Cryptography

Lecture 10

Announcements

- HW4 is up due Monday, 3/6

Agenda

- Last time:
  - CPA secure encryption from PRF (K/L 3.5)
  - Block Ciphers (K/L 3.5)
  - Modes of Operation (K/L 3.6)
    - Please read about Counter Mode on your own
- This time:
  - Introduction to MACs
  - Security Definition for MAC (K/L 4.2)
  - Constructing MAC from PRF (K/L 4.3)
  - Domain Extension for MACs (K/L 4.4)

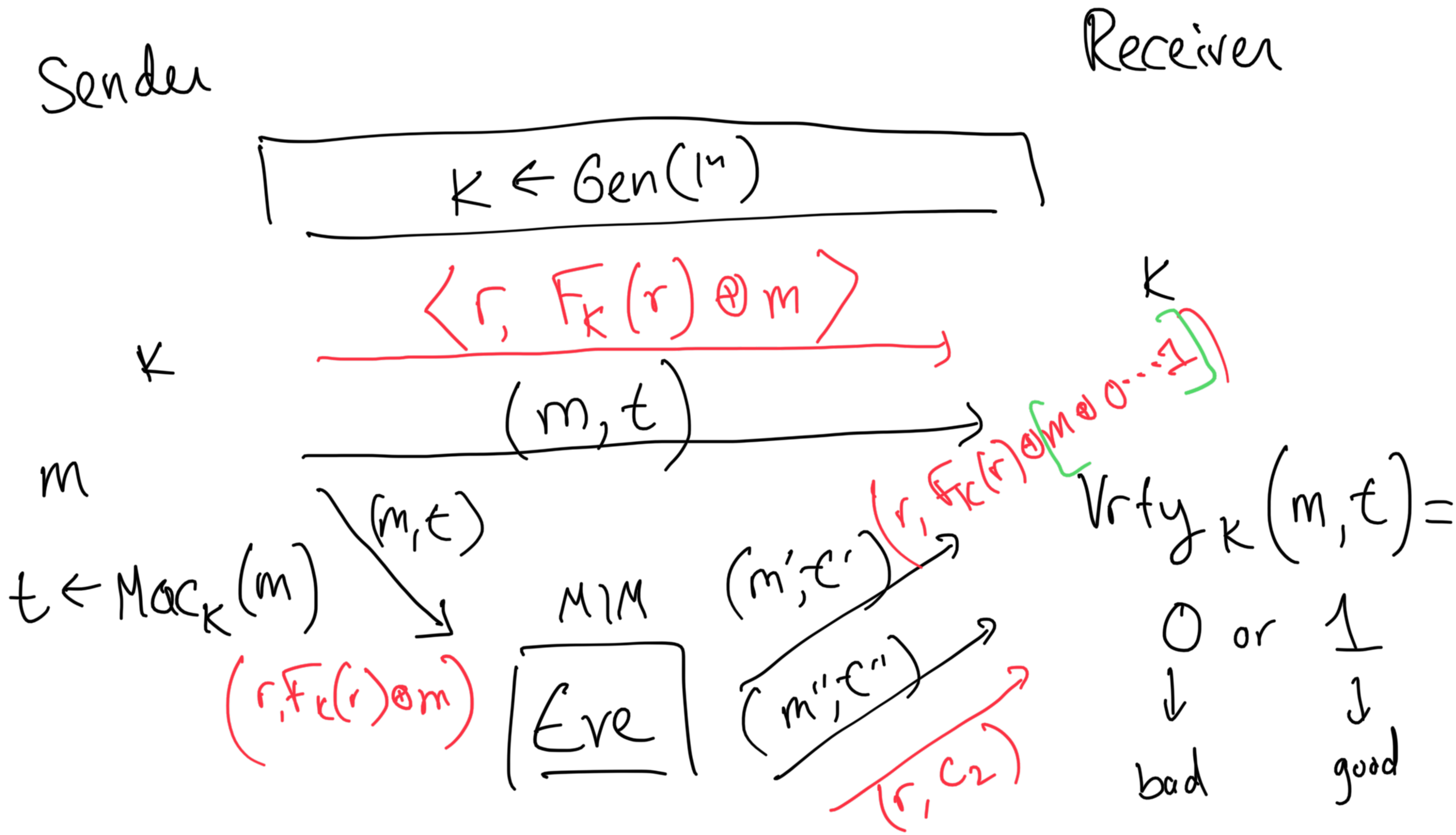
Agenda

- Last time:
  - PRF Class Exercise
  - Block Ciphers (K/L 3.5)
  - Modes of Operation (K/L 3.6)
- This time:
  - Introduction to MACs
  - Security Definition for MAC (K/L 4.2)
  - Constructing MAC from PRF (K/L 4.3)
  - Begin Discussing Domain Extension for MACs (K/L 4.4)
  - Class Exercise

MACs

Message Integrity

- Secrecy vs. Integrity
- Encryption vs. Message Authentication



Correctness: $\mathrm{Vrfy}_k(m, t \leftarrow \mathrm{Mac}_k(m)) = 1$

Message Authentication Codes

Definition: A message authentication code (MAC) consists of three probabilistic polynomial-time algorithms $(Gen, Mac, Vrfy)$ such that:

1. The key-generation algorithm Gen takes as input the security parameter 1^n and outputs a key k with $|k| \geq n$.
2. The tag-generation algorithm Mac takes as input a key k and a message $m \in \{0,1\}^*$, and outputs a tag t. $t \leftarrow Mac_k(m)$.
3. The deterministic verification algorithm $Vrfy$ takes as input a key k, a message m, and a tag t. It outputs a bit b with $b = 1$ meaning valid and $b = 0$ meaning invalid. $b \coloneqq Vrfy_k(m, t)$.

It is required that for every n, every key k output by $Gen(1^n)$, and every $m \in \{0,1\}^*$, it holds that $Vrfy_k\big(m, Mac_k(m)\big) = 1$.

Unforgeability for MACs

Consider a message authentication code $\Pi = (Gen, Mac, Vrfy)$, any adversary A, and any value n for the security parameter.

Experiment $MACforge_{A,\Pi}(n)$

Adversary $A(1^n)$

Challenger

Unforgeability for MACs

Consider a message authentication code $\Pi = (Gen, Mac, Vrfy)$, any adversary A, and any value n for the security parameter.

Experiment $MACforge_{A,\Pi}(n)$

Adversary $A(1^n)$

Challenger

$k \leftarrow Gen(1^n)$

Unforgeability for MACs

Consider a message authentication code $\Pi = (Gen, Mac, Vrfy)$, any adversary A, and any value n for the security parameter.

Experiment $MACforge_{A,\Pi}(n)$

Adversary $A(1^n)$

Challenger

$k \leftarrow Gen(1^n)$

$A^{Mac_k(\cdot)}$

m' →

← t'

⋮

Unforgeability for MACs

Consider a message authentication code $\Pi = (Gen, Mac, Vrfy)$, any adversary A, and any value n for the security parameter.

Experiment $MACforge_{A,\Pi}(n)$

Adversary $A(1^n)$

Challenger

$k \leftarrow Gen(1^n)$

$A^{Mac_k(\cdot)}$

m' →

← t'

⋮

Q is the set of all messages m' queried by A

Unforgeability for MACs

Consider a message authentication code $\Pi = (Gen, Mac, Vrfy)$, any adversary A, and any value n for the security parameter.

Experiment $MACforge_{A,\Pi}(n)$

Adversary $A(1^n)$

Challenger

$k \leftarrow Gen(1^n)$

$A^{Mac_k(\cdot)}$

m' →

t' ←

⋮

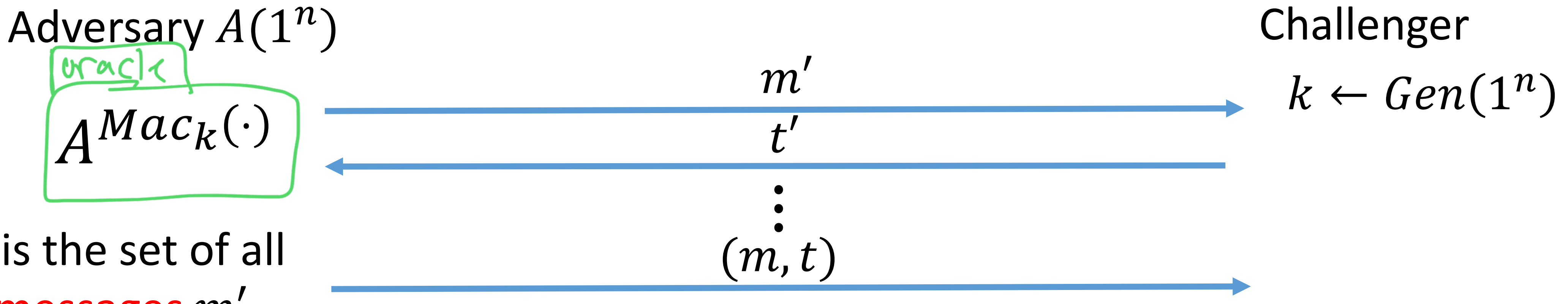
(m, t) →

Q is the set of all messages m' queried by A

Unforgeability for MACs

Consider a message authentication code $\Pi = (Gen, Mac, Vrfy)$, any adversary A, and any value n for the security parameter.

Experiment $MACforge_{A,\Pi}(n)$



Q is the set of all messages m' queried by A

$MACforge_{A,\Pi}(n) = 1$ if both of the following hold:

1. $m \notin Q$
2. $Vrfy_k(m, t) = 1$

Otherwise, $MACforge_{A,\Pi}(n) = 0$

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 Unforgeability for MACs

Consider a message authentication code $\Pi = (Gen, Mac, Vrfy)$, any adversary A, and any value n for the security parameter.

Experiment $MACsforge_{A,\Pi}(n)$

Adversary $A(1^n)$

Challenger

$k \leftarrow Gen(1^n)$

$A^{Mac_k(\cdot)}$

m'

t'

⋮

(m, t)

Q is the set of all message, tag pairs (m', t') queried/received by A

$MACsforge_{A,\Pi}(n) = 1$ if both of the following hold:

1. $(m, t) \notin Q$
2. $Vrfy_k(m, t) = 1$

Otherwise, $MACsforge_{A,\Pi}(n) = 0$

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:

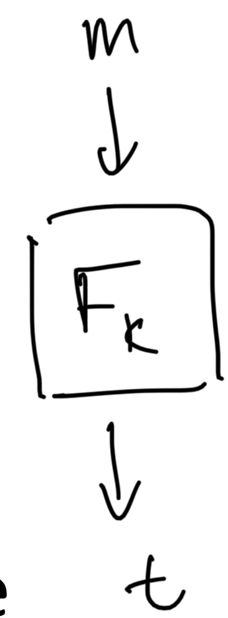
$$\Pr\left[MACsforge_{A,\Pi}(n) = 1\right] \leq neg(n).$$

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: Gen: output a key $K \in^R \{0,1\}^n$.

- 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.

High level: Assume Mac is not secure

Build a distinguisher against PRF.

Mac not secure: $\exists$ a ppt A st $\Pr[\mathrm{MacForge}_{A,\Pi}(n)=1] \geq \rho(n)$ non-negl

Prf not secure: $\exists$ a ppt D st.

$|\Pr[D^{F_k(\cdot)}(1^n)=1] - \Pr[D^{f(\cdot)}(1^n)=1]| \geq \rho'(n)$

Pseudorandom Function

Definition: Let $F: \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ be an efficient, length-preserving, keyed function. We say that F is a pseudorandom function if for all ppt distinguishers D, there exists a negligible function $negl$ such that:

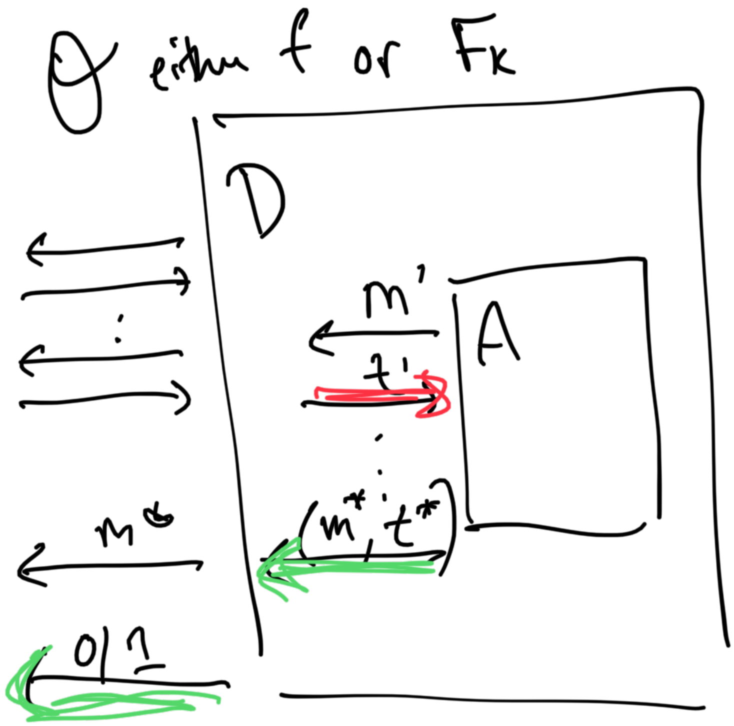
$$\left|\Pr\left[D^{F_k(\cdot)}(1^n) = 1\right] - \Pr\left[D^{f(\cdot)}(1^n) = 1\right]\right|$$
$$\leq negl(n).$$

where $k \leftarrow \{0,1\}^n$ is chosen uniformly at random and f is chosen uniformly at random from the set of all functions mapping n-bit strings to n-bit strings.

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:

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



To specify D.

1. How to answer Mac queries

(a) recive m'

(b) respond with $t' = \mathcal{O}(m')$

2. How to decide whether to output 0/1 given (m^*, t^*)

1. Check $m^* \notin Q$

   if $m^* \in Q$ output 0

2. Check $\mathcal{O}(m^*) \stackrel{?}{=} t^*$

   if yes output 1

   o/w output 0.

$$\Pr\left[D^{F_k(\cdot)}(1^n) = 1\right] = \Pr\left[\text{MacForge}_{A,\Pi}(n) = 1\right]$$

by assumption

$$\geq \varepsilon(n) \quad \text{where } \varepsilon \text{ is non-negl}$$

$$\Pr\left[D^{f(\cdot)}(1^n) = 1\right] \leq \frac{1}{2^n}$$

b/c when $m^* \notin Q$, f is undefined at m^* until the moment the query is made

$$\left| \varepsilon(n) - \underbrace{\frac{1}{2^n}}_{\text{negl}} \right| \geq \varepsilon'(n) \quad \text{where } \varepsilon' \text{ is non-negl.}$$

☑

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