Information Systems Security
Series 6 - Hash Functions and MACs

November 24th, 2021

Reminder : Hash Functions

A hash function $h : X \to Y$ has to be deterministic and easy to compute. But in order to be an effective cryptographic hash function, it also needs these properties :

- **Preimage resistance** : Given y, it is practically impossible to compute x such that $h(x) = y$.
- **Second preimage resistance** : Given $x \in X$ and $y \in Y$ such that $h(x) = y$, it is practically impossible to compute $x' \neq x$ such that $h(x') = h(x) = y$. This is also called "weak collision resistance".
- **Collision resistance** : It is practically impossible to compute two distinct $x, x' \in X$ such that $h(x) = h(x')$. This is also called "strong collision resistance".

Note that the collision resistance implies the second preimage resistance, but not the preimage resistance.

Exercise 1 : Hash Functions

For each following hash function, prove if they respect the three previous properties or not :

- $h_1(x) = x \mod n$, n a big prime number.
- $h_2(x) = x^d \mod n$, n a big prime number, d a big exponent.
- We have $x = x_1 ... x_n$, with each x_i as one byte. Let $+$ be the addition bit by bit, modulo 2, i.e. an xor. We define the hash function as $h_3(x) = x_1 + ... + x_n$.

- Let x and $+$ be defined the same way as previously, and $*$ be the multiplication by 4 bit blocks, modulo 16, i.e. to multiply a byte by a number, we multiply separately each half of 4 bits :
  For example, $5 * (7A)_{16} = (5 * (7)_{16} \mod 16) \parallel (5 * (A)_{16} \mod 16) = 35 \mod 16 \parallel 50 \mod 16 = (32)_{16}$.
  We define the hash function as $h_4(x) = n * x_1 + (n-1) * x_2 + ... + 1 * x_n$.

Exercise 2 : Message Authentication Codes

We are using a block cipher in CBC mode to create a MAC, with IV=0.

- Let the MAC be defined as follows :

$$\begin{cases} t_1 = E_k(m_1) \\ t_{i+1} = E_k(m_{i+1} \oplus t_i) \end{cases}$$

  You are given one pair (message, MAC) with one message of two blocks $m = m_1 \parallel m_2$, and the corresponding MAC $t = t_1 \parallel t_2$. Given these, show that you can build a falsified pair, i.e. you can create another message m' and falsify its MAC, without knowing the key.

- Now, let $M = m_1 \parallel m_2 \parallel ... \parallel m_n$ a message of n blocs, and $C = c_1 \parallel c_2 \parallel ... \parallel c_n$ the CBC block cipher computed as follows :

$$\begin{cases} c_1 = E_k(m_1) \\ c_{i+1} = E_k(m_{i+1} \oplus c_i) \end{cases}$$

  And let the corresponding MAC be defined as (with the same key k) :

$$MAC = E_k(m_n \oplus E_k(m_{n-1} \oplus E_k(... \oplus E_k(m_2 \oplus E_k(m_1))...)))$$

  1. You are given a message m = $m_1 \parallel ... \parallel m_n$, and the corresponding ciphers and MAC. Is it possible to falsify a MAC the same way as previously, i.e. finding a message m' such that you can build the MAC without the key ?
  2. Now, suppose you're intercepting and encrypted message with the MAC. You don't know the message, only the cipher and MAC. Can you modify the ciphers while conserving a valid MAC ?
  3. How would you modify this protocol in order to ensure integrity ?
  4. If we used this protocol as a hashing function, would we have the collision resistance property ?