Lecture 25: CBC-MAC
One-time MAC: We can construct from 2-wise independent hash function families. These exist even against adversaries with unbounded computational power.

General MAC: We can construct if One-way Functions Exist. For example, we use pseudorandom functions (using the GGM construction) for these constructions. The GGM construction uses length-doubling pseudorandom generators, and pseudorandom generators can be constructed from one-way functions.
Today’s Summary

Today we shall construct MACs using pseudorandom function (PRF) family and the Cipher Block Chaining (CBC) technique
What we shall use

- Pseudorandom Function Family \( \mathcal{F} = \{F_1, F_2, \ldots, F_\alpha\} \), where each function \( F_i : \{0, 1\}^B \rightarrow \{0, 1\}^B \)

What we shall construct

- Construct a MAC scheme for \( n \)-bit messages
Gen(). Create a key for the pseudorandom function family.
Return $sk \gets \{1, 2, \ldots, \alpha\}$

Mac$_{sk}(m)$. Interpret $m = (m_1, m_2, \ldots, m_\ell)$, where each $m_i$ is $B$-bits long and $\ell = \lceil n/B \rceil$

\[
\begin{align*}
F_{sk} & \quad F_{sk} & \quad F_{sk} & \quad \cdots & \quad F_{sk} \\
m_1 & \quad m_2 \quad \oplus & \quad m_3 \quad \oplus & \quad \cdots \quad \oplus & \quad m_\ell \\
 & \quad & \quad & \quad & \quad T
\end{align*}
\]
Ver<sub>sk</sub>(\(\tilde{m}, \tilde{\tau}\)). Let \(\tilde{m} = (\tilde{m}_1, \tilde{m}_2, \ldots, \tilde{m}_\ell)\), where each \(\tilde{m}_i\) is \(B\)-bit long. Return whether \(\tilde{\tau} = \tau'\) or not, where \(\tau'\) is calculated as below.

\[
\tilde{m}_1 \oplus F_{sk} \tilde{m}_2 \oplus F_{sk} \tilde{m}_3 \oplus F_{sk} \tilde{m}_\ell \oplus F_{sk} \tau'.
\]

CBC-MAC
Attack on this Scheme using Arbitrary-length Messages.

- The adversary sees the message-tag pair \((m, \tau)\), where 
  \[ m = (m_1, m_2, \ldots, m_\ell) \]
- The adversary sees the message-tag pair \((m', \tau')\), where 
  \[ m' = (m'_1, m'_2, \ldots, m'_{\ell'}) \]
- The adversary computes 
  \[ \tilde{m} = \left( m_1, \ldots, m_\ell, m'_1 \oplus \tau, m'_2, \ldots, m'_{\ell'} \right) \]
- The message-tag pair \((\tilde{m}, \tau')\) is a forgery (Check that this passes verification)
What we shall use

- Pseudorandom Function Family $\mathcal{F} = \{F_1, F_2, \ldots, F_\alpha\}$, where each function $F_i : \{0, 1\}^B \rightarrow \{0, 1\}^B$

What we shall construct

- Construct a MAC scheme for $\{0, 1\}^*$
Intuition for the construction.

- We shall use separate sk for each message length to “chain”
- The Gen() returns a random sk $\xleftarrow{\$} \{1, 2, \ldots, \alpha\}$.
- The pictorial summary of $\text{Mac}_{sk}(m)$ is provided in the next slide.
Suppose the message is $m \in \{0, 1\}^n$. It is interpreted as $(m_1, m_2, \ldots, m_\ell)$, where each $m_i$ is a $B$-bit string and $\ell = \lceil n/B \rceil$. Let $[n]_2$ represent the $B$-bit binary string that represents the length of $m$ in binary.
Note. You can use the same sk to sign messages of different length using the algorithm presented above!
We append the binary representation of the length of $m$ at the beginning and CBC-MAC the new message. See the construction below.

\[
\begin{align*}
[n]_2 & \quad m_1 \quad m_2 \quad m_3 \quad \cdots \quad m_\ell \\
F_{sk} & \quad F_{sk} \quad F_{sk} \quad F_{sk} \quad F_{sk} \\
\end{align*}
\]
Adding the length at the end is INSECURE! The following scheme is insecure.

\[ m_1 \oplus F_{sk} \oplus m_2 \oplus F_{sk} \oplus m_3 \oplus F_{sk} \oplus \cdots \oplus m_\ell \oplus F_{sk} \oplus [n]_2 \oplus F_{sk} \]
Students are strongly recommended to construct the attack on their own.

Suppose the adversary the message-tag pairs on two different $n$-bit messages $p$ and $q$. Let the message tag pairs be

\[
(p = (p_1, p_2, \ldots, p_\ell), \tau_p) \\
(q = (q_1, q_2, \ldots, q_\ell), \tau_q)
\]

The adversary requests to see the tag $\tau_m$ for the message $m$ as defined below:

\[
m = (p_1, p_2, \ldots, p_\ell, [n]_2, r_1, r_2, \ldots, r_t)
\]

We emphasize that here the adversary requests to see the signature on a particular message. All previous attacks had the adversary obtain message-tag pairs for arbitrary messages.
Now, the adversary can splice out \((p_1, \ldots, p_\ell)\) to replace \((q_1, \ldots, q_\ell)\) in the message \(m\) as follows

\[
m' = ( q_1, q_2, \ldots, q_\ell, [n]_2, r_1 \oplus \tau_p \oplus \tau_q, r_2, \ldots, r_t )
\]

Note that the tag of the message \(m'\) is identical to the tag \(\tau_m\)
But a small change to the above-mentioned insecure construction is secure.
All we need to ensure is that the key for the pseudorandom function used to chain the message-blocks is different from the key for the pseudorandom function used on $[n]_2$. Let $\text{key} = F_{sk}(0)$ and $\text{key}' = F_{sk}(1)$. Now, consider the following construction.
MAC-ing Arbitrary-length Messages, Third Construction II

\[ m_1 \oplus F_{\text{key}} \]

\[ m_2 \oplus F_{\text{key}} \]

\[ m_3 \oplus F_{\text{key}} \]

\[ \cdots \]

\[ m_\ell \oplus F_{\text{key}} \]

\[ [n]_2 \oplus F_{\text{key}}' \]

CBC-MAC
Check how this new construction prevents the adversarial attack where the message length was at the end. This is crucial to ensure that you have a good understanding of this new MAC scheme.

**Benefit of having the message-length at the end.** We do not need the length of the message ahead of time. We can even MAC messages that are coming as a stream!