Signature Schemes

Truong Tuan Anh
CSE-HCMUT

Outline

- Signature schemes
- RSA signature scheme

Context

- A “conventional” handwritten signature
  - Attached to a document
  - Specify the person responsible for the document
  - Used in everyday situations: writing a letter. Withdrawing money, signing a contract, …
- Electronic document?

→ Digital Signatures = Signature Schemes

- A method of signing a message stored in electronic form and can be transmitted over a computer network

Digital vs. Conventional Signatures

- Signing a document
  - Conventional Signatures: a part of the physical document being signed
  - Digital Signatures: not attached physically to the message being signed; There is an algorithm to "bind" the signature to the message
- Verification
  - Conventional Signatures: compare to authentic signatures; Not a very secure method
  - Digital Signatures: can be verified by a public verification algorithm; Prevent the possibility of forgeries

Signature Schemes

- A signature Scheme: consists of two components
  - A signing algorithm
  - A verification algorithm
- Alice signs a message using the private signing algorithm
- Bob verifies the signature using Alice's corresponding public verification algorithm

Signature Schemes

A *signature scheme* is a five-tuple $(\mathcal{P}, \mathcal{A}, \mathcal{K}, \mathcal{S}, \mathcal{V})$, where:

1. $\mathcal{P}$ is a finite set of possible *messages*
2. $\mathcal{A}$ is a finite set of possible *signatures*
3. $\mathcal{K}$, the *keyspace*, is a finite set of possible *keys*
4. For each $K \in \mathcal{K}$, there is
   - a *signing algorithm* $\mathbf{sig}_K \in \mathcal{S}$ and
   - a corresponding *verification algorithm* $\mathbf{ver}_K \in \mathcal{V}$.

Signature Schemes (cont.)

Each

$$\boxed{\begin{aligned} \mathbf{sig}_K &: \mathcal{P} \rightarrow \mathcal{A} \text{ and} \\ \mathbf{ver}_K &: \mathcal{P} \times \mathcal{A} \rightarrow \{true, false\} \end{aligned}}$$

are functions such that the following equation is satisfied:

for every message $x \in \mathcal{P}$ and for every signature $y \in \mathcal{A}$:

$$\boxed{\mathbf{ver}_K(x, y) = \begin{cases} true & \text{if } y = \mathbf{sig}_K(x) \\ false & \text{if } y \neq \mathbf{sig}_K(x). \end{cases}}$$

A pair (x, y) with $x \in \mathcal{P}$ and $y \in \mathcal{A}$ is called a *signed message*.

Signature Schemes: Notes

- The functions $\mathbf{sig}_K$ and $\mathbf{ver}_K$ should be polynomial-time functions
- $\mathbf{sig}_K$ is the private function and $\mathbf{ver}_K$ is the public function
- It should be computationally infeasible for anyone than Alice to compute a signature y such that $\mathbf{ver}_K(x,y) = true$
- If Oscar can compute y such that $\mathbf{ver}_K(x,y) = true$, then y is a forgery

Signature Schemes: An Example

- RSA cryptosystem can be used to provide digital signatures

→**RSA signature scheme**

- Alice signs a message using the RSA decryption rule $\mathbf{d}_K = \mathbf{sig}_K$
- Anyone can verify the signature using the RSA encryption rule $\mathbf{e}_K = \mathbf{ver}_K$
- Anyone can forge Alice's RSA signature by randomly choosing y and computing if $x = \mathbf{e}_K(y)$

Signature Schemes: An Example

RSA Signature Scheme

Let $n = pq$, where p and q are primes. Let $\mathcal{P} = \mathcal{A} = \mathbb{Z}_n$, and define

$$\mathcal{K} = \{(n, p, q, a, b) : n = pq, p, q \text{ prime}, ab \equiv 1 \pmod{\phi(n)}\}.$$

The values n and b are the public key, and the values p, q, a are the private key.

For $K = (n, p, q, a, b)$, define

$$\boxed{\mathbf{sig}_K(x) = x^a \bmod n}$$

and

$$\boxed{\mathbf{ver}_K(x, y) = \text{true} \Leftrightarrow x \equiv y^b \pmod{n}}$$

$$(x, y \in \mathbb{Z}_n).$$

RSA Signature Scheme: How it works

- Alice wishes to send an encrypted, signed message to Bob (given the plaintext is x)
  1. Alice computes her signature $y = \mathbf{sig}_{Alice}(x)$
  2. Alice encrypts both x and y using $\mathbf{e}_{Bob}$, then $z = \mathbf{e}_{Bob}(x, y)$
  3. Alice sends z to Bob
  4. Bob receives z
  5. Bob decrypts z using $\mathbf{d}_{Bob}$ to get x, y
  6. Bob uses Alice's <u>public verification function</u> to check that $\mathbf{ver}_{Alice}(x, y) = \text{true}$

RSA Signature Scheme: How it works

- Alice wishes to send an encrypted, signed message to Bob (given the plaintext is x)
  1. Alice encrypts both x using $\mathbf{e}_{Bob}$, then $z = \mathbf{e}_{Bob}(x)$
  2. Alice computes her signature $y = \mathbf{sig}_{Alice}(z)$
  3. Alice sends (z, y) to Bob
  4. Bob receives (z, y)
  5. Bob decrypts z using $\mathbf{d}_{Bob}$ to get x
  6. Bob uses Alice's <u>public verification function</u> to check that $\mathbf{ver}_{Alice}(z, y) = \text{true}$
- What is the problem?

Signature Schemes: In Reality