Introduction to Cryptology

Lecture 13

Announcements

- Homework 6 up on course webpage, due on Thursday, 4/2.

Agenda

- Last time:
  - CCA Security (3.7)
  - New topic: Message Integrity (4.1)
  - Message Authentication Codes (MAC) (4.2)

- This time:
  - More MAC definitions (4.2)
  - Constructing a fixed-length MAC (4.3)
  - Domain extension with CBC-MAC (4.4)
  - Authenticated Encryption (4.5)

Security of MACs

The message authentication experiment $MACforge_{A,\Pi}(n)$:

1. A key k is generated by running $Gen(1^n)$.
2. The adversary A is given input 1^n and oracle access to $Mac_k(\cdot)$. The adversary eventually outputs (m, t). Let Q denote the set of all queries that A asked its oracle.
3. A succeeds if and only if (1) $Vrfy_k(m, t) = 1$ and (2) $m \notin Q$. In that case, the output of the experiment is defined to be 1.

Security of MACs

Definition: A message authentication code $\Pi = (Gen, Mac, Vrfy)$ is existentially unforgeable under an adaptive chosen message attack if for all probabilistic polynomial-time adversaries A, there is a negligible function neg such that:

$$\Pr\left[MACforge_{A,\Pi}(n) = 1\right] \leq neg(n)\,.$$

Strong MACs

The strong message authentication experiment $MACsforge_{A,\Pi}(n)$:

1. A key k is generated by running $Gen(1^n)$.
2. The adversary A is given input 1^n and oracle access to $Mac_k(\cdot)$. The adversary eventually outputs (m, t). Let Q denote the set of all pairs (m, t) that A asked its oracle.
3. A succeeds if and only if (1) $Vrfy_k(m, t) = 1$ and (2) $(m, t) \notin Q$. In that case, the output of the experiment is defined to be 1.

Strong MACs

Definition: A message authentication code $\Pi = (Gen, Mac, Vrfy)$ is a strong MAC if for all probabilistic polynomial-time adversaries A, there is a negligible function neg such that:

$$\Pr\left[MACsforge_{A,\Pi}(n) = 1\right] \leq neg(n).$$

Constructing Secure Message Authentication Codes

A Fixed-Length MAC

Let F be a pseudorandom function. Define a fixed-length MAC for messages of length n as follows:

- Mac: on input a key $k \in \{0,1\}^n$ and a message $m \in \{0,1\}^n$, output the tag $t := F_k(m)$.
- $Vrfy$: on input a key $k \in \{0,1\}^n$, a message $m \in \{0,1\}^n$, and a tag $t \in \{0,1\}^n$, output 1 if and only if $t = F_k(m)$.

Security Analysis

Theorem: If F is a pseudorandom function, then the construction above is a secure fixed-length MAC for messages of length n.

Security Analysis

Let A be a ppt adversary trying to break the security of the construction. We construct a distinguisher D that uses A as a subroutine to break the security of the PRF.

Distinguisher D:

D gets oracle access to oracle O, which is either F_k, where F is pseudorandom or f which is truly random.

1. Instantiate $A^{Mac_k(\cdot)}(1^n)$.
2. When A queries its oracle with message m, output $O(m)$.
3. Eventually, A outputs (m^*, t^*) where $m^*, t^* \in \{0,1\}^n$.
4. If $m^* \in Q$, output 0.
5. If $m^* \notin Q$, query $O(m^*)$ to obtain output z^*.
6. If $t^* = z^*$ output 1. Otherwise, output 0.

Security Analysis

Consider the probability D outputs 1 in the case that O is truly random function f vs. O is a pseudorandom function F_k.

- When O is pseudorandom, D outputs 1 with probability $\Pr\left[MACforge_{A,\Pi}(n) = 1\right] = \rho(n)$, where ρ is non-negligible.
- When O is random, D outputs 1 with probability at most $\frac{1}{2^n}$. Why?

Security Analysis

D's distinguishing probability is:

$$\left|\frac{1}{2^n} - \rho(n)\right| = \rho(n) - \frac{1}{2^n}.$$

Since, $\frac{1}{2^n}$ is negligible and $\rho(n)$ is non-negligible,

$\rho(n) - \frac{1}{2^n}$ is non-negligible.

This is a contradiction to the security of the PRF.

Domain Extension for MACs

CBC-MAC

Let F be a pseudorandom function, and fix a length function ℓ. The basic CBC-MAC construction is as follows:

- Mac: on input a key $k \in \{0,1\}^n$ and a message m of length $\ell(n) \cdot n$, do the following:
  1. Parse m as $m = m_1, \ldots, m_\ell$ where each m_i is of length n.
  2. Set $t_0 := 0^n$. Then, for $i = 1$ *to* ℓ:

     Set $t_i := F_k(t_{i-1} \oplus m_i)$.

  Output t_ℓ as the tag.
- $Vrfy$: on input a key $k \in \{0,1\}^n$, a message m, and a tag t, do: If m is not of length $\ell(n) \cdot n$ then output 0. Otherwise, output 1 if and only if $t = Mac_k(m)$.

CBC-MAC

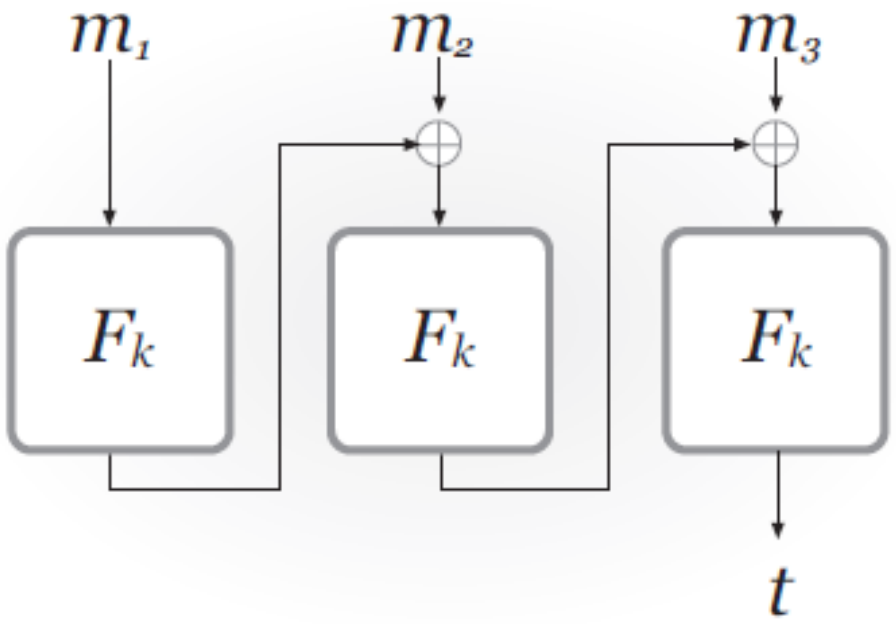

FIGURE 4.1: Basic CBC-MAC (for fixed-length messages).