{B+1} := L$ (with l bits).
2. $z_0 := 0^l$.
3. For $i = 1, \ldots, B+1$, compute $z_i := h^s(z_{i-1} \| x_i)$.
4. Output z_{B+1}.

Arbitrary length hash functions

Construction 2.13 $(\text{Merkle-Damgård})\ \Pi' = (\text{Gen}', h)$ fixed-length hash-function with input length $2l(n)$, output length $l(n)$.

$\Pi = (\text{Gen}, H)$ defined as:

$\text{Gen}:$ same as Gen'.

$H:$ on input key s and $x \in \{0,1\}^*$, $|x| = L < 2^{l(n)}$ do:

1. $B := \lceil L/l \rceil$ and pad x with 0's so its length is multiple of l, $x := x_1 \ldots x_B, x_{B+1} := L$ (with l bits).
2. $z_0 := 0^l$.
3. For $i = 1, \ldots, B+1$, compute $z_i := h^s(z_{i-1} \| x_i)$.
4. Output z_{B+1}.

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

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 2.15 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 2.16 If Υ' is secure and Π is collision-resistant, then Υ is secure.

Hash-and-Sign

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

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

Construction 2.15 Sig. scheme $\Upsilon = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ defined as:

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

$pk = (pk', s),\ sk = sk'$

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

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

Theorem 2.17 If Υ' is a one-time signature and Π is collision-resistant, then Υ is a one-time signature.

Hash-and-Sign

$A :=$ adversary against Υ

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

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

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

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

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 $Sign'_{sk'}(\cdot)$

1. Run Gen_H to obtain key s.
2. Simulate A. Whenever A queries its Sign-oracle $Sign_{sk}(\cdot)$ on a message m', do:
   a) Compute $h := H^s(m')$.
   b) Query $Sign'_{sk'}(\cdot)$ on input h to obtain $\sigma' := Sign'_{sk'}(h)$, 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".