Lecture 5
Digital Signatures
Plan

- Definition
- One-time signatures from OWF.
- Many-time

Last time: MAC

This time: Signatures

Being able to authenticate gives you ability to produce new MAC'd msgs.

Can verify/authenticate msgs without being able to sign new ones.
First, recap:

\[ \text{MAC}: K \times M \rightarrow T \]

**MAC Security:** EUF - CMA

Defined by security game we saw last time

**MAC for short msgs:**

\[ \text{PRF } F: K \times X \rightarrow Y \]

\[ \text{AES is example of PRF} \]

\[ \text{MAC}(k, m) := F(k, m) \quad (\text{msg space } X) \]

*PRF security relies on key being completely random.*

\[ F(\text{"0000","0000"}) \]

**MAC for long msgs:**

\[ \text{UHF Hash: } K \times X^2 \rightarrow X \]

\[ (\text{msg space } X^2) \]

\[ \text{MAC}((k_1, k_2), m) := F(k_1, \text{Hash}(k_2, m)) \]

UHF is a fast "non-cryptographic" hash fn.

**Intuition:** Until attacker finds two msgs that hash to same value, has no info that can help it forge.

[See Book Thm 2.7]
**Digital Signatures**

- Used everywhere on the web for authentication
  - Certificates, HTTPS, SSH, ...
- Unlike a pen-and-paper signature, can't cut & paste
  - $\text{Sig}$ is bound to data signed.
- Unlike MAC, there are two keys
  - **Private** signing key $sk$
  - **Public** verification key $vk/pk$

The idea of using two keys (public & private) was the revolutionary idea in cryptography in the 20th century (Diffie & Hellman)

- Followed thousands of years of shared secrets
Syntax

\[ \text{Gen}(\cdot) \rightarrow (sk, pk) \]

\[ \text{Sign}(sk, m) \rightarrow \sigma \]

\[ \text{Verify}(pk, m, \sigma) \rightarrow \{0, 1\} \]

**Correctness:**

Honest verifier accepts honest sigs:

\[
\forall (sk, pk) \leftarrow \text{Gen}() \\
\forall m \in \mathcal{M} \\
\forall \sigma \leftarrow \text{Sign}(sk, m) \\
\text{Verify}(pk, m, \sigma) = 1.
\]
Security: Almost identical to MAC security

EUF-CMA: Attacker sees sigs on many msgs of its choosing. Can't forge a sig on new msg of its choice.

For sig scheme $\Xi = (\text{Gen}, \text{Sign}, \text{Ver})$, adv $A$, let
$$\text{Sig Adv}[A, \Xi] = \Pr[\text{game output is 1}].$$

A sig scheme $\Xi = (\text{Gen}, \text{Sign}, \text{Ver})$ is secure if $A$ is not adv
$$\text{Sig Adv}[A, \Xi] \leq \text{negl}.$$
**Subtle point:**

This design admits schemes in which it is easy to cook up new valid sigs on msg \( m \) given one sig on \( m \). [Has led to attacks!]

\( \rightarrow \) ECDSA has this malleability property.

\( \rightarrow \) Think: How can you tweak deti to prevent?

**Efficiency** We typically care about

- Signature size
- Size of \( ph \)
- Signing time
- Verification time
- Assumptions (Safetorv/?RSA?)

No one signature scheme dominates all others in all metrics 😞
Recall: One-way Function

A \text{ fn } f: \mathcal{X} \rightarrow \mathcal{Y} \text{ is one way if }

\forall \text{ ess adv } A

\Pr \left[ f(x^*) = x : x^* \leftarrow A(f(x)) \right] \leq \text{negl}

(Reminder: IS \text{ P} = \text{NP} \Rightarrow \exists \text{ OWF})

Or, in game form...

\[
\begin{array}{c}
\text{Chal} \\
\begin{array}{c}
x \leftarrow \mathcal{X} \\
\end{array} \\
\begin{array}{c}
f(f(x^*)) = f(x) \\
o \text{ o.u.}
\end{array}
\end{array}
\quad \begin{array}{c}
f(x) \\
x^* \leftarrow A(f(x))
\end{array}
\begin{array}{c}
\text{Adv}
\end{array}
\]
Example candidate OWFs:

\[ f_{SHA}(x) = SHA256(x) \]

\[ f_{PRF}(x) = PRF(k, "000000") \]

\[ f_{MAC}(x) = MAC(k, "000000") \]

OWFs are essentially the weakest/simplest crypto tool.

⇒ Surprise? That possible to use then to get signatures

* OWF only hard to invert on a randomly sampled input.

If \( f: \{0,1\}^n \rightarrow \{0,1\}^n \) is OWF, no guarantee that

\[ f(000000 \cdots 011 \times) \text{ is hard to invert} \]

\[ f(011 \times) \text{ is hard to invert} \]
One-time-secure signatures (Lamport)

One-time secure = secure if adv only sees one sig under a given sk

Msg space \( M = \{0,1\}^n \)
Sec Param \( \lambda = 128 \)
O.U.F. \( f : \{0,1\}^\lambda \rightarrow \{0,1\}^\lambda \)

Gen(\() \rightarrow (sk, vk)\)

Choose \( 2n \) random \( \lambda \)-bit strings...

\[ sk = \begin{pmatrix} x_{01} & x_{02} & \cdots & x_{0n} \\ x_{11} & x_{12} & \cdots & x_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nn} \end{pmatrix} \]

\[ vk = \begin{pmatrix} f(x_{01}) & f(x_{02}) & \cdots & f(x_{0n}) \\ f(x_{11}) & f(x_{12}) & \cdots & f(x_{1n}) \\ \vdots & \vdots & \ddots & \vdots \\ f(x_{n1}) & f(x_{n2}) & \cdots & f(x_{nn}) \end{pmatrix} = \begin{pmatrix} y_{01}, \ldots, y_{0n} \\ y_{11}, \ldots, y_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ y_{n1}, \ldots, y_{nn} \end{pmatrix} \]

Sign \((sk, m_1||m_2||\ldots||m_n \in \{0,1\}^n) \rightarrow \sigma\)

Output sk values corresponding to bits of msg

\[ \sigma = \begin{pmatrix} x_{m_1,1} & x_{m_2,2} & \cdots & x_{m_n, n} \end{pmatrix} \]

Verify \((pk, m_1||\ldots||m_n \in \{0,1\}^n, \sigma) \rightarrow \{0,1\}\)

Check \( \forall i \in [n] : y_{m_i} = f(\sigma_i) \)

Output "1" if all accept.
Why are Lamport sigs only one-time secure?

$sk \leftarrow (X_0, X_1, X_2, X_3, \ldots, X_n)$

Each sig gives away $\frac{1}{2}$ of secret key. With 2 sigs can recover all.

Why is it secure for one-time use?

Intuition: To sign $m^*$, need to invert $f$ at least one point.

Beware: Intuition is often wrong. Devil in details. Security proofs are a useful tool: more or less essential in modern crypto.
Properties

Correctness - By construction.

Security - ONE-TIME secure.

Strategy

\exists \text{ adversary that } \implies \exists \text{ adv that inverts w.p. } \Pr \leq \epsilon \implies \text{contradicts our security.}

OWF

Challenger

\begin{align*}
\mathbf{x^*} & \leftarrow \mathbf{x} \\
y & = \mathbf{s}(\mathbf{x})
\end{align*}

Us

\begin{align*}
\mathbf{x_0}, \ldots, \mathbf{x_n} & \leftarrow \mathbf{x} \\
\mathbf{x'}, \ldots, \mathbf{x'}_n & \leftarrow \mathbf{x'} \\
\mathbf{p_k} & = (\mathbf{s}(\mathbf{x_0}), \mathbf{s}(\mathbf{x_1}), \ldots, y) \\
\text{w.p. } \frac{1}{2} & \text{ fail } \begin{cases} & \text{ can't answer query} \\
& \text{ w.p } \frac{1}{2} \text{ fail, else get } \mathbf{x'} \end{cases}
\end{align*}

Adv

\begin{align*}
\mathbf{m} & \rightarrow \mathbf{c} \\
\mathbf{m^*} & \rightarrow \mathbf{c^*}
\end{align*}

\begin{align*}
\mathbf{m} & \leftarrow \mathbf{s} \\
\mathbf{m^*} & \leftarrow \mathbf{s^*}
\end{align*}

Efficiency

* OWFs are very fast = 60m/s for AES
* Sigs are large-ish = \lambda^2 bits for \lambda = 256.
Extension: Signing long msgs: “Hash & sign”

Use $H : \{0, 1\}^* \rightarrow \{0, 1\}^8 \Rightarrow CRHTF$

$\text{Sig} : (sk, m) \rightarrow \text{Sig}(sk, H(m))$

$\text{Ver} : (pk, m, \sigma) \rightarrow \text{Ver}(pk, H(m), \sigma)$

Q: Why not use a (keyed) universal hash fn?
Vendor

**Application**: Software Updates

Vendor signs each update with a fresh sk.

*Only need one-time security*  
[Not typically used in practice.]
Many-Time Secure Sigs

We will show how to construct a full-blown (many-time secure) sig scheme from any one-time secure scheme + PRF.

\[ \Rightarrow \text{Sig from OUF.} \quad \text{[Since } \exists \text{ OUF } \Rightarrow \exists \text{ PRF]} \]

**Observation:** Can use one PRF secret key to implicitly generate a gigantic tree of sig sks (tree has \(2^n\) leaves).

**PRF:**

\[ F : K \times \{0, 1\}^n \rightarrow sk \]

All keys "as good as random" by PRF security.

\[ F(k, \"0\") \quad \text{pk}_0 \]

\[ F(k, \"1\") \quad \text{pk}_1 \]

\[ F(k, \"00\") \quad \text{pk}_{00} \]

\[ F(k, \"01\") \quad \text{pk}_{01} \]

\[ F(k, \"10\") \quad \text{pk}_{10} \]

\[ F(k, \"11\") \quad \text{pk}_{11} \]

From \(sk\), can generate \(pk\) for each node.
Many-time Sigs ... will be informal. See Book for full description.

\[ \text{Gen}() \rightarrow \exists \text{ pk}<\text{ pke} \text{ (at root)} \]
\[ \text{sk} \rightarrow \text{PRF key k} \]

\[ \text{Sign}(\text{sk}, m) \rightarrow \sigma \]

* Pick a leaf \( e \) at random.
* Sign using \( \text{sk} \).
* Sign each pair of \( \text{vks} \) using \( \text{sk} \) of "parent node" in tree.
* Publish all \( \text{signs} \) and \( \text{vks} \) on path to root + siblings.

Signature consists of 
\[ \sigma_0, \text{ pk}_0, \text{ pk}_1, \text{ pk}_{10}, \text{ pk}_{11}, \sigma_2 \]
Many-time Sigs (cont'd)

\[ \text{Ver}(pk, m, \sigma) : \]

Verify chain of sigs down to leaf