Introduction to Cryptography: Homework 5

October 13, 2021

Requirements about the delivery of this assignment:

- Submit a pdf-document via Brightspace;
- Any additional files (e.g., Python scripts) can be added as well;
- Make sure that you write both name and student number on all documents (not only in the file name).

Deadline: Monday, October 25, 17:00 sharp!

Grading: You can score a total of 100 points for the hand-in assignments. To get full points, please **explain all answers clearly**.

Exercises:

1. **Length extension of CBC-MAC with padding.** In this exercise, we will explore how to make use of the length-extension property of CBC-MAC to construct a MAC forgery, even when padding is applied to messages. We assume here that the regular padding 10^* is used (a message that has a length that is a multiple of the block length will get padded with an additional block). The padding will be applied by the device that you can query. The used block cipher is AES. Throughout the exercise, assume that the input message is represented as a sequence of bits. We do not truncate the tag, so the tag consists of a string of 128 bits.

   (a) What is the minimal amount of padding?

   We query a message m_1 of 127 bits. We then write T_1 for the tag corresponding to that message. That is, $\text{CBC-MAC}_{\text{AES}_\text{K}}(m_1) = \text{AES}_K(m_1\|1) = T_1$. To perform the length extension attack as in the lecture, we then want the second block to look like $(m_1\|1) \oplus T_1$, after the padding has been applied to the message.

   (b) Explain how you can find a message m'_1 such that $(m'_1\|10^*) = (m_1\|1) \oplus T_1$.

   (c) Show how you can create a forgery for CBC-MAC (with padding) using a single generation query. [Hint: Use (b).]

2. **Merkle-Damgård properties.** Consider a hash function h (see Figure 1) based on the Merkle–Damgård mode and a compression function $\text{F}\colon \{0,1\}^{256} \to \{0,1\}^{128}$, where $\text{len}(M)$ is the encoding of the bit length of $M = M_1 \mid\mid M_2 \mid\mid \cdots \mid\mid M_k$.

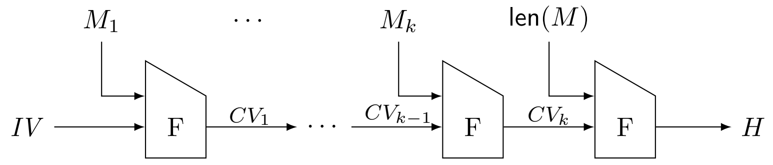

Figure 1: The hash function h.

   (a) Explain that, in practice, $\text{len}(M)$ always fits in one block.

   In the following two subquestions, we are going to show that if we can find a collision for h, i.e., find messages $M \neq M'$ such that $\text{h}(M) = \text{h}(M')$, we can find a collision for F. This shows that if F is collision resistant, that h is collision resistant as well. So, assume that $M \neq M'$ are such that $\text{h}(M) = \text{h}(M')$.

   (b) Assume first that $|M| \neq |M'|$. Give explicit expressions for a collision for F in terms of M and M'.

   (c) Assume now that $|M| = |M'|$. Explain, using the Merkle-Damgård construction of h, how we can construct a collision for F.

Hand in assignments:

1. **(50 points) Bad MAC design.** In this question, we are confronted with bad MAC design. We have a key K of length n bits. Let $\mathrm{B}_K \colon \{0,1\}^n \to \{0,1\}^n$ be a PRP-secure block cipher. Consider messages m of length $3n$ bits. Write $m = m_1 \| m_2 \| m_3$, where $|m_1| = |m_2| = |m_3| = n$. Consider the following two MAC-functions:

$$\mathrm{MAC1}_K(m) = \mathrm{B}_K(m_1) \| \mathrm{B}_K(\mathrm{B}_K(m_2)) \| \mathrm{B}_K(\mathrm{B}_K(m_3));$$
$$\mathrm{MAC2}_K(m) = (\mathrm{B}_K(\overline{m_1}) \oplus \mathrm{B}_K(m_2)) \| \mathrm{B}_K(m_1 \oplus \overline{m_2} \oplus m_3).$$

[The notation $\overline{m_1}$ is used for the complement of a bit-string. It can be read as $\overline{m_1} = m_1 \oplus 1^n$, here.]

   (a) Show that you can create a forgery for MAC1 using one generation query. [Give the message m for which you query the MAC generation (generation query). Then, create a forgery by giving a message-tag pair (m', T') with $m' \neq m$, and show that it verifies correctly, i.e., doing a verification query would yield 1.] 20 pt

   (b) Show that you can create a forgery for MAC2 using at most two generation queries. [It is possible to do it with just one, but with two might be easier.] 20 pt

   (c) Explain why your attack in (a) means that MAC1 is not PRF-secure. [Hint: Use (a) to create a distinguisher that shows that MAC1 is distinguishable from $\mathcal{RO}$.] 10 pt

2. **(50 points) Merkle-Damgård keyed hash function.** Consider the following MAC function built by keying a hash function that is based on the Merkle-Damgård construction:

$$\mathrm{MAC}_K(m) = \mathrm{h}(K \| m)$$

Let $\mathrm{h} \colon \{0,1\}^* \to \{0,1\}^{128}$ be a hash function based on the unpadded Merkle-Damgård mode[1] with compression function $\mathrm{F} \colon \{0,1\}^{128} \times \{0,1\}^{128} \to \{0,1\}^{128}$. Assume that the key K is of fixed length 128 bits.

   (a) Use the length-extension weakness of Merkle-Damgård to create a forgery, using one *generation* query on a message m_1 of 128 bits. Show that it verifies correctly, i.e., a verification query on this forgery would yield output 1. 10 pt

   (b) Explain why we cannot query F with (m', T') if $|m'| > 128$ and $|T'| = 128$. 10 pt

   (c) Suppose that you now obtain a real-life message-tag pair (m'_1, T_1) of $128n$ bits for some $n \geq 2$. Show that you can create a forgery for $m'_1 \| m'_2$ for MAC using an arbitrary length message $m'_2 = m_{2,0} \| m_{2,1} \| ... \| m_{2,n'-1}$ with $n' \geq 1$ and $|m'_{2,i}| = 128$ for all $i \in \{0, \ldots, n'-1\}$. [You can make offline queries to the compression function $\mathrm{F}(-)$ in the Merkle-Damgård construction. If you do, clearly state how many and which you need.] 20 pt

   (d) Explain why it is not secure to use this MAC function as a key derivation function (see slide 7 in slides_6_hashing). 10 pt

[1] Technically the domain of h then is not $\{0,1\}^*$, but instead $\bigcup_{n=0}^{\infty} \{0,1\}^{128n}$.