Introduction to cryptology
TD#4

2018-W11

Exercise 1: MACs

Q. 1: Let $\mathcal{M} : \{0,1\}^\kappa \times \{0,1\}^* \to \{0,1\}^\tau$ be a "perfect" MAC whose outputs are uniformly and independently random. An adversary is given a single message m and is asked to find the corresponding tag $\mathcal{M}(k, m)$ when k is unknown. What is his success probability (in function of κ and τ)?

Q. 2: Let $\mathcal{M}$ be as above, but with the constraint that it is linear. Give a universal forgery attack on $\mathcal{M}$ with small time and query complexity. Does your attack still work if $\mathcal{M}$ takes an additional "nonce" input r that is never reused from one call to another?

Q. 3: Let $\mathcal{M}$ be as in **Q. 1**. What is the problem with the following scheme

$$k_e, r, k_a, m \mapsto \mathsf{CBC\text{-}Encrypt}(k_e, r, m) || \mathcal{M}(k_a, m),$$

that combines encryption and authentication?

Exercise 2: MACs bis: CBC-MAC

We define a vanilla CBC-MAC with zero IV as $k, m \mapsto \lfloor \mathsf{CBC\text{-}Encrypt}(k, 0, m) \rfloor_{\text{last}}$, where $\lfloor \cdot \rfloor_{\text{last}}$ truncates its input to its last block (for the sake of simplicity, we assume that the input message always has a length multiple the block size).

Q. 1: Why is this scheme not secure?

Hint: Notice that the tag of a single-block message m_0 appears as intermediate value when computing the tag of $m_0||m_1$, for any value of m_1. If you know m_0 and its associated tag t, how can you pick m_1 to ensure that the two-block message $m_0||m_1$ also has tag t?

Q. 2: One proposes to solve the above issue by composing vanilla CBC-MAC with a one-block encryption $\mathcal{E}(k, \cdot)$ with a key k independent from the one used in vanilla CBC-MAC. Do you think that this makes sense?

Q. 3: Is it possible to extract a similar MAC scheme from the CTR mode?

Exercise 3: MACs ter: MAC with a small state

A designer wants to design a MAC using a block cipher $\mathcal{E} : \{0,1\}^{128} \times \{0,1\}^{32} \to \{0,1\}^{32}$. He wants to use a variant of CBC-MAC, but with larger tags than what a direct application using $\mathcal{E}$ would allow. Specifically, he wishes for 128-bit tags. The result is the following. On input $(k, k_0, k_1, k_2, k_3, m)$, compute:

$$x := \mathsf{CBC\text{-}Encrypt}[\mathcal{E}](k, 0, m) \quad y_0 := \mathcal{E}(k_0, x) \quad y_1 := \mathcal{E}(k_1, x) \quad y_2 := \mathcal{E}(k_2, x) \quad y_3 := \mathcal{E}(k_3, x),$$

and output $y := y_0||y_1||y_2||y_3$.

Q. 1: How many possible values can be taken by x (for any k, m)?

Q. 2: How many possible values can be taken by y, for a fixed MAC key (k, k_0, k_1, k_2, k_3)?

Q. 3: Give a strategy that allows to gather all possible tags for a fixed MAC key, with time, memory and query complexity 2^{32} (assuming for simplicity that if the input message is 32-bit long, no padding is performed in the CBC encryption).

Q. 4 Assuming that the precomputation of the previous question has been performed, what is the forgery probability for a random message? Is this MAC a good MAC?

Q. 5 Is the modified scheme that on input $(k, k_0, k_1, k_2, k_3, m)$ computes:

$$x := \mathsf{CBC\text{-}Encrypt}[\mathcal{E}](k, 0, m) \quad y_0 := \mathcal{E}(k_0, x) \quad y_1 := \mathcal{E}(k_1, y_0) \quad y_2 := \mathcal{E}(k_2, y_1) \quad y_3 := \mathcal{E}(k_3, y_2),$$

and outputs $y := y_0||y_1||y_2||y_3$ protected against the above attack?