2201-COL759
IIT Delhi — Lecture 20 — Venkata K

1 Plan for today's lecture

Last lecture, we discussed a few approaches for signing unbounded length messages. Many of those attempts were broken. Today, we will discuss how to fix two of the attempts discussed last class.

2 MAC for long messages

We saw a few attempts for building a MAC scheme for unbounded length messages. Let us break down the problem into two steps. First, we will construct MAC for *long but bounded-length* messages. Here the goal is to keep the keys (and, if possible, even the signatures) short, but handle much longer messages. In the next section, we will extend these constructions to handle messages of *arbitrary length*.

Let $\mathcal{M} = \{0,1\}^{n \cdot \ell}$ be the desired *long-but-bounded* message space.

2.1 MAC using counter mode

Recall the counter-mode MAC construction attempt from last class. It uses a MAC scheme $\mathsf{MAC}_{\mathrm{bdd}} = (\mathsf{Sign}_{\mathrm{bdd}}, \mathsf{Verify}_{\mathrm{bdd}})$ for bounded-length n-bit messages, and works as follows.

Let $m = m_1 \mid\mid \ldots \mid\mid m_{2\ell}$ be the message to be signed, where each m_i is $n/2$ bits long. The signing algorithm outputs

$$\sigma = \Big(\mathsf{Sign}_{\mathrm{bdd}}(1 \mid\mid m_1, k), \mathsf{Sign}_{\mathrm{bdd}}(2 \mid\mid m_2, k), \ldots, \mathsf{Sign}_{\mathrm{bdd}}(2\ell \mid\mid m_{2\ell}, k)\Big)$$

as the signature. [1]

The above construction takes care of 'rearrangement' attacks, where the adversary queries for a signature σ on message m and produces a new (message, signature) pair by simply shuffling the components of m and σ. However, it does not prevent 'mix-and-match' attacks. For simplicity, let us take $\ell = 1$. The adversary can query for a signature on $m = (m_1 \mid\mid m_2)$ and $m' = (m'_1 \mid\mid m'_2)$. It receives $\sigma = (\sigma_1, \sigma_2)$ and $\sigma' = (\sigma'_1, \sigma'_2)$ respectively. Finally, it sends $m^* = m_1 \mid\mid m'_2$ and $\sigma^* = (\sigma_1, \sigma'_2)$ as the forgery.

Qn: Consider the PRF based MAC scheme for bounded messages. Can we break the message into chunks, compute the signature on each chunk (appended with the counter), and compute the XOR of all signatures? That is, suppose we define $\mathsf{Sign}'((m_1, \ldots, m_\ell), k) = \oplus_i F(i \mid\mid m_i, k)$. Is this a secure MAC scheme?

Ans: The 'mix-and-match' attack doesn't work directly here. However, a different attack was proposed in class, which requires three queries to the signing algorithm. Suppose σ_1 is a signature on (m_1, m_2), σ_2 a signature on (m'_1, m_2) and σ_3 a signature on (m_1, m'_2). Then $\sigma_1 \oplus \sigma_2 \oplus \sigma_3$ is a signature on (m'_1, m'_2).

There is a simple fix to prevent these mix-and-match attacks, one that is **provably secure**. The signing algorithm can choose a random $n/3$-bit string r. The message is broken into chunks of $n/3$ bits, and the signature on $m = (m_1 \mid\mid \ldots \mid\mid m_{3\ell})$ is

$$\sigma = \Big(r, \ \ \mathsf{Sign}_{\mathrm{bdd}}(1 \mid\mid r \mid\mid m_1, k), \ldots, \mathsf{Sign}_{\mathrm{bdd}}(3\ell \mid\mid r \mid\mid m_{3\ell}, k)\Big).$$

Note that r also needs to be output as part of the signature (otherwise the verification algorithm cannot verify valid signatures).

If the underlying bounded signature scheme is secure, then we can show that the new resulting scheme $\mathsf{MAC}_{\mathrm{long}}$ is also secure. There will be an additive negligible factor loss, due to the small probability that for two of the queries, the randomness might be equal.

[1] Here the counters $1, 2, \ldots, 2\ell$ are expressed using $n/2$ bits.

Theorem 20.01. Suppose there exists a p.p.t. adversary $\mathcal{A}$ that makes q signing queries, and breaks the strong unforgeability of $\mathsf{MAC}_{\text{long}}$ with probability ϵ. Then there exists a p.p.t algorithm $\mathcal{B}$ that makes $O(\ell \cdot q)$ signing queries, and breaks the strong unforgeability of $\mathsf{MAC}_{\text{bdd}}$ with probability $\epsilon - q^2/2^{n/3}$.

Here, the $q^2/2^{n/3}$ comes from 'birthday bound'.

Proof. Suppose there exists an adversary that can win the strong unforgeability game with probability ϵ. We will first define an altered game that is indistinguishable from the strong unforgeability game. Next, we prove that if an adversary can win in the altered game, then there exists a reduction algorithm that can break the base 'bounded-message' scheme.

- **Game-0:** This is the strong unforgeability game.
  - Setup: Challenger chooses a signing key k.
  - Signing queries: The adversary makes polynomially many signing queries. For the i^{th} signing query $m_i = (m_{i,1}, \ldots, m_{i,\ell})$, the challenger chooses a uniformly random string $r_i \leftarrow \{0,1\}^{n/3}$. Next, it computes $\sigma_{i,j} \leftarrow \mathsf{Sign}((j \mid\mid r_i \mid\mid m_{i,j}), k)$ for each $j \in [\ell]$ and sends $(r_i, \sigma_{i,1}, \ldots, \sigma_{i,\ell})$.
  - Forgery: Finally, the adversary outputs $m^* = (m^*_1, \ldots, m^*_\ell)$, $\sigma^* = (r^*, \sigma^*_1, \ldots, \sigma^*_\ell)$, and wins if $(m^*, \sigma^*) \neq (m_i, \sigma_i)$ for all i and $\mathsf{Verify}(m^*, \sigma^*, k) = 1$.
- **Game-1:** This is similar to the strong unforgeability game, except that the challenger chooses randomness r_i without replacement.
  - Setup: Challenger chooses a signing key k.
  - Signing queries: The adversary makes polynomially many signing queries. For the i^{th} signing query $m_i = (m_{i,1}, \ldots, m_{i,\ell})$, the challenger chooses a uniformly random string $r_i \leftarrow \{0,1\}^{n/3}$ without replacement.
    Next, it computes $\sigma_{i,j} \leftarrow \mathsf{Sign}((j \mid\mid r_i \mid\mid m_{i,j}), k)$ for each $j \in [\ell]$ and sends $(r_i, \sigma_{i,1}, \ldots, \sigma_{i,\ell})$.
  - Forgery: Finally, the adversary outputs $m^* = (m^*_1, \ldots, m^*_\ell)$, $\sigma^* = (r^*, \sigma^*_1, \ldots, \sigma^*_\ell)$, and wins if $(m^*, \sigma^*) \neq (m_i, \sigma_i)$ for all i and $\mathsf{Verify}(m^*, \sigma^*, k) = 1$.

$$
\begin{aligned}
&\Pr\left[\mathcal{A} \text{ wins in Game-0}\right] \\
=&\Pr\left[(\mathcal{A} \text{ wins in Game-0}) \wedge (\forall\ i \neq j, r_i \neq r_j)\right] + \Pr\left[(\mathcal{A} \text{ wins in Game-0}) \wedge (\exists i \neq j \text{ s.t. } r_i = r_j)\right] \\
\leq&\Pr\left[\mathcal{A} \text{ wins in Game-0}) \wedge (\forall i \neq j, r_i \neq r_j)\right] + q^2/2^{n/3} \\
=&\Pr\left[\mathcal{A} \text{ wins in Game-1}\right] + q^2/2^{n/3}
\end{aligned}
$$

Here, the second last step follows from the birthday bound. Next, we will show that the probability of $\mathcal{A}$ winning in Game-1 is negligible.

Suppose, on the contrary, $\mathcal{A}$ wins Game-1 with probability ϵ. We will construct a reduction algorithm that breaks strong unforgeability of $\mathsf{MAC}_{\text{bdd}}$ with probability ϵ. Our reduction algorithm works as follows:

- For the i^{th} signing query $m_i = (m_{i,1} \mid\mid \ldots \mid\mid m_{i,3\ell})$, the reduction algorithm chooses a uniformly random string r_i subject to the restriction that $r_i \neq r_j$ for all $j < i$. Next, it makes 3ℓ queries to the challenger. The j^{th} query is $j \mid\mid r_i \mid\mid m_{i,j}$, and it receives $\sigma_{i,j}$. Finally, after these 3ℓ queries, the reduction algorithm sends $\sigma_i = (r_i, \ \sigma_{i,1}, \ldots, \sigma_{i,3\ell})$ to the adversary.
- Finally, after q queries, the adversary sends a forgery (m^*, σ^*) such that $(m^*, \sigma^*) \neq (m_i, \sigma_i)$ for all $i \in [q]$. Suppose $\mathsf{Verify}(m^*, \sigma^*, k) = 1$. Let $\sigma^* = (r^*, \sigma^*_1, \ldots, \sigma^*_\ell)$. There are two cases:

- $r^* \neq r_i$ for all i. In this case, the reduction algorithm can send $((1 \mid\mid r^* \mid\mid m_1), \sigma_1^*)$ as a forgery.
- $r^* = r_i$ for some (unique) $i \in [q]$. Since $(m^*, \sigma^*) \neq (m_i, \sigma_i)$, we know that there exists some index $j \in [\ell]$ such that $(m_j^*, \sigma_j^*) \neq (m_{i,j}, \sigma_{i,j})$. Then the reduction algorithm can send $((j \mid\mid r^* \mid\mid m_j^*), \sigma_j^*)$ as a forgery to the challenger.

  For this to be a valid forgery for the $\mathsf{MAC}_{\text{bdd}}$ scheme, we also need to argue that the reduction never received a signature σ_j^* on $(j \mid\mid r^* \mid\mid m_j^*)$ from the challenger. Since i is the unique index such that $r_i = r^*$, we only need to focus on the j^{th} signature query made by $\mathcal{B}$ when responding to $\mathcal{A}$'s i^{th} signature query.

  If m_j^* is equal to $m_{i,j}$, then $\sigma_j^* \neq \sigma_{i,j}$, and hence this is a valid forgery. If $m_{i,j} \neq m_j^*$, the tuple $(j \mid\mid r^* \mid\mid m_j^*)$ was never queried to the challenger for a signature. As a result, we again have a valid forgery.

□

Remark: This scheme has a *randomized signing algorithm*. Sometimes, it is desirable to have deterministic signing algorithms, especially on low-power devices which have limited source of randomness. One option is to generate the randomness using a PRF. The PRF key is also part of the MAC key, and the randomness is generated by applying the PRF on the message.

How to construct PRFs with large input space? Suppose you are given a PRF/PRP with bounded input space. Can we obtain a PRF that can handle longer inputs? You can construct one along the lines of the CBC-MAC construction (outlined below). Can you use the GGM construction here? We will see other constructions (in Assignment 4).

2.2 CBC-MAC

The next construction (one that is widely used in practice) is closely related to the cipher-block chaining mode of encryption. Let $\mathcal{M} = (\{0,1\}^n)^\ell$ be the desired message space for our MAC scheme. The scheme $\mathsf{CBC\text{-}MAC}_\ell$ is defined as follows:

- $\mathsf{Sign}_\ell(m = (m_1, \ldots, m_\ell), k)$: Set $y_1 = F(m_1, k)$. For all $i \in [2, \ell]$, set $y_i = F(m_i \oplus y_{i-1}, k)$. Finally, output $\sigma = y_\ell$ as the signature.
- $\mathsf{Verify}_\ell(m = (m_1, \ldots, m_\ell), \sigma, k)$: Set $y_1 = F(m_1, k)$. For all $i \in [2, \ell]$, set $y_i = F(m_i \oplus y_{i-1}, k)$. Finally, output 1 if and only if $\sigma = y_\ell$.

The above scheme is a provably secure scheme for fixed block-length message space. The proof is not very complicated, but it is not included in the course syllabus.

3 MAC for unbounded length messages

In this section, we will discuss how to modify the above constructions to handle unbounded length messages. For simplicity of exposition, we will assume that the message length is a multiple of n.

3.1 Counter mode MAC for unbounded length messages

At first sight, it appears that the scheme described in Section 2.1 already handles unbounded length messages.

- $\mathsf{Sign}(m, k)$: Let $|m| = \ell \cdot n/3$ for some integer ℓ. Choose a random $n/3$-bit string r, and output $\Big(r, \mathsf{Sign}_{\text{bdd}}(1 \mid\mid r \mid\mid m_1, k), \ldots, \mathsf{Sign}_{\text{bdd}}(\ell \mid\mid r \mid\mid m_\ell, k)\Big)$.
- $\mathsf{Verify}(m = (m_1 \mid\mid \ldots \mid\mid m_\ell), \sigma = (r, \sigma_1, \ldots, \sigma_\ell), k)$: For each $i \in [\ell]$, check $\mathsf{Verify}_{\text{bdd}}(i \mid\mid r \mid\mid m_i, \sigma_i, k) = 1$. If all verifications pass, output 1.

However, the scheme is not secure. Given a signature on $m = (m_1 \mid\mid m_2)$, the adversary can output a valid signature on $m' = m_1$ (why?).

There is a simple fix: just include the message length in each of the strings that are signed. More formally, the secure construction looks as follows.

- $\mathsf{Sign}(m, k)$: Let $|m| = \ell \cdot n/4$ for some integer ℓ. Choose a random $n/4$-bit string r, and output $\Big(r, \mathsf{Sign}_{\mathrm{bdd}}(1 \mid\mid r \mid\mid \ell \mid\mid m_1, k), \ldots, \mathsf{Sign}_{\mathrm{bdd}}(\ell \mid\mid r \mid\mid \ell \mid\mid m_\ell, k)\Big)$.
- $\mathsf{Verify}(m = (m_1 \mid\mid \ldots \mid\mid m_\ell), \sigma = (r, \sigma_1, \ldots, \sigma_\ell), k)$: For each $i \in [\ell]$, check $\mathsf{Verify}_{\mathrm{bdd}}(i \mid\mid r \mid\mid \ell \mid\mid m_i, \sigma_i, k) = 1$. If all verifications pass, output 1.

Finally, we can compress the signature by computing the XOR of all these ℓ signature blocks. This will be one of the problems in the next assignment.

3.2 CBC-MAC for unbounded length messages

The CBC-MAC scheme is insecure if the message space is $(\{0,1\}^n)^*$. However, there are a few ways to make it secure.

- The signing key is a PRF key k. To sign a message with ℓ blocks, first compute $k_\ell = F(\ell, k)$. Then use k_ℓ as the key for $\mathsf{CBC\text{-}MAC}_\ell$ (described in Section 2.2 above).
- Given message $m = (m_1, \ldots, m_\ell)$, let $m' = (\ell, m_1, \ldots, m_\ell)$ be an $(\ell + 1)$-block message. Use $\mathsf{Sign}_{\ell+1}$ (described in Section 2.2) on message m'. Note that it is important to **prepend** the block-length. If the block-length is appended to the message, then the resulting MAC scheme is insecure (you will show this in Assignment 3).
- The signing key consists of two PRF keys k_1, k_2. To sign a message $m = (m_1, \ldots, m_\ell)$, compute $t = \mathsf{Sign}_\ell(m, k_1)$. Output $F(t, k_2)$. This is used in many real-world systems, and is referred to as 'encrypted' CBC-MAC, since $F(t, k_2)$ is seen as an encryption of the CBC-MAC output.

4 Lecture summary, plan for next lecture, additional resources

Summary: In this lecture, we first saw two MAC schemes for handling long-but-bounded length messages. Next, we discussed how to extend them to handle unbounded length messages.

Next lecture: Next lecture, we will discuss hash functions. This is a third approach for signing long messages: apply the hash function, and then sign the resulting digest.