Introduction to Modern Cryptography

Michele Ciampi

(Slides courtesy of Prof. Jonathan Katz)

Lecture 10, Part 2
Message Authentication Code (MAC)
So Far

Last lecture

- Introduced **message integrity**
- Introduced **message authentication codes (MAC)**

This lecture

MAC algorithms and proof of security
A Fixed-length MAC: Intuition

We need a keyed function Mac such that:

▶ Given \( \text{Mac}_k(m_1), \text{Mac}_k(m_2), \ldots \)
▶ ...it is infeasible to predict the value \( \text{Mac}_k(m) \) for any \( m \not\in \{m_1, \ldots \} \)

PRF

Let \( f \) be PRF. Knowledge of \( f(x_1), f(x_2), \ldots \) does not reveal any information on \( f(x): x \not\in \{x_1, x_2, \ldots \} \).

Idea

Let Mac be a PRF i.e. set \( \text{Mac}_k \equiv F_k \)
A Fixed-length MAC Construction

<table>
<thead>
<tr>
<th>Fixed-length MAC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Let $F$ be a length-preserving PRF (i.e. block cipher). Construct the following MAC $\Pi$:</td>
</tr>
<tr>
<td>▶ Gen: choose a uniform key $k$ for $F$</td>
</tr>
<tr>
<td>▶ $\text{Mac}_k(m)$: output $F_k(m)$</td>
</tr>
<tr>
<td>▶ $\text{Vrfy}_k(m, t)$: output 1 iff $F_k(m) = t$</td>
</tr>
</tbody>
</table>
A Fixed-length MAC Construction

<table>
<thead>
<tr>
<th>Theorem</th>
</tr>
</thead>
<tbody>
<tr>
<td>( F ) is a PRF ( \implies \Pi ) is a secure MAC</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Proof</th>
</tr>
</thead>
<tbody>
<tr>
<td>By reduction</td>
</tr>
</tbody>
</table>
Proof by Reduction
Proof by Reduction (see CPA-security)

High level

- Attacker $A$ attacks MAC $\Pi$ (as was defined)
- Design distinguisher $D$ that uses $A$ as a subroutine to attack the PRF $F$
  - i.e. $D$ tries to distinguish $F$ from a random function (RF)
- $D$ simulates to $A$ the steps in the $\text{Forge}_{A,\Pi}(n)$ experiment for $F$ and for a RF
- Relate the success $\Pr$ of $A$ to the success $\Pr$ of $D$
- If $A$ succeeds $\implies D$ succeeds $\implies F \neq \text{PRF}$
- contradicts $F \text{ PRF} \implies A$ can not succeed $\implies \Pi$ is a secure MAC
The $\text{Forge}_{A,\Pi}(n)$ Experiment (Recall)

Fix $A, \Pi$. Define randomized experiment $\text{Forge}_{A,\Pi}(n)$:

$\begin{align*}
\text{▶ } & k \leftarrow \text{Gen}(1^n) \\
\text{▶ } & A \text{ interacts with an oracle } \text{Mac}_k(\cdot):
\text{▶ } & A \text{ submits } m_1, \ldots, m_i \text{ to } \text{Mac}_k(\cdot) \\
\text{▶ } & A \text{ collects back } t_1, \ldots, t_i \text{ from } \text{Mac}_k(\cdot) \\
\text{▶ } & \text{Let } M = \{m_1, \ldots, m_i\} \text{ be the set of messages submitted to the oracle} \\
\text{▶ } & A \text{ outputs } (m, t) \\
\text{▶ } & A \text{ succeeds, and the experiment evaluates to } 1, \text{ if } \\
\text{Vrfy}_k(m, t) = 1 \text{ and } m \notin M
\end{align*}$

$\Pi$ is secure if $\forall \text{ PPT } A, \exists \epsilon \text{ negl. such that}$

$\Pr[\text{Forge}_{A,\Pi}(n) = 1] \leq \epsilon(n)$
Proof by Reduction (in Picture)

A attacks the MAC Π
Proof by Reduction (in Picture)

$D$ uses $A$ as a subroutine in distinguishing between RF $f$ and PRF $F_k$ for uniform $k$. 
Proof by Reduction (in Picture)

A requests the tag on message \( m_1 \)
Proof by Reduction (in Picture)

\[ D \text{ forwards } m_1 \text{ to the oracle } \mathcal{O} \in \{f, F_k\} \]
Proof by Reduction (in Picture)

$D$ gets back $t_1 = \mathcal{O}(m_1)$
Proof by Reduction (in Picture)

$D$ forwards $t_1 = \mathcal{O}(m_1)$ to $A$. From the perspective of $A$, $t_1$ is the tag of $m_1$
Proof by Reduction (in Picture)

PRF/random

A

D
Proof by Reduction (in Picture)
Proof by Reduction (in Picture)
Proof by Reduction (in Picture)
Proof by Reduction (in Picture)
Proof by Reduction (in Picture)

A outputs its forgery \((m, t)\): \(m \notin \{m_1, m_2 \ldots\}\), \(t\) – tag for \(m\)
Proof by Reduction (in Picture)

\[ D \text{ forwards } m \text{ to the oracle } O \in \{ f, F_k \} \]
Proof by Reduction (in Picture)

\[ D \text{ gets back } t^* = \mathcal{O}(m) \]
Proof by Reduction (in Picture)

If $t^* = t \implies D$ outputs 1; otherwise 0;
Proof by Reduction

The Simulation

$D$ simulates $\text{Forge}_{A,\Pi}(n)$ for $A$ with $f$–RF or $f$–PRF:

1. $A$ submits $m_i : i = 1, 2 \ldots$ to the MAC $O$
2. $D$ simulates the interaction with the MAC $O$ for $A$:
   - $D$ forwards $m_i$ to $f$; receives $t_i = f(m_i)$
   - $D$ returns $t_i$ to $A$
3. $A$ outputs $(m, t)$; $m \notin \{m_1, m_2, \ldots\}$
4. $D$ forwards $m$ to $f$; receives $t^* = f(m)$
5. If $t^* = t \implies D$ outputs 1 (success); otherwise 0 (fail)
World 0: \(D\) with a Truly Random Function \(f\)

\(D^f\) simulates \(\text{Forge}_{A,\Pi}(n)\) for \(A\) with truly random \(f\)

- By definition of RF observing \(f(m_1), f(m_2), \ldots\) does not reveal information on \(f(m) : m \not\in \{m_1, m_2, \ldots\}\)

- Therefore

\[
\Pr[D^f(\cdot) = 1] = \Pr[f(m) = t] = \Pr[t^* = t] = 2^{-n}
\]

where \(n = |m|\)
World 1: $D$ with a Pseudo-random Function $f = F_k$

<table>
<thead>
<tr>
<th>$D^F_k$ simulates $\text{Forge}_{A,\Pi}(n)$ for $A$ with truly random $F_k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>► The view of $A$ in this case is <strong>exactly</strong> as in the $\text{Forge}_{A,\Pi}(n)$ experiment</td>
</tr>
<tr>
<td>► Therefore</td>
</tr>
</tbody>
</table>

$$\Pr[D^F_k(\cdot) = 1] = \Pr[\text{Forge}_{A,\Pi}(n) = 1]$$
The Reduction

Proof.

By the assumption that $F$ is a PRF $\exists \epsilon(n) = \text{negl}$:

$$|\Pr_{k \leftarrow \{0,1\}^n}[D^{F_k}(\cdot) = 1] - \Pr_{f \leftarrow \mathcal{F}_n}[D^f(\cdot) = 1]| \leq \epsilon(n)$$

By the simulation of $\text{Forge}_{A,\Pi}(n)$ by $D^f$ with RF:

$$\Pr_{f \leftarrow \mathcal{F}_n}[D^f(\cdot) = 1] = \Pr[f(m) = t] = 2^{-n}$$

By the simulation of $\text{Forge}_{A,\Pi}(n)$ by $D^{F_k}$ with PRF:

$$\Pr_{k \leftarrow \{0,1\}^n}[D^{F_k}(\cdot) = 1] = \Pr[\text{Forge}_{A,\Pi}(n) = 1]$$

Therefore

$$\Pr[\text{Forge}_{A,\Pi}(n) = 1] \leq \epsilon(n) + 2^{-n} = \text{negl}(n)$$

$\implies$ $\Pi$ is a secure MAC
Limitations of the MAC $\Pi$

- Block ciphers (i.e. PRFs) have short, fixed-length block size
- e.g. AES has a 128-bit block size (shorter than a tweet!)
- Therefore $\Pi$ is limited to authenticating only short, fixed-length messages
- In practise we want to be able to send messages much longer than 128 bits
- We also want to be able to send messages of different (i.e. not fixed) length
- A solution: CBC-MAC (next)
Variable-length MAC

Suggestion

Can you construct a secure MAC for variable-length messages from a MAC for fixed-length messages?

Idea

\[
\text{Mac}'_k(m_1 \ldots m_l) = \text{Mac}_k(m_1) \ldots \text{Mac}_k(m_l)
\]

\[
\text{Vrfy}'_k(m_1 \ldots m_l, t_1 \ldots t_l) = 1 \iff \forall i : \text{Vrfy}_k(m_i, t_i) = 1
\]

Is this secure?
A Construction

Problem

Need to prevent (at least):
- Block reordering
- Truncation
- Mixing-and-matching blocks from multiple messages

One solution

\[ \text{Mac}'_k(m_1 \ldots m_l) = r, \text{Mac}_k(r | l | 1 | m_1), \text{Mac}_k(r | l | 2 | m_2), \ldots \]

Not very efficient – can we do better? Yes: CBC-MAC.
Basic CBC-MAC

\[ \mathbf{m}_1 \xrightarrow{F_k} \mathbf{m}_2 \oplus \cdots \oplus \mathbf{m}_l \xrightarrow{F_k} \mathbf{t} \]
CBC-MAC vs. CBC-mode

- CBC-MAC is deterministic (no IV)
  - MACs do not need to be randomized to be secure
  - Verification is done by re-computing the result
- In CBC-MAC, only the final value is output
- Both are essential for security
Theorem

If $F$ is a length-preserving PRF with input length $n$, then for any fixed $l$ basic CBC-MAC is a secure MAC for messages of length $ln$.

Proof

By reduction (omitted)

Note

- The sender and receiver must agree on the length parameter $l$ in advance.
- Basic CBC-MAC is not secure if this is not done!
## CBC-MAC for Variable Length Messages

### Method 1
Prepend the message with its block length \( l \)

### Method 2
- Apply \( F_k \) to the block length \( l \) to obtain key \( k_l \)
- Compute the tag with Basic CBC-MAC and key \( k_l \)
- Send \((t, l)\)

### Method 3
- Choose two keys \( k_1 \leftarrow \{0, 1\}^n \), \( k_2 \leftarrow \{0, 1\}^n \)
- Compute \( t_1 \) with Basic CBC-MAC using key \( k_1 \)
- Compute final tag using \( k_2 \) as \( t = F_{k_2}(t_1) \)
CBC-MAC for Variable Length Messages: Method 1

Prepend the message with its block length $l$
Hash Functions

<table>
<thead>
<tr>
<th>Hash functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Another way for constructing MACs for variable length messages</td>
</tr>
</tbody>
</table>

⇒ next lecture
End

References: Sec. 4.3 (not Theorem 4.8) and Sec 4.4.1