1 Last lecture, and plan for today's lecture

Last two lectures, we saw the security definition(s) for signature scheme, and saw a provably secure construction in the random oracle model (based on the RSA assumption). Today, we will discuss a construction that's provably secure in the standard model (assuming one way functions + CRHFs). We will first see a one-time secure signature scheme. Next, we will bootstrap this construction to achieve unbounded security (at the cost of very large signing keys/stateful signing algorithm). Finally, we will see how to make it stateless.

2 One-time secure digital signature

Let $f : \{0,1\}^\ell \to \{0,1\}^\ell$ be a secure one-way function (think of discrete log, RSA, etc). First, we will consider a signature scheme with message space $\{0,1\}^n$, whose security relies on the one-wayness of f.

Lamport's one-time signatures

- KeyGen : Choose $2n$ strings $a_{i,b} \leftarrow \{0,1\}^\ell$ for each $i \in [n]$, $b \in \{0,1\}$. The secret key is $\mathsf{sk} = (a_{i,b})_{i,b}$ and the verification key is $\mathsf{vk} = (f(a_{i,b}))_{i,b}$.
- $\mathsf{Sign}(m = (m_1, \ldots, m_n), \mathsf{sk} = (a_{i,b})_{i,b})$: The signing algorithm outputs $\sigma = (a_{i,m_i})$ as the signature.
- $\mathsf{Verify}(m = (m_1, \ldots, m_n), \sigma = (z_i), \mathsf{vk} = (\mathsf{vk}_{i,b}))$: The verification algorithm checks if $f(z_i) = \mathsf{vk}_{i,m_i}$ for all $i \in [n]$. If so, it outputs 1, else it outputs 0.

First, check that correctness holds. The scheme is deterministic.

Is this a scheme with unique signatures? Would this be a scheme with unique signatures if Discrete Log/RSA are used?

In Question 35.01, we discuss why the above construction may not always give a signature scheme with strong unforgeability.

2.1 One-time unforgeability

This scheme is one-time weakly unforgeable. Intuitively, this is because the adversary queries for a signature on m, and must send a forgery on a different message m^*. Since these two strings differ in at least one position, the reduction can 'guess' this position, and plant the OWF challenge at this position. The formal reduction is given below.

Claim 37.01. Assuming f is a secure one way function, the construction described above is a weakly unforgeable one-time secure signature scheme.

Proof. The reduction algorithm receives y from the OWF challenger. It guesses an index $i^* \leftarrow [n]$ and a bit $b^* \leftarrow \{0,1\}$. For all $(i,b) \neq (i^*, b^*)$, it chooses $a_{i,b}$ and sets $\mathsf{vk}_{i,b} = f(a_{i,b})$. It sets $\mathsf{vk}_{i^*,b^*} = y$ and sends vk to the adversary. The adversary sends a message $m \in \{0,1\}^n$. If $m_{i^*} = b^*$, then the reduction algorithm quits. Else, it sends $\sigma = (a_{i,m_i})_i$ as the signature. Finally the adversary outputs a forgery (m^*, σ^*). If $m^* \neq m$ and $\mathsf{Verify}(m^*, \sigma^*, \mathsf{vk}) = 1$, the reduction checks if $m^*_{i^*} = b^*$. If so, it sends $x = \sigma^*_{i^*}$ to the OWF challenger. Note that since the forgery verifies, if $m^*_{i^*} = b^*$, then $f(x) = y$. If the adversary breaks weak unforgeability with probability ϵ, then the reduction breaks the OWF security with probability $O(\epsilon/n)$. □

2.2 One-time unforgeable signatures with unbounded message space

Given a one-time unforgeable signature scheme with bounded message space, we can combine it with a CRHF to get a one-time signature scheme with unbounded messages, satisfying one-time security. Simply hash the message and compute a signature on the digest. The hash-and-sign approach also works for one-time security.

2.3 Two-time security

This scheme is not two-time secure. An adversary can query for 0^n, followed by 1^n, and it would learn the entire secret key. Can we achieve any heuristic guarantees here? Question 35.02 explores this possibility.

3 From one-time security to unbounded security

Let $\mathcal{S}_{\text{OT}} = (\mathsf{KeyGen}_{\text{OT}}, \mathsf{Sign}_{\text{OT}}, \mathsf{Verify}_{\text{OT}})$ be a one-time secure signature scheme with message space $\{0,1\}^*$. Our aim in this section is to construct a signature scheme with unbounded security. As a first step, we will construct a signature scheme with the following properties:

- the message space is $\{0,1\}^n$.
- the signing key has size exponential in n. However, the signature size is $\mathsf{poly}(n)$.
- the verification key has size $\mathsf{poly}(n)$, and verification is efficient ($\mathsf{poly}(n)$).

3.1 From one-time security to unbounded security with large signing keys

For this construction, we will use a 'binary tree structure'. The depth of the tree is n, and the leaves are the message space. Corresponding to every internal node, we have a fresh one-time signing/verification key. The construction is formally described below.

Tree-based signatures

- KeyGen : Let ϵ denote the empty string, and let $T = \{0,1\}^{<n} \cup \epsilon$. For every string $w \in T$, choose $(\mathsf{sk}_w, \mathsf{vk}_w) \leftarrow \mathsf{KeyGen}_{\text{OT}}$. The final verification key $\mathsf{VK} = \mathsf{vk}_\epsilon$. The signing key consists of all signing/verification keys $(\mathsf{sk}_w, \mathsf{vk}_w)_{w \in T}$.
- $\mathsf{Sign}(m, \mathsf{SK})$: Let $m = (m_1, m_2, \ldots, m_n)$ be the message. Let $y_0 = \epsilon$, and for all $i < n$, $y_i = (m_1, \ldots, m_i)$. The signing algorithm computes the following signatures for $i = 1$ to $n-1$: $\sigma_i = \mathsf{Sign}_{\text{OT}}(\mathsf{vk}_{y_{i-1}|0} \,||\, \mathsf{vk}_{y_{i-1}|1}, \mathsf{sk}_{y_{i-1}})$. Finally, it computes $\sigma_n = \mathsf{Sign}_{\text{OT}}(m, \mathsf{sk}_{y_{n-1}})$. The final signature is $\sigma = ((\mathsf{vk}_{y_{i-1}|0}, \mathsf{vk}_{y_{i-1}|1}, \sigma_i))_{i=1}^n$.
- $\mathsf{Verify}(m, \sigma = ((\mathsf{vk}_{i,0}, \mathsf{vk}_{i,1}, \sigma_i))_{i=1}^n, \mathsf{VK} = \mathsf{vk}_\epsilon)$: The verification algorithm sets $\mathsf{vk}_0 = \mathsf{VK}$, and does the following for $i = 1$ to n:
  - If $i < n$, set $m_i = \mathsf{vk}_{i,0} \,||\, \mathsf{vk}_{i,1}$, else set $m_n = m$.
  - Check $\mathsf{Verify}_{\text{OT}}(m_i, \mathsf{vk}_{i-1}, \sigma_i) = 1$. If so, set $\mathsf{vk}_i = \mathsf{vk}_{i,m_i}$, else output 0.

  If all these checks pass, output 1.

Verify the following properties of this construction:

- There are $O(2^n)$ key-pairs in the secret key. Even if an adversary queries for many signatures, corresponding to different messages, every one-time key-pair is used **at most once**. Note that it is crucial that, at each node w, we sign both the verification keys at $w|0$ and $w|1$ (that is, $\sigma_i = \mathsf{Sign}_{\text{OT}}(\mathsf{vk}_{y_{i-1}|0} \,||\, \mathsf{vk}_{y_{i-1}|1}, \mathsf{sk}_{y_{i-1}})$. (why?)
- Even though there are exponentially many components in the signing key, we only require $O(n)$ of the signing/verification keys for any signing query. Therefore, if the signing party is **stateful** and chooses these keys on-the-fly, then the signing algorithm is a polynomial time algorithm. The state of the signing algorithm grows with the number of signing queries.

One can prove that the above construction results in an unforgeable signature scheme in the standard model (with a stateful signing algorithm). The proof (which is not included in the syllabus) is not complicated, but requires careful book-keeping. See Section 12.6.2 of the Katz-Lindell textbook for more details (Chapter 12 of the Katz-Lindell textbook is available on Moodle/shared Dropbox folder).

4 From unbounded security with large secret keys to unbounded security with efficient secret keys

In this section, we will modify the above construction to achieve a signature scheme where the signing key is polynomial in size. The modification is pretty simple: generate the keys $(\mathsf{sk}_w, \mathsf{vk}_w)$ on-the-fly, **pseudorandomly**.

Tree-based signatures with efficient signing keys

Let $\mathcal{S}_{\mathrm{OT}}$ be a one-time secure signature scheme, where the key generation algorithm uses r bits of randomness. Let $F : \{0,1\}^{<n} \times \mathcal{K} \to \{0,1\}^r$ be a secure pseudorandom function.

- KeyGen : Choose $(\mathsf{sk}_\epsilon, \mathsf{vk}_\epsilon) \leftarrow \mathsf{KeyGen}_{\mathrm{OT}}$, and a PRF key $k \leftarrow \mathcal{K}$.
- $\mathsf{Sign}(m, \mathsf{SK})$: Let $m = (m_1, m_2, \ldots, m_n)$ be the message.

  Let ϵ denote the empty string, $y_0 = \epsilon$, and for all $i < n$, $y_i = (m_1, \ldots, m_i)$.

  For each $i < n$ and $b \in \{0,1\}$, compute $r_{i,b} = F(y_i|b, k)$, $(\mathsf{sk}_{y_i \mid b}, \mathsf{vk}_{y_i \mid b}) \leftarrow \mathsf{KeyGen}_{\mathrm{OT}}(1^n; r_{i,b})$.

  The signing algorithm computes the following signatures for $i = 1$ to $n-1$: $\sigma_i = \mathsf{Sign}_{\mathrm{OT}}(\mathsf{vk}_{y_{i-1}|0} \,||\, \mathsf{vk}_{y_{i-1}|1}, \mathsf{sk}_{y_{i-1}})$. Finally, it computes $\sigma_n = \mathsf{Sign}_{\mathrm{OT}}(m, \mathsf{sk}_{y_{n-1}})$.

  The final signature is $\sigma = ((\mathsf{vk}_{y_{i-1}|0}, \mathsf{vk}_{y_{i-1}|1}, \sigma_i))_{i=1}^n$.
- $\mathsf{Verify}(m, \sigma = ((\mathsf{vk}_{i,0}, \mathsf{vk}_{i,1}, \sigma_i))_{i=1}^n, \mathsf{VK} = \mathsf{vk}_\epsilon)$: same as the stateful verification (note that verification in both cases depends only on vk_ϵ).

High-level outline of the proof The proof of this scheme will be very similar to the proof of the stateful signatures scheme, with one additional jump : replacing the PRF with a truly random function. Once the PRF is replaced with a random function, the one-time signing/verification keys at every node are sampled uniformly at random, and therefore this is identical to the stateful signing algorithm (where the signing/verificatiton keys are chosen at random, on-the-fly).

5 Lecture summary

Summary: We saw a standard model construction of digital signatures. As a first step, we constructed a one-time secure signature scheme. Next, we saw a tree-based construction, where the secret key was exponentially large. Finally, we saw how to make the key succinct using a PRF key.

Additional resources Chapter 12.6 of the Katz-Lindell textbook (chapter 12 uploaded on Moodle).

6 Questions

Question 35.01. The construction described in Section 2 may not be a one-time strongly unforgeable signature scheme. Let f be a secure one-way function. Construct a new one-way function f' such that the construction in Section 2 is not a one-time strongly unforgeable signature scheme.

What happens if we use a collision-resistant hash function family $\mathcal{H}$ instead of a one-way function in the above construction?

Question 35.02. Consider the following heuristic signature scheme. It supports message space $\{0,1\}^*$, and uses a hash function $H : \{0,1\}^* \to \{0,1\}^n$ (to be modeled as a hash function) and a one way function $f : \{0,1\}^\ell \to \{0,1\}^\ell$.

- KeyGen: The key generation algorithm chooses $2n$ strings $(a_{i,b})_{i\in[n],b\in\{0,1\}}$ and set this as the signing key. It set $\mathsf{vk}_{i,b} = f(a_{i,b})$ and $\mathsf{vk} = (\mathsf{vk}_{i,b})_{i,b}$.
- $\mathsf{Sign}(m, \mathsf{sk})$: Let $y = H(m)$. Note that $y = (y_1, \ldots, y_n)$ is an n bit string. The signature is $\sigma = (a_{i,y_i})_i$.
- $\mathsf{Verify}(m, \sigma, \mathsf{vk})$: The algorithm first computes $y = H(m)$. Next, it checks if $f(\sigma_i) = \mathsf{vk}_{i,y_i}$ for all $i \in [n]$. If so, it outputs 1.

Prove that for any constant c, this scheme is c-time secure in the random oracle model. What is the success probability of the reduction algorithm, and what properties of the random oracle model are being used here? What can we say about unbounded weak-unforgeability?