Previously...
Digital Signatures

Algorithms:
• $\text{Gen}() \rightarrow (sk,pk)$
• $\text{Sign}(sk,m) \rightarrow \sigma$
• $\text{Ver}(pk,m,\sigma) \rightarrow 0/1$

Correctness:
$\Pr[\text{Ver}(pk,m,\text{Sign}(sk,m))=1: (sk,pk)\leftarrow \text{Gen()}]=1$
Many-time Signatures

\[ \text{CMA-Adv}(\cdot) = \Pr[ \text{outputs 1} ] \]

\[ \text{Output 1 iff:} \]
\[ \begin{align*}
& \bullet \ m^* \notin \{m_1, \ldots\} \\
& \bullet \ \text{Ver}(pk, m^*, \sigma^*) = 1
\end{align*} \]

\((sk, pk) \leftarrow \text{Gen}()\)

\(\sigma \leftarrow \text{Sign}(sk, m)\)
Strong Security

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

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

Output 1 iff:
- \((m^*, \sigma^*) \notin \{(m_1, \sigma_1)\ldots}\)
- \(\text{Ver(pk, m^*, \sigma^*)} = 1\)

\(\text{CMA-Adv(ぞっとしないもの)} = \Pr[\text{ぞっとしないもの outputs 1}]\)
Signatures from TDPs

\[ \text{Gen}_{\text{Sig}}() = \text{Gen}() \]

\[ \text{Sign}(sk,m) = F^{-1}(sk, H(m)) \]

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

**Theorem:** If \((\text{Gen}, F, F^{-1})\) is a secure TDP, and \(H\) is modeled as a random oracle, then \((\text{Gen}_{\text{Sig}}, \text{Sign}, \text{Ver})\) is (strongly) CMA-secure
Basic Rabin Signatures

\[ \text{Gen}_{\text{Sig}}(): \text{ let } p, q \text{ be random large primes} \]
\[ sk = (p, q), \; pk = N = pq \]

\[ \text{Sign}(sk, m): \text{ Solve equation } \sigma^2 = H(m) \mod N \]
using factors \( p, q \)
• Output \( \sigma \)

\[ \text{Ver}(pk, m, \sigma): \sigma^2 \mod N = H(m) \]
Signatures from One-way Functions

One-way functions are sufficient to build signature schemes

Therefore, can build signatures from:
• RSA, DDH, Block Ciphers, CRHF, etc.

Limitation:
• Poor performance in practice
Lamport Signatures

Let $F: X \rightarrow Y$ be a one-way function

Let $M = \{0,1\}^n$ be message space

Gen():

$$\begin{align*}
\text{Gen():} & \quad X \\
& \downarrow \\
\begin{array}{c}
X_{1,0} \\
X_{1,1} \\
X_{2,0} \\
X_{2,1} \\
X_{3,0} \\
X_{3,1} \\
X_{4,0} \\
X_{4,1} \\
X_{5,0} \\
X_{5,1}
\end{array} \\
\downarrow \\
\begin{array}{c}
y_{1,0} \\
y_{1,1} \\
y_{2,0} \\
y_{2,1} \\
y_{3,0} \\
y_{3,1} \\
y_{4,0} \\
y_{4,1} \\
y_{5,0} \\
y_{5,1}
\end{array}
\end{align*}$$

$y_{i,b} = F(x_{i,b})$

$sk \rightarrow F \rightarrow pk$
Lamport Signatures

Sign\( (sk, m)\): \((x_{i,m_i})_{i=1,\ldots,n}\)

Ver\( (pk, m, \sigma)\): \(F(x_{i,m_i}) = y_{i,m_i}\)
Lamport Signatures

**Theorem:** If $F$ is a secure OWF, then $(\text{Gen,Sign,Ver})$ is a (weakly) secure one-time signature scheme.
Proof
Proof

Since $m^* \neq m$, $\exists i$ s.t. $m^*_i \neq m_i$

Suppose we know $i$, $m_i = 1-b$, $m^*_i = b$

Construct adversary that inverts OWF
Proof

\[
\begin{align*}
\mathbf{y}^* & = \begin{bmatrix} y_{1,0} & y_{2,0} & y_{2,1} & y_{3,1} & y_{4,0} & y_{4,1} & y_{5,0} & y_{5,1} \end{bmatrix} \\
\end{align*}
\]

\[
\begin{align*}
\begin{bmatrix} x_{1,0} & x_{1,1} & x_{2,0} & x_{2,1} & x_{3,0} & x_{3,1} & x_{4,0} & x_{4,1} & x_{5,0} & x_{5,1} \end{bmatrix} \& = \begin{bmatrix} x_{1,0} & x_{2,0} & x_{2,1} & x_{3,1} & x_{4,1} & x_{5,1} \end{bmatrix} \\
\end{align*}
\]
Proof

View of exactly as in 1-time CMA experiment, assuming

• \text{ith bit of } m = b
• \text{ith bit of } m^* = 1-b

If always chooses m,m* with these properties, and forges with probability \( \varepsilon \), then inverts with probability \( \varepsilon \)
Proof

In general, \( m, m^* \) may choose \( m, m^* \) to differ at arbitrary places
• May be randomly chosen, may depend on \( pk \), may even depend on \( \sigma \)
• May never be at certain places

How do we make \( \text{\textbullet} \) still succeed?
Proof

\[
i, b \leftarrow [n] \times \{0, 1\}
\]

\[
\begin{array}{cccccc}
 y_{1,0} & y_{2,0} & y^* & y_{4,0} & y_{5,0} \\
y_{1,1} & y_{2,1} & y_{3,1} & y_{4,1} & y_{5,1} \\
\end{array}
\]

\[
\begin{array}{cccccc}
 x_{1,0} & x_{2,0} & x_{3,0} & x_{4,0} & x_{5,0} \\
x_{1,1} & x_{2,1} & x_{3,1} & x_{4,1} & x_{5,1} \\
\end{array}
\]

If need \( x_{i,b} \), abort

If no \( x_{i,b} \), abort
Proof

\( \text{pk} \) independent of \((i, b)\)
\( \text{m} \) independent of \((i, b)\)
Therefore, \( \Pr[m_i = 1-b] = \frac{1}{2} \)

Conditioned on \( m_i = 1-b \),
\( \text{Signing succeeds} \)
\( \sigma \) independent of \( i \)
\( \text{\& forgives with probability } \varepsilon, \text{ independent of } i \)
Proof

We know if \( \text{\texttt{\textbullet}} \) forges, then \( m^* \neq m \)

Since \( m^* \) independent of \( i \), have prob at least \( \frac{1}{n} \) that \( m^*_i = 1 - m_i = b \)

In this case, \( \text{\texttt{\textbullet}} \) succeeds in inverting \( y^* \)
- \( \text{Prob} = \frac{1}{2} \times \varepsilon \times \frac{1}{n} = \frac{\varepsilon}{2n} \)
Limitations of Lamport Signatures

Only weakly secure
• Why?
• How to fix?

\[ |pk|,|\sigma| \gg |m| \]
• How to fix?
Theorem: Given a secure OWF, it is possible to construct a strongly secure 1-time signature scheme where $|m| \gg |pk|,|\sigma|$
Signing Multiple Messages

Once adversary sees two signed messages, security is lost (why?)

How do we sign multiple messages?
Signature Chaining

\[ m_1, \sigma_1 \leftarrow \text{Sign}(sk_1, m_1) \]

\[ \text{Ver}(pk_1, m_1, \sigma_1) \]
Signature Chaining

\[
m_1, \sigma_1 = (pk_2, \sigma_1')
\]

\[
\sigma_1' \leftarrow \text{Sign}(sk_1, (m_1, pk_2))
\]

\[
\text{Ver}(pk_1, (m_1, pk_2), \sigma_1')
\]
Signature Chaining

\[
\begin{align*}
&m_2, \sigma_2 \\
\sigma_1' \leftarrow & \text{Sign}(sk_2, m_2) \\
&(sk_2, pk_2) \leftarrow \text{Gen()}
\end{align*}
\]

Ver(pk_2, m_2, \sigma_2)
Signature Chaining

Idea: Bob can be assured that $pk_2$ was in fact generated by Alice
• If Eve tampered with $pk_2$, then signature on first message would have been invalid

Therefore, Alice can sign $m_2$ using $sk_2$, and Eve cannot produce a forgery $m_2'$ with valid signature

Can repeat process to sign arbitrarily many messages
Signature Chaining

\[ m_2, \sigma_2 = (pk_3, \sigma_2') \]

\[ \sigma_1' \leftarrow \text{Sign}(sk_2, (m_2, pk_3)) \]

\[ (sk_2, pk_2) \leftarrow \text{Gen}() \]

\[ (sk_3, pk_3) \leftarrow \text{Gen}() \]

\[ \text{Ver}(pk_2, (m_2, pk_3), \sigma_2') \]
Limitations

Alice and Bob must stay synchronized
• Else, Bob won’t be using correct public key to verify

If many users, every pair needs to be synchronized
• What if Alice is sending messages to Bob and Charlie?
(Almost) Stateless Signature Chaining

\[
\begin{align*}
\text{Ver}(pk_1, (m_1, pk_2), \sigma_1') & \\
\sigma_1' & \leftarrow \text{Sign}(sk_2, (m_2, pk_3)) \\
\sigma_2' & \leftarrow \text{Sign}(sk_2, (m_2, pk_3)) \\
\end{align*}
\]

\[
\begin{align*}
(m_2, \sigma_2) & = (m_1, pk_2, \sigma_1', pk_3, \sigma_2') \\
\end{align*}
\]
Still Limitations

Now Bob (and Charlie, etc) are stateless

However, Alice is still stateful
• Needs to remember all messages sent
• Signature length grows with number of messages signed
Signature Trees

\[ \sigma_\emptyset \leftarrow \text{Sign}(sk_\emptyset, (pk_0, pk_1)) \]
\[ \sigma_0 \leftarrow \text{Sign}(sk_0, (pk_{00}, pk_{01})) \]
\[ \sigma_1 \leftarrow \text{Sign}(sk_1, (pk_{10}, pk_{11})) \]

\[ \sigma_{00}, \sigma_{01}, \sigma_{10}, \sigma_{11} \]
Signature Trees

To sign $m_i$,

- Compute $\sigma_i \leftarrow \text{Sign}(sk_i, m_i)$, where $sk_i$ is the $i$th leaf.
- Must include $pk_i$ in signature so Bob can verify $\sigma_i$.
- Must authenticate $pk_i$, so include $\sigma_{P(i)}$ (and $pk_{S(i)}$).
- Must include $pk_{P(i)}$ so Bob can verify $\sigma_{P(i)}$.
- Must auth $pk_{P(i)}$, so include $\sigma_{P(P(i))}$ (and $pk_{S(P(i))}$).
- ...
Comparison to Chaining

Limitations:
• Bounded number of messages ($2^d$)
• Still requires Alice to keep state (all the $sk$’s, $pk$’s).
  Size of state $\approx 2^d$

Advantages:
• Signature size $\approx d$, logarithmic in number of messages signed
Avoid Large State?

Alice keeps PRF key $k$ as part of secret key

- For all internal nodes or leaves $i$,

$$\begin{array}{c}
(sk_i, pk_i) \leftarrow \text{Gen}(; \text{PRF}(k, i))
\end{array}$$

- Alice never stores signatures or public keys
- Instead, she computes needed signatures/public keys on the fly
Unbounded Messages

Set $d=128$ or $256$

- Can now sign up to $2^{128}$ messages
- Signature size $\propto d = 128$, so shortish signatures
- Size of state independent of $d$, so short
- Time to compute signature?
  - Only need pk’s, σ’s on path from root to leaf, plus neighbors
  - Only $O(d)$ terms
  - Can efficiently compute from PRF key $k$
Fully Stateless?

So far, still need to keep state to remember which leaf we should use next

However, now we can do something different:
• Instead of choosing leaves sequentially, just choose leaf at random
• Except with probability $O(|messages|^2/2^d)$, never use the same leaf twice
Putting it Together

\[ pk_\emptyset \quad \text{sk}=(sk_\emptyset, k) \]

\[ i \leftarrow \{0, \ldots, 2^d-1\} \]
Putting it Together

**sk = (sk∅, k)**

(sk₀, pk₀) ← Gen(; PRF(k, 0))
(sk₁, pk₁) ← Gen(; PRF(k, 1))
(sk₀₀, pk₀₀) ← Gen(; PRF(k, 00))
(sk₀₁, pk₀₁) ← Gen(; PRF(k, 01))
...

σ∅ ← Sign(sk∅, (pk₀, pk₁))
σ₀ ← Sign(sk₀, (pk₀₀, pk₀₁))
...

σ ← Sign(skᵢ, m)

Output all pkᵢ’s and all σ’s as signature
Putting it Together

OWF to get 1-time signatures (with large \( pk \)'s, \( \sigma \)'s)

Hash message
- 1-time signatures with small \( pk \)'s, \( \sigma \)'s
- Can accomplish using just OWFs

Create tree of signatures (stateful scheme)

Make stateless by using a PRF
What's Known

OWP
CRH
CPA-PKE
CCA-PKE
OWF
PRG
Com
PRF
PRP
MAC
SKE
Auth Enc
Sig
What’s Known

OWP
CRH
CPA-PKE
CCA-PKE

TCR
OWF
PRG
Com
Sig
PRF
MAC
Auth
Enc
PRP
SKE
Theorem: Given a secure OWF, it is possible to construct a strongly CMA-secure signature scheme.
Practical Use?

Lamport signatures are fast:
• Signing is just revealing part of your secret key
• Verifying is just a few OWF evaluations

Tree-based signatures are a bit slower
• Need to generate many signatures
• Need to generate many public keys
• Need many PRF evals
Practical Use?

Main limitation: Signature size
• Basic Lamport: 128 bits per message bit
• With hashing, need to sign 256 bit messages
• For signature trees, signature consists of $d$ Lamport signatures (plus public keys)
  • $d$ must be big enough to prevent collisions
  • E.g. $d = 128$

Overall signature size: around a **megabit**
What’s the Smallest Signature?

Signature Trees: 1megabits

RSA Hash-and-Sign: 2 kilobits

ECDSA: around 512 bits

BLS: 256 bits

Are 128-bit signatures possible?
Obfuscation-Based Signatures

Let $(\text{MAC,Ver})$ be a message authentication code

$\text{Gen()}$: $k \leftarrow K$
- $\text{sk} = k$
- $\text{pk} = \text{Obf}(\text{Ver}(k, . , . ))$

$\text{Sign}(\text{sk},m) = \text{MAC}(k,m)$
$\text{Ver}(\text{pk},m,\sigma) = \text{pk}(m,\sigma)$

Signature size: 128 bits!
- But running time, public key size is horrible
Next Time

Identification protocols: how to prove you are who you say you are
Reminders

HW6 Due Wednesday

HW7 out Tonight