Lecture 10

Douglas Wikström
KTH Stockholm
dog@csc.kth.se

April 17, 2015

Signature Schemes

Digital Signature

- A digital signature is the **public-key** equivalent of a MAC; the receiver verifies the integrity and authenticity of a message.
- Does a digital signature replace a real handwritten one?

Textbook RSA Signature (1/2)

- Generate RSA keys $((N, e), (p, q, d))$.
- To sign a message $m \in \mathbb{Z}_N$, compute $\sigma = m^d \bmod N$.
- To verify a signature σ of a message m, verify that $\sigma^e = m \bmod N$.

Textbook RSA Signature (2/2)

- Are Textbook RSA Signatures any good?

Textbook RSA Signature (2/2)

- Are Textbook RSA Signatures any good?
- If σ is a signature of m, then $\sigma^2 \bmod N$ is a signature of $m^2 \bmod N$.

Textbook RSA Signature (2/2)

- Are Textbook RSA Signatures any good?
- If σ is a signature of m, then $\sigma^2 \bmod N$ is a signature of $m^2 \bmod N$.
- If σ_1 and σ_2 are signatures of m_1 and m_2, then $\sigma_1\sigma_2 \bmod N$ is a signature of $m_1 m_2 \bmod N$

Textbook RSA Signature (2/2)

- Are Textbook RSA Signatures any good?
- If σ is a signature of m, then $\sigma^2 \bmod N$ is a signature of $m^2 \bmod N$.
- If σ_1 and σ_2 are signatures of m_1 and m_2, then $\sigma_1\sigma_2 \bmod N$ is a signature of $m_1 m_2 \bmod N$
- We can also pick a signature σ and compute the message it is a signature of by $m = \sigma^e \bmod N$.

Textbook RSA Signature (2/2)

- Are Textbook RSA Signatures any good?
- If σ is a signature of m, then $\sigma^2 \bmod N$ is a signature of $m^2 \bmod N$.
- If σ_1 and σ_2 are signatures of m_1 and m_2, then $\sigma_1\sigma_2 \bmod N$ is a signature of $m_1 m_2 \bmod N$
- We can also pick a signature σ and compute the message it is a signature of by $m = \sigma^e \bmod N$.

We must be more careful!

Signature Scheme

- Gen **generates a key pair** $(\mathsf{pk}, \mathsf{sk})$.
- Sig takes a secret key sk and a message m and **computes a signature** σ.
- Vf takes a public key pk, a message m, and a candidate signature σ, **verifies the candidate signature**, and outputs a single-bit verdict.

Existential Unforgeability

Definition. A signature scheme $(\mathsf{Gen}, \mathsf{Sig}, \mathsf{Vf})$ is **secure against existential forgeries** if for every polynomial time algorithm and a random key pair $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^n)$,

$$\Pr\left[A^{\mathsf{Sig}_{\mathsf{sk}}(\cdot)}(\mathsf{pk}) = (m, \sigma) \wedge \mathsf{Vf}_{\mathsf{pk}}(m, \sigma) = 1 \wedge \forall i : m \neq m_i\right]$$

is negligible where m_i is the ith query to $\mathsf{Sig}_{\mathsf{sk}}(\cdot)$.

Trapdoor One-Way Permutations

Let $f = \{f_\alpha\}$ be an ensemble of **permutations** (bijections).

- Gen **generates a random key pair** $\alpha = (\mathsf{pk}, \mathsf{sk})$.
- Eval takes pk and x as input and **efficiently evaluates** $f_\alpha(x)$.
- Invert takes sk and y as input and **efficiently evaluates the inverse** $f_\alpha^{-1}(y)$.

One-way if $\mathsf{Eval}_{\mathsf{pk}}(\cdot)$ is one-way for a random pk.

Trapdoor One-Way Permutations

Let $f = \{f_\alpha\}$ be an ensemble of **permutations** (bijections).

- Gen **generates a random key pair** $\alpha = (\mathsf{pk}, \mathsf{sk})$.
- Eval takes pk and x as input and **efficiently evaluates** $f_\alpha(x)$.
- Invert takes sk and y as input and **efficiently evaluates the inverse** $f_\alpha^{-1}(y)$.

One-way if $\mathsf{Eval}_{\mathsf{pk}}(\cdot)$ is one-way for a random pk.

RSA is a trap-door permutation over $\mathbb{Z}_N^*$.

Trapdoor One-Way Permutations (Less Formal)

Let $f = \{f_\alpha\}$ be an ensemble of **permutations** (bijections).

- Gen **generates a pair** $(f_\alpha, f_\alpha^{-1})$.
- Eval takes pk and x as input and **efficiently evaluates** $f_\alpha(x)$.
- Invert takes sk and y as input and **efficiently evaluates the inverse** $f_\alpha^{-1}(y)$.

One-way if f_α is one-way when chosen randomly.

RSA is a trap-door permutation over $\mathbb{Z}_N^*$.

Full Domain Hash Signature In ROM

Let $f = \{f_\alpha\}$ be a trapdoor permutation (family) and let $H : \{0,1\}^* \to \{0,1\}^n$ be a random oracle.

- Gen samples a pair $(f_\alpha, f_\alpha^{-1})$.
- Sig takes f_α^{-1} and a message m as input and outputs $f_\alpha^{-1}\big(H(m)\big)$.
- Vf takes f_α, a message m, and a candidate signature σ as input, and outputs 1 if $f_\alpha(\sigma) = H(m)$ and 0 otherwise.

Proof of Knowledge of Exponent

In an **identification scheme** one party convinces another that it holds some special token.

Proof of Knowledge of Exponent

In an **identification scheme** one party convinces another that it holds some special token.

- Let G_q be a group of prime order q with generator g.

Proof of Knowledge of Exponent

In an **identification scheme** one party convinces another that it holds some special token.

- Let G_q be a group of prime order q with generator g.
- Let $x \in \mathbb{Z}_q$ and define $y = g^x$.

Proof of Knowledge of Exponent

In an **identification scheme** one party convinces another that it holds some special token.

- Let G_q be a group of prime order q with generator g.
- Let $x \in \mathbb{Z}_q$ and define $y = g^x$.
- Can we prove knowledge of x without disclosing anything about x?

Schnorr's Signature Scheme (1/3)

1. The prover chooses $r \in \mathbb{Z}_q$ randomly and hands $\alpha = g^r$ to the verifier.

Schnorr's Signature Scheme (1/3)

1. The prover chooses $r \in \mathbb{Z}_q$ randomly and hands $\alpha = g^r$ to the verifier.
2. The verifier chooses $c \in \mathbb{Z}_q$ randomly and hands it to the prover.

Schnorr's Signature Scheme (1/3)

1. The prover chooses $r \in \mathbb{Z}_q$ randomly and hands $\alpha = g^r$ to the verifier.
2. The verifier chooses $c \in \mathbb{Z}_q$ randomly and hands it to the prover.
3. The prover computes $d = cx + r \bmod q$ and hands d to the verifier.

Schnorr's Signature Scheme (1/3)

1. The prover chooses $r \in \mathbb{Z}_q$ randomly and hands $\alpha = g^r$ to the verifier.
2. The verifier chooses $c \in \mathbb{Z}_q$ randomly and hands it to the prover.
3. The prover computes $d = cx + r \mod q$ and hands d to the verifier.
4. The verifier accepts if $y^c\alpha = g^d$.

Schnorr's Signature Scheme (1/3)

1. The prover chooses $r \in \mathbb{Z}_q$ randomly and hands $\alpha = g^r$ to the verifier.
2. The verifier chooses $c \in \mathbb{Z}_q$ randomly and hands it to the prover.
3. The prover computes $d = cx + r \bmod q$ and hands d to the verifier.
4. The verifier accepts if $y^c \alpha = g^d$.

Suppose that a machine convinces us in the protocol with probability δ. Does it mean that it knows x such that $y = g^x$?

Schnorr's Signature Scheme (2/3)

Schnorr's Signature Scheme (2/3)

1. Run the machine to get α.

Schnorr's Signature Scheme (2/3)

1. Run the machine to get α.

2. Complete the interaction twice using **the same** α, once for a challenge c and once for a challenge c', where $c, c' \in \mathbb{Z}_q$ are chosen randomly.

Schnorr's Signature Scheme (2/3)

1. Run the machine to get α.

2. Complete the interaction twice using **the same** α, once for a challenge c and once for a challenge c', where $c, c' \in \mathbb{Z}_q$ are chosen randomly.

3. Repeat from (1) until the resulting interactions (α, c, d) and (α, c', d') are accepting and $c \neq c'$.

Schnorr's Signature Scheme (2/3)

1. Run the machine to get α.

2. Complete the interaction twice using **the same** α, once for a challenge c and once for a challenge c', where $c, c' \in \mathbb{Z}_q$ are chosen randomly.

3. Repeat from (1) until the resulting interactions (α, c, d) and (α, c', d') are accepting and $c \neq c'$.

4. Note that:

$$y^{c-c'} = \frac{y^c}{y^{c'}} = \frac{y^c\alpha}{y^{c'}\alpha} = \frac{g^d}{g^{d'}} = g^{d-d'}$$

which gives the logarithm $x = (d - d')(c - c')^{-1} \bmod q$ such that $y = g^x$.

Schnorr's Signature Scheme (3/3)

- Anybody can sample $c, d \in \mathbb{Z}_q$ randomly and compute $\alpha = g^d / y^c$.
- The resulting tuple (α, c, d) has **exactly** the same distribution as the transcript of an interaction!

Such protocols are called (honest verifier) **zero-knowledge proofs of knowledge**.