Message integrity: Confidentiality alone is not sufficient; also need message integrity. Otherwise adversary can tamper with the message (e.g., “Said $100 to Bob” → “Said $100 to Eve”)

In some cases (e.g., software patches), integrity more important than confidentiality.

Idea: Append a “tag” (also called “signature”) to the message to prove integrity (property we want: tags should be hard to forge).

Observation: The tag should be computed using a keyed-function.

Example of keyless integrity check: CRC (cyclic redundancy check) [simple example is to set tag to be the parity]

→ Example of keyless integrity check: CRC (cyclic redundancy check)
→ This was used in SSH v1 (1995) for data integrity. Fixed in SSH v2 (1996)
→ Also used in WEP (802.11b) protocol for integrity—also broken!

Problem: If there is no key, anyone can compute it! Adversary can tamper with message and compute the new tag.

Definition. A message authentication code (MAC) with key-space $K$, message space $M$ and tag space $T$ is a tuple of algorithms $\text{Timac} = (\text{Sign}, \text{Verify})$:

$$\begin{align*}
\text{Sign} &: K \times M \to T \\
\text{Verify} &: K \times M \times T \to \{0,1\}
\end{align*}$$

$\{0,1\}$ Must be efficiently-computable.

Correctness: $\forall k \in K, \forall m \in M$:

$$\Pr [ \text{Verify}(k,m, \text{Sign}(k,m)) = 1 ] = 1$$

→ Sign can be a randomized algorithm.

Defining security: Intuitively, adversary should not be able to compute a tag on any message without knowledge of $k$ by $k$.

→ Moreover, since adversary might be able to see tags on existing messages (e.g., signed software updates), it should not help towards creating a new MAC.

Definition. A MAC $\text{Timac} = (\text{Sign}, \text{Verify})$ satisfies $\text{chosen message attacks}$ (EUF-CMA) if for all efficient adversaries $A$, $\text{MACAdv}[A, \text{Timac}] = \Pr[W=1] = \text{negl}(\lambda)$, where $W$ is the output of the following security game:

As usual, $\lambda$ denotes the length of the MAC secret key (e.g., $\log |K| = \text{poly}(\lambda)$)

Note: the key can also be sampled by a special KeyGen algorithm (for simplicity, we just define $k$ to be uniformly random).

Let $m_1, ..., m_n$ be the signing queries the adversary submits to the challenger, and let $t_1, ..., t_n \leftarrow \text{Sign}(k, m)$ be the challenger’s responses. Then $W = 1$ if and only if:

$$\text{Verify}(k, m_i, t_i) = 1 \quad \text{and} \quad (m_i, t_i) \not\in \{(m_1, t_1), ..., (m_n, t_n)\}$$

MAC security notion says that adversary cannot produce a new tag on any message even if it gets to obtain tags on messages of its choosing.

First, we show that we can directly construct a MAC from any PRF.
Let $F: K \times M \rightarrow T$ be a PRF. We construct a MAC $\text{Imac}$ over $(K, M, T)$ as follows:

\begin{align*}
\text{Sign}(k, m): \text{output } t = F(k, m) \\
\text{Verify}(k, m, t): \text{output } 1 \text{ if } t = F(k, m) \text{ and } 0 \text{ otherwise}
\end{align*}

Theorem. If $F$ is a secure PRF with a sufficiently large range, then $\text{Imac}$ defined above is a secure MAC. Specifically, for every efficient MAC adversary $A$, there exists an efficient PRF adversary $B$ such that

$$\text{MACAdv}[A, \text{Imac}] \leq \text{PRFAdv}[B, F] + \frac{1}{|T|}.$$

Intuition for proof: 1. Output of PRF is computationally indistinguishable from that of a truly random function.

2. If we replace the PRF with a truly random function, adversary wins the MAC game only if it correctly predicts the random function at a new point. Success probability is then exactly $1/|T|$.

Proof. We define the following sequence of hybrid experiments:

- $\text{Hybo}_0$: This is the MAC security game:

  - adversary
  - $m \in M$
  - $t \leftarrow F(k, m)$
  - $C$
  - $(m^*, t^*)$

  Goal: Show for all efficient $A$:
  
  $$\Pr[\text{Hybo}_0(A) = 1] = \text{negl}.$$

- $\text{Hybo}_1$: Same as $\text{Hybo}_0$ except we replace $F(k, \cdot)$ with $f(\cdot)$ where $f \in \text{Funcs}[M, T]$.

Lemma 1. If $F$ is a secure PRF, then for all efficient adversaries $A$,

$$\Pr[\text{Hybo}_0(A) = 1] - \Pr[\text{Hybo}_1(A) = 1] = \text{negl}.$$

Proof. Suppose there exists efficient $A$ such that above probability is $E$. We construct $B$ as follows:

- $\text{Hyb}_0$: This is the MAC security game:

  - adversary
  - $m \in M$
  - $t \leftarrow F(k, m)$
  - $C$
  - $(m^*, t^*)$

  - output $1$ if adversary did not query on $m^*$ and $t^* = F(k, m^*)$

- $\text{Hyb}_1$: Same as $\text{Hyb}_0$ except we replace $F(k, \cdot)$ with $f(\cdot)$ where $f \in \text{Funcs}[M, T]$.

- $\text{Hyb}_2$: Same as $\text{Hyb}_1$ except we replace $F(k, \cdot)$ with $f(\cdot)$ where $f \in \text{Funcs}[M, T]$.

Lemma 2. For all adversaries $A$, $\Pr[\text{Hyb}_0(A) = 1] = 1/|T|$.

Proof. $\text{Hyb}_0(A)$ outputs $1$ if $A$ predicts value of $f$ at $m^*$. Since $f$ is uniform, $A$ succeeds with probability at most $1/|T|$.
Implication: Any PRF with large output space can be used as a MAC.

AES has 128-bit output space, so can be used as a MAC.

Drawback: Domain of AES is 128-bits, so can only sign 128-bit (16-byte) messages.

How do we sign longer messages? We will look at two types of constructions:

1. Constructing a large-domain PRF from a small-domain PRF (i.e., AES)
2. Hash-based constructions

Approach 1: use CBC

\[ m_1 \rightarrow F(k) \rightarrow m_2 \rightarrow F(k) \rightarrow \ldots \rightarrow m_n \rightarrow F(k) \rightarrow \text{output} \]

Not encrypting messages so no need for IV (or intermediate blocks)

\[ \Rightarrow \text{Mode often called "raw-CBC"} \]

Raw-CBC is a way to build a large-domain PRF from a small-domain one

\[ \Rightarrow \text{Can show security for "prefix-free" messages [more precisely, raw-CBC is a prefix-free PRF: pseudorandom as long as PRF never evaluated on two values where one is a prefix of other] includes fixed-length messages as a special case.} \]

But not secure for variable-length messages: “Extension attack”

1. Query for MAC on arbitrary block \( X \):

\[ \begin{align*}
X & \downarrow \\
F(k) & \rightarrow F(k, X) \\
\text{tag } t & \rightarrow
\end{align*} \]

2. Output forgery on message \( (X, X \oplus t) \) and tag \( t \)

\[ \Rightarrow \text{t is a valid tag on extended message } (X, t) \]

\[ \Rightarrow \text{Adversary succeed with advantage } \frac{1}{2} \]

\[ \text{includes fixed-length messages as a special case.} \]