Lecture 24: MAC for Arbitrary Length Messages

Previous lecture, we constructed MACs for fixed length messages

- The GGM Pseudo-random Function (PRF) Construction
  - **Given.** Pseudo-random Generator (PRG) $G\colon \{0,1\}^k \to \{0,1\}^{2k}$
  - **We Constructed.** PRF $F_{\mathsf{sk}}(m)$ using the GGM construction from the domain $\{0,1\}^n$ to the range $\{0,1\}^k$
- The MAC scheme was provided by $(\mathsf{Gen}, \mathsf{Mac}, \mathsf{Ver})$

  - $\mathsf{Gen}()$. Return $\mathsf{sk} \xleftarrow{\$} \{0,1\}^k$
  - $\mathsf{Mac}_{\mathsf{sk}}(m)$. Return $\tau = F_{\mathsf{sk}}(m)$
  - $\mathsf{Ver}_{\mathsf{sk}}(\widetilde{m}, \widetilde{\tau})$. Return $\widetilde{\tau} == F_{\mathsf{sk}}(\widetilde{m})$

- This construction is secure only for fixed-length messages
- We need different secret key for every length of the message. Otherwise, we can perform length-extension attacks

Goal of this Lecture

- Suppose we are given a MAC scheme $(\mathsf{Gen}', \mathsf{Mac}', \mathsf{Ver}')$ that is secure only for B-bit messages and generates tags of length k
- We want to construct a MAC scheme $(\mathsf{Gen}, \mathsf{Mac}, \mathsf{Ver})$ that is secure for arbitrary length messages

First Proposal I

The students proposed the following construction

$\mathsf{Gen}()$. Return $\mathsf{sk} = \mathsf{Gen}'()$

$\mathsf{Mac}_{\mathsf{sk}}(m)$.

1. Pad the message m so that the length of m is a multiple of B
2. Break m into n/B blocks $(m^{(1)}, m^{(2)}, \dots, m^{(n/B)})$ such that each block $m^{(i)}$ has size B
3. For $i \in \{1, 2, \dots, n/B\}$: Compute $\tau^{(i)} = \mathsf{Mac}'_{\mathsf{sk}}(m^{(i)})$
4. Return $\tau = (\tau^{(1)}, \tau^{(2)}, \dots, \tau^{(n/B)})$

$\mathsf{Ver}_{\mathsf{sk}}(\widetilde{,}\widetilde{\tau})$.

1. Let $\widetilde{m} = (\widetilde{m}^{(1)}, \widetilde{m}^{(2)}, \dots, \widetilde{m}^{(t)})$ (each block of length B)
2. Let $\widetilde{\tau} = (\widetilde{\tau}^{(1)}, \widetilde{\tau}^{(2)}, \dots, \widetilde{\tau}^{(t)})$ (each block of length k)
3. Return true if and only if for all $i \in \{1, \dots, t\}$, the expression $\mathsf{Ver}'_{\mathsf{sk}}(\widetilde{m}^{(i)}, \widetilde{\tau}^{(i)})$ is true

First Proposal II

Attacks on the Scheme. This scheme is insecure. The students proposed the following attacks.
Suppose the adversary sees the message $m = (m^{(1)}, m^{(2)}, \dots, m^{\left(n/B\right)})$ and the tag $\tau = (\tau^{(1)}, \tau^{(2)}, \dots, \tau^{\left(n/B\right)})$.

1. Permuting.
   The adversary can construct m' from m by arbitrarily permuting the blocks of the message m. The adversary can construct the corresponding τ' from τ by performing the same permutation of blocks on the tag τ.

First Proposal III

2. Shortening/Extending.
   The adversary can construct m' from m by dropping blocks from m. The adversary can construct the corresponding τ' from τ by dropping the tags for those blocks in the tag τ. The adversary can also construct m' from m by repeating some blocks of m. The adversary can construct the corresponding τ' from τ by repeating the tags for those blocks in the tag τ

First Proposal IV

3. Splicing.
   Suppose the adversary sees another message $\widehat{m} = (\widehat{m}^{(1)}, \widehat{m}^{(2)}, \dots, \widehat{m}^{\left(n/B\right)})$ and its tag $\widehat{\tau} = (\widehat{\tau}^{(1)}, \widehat{\tau}^{(2)}, \dots, \widehat{\tau}^{\left(n/B\right)})$
   The adversary can construct m' from the messages m and $\widehat{m}$ by splicing blocks. The adversary can construct the corresponding τ' from the tags τ and $\widehat{\tau}$ by splicing the corresponding blocks from the tags τ and $\widehat{\tau}$.
   For example, the tag of the message $(m^{(1)}, \widehat{m}^{(2)}, \dots, \widehat{m}^{\left(n/B\right)})$ is $(\tau^{(1)}, \widehat{\tau}^{(2)}, \dots, \widehat{\tau}^{\left(n/B\right)})$

First Proposal V

4. Combination of the attacks mentioned above.
   For example, the adversary can splice, extend, and permute!

Preventing the Attacks

Students proposed the following fixes.

- Prevent Permutation. We can include the information in each $\tau^{(i)}$ that it is the tag of the message-block at position i, then permutation attacks cannot be performed.
- Prevent Shortening/Extension. We can include the information in each $\tau^{(i)}$ that it is the tag of the message of total length n, then shortening/extension attacks cannot be performed.
- Prevent Splicing. We can include a time-stamp in each $\tau^{(i)}$ so that we cannot splice messages at two different time instances. (We shall use a *slightly different technique* but I wanted to summarize the proposals of the students)

Secure MAC Scheme for Arbitrary-length Messages I

- Gen(). Return $\mathsf{sk} = \mathsf{Gen}'()$

Secure MAC Scheme for Arbitrary-length Messages II

- $\mathsf{Mac}_{sk}(m)$.
  1. Pad m to make its length a multiple of $B/4$
  2. Break m into $4n/B$ blocks of size $B/4$ each. We represent this as $m = (m^{(1)}, m^{(2)}, \dots, m^{(4n/B)})$
  3. Pick $r \xleftarrow{\$} \{0,1\}^{B/4}$
  4. Let $[n]_2$ represent the $(B/4)$-bit representation of the number n in binary
  5. Let $[i]_2$ represent the $(B/4)$-bit representation of the number i in binary
  6. For each $i \in \{1, 2, \dots, 4n/B\}$ create the following block

$$m^{+(i)} = \Big(\overbrace{r}^{B/4\text{-bits}}, \overbrace{[n]_2}^{B/4\text{-bits}}, \overbrace{[i]_2}^{B/4\text{-bits}}, \overbrace{m^{(i)}}^{B/4\text{-bits}} \Big)$$

  7. For each $i \in \{1, 2, \dots, 4n/B\}$ create the tag $\tau^{(i)} = \mathsf{Mac}'_{sk}(m^{+(i)})$
  8. Return $\tau = (r, \tau^{(1)}, \tau^{(2)}, \dots, \tau^{(4n/B)})$

- $\mathsf{Ver}_{\mathsf{sk}}(\widetilde{m}, \widetilde{\tau})$.
  1. Interpret $\widetilde{m} = (\widetilde{m}^{(1)}, \widetilde{m}^{(2)}, \dots, \widetilde{m}^{(t)})$, where each block is of length B
  2. Interpret $\widetilde{\tau} = (\widetilde{r}, \widetilde{\tau}^{(1)}, \widetilde{\tau}^{(2)}, \dots, \widetilde{\tau}^{(t)})$, where $\widetilde{r}$ is of length $B/4$ and each block $\widetilde{\tau}^{(i)}$ is of length k
  3. Let $\widetilde{n} = t \times (B/4)$, the length of the message $\widetilde{m}$
  4. For each $i \in \{1, 2, \dots, t\}$ construct the following block

$$\widetilde{m}^{+(i)} = (\widetilde{r}, [\widetilde{n}]_2, [i]_2, \widetilde{m}^{(i)})$$

  5. Accept $(\widetilde{m}, \widetilde{\tau})$ if for each $i \in \{1, 2, \dots, t\}$ the test $\mathsf{Ver}'_{\mathsf{sk}}(\widetilde{m}^{+(i)}, \widetilde{\tau}^{(i)})$ accepts

Notes

- Note that we can assume that all random strings "r" are unique. Because, by birthday bound, we need to tag roughly $\sqrt{\left|2^{B/4}\right|} = 2^{B/8}$ messages before we can expect one collision. If we choose $B = 800$, then we will (with high probability) see unique "r" strings. This, effectively, serves as a "time stamp"
- Check that all the attacks that we discussed cannot be performed against this new MAC scheme $(\mathrm{Gen}, \mathrm{Mac}, \mathrm{Ver})$
- **Very Important.** We emphasize that "protecting against known attacks" does not imply security of a MAC scheme. We can formally prove the security of the MAC scheme that we have described above. The proof is beyond the scope of this course.