Message Authentication Code

Viet Tung Hoang

The slides are loosely based on those of Prof. Mihir Bellare, UC San Diego.
Agenda

1. MAC and Authenticity

2. MAC Constructions

3. How to Construct Good MAC
The Need for Authenticity

Transfer $5 to account 12345

Transfer $1000 to account 99999

Classical encryptions (CTR, CBC) don’t provide authenticity
MAC Syntax

**Key Gen**

\[ K \rightarrow \$ \rightarrow K \]

**MAC**

\[ M \rightarrow T \rightarrow T \]

Tag has fixed (short) length

\[ K \rightarrow T \rightarrow T \]

**Verify**

\[ M, T \rightarrow V \rightarrow 0 \text{ or } 1 \]

Canonical implementation:

Return \( T = T_K(M) \)
MAC Usage

\[ T \leftarrow \mathcal{T}_K(M) \]

\[ b \leftarrow \mathcal{V}_K(M', T') \]
Formalizing Security

\[ \text{MAC}_T \]

**procedure Initialize()**

\[ K \leftarrow \mathcal{K} \]

Return \( T_K(M) \)

**procedure Tag(M)**

**procedure Finalize(T', M')**

Return \( (T' = T_K(M')) \)

\[ \text{Adv}^{\text{mac}}_T(A) = \Pr[\text{MAC}_T^A \Rightarrow 1] \]
Exercise: Breaking MAC Security With No Query

\[ M_1 \xrightarrow{E_K} M_2 \xrightarrow{E_K} M_3 \xrightarrow{E_K} M_4 \xrightarrow{T} \]
Replay Attack

Bob transfers $10 instead of $5 !

MAC wasn’t defined to handle replay attack.
Replay is best addressed as an add-on to standard msg authentication.
Prevent Replay Attack Using Timestamp

\[ T \leftarrow \mathcal{T}_K (\text{Time}_A \| M) \]

Accept if:
\[ T = \mathcal{T}_K (\text{Time}_A \| M) \]
\[ |\text{Time}_A - \text{Time}_B| \leq \Delta \]
small interval
Prevent Replay Attack Using Counter

\[ T \leftarrow \mathcal{T}_K(\text{counter}_A \parallel M) \]
\[ \text{counter}_A \leftarrow \text{counter}_A + 1 \]

\[ \text{counter}_B \]

\[ K \]

If \( T = \mathcal{T}_K(\text{counter}_B \parallel M) \)
\[ \text{counter}_B \leftarrow \text{counter}_B + 1 \]
accept

Counters need to be synchronized
Agenda

1. MAC and Authenticity

2. MAC Constructions

3. How to Construct Good MAC
An Insecure Construction: Plain CBC-MAC

Question: Break CBC-MAC with a single Tag query
An Incorrect Fix of CBC-MAC

Exercise: Break this version using 3 Tag queries

Encoding the number of blocks
A Good Construction: Encrypted CBC-MAC

\[ M_1 \xrightarrow{E_K} M_2 \xrightarrow{E_K} M_3 \xrightarrow{E_K} M_4 \xrightarrow{E_{K'}} T \]

Different key
Dealing with Fragmentary Data

Solution: Padding with 10*

Question: Can we instead use padding with 0*?

Example: Suppose that the block length is 16 bytes.

Answer: No, can break this with a single Tag query
Agenda

1. MAC and Authenticity
2. MAC Constructions
3. How to Construct Good MAC
PRF Is a Good MAC

**Intuition:** - A good MAC means the output should be unpredictable
  - Random strings are unpredictable

**Question:** Given a good MAC $F$, construct $F'$ that is still a good MAC but has a trivial PRF attack.
PRF Extension

**Blockcipher:** Good PRF with **small** domain \( \{0, 1\}^n \)

\[ E_K \]

How to extend the domain of a PRF?

\[ F_{K'} \]

**Want:** Good PRF with **large** domain \( \{0, 1\}^* \)
Extending Domain: Carter-Wegman Paradigm

Condensing msg using a (keyed) hash

What’s the needed property for the hash?
Computationally Almost Universal Hash

\[ \text{Adv}_{h}^{\text{cau}}(A) = \Pr_{L \leftarrow \mathcal{L}}[h_L(X_1) = h_L(X_2)] \]

Must be distinct
Building A PRF Via Carter-Wegman

**Encrypted CBCMAC**

\[ M_1 \xrightarrow{0^n} E_L \xrightarrow{} M_2 \xrightarrow{} E_L \xrightarrow{} M_3 \xrightarrow{} E_L \xrightarrow{} M_4 \xrightarrow{} E_L \xrightarrow{} E_K \xrightarrow{} T \]

CBC-MAC is computationally universal