Today: Continue the construction of a signature scheme from one-way functions

Note: One-way fucntion is a minimal assumption since if OWFs do not exist then Gen can be inverted.

Gen: $r \longmapsto VK$

The construction proceeds in three steps:

Step 1: Construct a signature scheme that is one-time secure for msgs of bounded length. ← Covered last class

Step 2: From bounded length msgs to unbounded length msgs.

Step 3: From one-time security for unbounded length msgs to standard (many-time) security.

Step 2: From bounded length msgs to unbounded length msgs.

Hash-then-sign: Given a collision resistant hash function $h: \{0,1\}^* \to \{0,1\}^n$

Can convert a one-time secure signature scheme (Gen, Sign, Ver) with msg space $\{0,1\}^n$

into a one-time secure signature scheme (Gen, Sign', Ver') with msg space $\{0,1\}^*$, as follows:

$Sign'(sk,m)=Sign(sk,h(m))$

$Ver'(vk,m,\sigma)=1$ iff $Ver(vk,h(m),\sigma)=1$

Theorem: If (Gen, Sign, Ver) is one-time (resp. many time) secure with msg space $\{0,1\}^n$

and if $h:\{0,1\}^* \longrightarrow \{0,1\}^n$ is a collision resistant hash function,

then (Gen, Sign', Ver') is one-time (resp. many time) secure with msg space $\{0,1\}^*$.

"Proof": Suppose there exists an adv A that breaks the security of (Gen, Sign', Ver').

Denote the signing queries made by A by $m_1,\dots,m_t$ and suppose it generates an accepting pair (m^*,σ^*)

s.t. $m^* \neq m_i$ for every $i \in [t]$.

Case 1: There exists $i \in [t]$ s.t. $h(m^*)=h(m_i)$. In this case we can use A to find a collision, contradiction.

Case 2: For every $i \in [t]$ $h(m^*) \neq h(m_i)$. In this case we can use A to break the (one-time) security of (Gen, Sign, Ver).

∎

Note: Hash-and-sign is not only useful to enlarge the msg space, but it is also useful for:

1. <u>Enhancing efficiency:</u>

   signing shorter msgs is faster than signing long ones, and signing is typically much slower than hashing.

2. <u>Enhancing security:</u>

   If we think of the hash function as a random oracle (i.e., indistinguishable from a truly random function), then even though the adversary can make Alice sign any msg m of his choice,
   if we use the hash-then-sign paradigm, then the adversary will obtain a signature for $H(m)$ which is a random message. This motivates weaker security definitions.

Def: A signature scheme (Gen, Sign, Ver) is existentially unforgeable against random message attack if for every (polynomial) t, the adversary, given polynomially many valid msg-signature pairs $\{(m_i, \sigma_i)\}_{i\in[t]}$ for random msgs $m_1, \ldots, m_t$, outputs a valid msg-signature pair (m^*, σ^*) s.t. $m^* \notin \{m_1, \ldots, m_t\}$, only with negligble prob.

An even weaker security notion requires the adversary to sign a random msg as opposed to a msg of his choice.

Def: A signature scheme (Gen, Sign, Ver) is secure for random messages against random message attack if for every (polynomial) t, the adversary, given polynomially many valid msg-signature pairs $\{(m_i, \sigma_i)\}$ for random msgs $m_1, \ldots, m_t$, and given a random msg m^*, outputs a valid signature σ^* only with negligble prob.

See PSet 2 for a question about the hash-and-sign paradigm and its security benifits!

So far: one-time secure signature scheme for unbounded msg space

Step 3: From one-time security for unbounded length msgs to standard (many-time) security.

tree-based signature scheme!

Use a one-time secure scheme (Gen, Sign, Ver) for msgs of unbounded length, and a PRF F,

to construction a many-time secure signature scheme (Gen', Sign', Ver').

Take 1: Inefficient construction

for a many-time secure signature scheme with msg space $\{0,1\}^n$.

Generate $N=2^n$ pairs $\{(sk_i^n, vk_i^n)\}_{i\in\{0,1\}^n}$ for the one-time scheme.

These keys are going to be in the leaves of tree.

Use sk_i^n only to sign msg $i \in \{0,1\}^n$.

Generate 2^{n-1} pairs $\{(sk_i^{n-1}, vk_i^{n-1})\}_{i\in\{0,1\}^{n-1}}$ for the one-time scheme.

These keys are going to be the parents of the leaves in the tree.

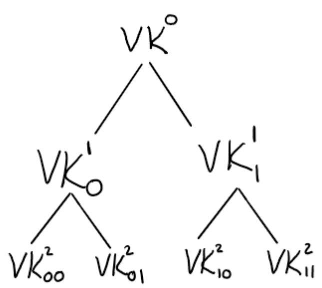
For every $i\in\{0,1\}^{n-1}$ use sk_i^{n-1} only to sign (vk_{i0}^n, vk_{i1}^n).

More generally, for every $j\in[n]$, generate 2^j pairs $\{(sk_i^j, vk_i^j)\}_{i\in\{0,1\}^j}$

for the one-time scheme.

These keys are going to be at layer n-j in the tree (where leaves are at level n).

Use sk_i^j only to sign $(vk_{i0}^{j+1}, vk_{i1}^{j+1})$.



Gen': Outputs VK^0 as the verification key and keeps all the keys $\{SK^j_{b_1..b_j}, VK^j_{b_1..b_j}\}_{\substack{j\in[n] \\ b_1..b_j\in\{0,1\}}}$ and SK^0 as the secret key

Sign'(SK,i): $Sign(SK^n_i, i), (VK^{j+1}_{i_1..i_j0}, VK^{j+1}_{i_1..i_j1})^{n-1}_{j=0}, (Sign(SK^j_{i_1..i_j}, VK^{j+1}_{i_1..i_j0}, VK^{j+1}_{i_1..i_j1}))^{n-1}_{j=0}$

Can also be SK^n_i

Ver': Verifies the path of signatures.

Main downside: Efficiency!

The signer needs to prepare and store an exponential size tree of keys!

Final Construction:

Prepare the tree as needed! No need to store the entire tree!

Main idea: Use PRF!

Gen': 1. Run Gen to obtain a key pair (sk, vk) for the one-time scheme.

2. Choose a random PRF key K.

Ouput vk'=vk and sk'=(sk, K).

Sign': Given a secret key sk'=(sk, K) and a msg $i\in\{0,1\}^n$

1. For every $b\in\{0,1\}$, let $r_b = F(K,b)$.

Let (sk'_b, vk'_b) be the key pair generated by Gen with randomness r_b.

2. For every $j\in[n-1]$ and every $b\in\{0,1\}$, generate $(SK^{j+1}_{i_1..i_jb}, VK^{j+1}_{i_1..i_jb})$ by running Gen with randomness $F(K,i_1,\dots,i_j,b)$

3. Sign as before.

Ver': Verifies the path of signatures (as before).