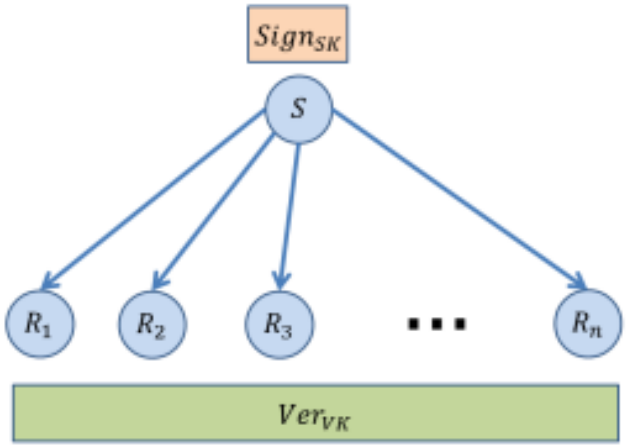
Introduction to Modern Cryptography
Recitation 9

Orit Moskovich
Tel Aviv University
December 28, 2016

Digital Signatures

- Digital signature schemes allow a *signer* with a private key sk to sign a message such that:
- Any other party who knows pk can verify that the message originated from the signer and has not been modified

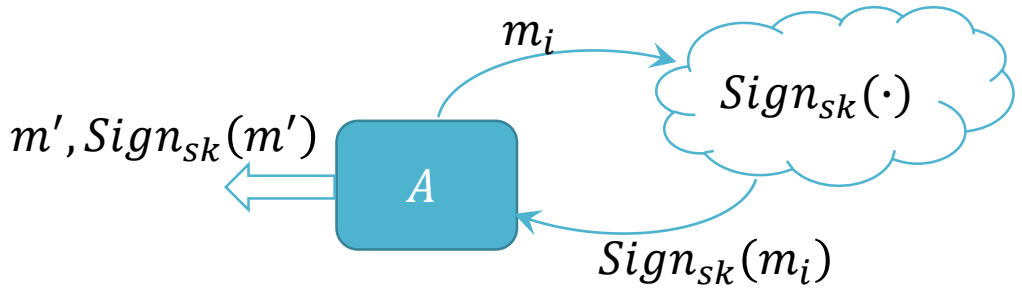


Digital Signatures

Definition. A **digital signature** is a tuple of probabilistic polynomial-time algorithms $(Gen, Sign, Vrfy)$ such that:

1. Gen outputs a private signature key sk and a public verification key pk
2. $Sign$ receives for input a key sk and a message m and outputs a signature $\sigma = Sign_{sk}(m)$
3. $Vrfy$ receives for input a key pk, a signature σ and a message m and verify its correctness, i.e., $Vrfy_{pk}(m, \sigma) = b$ such that $Vrfy_{pk}\big(m, Sign_{sk}(m)\big) = 1$

Security of Digital Signatures



The signature experiment for A:

1. A random key pk, sk is chosen
2. The adversary A is given pk and oracle access to $Sign_{sk}(\cdot)$. Let Q denote the queries asked by A during the execution
3. A outputs a pair (m, σ)

$$A \text{ forges a valid signature} \Leftrightarrow Vrfy_{pk}(m, \sigma) = 1 \text{ and } m \notin Q$$

Definition. A signature scheme is (ε, t)-**existentially unforgeable under an adaptive chosen message attack** $(t = |Q|)$ if for any PPT adversary A:

$$\Pr[A \text{ forges a valid singature}] < \varepsilon$$

RSA

- Gen: generates N, e, d (where $N = pq$, and $ed = 1 \, mod \, \phi(N)$)

$$pk = N, e$$

$$sk = d$$

- $Sign$: to sign a message $m \in \mathbb{Z}_N^*$:

$$\sigma = m^d \, mod \, N$$

- $Vrfy$: to verify a message $m \in \mathbb{Z}_N^*$ and a signature σ:

$$m =^? \sigma^e \, mod \, N$$

RSA

- A no-message attack:

- Choose arbitrary σ
- Compute $m := \sigma^e \ mod \ N$
- Output the forgery m, σ

RSA

- Forging a signature on an arbitrary message:

- Say the adversary want to forge a signature for a message m
- Chose a random m_1 and set m_2 such that $m = m_1 \cdot m_2$ ($m_2 := m \cdot m_1^{-1}$)
- Obtain signatures $\sigma_1 = m_1^d, \sigma_2 = m_2^d$ on m_1, m_2
- Output the forgery:

  $$m, \sigma_1 \cdot \sigma_2$$

- Correctness:
- $(\sigma_1 \cdot \sigma_2)^e \ mod\ N = \left(m_1^d \cdot m_2^d\right)^e \ mod\ N = m_1^{ed} \cdot m_2^{ed} \ mod\ N = m_1 m_2$

RSA- Bad Improvement

- Gen: generates N, e, d (where $N = pq$, and $ed = 1 \; mod \; \phi(N)$)

$$pk = N, e$$

$$sk = d$$

- $Sign$: to sign a message $m \in \mathbb{Z}_N^*$:

$$\sigma = (m + 1)^d \; mod \; N$$

- $Vrfy$: to verify a message $m \in \mathbb{Z}_N^*$ and a signature σ:

$$m + 1 =^? \sigma^e \; mod \; N$$

RSA- Bad Improvement

- Is this secure?

- Chose a random m_1 and m_2
- Obtain signatures $\sigma_1 = (m_1 + 1)^d, \sigma_2 = (m_2 + 1)^d$ on m_1, m_2
- Output the forgery:

$$m_1 m_2 + m_1 + m_2, \sigma_1 \cdot \sigma_2$$

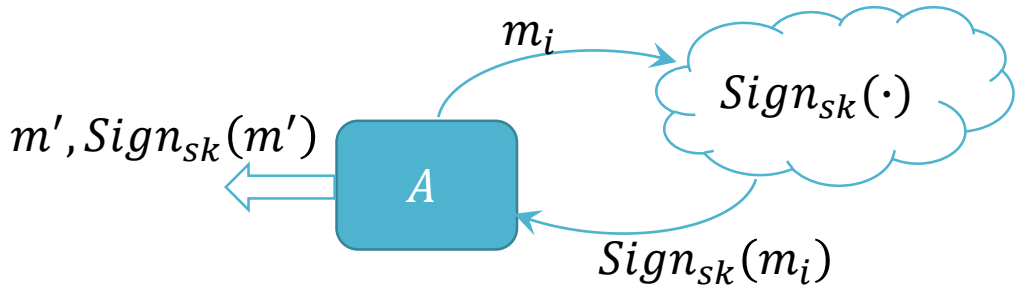
- Correctness:
- $\sigma_1 \cdot \sigma_2 \; mod \; N = (m_1 + 1)^d \cdot (m_2 + 1)^d \; mod \; N =$

$$\big((m_1 + 1)(m_2 + 1)\big)^d = (m_1 m_2 + m_1 + m_2 + 1)^d$$

One-Time Signature Scheme

- We define one-time signature schemes which are secure as long as they are used to sign only a single message

One-Time Signature Scheme



The signature experiment for A_1:

1. A random key pk, sk is chosen
2. The adversary A_1 is given pk and oracle access to $Sign_{sk}(\cdot)$.
3. A_1 is allowed to query for a single message m'
4. A_1 outputs a pair (m, σ)

A_1 forges a valid signature $\Leftrightarrow Vrfy_{pk}(m, \sigma) = 1$ and $m \neq m'$

Definition. A signature scheme is $(\varepsilon, 1)$-**existentially unforgeable under a single message attack** if for any PPT adversary A_1:

$$\Pr[A \text{ forges a valid singature}] < \varepsilon$$

Lamport's One-Time Signature Scheme

- Let f be a OWF
- Gen: Choose at random values for sk
- sk: $\begin{pmatrix} x_{1,0} & x_{2,0} & \cdots & x_{t,0} \\ x_{1,1} & x_{2,1} & \cdots & x_{t,1} \end{pmatrix}$
- And set:
- pk: $\begin{pmatrix} y_{1,0} & y_{2,0} & \cdots & y_{t,0} \\ y_{1,1} & y_{2,1} & \cdots & y_{t,1} \end{pmatrix} = \begin{pmatrix} f(x_{1,0}) & f(x_{2,0}) & \cdots & f(x_{t,0}) \\ f(x_{1,1}) & f(x_{2,1}) & \cdots & f(x_{t,1}) \end{pmatrix}$

Lamport's One-Time Signature Scheme

- Let f be a OWF
- sk: $\begin{pmatrix} x_{1,0} & x_{2,0} & \cdots & x_{t,0} \\ x_{1,1} & x_{2,1} & \cdots & x_{t,1} \end{pmatrix}$
- pk: $\begin{pmatrix} y_{1,0} & y_{2,0} & \cdots & y_{t,0} \\ y_{1,1} & y_{2,1} & \cdots & y_{t,1} \end{pmatrix} = \begin{pmatrix} f(x_{1,0}) & f(x_{2,0}) & \cdots & f(x_{t,0}) \\ f(x_{1,1}) & f(x_{2,1}) & \cdots & f(x_{t,1}) \end{pmatrix}$

Lamport's One-Time Signature Scheme

- Let f be a OWF
- sk: $\begin{pmatrix} x_{1,0} & x_{2,0} & \cdots & x_{t,0} \\ x_{1,1} & x_{2,1} & \cdots & x_{t,1} \end{pmatrix}$
- pk: $\begin{pmatrix} y_{1,0} & y_{2,0} & \cdots & y_{t,0} \\ y_{1,1} & y_{2,1} & \cdots & y_{t,1} \end{pmatrix} = \begin{pmatrix} f(x_{1,0}) & f(x_{2,0}) & \cdots & f(x_{t,0}) \\ f(x_{1,1}) & f(x_{2,1}) & \cdots & f(x_{t,1}) \end{pmatrix}$
- $sign_{sk}(m) = sign_{sk}(m_1 \dots m_t) = x_{1,m_1} \dots x_{t,m_t}$
- $vrfy_{pk}(m, x_{1,m_1} \dots x_{t,m_t}) = 1 \Leftrightarrow \forall_i . f(x_{i,m_i}) = y_{i,m_i}$

Lamport's One-Time Signature Scheme

- How do we prove security?
- Let A_1 be a PPT adversary such that

1. A random key pk, sk is chosen (according to Lempart's scheme)
2. The adversary A_1 is given pk a one time oracle access to $Sign_{sk}(\cdot)$.
3. A_1 is allowed to query for a single message m'
4. A_1 outputs a pair (m, σ)

- We'll prove:
  If $\Pr[A_1 \text{ forges a valid singature}] > \varepsilon$ ➔
  There exists an inverter A_f such that $\Pr[A_f \text{ inverts } f] > \frac{\varepsilon}{2t}$

Lamport's One-Time Signature Scheme

- Let A_f be a PPT adversary that given $f(x)$ finds x:

1. For every $i \in \{0, \dots, t\}, b \in \{0,1\}$:
   - Choose a random $x_{i,b}$ and set $y_{i,b} = f(x_{i,b})$
2. A_f chooses a random $i^* \leftarrow \{0, \dots, t\}, b \leftarrow \{0,1\}$ and set $y_{i^*,b^*} = f(x)$
3. Run A_1 on $pk = \begin{pmatrix} y_{1,0} & y_{2,0} & \cdots & y_{t,0} \\ y_{1,1} & y_{2,1} & \cdots & y_{t,1} \end{pmatrix}$
4. When A_1 requests for a signature on a message m':
   - If $m_{i^*} = b^*$, stop
   - Otherwise, return the correct signature $\sigma = (x_{1,m_1}, \dots, x_{t,m_t})$
5. A_1 outputs a pair (m, σ):
   - If $m_{i^*} = b^*$ (A_1 outputs a forgery at (i^*, b^*)), then output x_{i^*}

Lamport's One-Time Signature Scheme

- Assume $\Pr[A_1 \text{ forges a valid singature}] > \varepsilon$
- $\Pr[A_f \text{ inverts } f] =$

$$\Pr[A_f \text{ passes step 4}] \cdot \Pr[A_1 \text{ forges a valid signature}] \cdot \Pr[A_1 \text{ forges signature on } (i^*, b^*)]$$

$$> \frac{1}{2} \cdot \varepsilon \cdot \frac{1}{t} = \frac{\varepsilon}{2t}$$

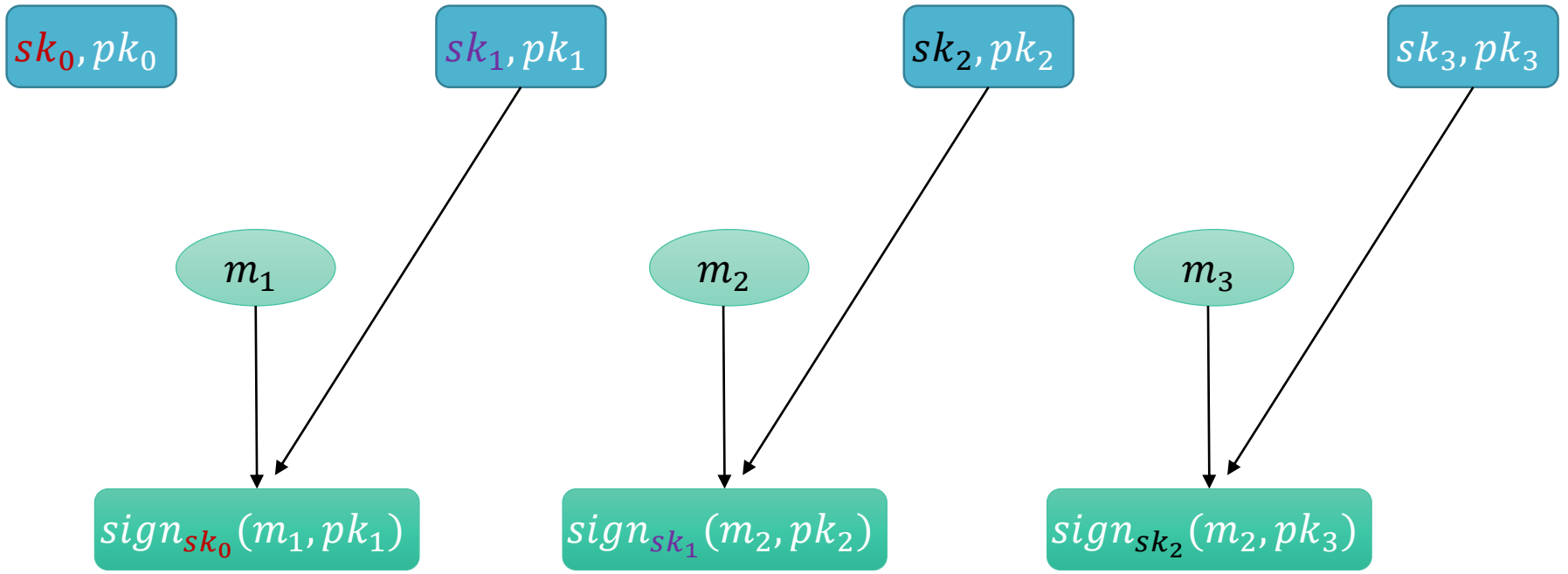
Lamport's One-Time Signature Scheme

- Issues with the scheme:
  - Good for one message
  - Signature length increases according to m – solved using a CRH

“Chain-Based” Signatures

- Goal: To construct a signature scheme that is existentially unforgeable under an adaptive chosen-message attack
- We show an inefficient construction

"Chain-Based" Signatures



- $Vrfy: \{Vrfy_{pk_{j-1}}(m_j, pk_j, \sigma_j)\}_{j \leq i}$

"Chain-Based" Signatures

- A sequential multi-message stateful signature scheme:
- Gen: Initially one-time keys are sampled (sk_0, pk_0)
  - The initial state of the signer is $state_0 = sk_0$
- $Sign$:
  - $state_{i-1}$ includes:
    - All previous messages $m_1, \dots m_{i-1}$
    - Previous one-time keys $sk_0, \dots, sk_{i-1}$ and $pk_0, \dots, pk_{i-1}$
    - Previous one-time signatures $\sigma_1, \dots, \sigma_{i-1}$
  - To sign m_i:
    - Sample a new pair (sk_i, pk_i)
    - $\sigma_i = sign_{sk_{i-1}}(m_i, pk_i)$
    - Add to $state_i$: $m_i, sk_i, pk_i, \sigma_i$
    - Publish: $\{pk_i, m_i, \sigma_i\}_{j \le i}$
- $Vrfy$: verifying all signatures along the chain: $\left\{Vrfy_{pk_{j-1}}\left(m_j, pk_j, \sigma_j\right)\right\}_{j \le i}$

“Chain-Based” Signatures

- Security - HW

El Gamal Signature Scheme

El Gamal PKE Scheme. Let $g \in \mathbb{Z}_p^*$ be a generator and H a CRH.

1. Key generation algorithm Gen chooses a random $x \in \mathbb{Z}_p$.
   $pk = (G, q, g, g^x), sk = (G, q, g, x)$
2. To sign a message M using $sk = (G, q, g, x)$:
   - $m = H(M)$
   - Choose random $k \in \mathbb{Z}_p$ such that $\gcd(k, p-1) = 1$
   - Compute $r = g^k \bmod p$ and $s = (m - xr) \cdot k^{-1} \bmod p - 1$
   - Output $M, (r, s)$
3. To verify a message M and a signature r, s using $pk = (G, q, g, h)$:
   - Compute $m = H(M)$
   - Verify $h^r r^s =^? g^m \bmod p$ and $0 < r < p, 0 < s < p - 1$

El Gamal Signature Scheme

El Gamal PKE Scheme. Let $g \in \mathbb{Z}_p^*$ be a generator and H a CRH.

1. Key generation algorithm Gen chooses a random $x \in \mathbb{Z}_p$.
   $pk = (G, q, g, g^x), sk = (G, q, g, x)$
2. To sign a message M using $sk = (G, q, g, x)$:
   - $m = H(M)$
   - Choose random $k \in \mathbb{Z}_p$ such that $\gcd(k, p-1) = 1$
   - Compute $r = g^k \bmod p$ and $s = (m - xr) \cdot k^{-1} \bmod p - 1$
   - Output $M, (r, s)$
3. To verify a message M and a signature r, s using $pk = (G, q, g, h)$:
   - Compute $m = H(M)$
   - Verify $h^r r^s =^? g^m \bmod p$ and $0 < r < p$, $0 < s < p - 1$

$ks + xr = m \bmod p - 1$

El Gamal Signature Scheme

El Gamal PKE Scheme. Let $g \in \mathbb{Z}_p^*$ be a generator and H a CRH.

1. Key generation algorithm Gen chooses a random $x \in \mathbb{Z}_p$.
   $pk = (G, q, g, g^x), sk = (G, q, g, x)$
2. To sign a message M using $sk = (G, q, g, x)$:
   - $m = H(M)$
   - Choose random $k \in \mathbb{Z}_p$ such that $\gcd(k, p-1) = 1$
   - Compute $r = g^k \bmod p$ and $s = (m - xr) \cdot k^{-1} \bmod p - 1$

$ks + xr = m \bmod p - 1$

$h^r r^s = (g^x)^r (g^k)^s = g^{xr+ks} = g^m$

nd a signature r, s using $pk = (G, q, g, h)$:

$m = H(M)$

- Verify $h^r r^s =^? g^m \bmod p$ and $0 < r < p, 0 < s < p - 1$

El Gamal

- Show that signing two messages with the same k is insecure:
- Let $m_1 \neq m_2$
- $\sigma_1 = (H(m_1) - xr)k^{-1} \; mod \; p - 1$
- $\sigma_2 = (H(m_2) - xr)k^{-1} \; mod \; p - 1$

- Then, $\sigma_1 - \sigma_2 = \big(H(m_1) - H(m_2)\big)k^{-1} \; mod \; p - 1$
- Find the inverse of $\sigma_1 - \sigma_2$
- Recover k
- Now we can find x