TD 6: Authenticated Encryption

Exercise 1. [*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).

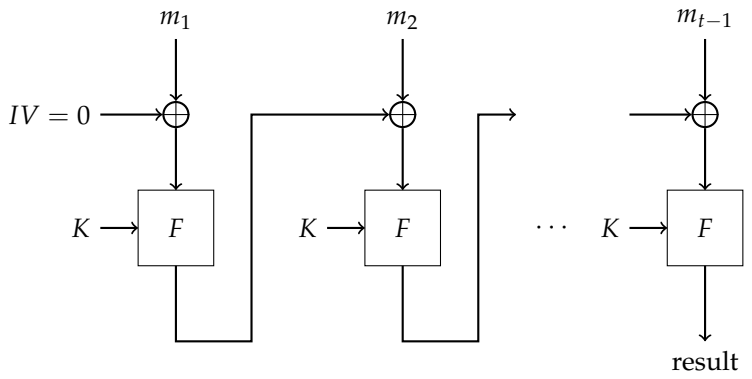

Figure 1: CBC-MAC

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: $m = m_1|\dots|m_{d-1}|(m_d\|\text{pad}(m))$.

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 $\text{pad}(m_0) \neq \text{pad}(m_1)$. The ISO norm is to pad with $10\cdots0$, and if the message length is a multiple of the block length, to add a new "dummy" block $10\cdots0$ 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.

Remark. The NIST standard is called CMAC, it is a variant of CBC-MAC with three keys (k, k_1, k_2). If the message length is not a multiple of the block length, then we append the ISO padding to it and then we also XOR this last block with the key k_1. If the message length is a multiple of the block length, then we XOR this last block with the key k_2. After that, we perform a last encryption with $F(k, .)$ to obtain the tag.

Exercise 2. [*Nested Encryption*]

Let (E, D) be an AE-secure cipher. Consider the following derived cipher (E', D'):

- $E'((k_1, k_2), m) := E(k_2, E(k_1, m))$

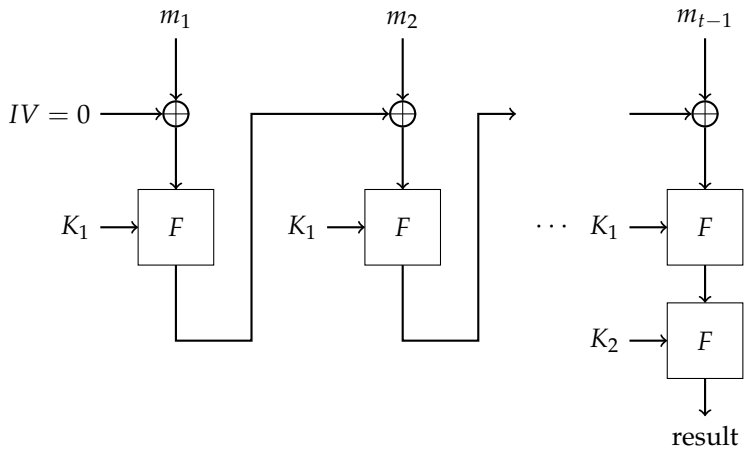

Figure 2: ECBC-MAC

- $D'((k_1, k_2), c) := \begin{cases} D(k_1, D(k_2, c)), & \text{if } D(k_2, c) \neq \text{ reject} \\ \text{reject}, & \text{otherwise} \end{cases}$

1. Show that (E', D') is AE-secure even if the adversary knows k_1, but not k_2.
2. Show that (E', D') is not AE-secure if the adversary knows k_2, but not k_1.
3. Design a cipher built from (E, D) where keys are pairs $(k_1, k_2) \in \mathcal{K}^2$ and the cipher remains AE-secure even if the adversary knows one of the keys, but not the other.

Exercise 3. [*Using independent keys*]

Let us see an example of a CPA-secure cipher and a secure MAC that are insecure when used in "Encrypt-then-MAC" when the same secret key k is used for both the cipher and the MAC. Let (E, D) be a block cipher defined over $(\mathcal{K}, \mathcal{X})$ where $\mathcal{X} = \{0,1\}^n$ and $|\mathcal{X}|$ is super-poly. Consider randomized CBC mode encryption built from (E, D) as the CPA-secure cipher for single block messages: an encryption of $m \in \mathcal{X}$ is the pair $c := (r, E(k, r \oplus m))$ where r is the random IV. Use RawCBC built from (E, D) as the secure MAC. This MAC is secure in this context because it is only being applied to fixed length messages (messages in $\mathcal{X}^2$): the tag on a ciphertext $c \in \mathcal{X}^2$ is $t := E(k, E(k, c[0]) \oplus c[1])$. Show that using the same key k for both the cipher and the MAC in "Encrypt-then-MAC" results in a cipher that does not provide authenticated encryption. Both CPA security and ciphertext integrity can be defeated.

Exercise 4. [*Plaintext integrity*]

Consider a weaker notion of integrity called **plaintext integrity**, or simply PI. The PI game is identical to the CI game except that the winning condition is relaxed to:

$$D(k, c) \neq \textit{reject}, \text{ and } D(k, c) \notin \{m_1, m_2, \cdots\}$$

Prove that the following holds:

1. Show that "MAC-then-Encrypt" is both CPA and PI secure.
2. Prove that a system that is CCA- and PI-secure is also AE-secure. The proof only needs a weak version of CCA, namely where the adversary issues a single decryption query and is told whether the ciphertext is accepted or rejected. Also, you may assume a super-poly-sized message space.