TD 5: MACs and CCA Security

Exercise 0. [Homework discussion]

Exercise 1. [*Malleability of CBC*]

Let c be the CBC encryption of some message $m \in \mathcal{X}^l$, where $\mathcal{X} := \{0,1\}^n$. You do not know m. Let $\Delta \in \mathcal{X}$. Show how to modify the ciphertext c to obtain a new ciphertext c' that decrypts to m', where $m'[0] = m[0] \oplus \Delta$, and $m'[i] = m[i]$ for $i = 1, \cdots, l-1$. That is, by modifying c appropriately, you can flip bits of your choice in the first block of the decryption of c without affecting any of the other blocks.

Exercise 2. [*MAC with verification oracle*]

In the notion of existential **strong** unforgeability under chosen-message attacks, the adversary is given access to a MAC generation oracle $\mathsf{Mac}(k, .)$.

At each message query m, the challenger computes $t \leftarrow \mathsf{Mac}(k, m)$, returns t and updates the set of MAC queries $Q := Q \cup \{(t, m)\}$, which is initialized to $Q := \emptyset$. At the end of the game, the adversary outputs a pair $(m^\star, t^\star)$ and wins if: (i) $\mathsf{Verify}(k, m^\star, t^\star) = 1$; and (ii) $(m^\star, t^\star) \notin Q$ [1]

We consider an even stronger definition where the adversary is additionally given access to a verification oracle $\mathsf{Verify}(k, ., .)$. At each verification query, the adversary chooses a pair (m, t) and the challenger returns the output of $\mathsf{Verify}(k, m, t) \in \{0, 1\}$. In this context, the adversary wins if one of these verification queries (m, t) satisfies: (i) $\mathsf{Verify}(k, m, t) = 1$; and (ii) $(m, t) \notin Q$

Show that the verification oracle does not make the adversary any stronger. Namely, any strongly unforgeable MAC remains strongly unforgeable when the adversary has a verification oracle.

Exercise 3. [*CCA Security*]

Recall the definition of CCA security given in the lecture. We define the scheme "Encrypt and tag" by: for a message m, independent keys k and k', a CPA-secure encryption Enc and a secure MAC $Sign$, we let $c = Enc(k, m)$ and $t = Sign(k', m)$, and return (c, t). Is this scheme CCA-secure?

Exercise 4. [*Authenticated Encryption*]

Consider the following construction of symmetric encryption.

Gen(1^λ)**:** Choose a random key $K_1 \leftarrow U(\{0,1\}^\lambda)$ for an IND-CPA secure symmetric encryption scheme $(\mathsf{Gen}', \mathsf{Enc}', \mathsf{Dec}')$. Choose a random key $K_0 \leftarrow U(\{0,1\}^\lambda)$ for a MAC $\Pi = (\mathsf{Gen}, \mathsf{Mac}, \mathsf{Verify})$. The secret key is $K = (K_0, K_1)$

Enc(K, M)**:** To encrypt M, do the following.

1. Compute $c = \mathsf{Enc}'(K_1, M)$.
2. Compute $t = \Pi.\mathsf{Mac}(K_0, c)$.

Return $C = (t, c)$.

Dec(K, C)**:** Return $\perp$ if $\Pi.\mathsf{Verify}(K_0, c, t) = 0$. Otherwise, return $M = \mathsf{Dec}'(K_1, c)$.

[1] In the definition of **standard** unforgeability under chosen-message attacks, condition (ii) is replaced by $\forall (m_i, t_i) \in Q$, $M^\star \neq m_i$.

1. Show that the scheme is not IND-CCA secure if the MAC Π is only unforgeable (i.e., not strongly) under chosen-message attacks.

2. Prove that the scheme is IND-CCA secure assuming that: (i) $(\mathsf{Gen}', \mathsf{Enc}', \mathsf{Dec}')$ is IND-CPA-secure; (ii) Π is strongly unforgeable under chosen-message attacks.

Exercise 5. [*CBC-MAC*]

Prove that the following modifications of CBC-MAC do not yield a secure fixed-length MAC:

1. Modify the following CBC-MAC (Figure 1) so that a random IV (rather than $IV = \mathbf{0}$) is used each time a tag is computed (and the IV is output along with t_ℓ).

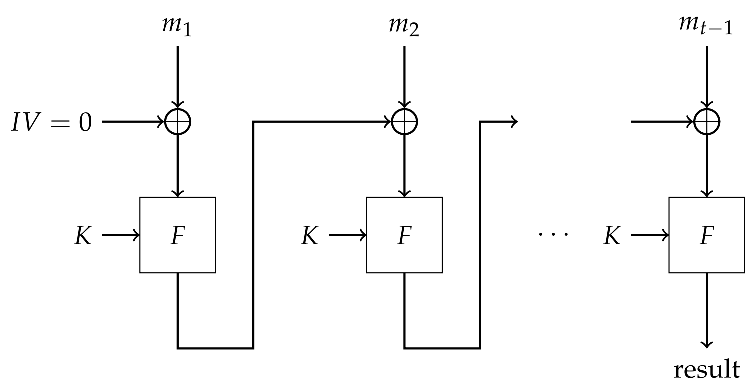

Figure 1: CBC-MAC

2. Modify CBC-MAC so that all the outputs of F are output, rather than just the last one.

We now consider the following ECBC-MAC scheme, let $F : K \times X \to X$ be a PRP, we define F_{ECBC} : $K^2 \times X^{\leq L} \to X$ as in Figure 2, where k_1 and k_2 are two independent keys.
If the message length is not a multiple of the block length n, we add a pad to the last block: $m = m_1 | \dots | m_{d-1} | (m_d \| \mathrm{pad}(m))$.

3. Show that there exists a padding for which this scheme is not secure.

For the security of the scheme, the padding must be invertible, and in particular for any message $m_0 \neq m_1$ we need to have $\mathrm{pad}(m_0) \neq \mathrm{pad}(m_1)$. The ISO norm is to pad with $10 \cdots 0$, and if the message length is a multiple of the block length, to add a new "dummy" block $10 \cdots 0$ of length n.

4. Explain why the scheme is not secure if this padding does not add a new block if the message length is a multiple of the block length.

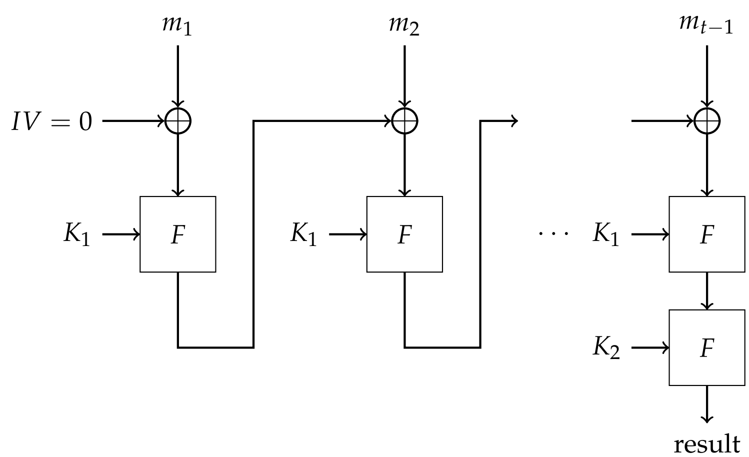

Figure 2: ECBC-MAC