Lecture 25: Signatures on Arbitrary-length Messages
Problem Statement

- Suppose we are given a \((\text{Gen}, \text{Sign}, \text{Ver})\) digital signature scheme for \(B\)-bit messages (i.e., messages in \(\{0, 1\}^B\)), for some fixed \(B \in \mathbb{N}\). We shall refer to this signature scheme as the basic signature scheme.

- Given this signature scheme \((\text{Gen}^*, \text{Sign}^*, \text{Ver}^*)\) for \(B\)-bit messages, construct a signature scheme for arbitrary-length messages (i.e., messages in \(\{0, 1\}^*\)).
Given a message $m \in \{0, 1\}^*$, we use standard padding technique to make its length a multiple of $B$ and, then, break it into $B$-bit blocks $(m_1, m_2, \ldots, m_\alpha)$, where $m_1, m_2, \ldots, m_\alpha \in \{0, 1\}^B$.

Our first strategy is to sign the blocks $m_1, m_2, \ldots, m_\alpha$ using the basic signature scheme. Suppose the signatures of $m_1, m_2, \ldots, m_\alpha$ are, respectively, $\sigma_1, \sigma_2, \ldots, \sigma_\alpha$.

Our first attempt generates the signature of the message $m \equiv (m_1, m_2, \ldots, m_\alpha)$ as the signature $\sigma = (\sigma_1, \sigma_2, \ldots, \sigma_\alpha)$.
Vulnerability: Prefix Attacks

Suppose we are given the signature of the message $m = (m_1, m_2, \ldots, m_\alpha)$ as the signature $\sigma = (\sigma_1, \sigma_2, \ldots, \sigma_\alpha)$.

We can generate the signature of the message $m' = (m_1, m_2, \ldots, m_i)$ as $\sigma' = (\sigma_1, \sigma_2, \ldots, \sigma_i)$, for any $1 \leq i < \alpha$.

**Solution.** We need to tie the “number of the blocks” into the message being signed by the basic scheme.
Given a message $m \in \{0, 1\}^*$, we use standard padding technique to make its length a multiple of $B/2$ and, then, break it into $B/2$-bit blocks $(m_1, m_2, \ldots, m_\alpha)$, where $m_1, m_2, \ldots, m_\alpha \in \{0, 1\}^{B/2}$.

Our second strategy is to sign the blocks $(\alpha \parallel m_1), (\alpha \parallel m_2), \ldots, (\alpha \parallel m_\alpha)$ using the basic signature scheme. We clarify that $(\alpha \parallel m_i)$ is the concatenation of (a) $B/2$-bit representation of the number of total blocks $\alpha$, and (b) the $B/2$-bit message $m_i$. Suppose the signatures are, respectively, $\sigma_1, \sigma_2, \ldots, \sigma_\alpha$.

Our second attempt generates the signature of the message $m \equiv (m_1, m_2, \ldots, m_\alpha)$ as the signature $\sigma \equiv (\sigma_1, \sigma_2, \ldots, \sigma_\alpha)$.
Suppose we are given the signature of the message $m = (m_1, m_2, \ldots, m_\alpha)$ as the signature $\sigma = (\sigma_1, \sigma_2, \ldots, \sigma_\alpha)$.

We can generate the signature of the message $m' = (m_2, m_1, \ldots, m_\alpha)$ as $\sigma' = (\sigma_2, \sigma_1, \ldots, \sigma_\alpha)$.

In general, we can permute the message blocks of $m$ and generate the signature of the permuted message.

**Solution.** We need to tie the “position of the message block” into the message being signed by the basic scheme.
Given a message $m \in \{0, 1\}^*$, we use standard padding technique to make its length a multiple of $B/3$ and, then, break it into $B/3$-bit blocks $(m_1, m_2, \ldots, m_{\alpha})$, where $m_1, m_2, \ldots, m_{\alpha} \in \{0, 1\}^{B/3}$.

Our second strategy is to sign the blocks $(\alpha\|1\|m_1), (\alpha\|2\|m_2), \ldots, (\alpha\|\alpha\|m_{\alpha})$ using the basic signature scheme. We clarify that $(\alpha\|m_i)$ is the concatenation of (a) $B/3$-bit representation of the number of total blocks $\alpha$, (b) $B/3$-bit representation of the position $i$, and (c) the $B/3$-bit message $m_i$. Suppose the signatures are, respectively, $\sigma_1, \sigma_2, \ldots, \sigma_{\alpha}$.

Our third attempt generates the signature of the message $m \equiv (m_1, m_2, \ldots, m_{\alpha})$ as the signature $\sigma = (\sigma_1, \sigma_2, \ldots, \sigma_{\alpha})$.
Vulnerability: Splicing Attacks

- Suppose we are given the signature of the message \( m = (m_1, m_2, \ldots, m_\alpha) \) as the signature \( \sigma = (\sigma_1, \sigma_2, \ldots, \sigma_\alpha) \)
- Suppose we are given the signature of another message (of the same number of blocks) \( m' = (m_1, m_2, \ldots, m_\alpha) \) as the signature \( \sigma' = (\sigma'_1, \sigma'_2, \ldots, \sigma'_\alpha) \)
- We can generate the signature of the message \( m'' = (m'_1, m_2, \ldots, m_\alpha) \) as \( \sigma'' = (\sigma'_1, \sigma_2, \ldots, \sigma_\alpha) \)
- In general, we can splice the blocks of \( m \) and \( m' \) and generate the message \( m'' \) and forge the signature on \( m'' \)
- **Solution.** We need to “tie together all blocks of a particular message” into the message being signed by the basic scheme
Fourth Attempt

- Given a message $m \in \{0, 1\}^*$, we use standard padding technique to make its length a multiple of $B/4$ and, then, break it into $B/4$-bit blocks $(m_1, m_2, \ldots, m_\alpha)$, where $m_1, m_2, \ldots, m_\alpha \in \{0, 1\}^{B/4}$

- Pick a random string $s \leftarrow \{0, 1\}^{B/4}$

- Our second strategy is to sign the blocks $(\alpha \| 1 \| s \| m_1), (\alpha \| 2 \| s \| m_2), \ldots, (\alpha \| \alpha \| s \| m_\alpha)$ using the basic signature scheme. We clarify that $(\alpha \| m_i)$ is the concatenation of (a) $B/4$-bit representation of the number of total blocks $\alpha$, (b) $B/4$-bit representation of the position $i$, (c) the random bit string $s$, and (d) the $B/4$-bit message $m_i$. Suppose the signatures are, respectively, $\sigma_1, \sigma_2, \ldots, \sigma_\alpha$

- Our fourth attempt generates the signature of the message $m \equiv (m_1, m_2, \ldots, m_\alpha)$ as the signature $\sigma = (\sigma_1, \sigma_2, \ldots, \sigma_\alpha)$.

- The idea is that all blocks of a message shall have the same random bit-string $s$. Furthermore, the bitstring corresponding to two messages shall be different with high probability (using the Birthday bound).
The fourth attempt ensures that prefix, permutation, and splicing attacks cannot forge signatures.

In fact, this scheme is secure against all forging strategies (not just the three forging strategies mentioned above). In a higher-level course, we can prove this stronger result.

It is left as an exercise to write the algorithms \((\text{Gen}^*, \text{Sign}^*, \text{Ver}^*)\) using the algorithms \((\text{Gen}, \text{Sign}, \text{Ver})\).