Today:

Message Authentication Codes

Read: Section 5 (pg. 133-149) in the lecture notes by Pass & Shelat

So far: We focused on secrecy:

* Encryption scheme w. perfect secrecy: One-time pad.
* Encryption scheme w. computational secrecy:
  OWF $\Rightarrow$ Hard-Core predicates $\Rightarrow$ PRG $\Rightarrow$ PRF $\Rightarrow$ Enc scheme
  (Goldreich-Levin Thm)
* Zero-knowledge + zero-knowledge proof-of-knowledge.

Today: We shift our focus from secrecy to authenticity.

Goal: Ensure that an adversary cannot alter a msg sent between two honest parties (without being detected)

Authenticity is very important in real world applications such as banking and online shopping.

In day-to-day life we use signatures or other (physical) methods such as special paper (for money), watermarks, etc.

Today we will discuss digital methods to authenticate msgs.

We will discuss two approaches:

1. Message Authentication codes: assumes that every
2 Digital Signatures: Authentication in the public key setting. Assumes each user is associated with a public key, and only knowledge of this public key is needed for authentication.

Note: In any digital authentication method, the "signature" is a string of bits, and therefore to ensure security the "signature" must depend on the msg to be signed.

Message Authentication Codes (MAC)

Definition: A MAC scheme with msg space $M$ consists of 3 ppt alg $(\text{Gen, MAC, Ver})$ with the following syntax:

- Gen: takes as input a security parameter $\lambda$ and outputs a secret key $K$.
  This alg must be probabilistic (for security).
- MAC takes as input a secret key $K$ and a msg $m \in M$ and outputs a tag $t$.
- Ver takes as input a secret key $K$, a msg
A MAC scheme is required to be correct

**Def:** A MAC scheme $\langle \text{Gen}, \text{MAC}, \text{Ver} \rangle$ for msg space $M$

is said to be correct if $\forall m \in M$

$$\Pr \left[ \text{Ver} \left( k, m, \text{MAC}(k, m) \right) = 1 \right] = 1$$

$k \leftarrow \text{Gen}(1^n)$

* The MAC alg can be prob. or deterministic. In the former case, the probability is also over the randomness of the MAC alg.

* The Ver alg is usually thought of as being deterministic.

**Security**

The goal of the adversary is to forge a signature.

**His power:** The adv. is PPT. He can see pairs $(m, \text{MAC}(k, m))$.

For which msgs $m$? Usually we assume worst case and allow the adversary to choose the msgs $m \in M$ for which he sees $\text{MAC}(k, m)$. This is called a "Chosen Msg Attack".

In addition, the adv. is given access to $\text{Ver}(k, \cdot, \cdot)$. 
Def: A MAC scheme $(\text{Gen}, \text{MAC}, \text{Ver})$ is said to be secure against adaptive chosen msg attacks if

\[
\forall \mathcal{A} \exists \text{ negl } \text{func } \mathcal{E}(\cdot) \text{ st. } \forall n \in \mathbb{N}
\]

\[
\Pr \left[ \mathcal{A}(\text{MAC}(k, \cdot), \text{Ver}(k, \cdot ; \cdot)) \left( 1^n \right) = (m, t) \right. \left. \text{ s.t. } \forall \mathcal{E}(n) \right]
\]

\[
\left. \text{Ver}(k, m, t) = 1 \right. \left. \text{A didn't query m} \right. \rightarrow \text{MAC}(k, \cdot)
\]

Note: In the security def, we require security against "existential forgery"; i.e., the adv should not be able to create a valid tag for any msg (that was not tagged by the oracle).

A weaker security def is against "universal forgery"; i.e., the adv should not be able to generate a valid tag for any msg (or for a random msg).

An even weaker security def is against a "total break"; i.e., the adv should not be able to recover the secret key $k$.

Q: Is a semantic secure encryption scheme a good MAC scheme? No!

An encryption can be malleable: given $\text{Enc}(k, m)$ it
may be easy to compute $\text{Enc}(k, M+1)$

\textbf{Ex:} OTP is malleable

\begin{align*}
\text{Enc}(k, m) &= (r, f_k(r) \oplus m) \text{ is malleable.} \\
\text{Given } \text{Enc}(k, m), \text{ one can easily compute } \text{Enc}(k, m \oplus m') &= (r, f_k(r) \oplus m \oplus m') \\
&\underline{\text{Enc}(k, m)}
\end{align*}

\textbf{Thm:} Assume OWF exists. Then there exists a MAC scheme that is secure against adaptive chosen msg attacks (ACKA) for msgs in $M_n = \{0, 1\}^n$ where $n$ is the security parameter.

\textbf{DF:} OWF $\Rightarrow$ PRF family $\mathcal{F} = \{F_n\}$ where $\forall n$

\begin{align*}
F_n &= \{ f_k : \{0, 1\}^n \rightarrow \{0, 1\}^* \} \\
&\text{seed generation alg of the PRF}
\end{align*}

Define the following MAC scheme $(\text{Gen}, \text{MAC}, \text{Ver})$

\begin{align*}
\text{Gen}(1^n) &\Rightarrow \text{Output } k \leftarrow \mathcal{G}(1^n) \\
\text{MAC}(k, m) &= f_k(m)
\end{align*}
Ver(k, m, t) = 1 \iff t = f_k(m).

It is easy to see that this MAC scheme is correct, i.e.

\forall n \in \mathbb{N} \quad \forall m \in \{0, 1\}^n

\Pr[\forall k \leftarrow G(1^n) \quad \text{Ver}(k, m, \text{MAC}(k, m)) = 1] = 1

Assume for conr that it is not secure (against ACMA). Namely, \exists PPT alg\ A \exists poly p \exists infinite set N \subseteq \mathbb{N} st

\forall n \in \mathbb{N}

\Pr[\forall k \leftarrow G(1^n) \quad A(1^n) = (m, f_k(m) : m \notin \text{query list})] > \frac{1}{p(n)}.

Note that this is not possible if \( f_k \) was replaced w.a.r. by \( R \). Thus, \( A \) can be used to distinguish BB access to \( f_k \) from PPs access to \( R \), as follows. Define PPT B:

\[ B(1^n) : \text{Generate} \quad (m, t) \leftarrow A(f_k(1^n)). \]

- If \( m \) was queried by \( A \) then output random bit \( b \leftarrow \{0, 1\} \)
- O.w. query the oracle on \( m \) to obtain \( f_k(m) \). If \( f_k(m) = t \) then output 1
- O.w. output a random bit \( b \leftarrow \{0, 1\} \).
It remains to note that \( \forall n \in \mathbb{N} \)
\[
\Pr [ B^f_k (Y^n) = 1 ] > \frac{1}{2} + \frac{1}{\rho(n)},
\]
whereas \( \Pr [ B^g_k (Y^n) = 1 ] \leq \frac{1}{2} + \text{negl}(n) \),

contradicting the PRF security.

\[\text{\footnotesize\textbf{Note:}}\hspace{1em} \text{This MAC is only for msgs of bounded length } n.\]

What if we want to authenticate longer msgs?

\textbf{Attempt 1}: \( \text{MAC}(k, (m_1, m_2)) = ( \text{MAC}(k, m_1), \text{MAC}(k, m_2) ) \)

each of length \( n \).

Insecure! Given \( \text{MAC}(k, (m_1, m_2)) \) can generate \( \text{MAC}(k, (m_2, m_1)) \).

\textbf{Attempt 2}: \( \text{MAC}(k, (m_1, m_2)) = \text{MAC}(k, m_1 \oplus m_2) \)

Insecure! Can easily find collisions to the \( \oplus \) operator.

\textbf{Attempt 3}: \( \text{MAC}(k, (m_1, m_2, \ldots, m_t)) \), where each \( m_i \in \{0, 1\}^{n/2} \):

choose a random \( r \leftarrow \{0, 1\}^{n/2 - \log t} \), and output:

\[
(r ( \text{MAC}(k, (r_i, m_i)) )_{i=1}^m, 10^{\log t}).
\]

Secure!

For a priori fixed \( t \).
Goal: Make the tags short, and allow arbitrarily long msgs.

Hash-then-MAC

Def: A family of functions $\mathcal{H} = \{h_h\}_{h \in \mathcal{H}}$ where $\forall h \in \mathcal{H}$, $h : \{0, 1\}^* \rightarrow \{0, 1\}^n$ is said to be collision resistant if $\forall PPT$ alg $A \in \text{negl}(\cdot)$ s.t.

$$\Pr[A(1^n, h) = (x, x') : x \neq x' \land h(x) = h(x')] \leq \text{negl}(n).$$

Let $(\text{Gen}, \text{MAC}, \text{Ver})$ be a MAC scheme for msg space $M_n = \{0, 1\}^n$ (where $n$ is the security parameter), that is secure against ACMA.

Let $\mathcal{H}$ be a collision resistant hash family (CRHF).

Define $(\text{Gen}, \text{MAC}, \text{Ver})$ to be the following MAC scheme, with msg space $M = \{0, 1\}^n$

$\text{Gen}(1^n)$: Generate $k \leftarrow \text{Gen}(1^n)$ and generate $h \leftarrow \mathcal{H}_n$.

Output $(k, h)$.

$\text{MAC}(k, h, m)$ outputs $\text{MAC}(k, h(m))$.

$\text{Ver}(k, h, m, t) = 1$ iif $t = \text{MAC}(k, h(m))$. 
Claim: If \((\text{Gen}, \text{MAC}, \text{Ver})\) is a secure MAC scheme w. msg space \(M = \{0, 1\}^n\) and \(H = \emptyset\) is a family of collision resistant hash functions, then \((\text{Gen}, \text{MAC}, \text{Ver})\) is a (ACMA) secure MAC scheme w. msg space \(M = \{0, 1\}^n\).

Moreover, the length of a tag is ind. of the msg length (and depends only on the security parameter).

"Pf sketch": Suppose for cont. that \((\text{Gen}, \text{MAC}, \text{Ver})\) is not secure. Namely, \(\exists \text{ PPT adv} \exists \text{ poly } p() \& \text{ infinite set } N \subseteq \mathbb{N} \text{ st. } \forall n \in N\)

\[
\Pr[A^{\text{MAC}(k_n, \cdot), \text{Ver}(k_n, \cdot); \cdot} (\mathcal{L}) = (M, t); \text{ t is valid } \& m \text{ was not queried } ] > \frac{1}{p(n)}.
\]

\(\forall n \in \mathbb{N}\) let \(Q_n(k_n, h)\) be the set of all queries that \(A\) sends its oracle \(\text{MAC}(k_n, \cdot)\).

Case 1: \(\Pr[\exists m' \in Q_n(k_n, h) \text{ s.t. } h(m) = h(m') \& m \neq m'] = \delta(n)\)

\((k_n, h) \leftarrow \text{Gen} (1^n)\)

where \(\delta(\cdot)\) is non-negl. In this case we use \(A\) to contradict the collision-resistant property of the hash function.

Case 2: \(\Pr[\exists m', m \in Q_n(k_n, h) \text{ s.t. } h(m) = h(m') \& m \neq m'] = \text{negl}(n)\).

In this case we use \(A\) to break the security of the underlying MAC.
Today: Digital Signatures = public key version of MACs