Cryptography

Cunsheng DING, HKUST

COMP3631

Lecture 16: A Digital Signature Scheme

Outline of This Lecture

- Introduce some basic requirements for digital signature.
- Define hash functions.
- Introduce a digital signature scheme involving a public-key cipher and a hash function.
- Present three forgery attacks on the digital signature scheme.
- Investigate security requirements for the underlying hash function and public-key cipher.

Handwritten Signature and Digital Signature

- The singer nonrepudiation security service was introduced in Lecture 3.

  — It means the **detection** of the repudiation of a signature on a document by the signer.

- While our handwritten signature or our finger print on hard copy documents is used for singer nonrepudiation, digital signature on electrical documents can be created also for signer nonrepudiation.

- A digital signature on an electrical document is a string of characters computed from the document.

Basic Requirements for Digital Signature

- The digital signature must depend on the digital document being signed and thus should vary from document to document.
- The digital signature must use some information unique to the signer, to prevent both forgery and denial.
- It must be computationally infeasible to forge a digital signature,
  – either by constructing a new message for an existing digital signature
  – or by constructing a fraudulent digital signature for a given message.
- It must be relatively easy to produce the digital signature.
- It must be relatively easy to recognize and verify the digital signature.
- It must be practical to retain a copy of the digital signature in storage.

Hash Functions

Hash Functions

Definition: A **hash function** h is a function from a **subset** of the set of all finite strings of characters from an alphabet A_1 (denoted by A_1^*) to the set of all strings of characters from an alphabet A_2 **with a fixed length** ℓ (denoted by A_2^ℓ).

For any x, $h(x)$ is called the **hash value**, or **message digest**.

Attentions:

- The domain of h is a subset of A_1^*. The input message x for $h(x)$ has a variable length and does not have an upper bound on its length in general.
- The range of h is a subset of A_2^ℓ for some fixed integer ℓ. The length of each hash value $h(x)$ is always ℓ.

Remark: A hash function h is publicly known for many applications.

A Digital Signature Scheme

The Building Blocks of the Digital Signature Scheme

Building blocks: A public-key cipher $(\mathcal{M}, \mathcal{C}, \mathcal{K}_e, \mathcal{K}_d, E_{k_e}, D_{k_d})$, and a hash function h.

A basic condition on the two building blocks:
The **range** of h is a subset of the **domain** of D_{k_d}. This guarantees that the function composition $D_{k_d} \circ h$ is well defined.

Attention: Both the public key cipher and the hash function are put in the public domain, so that signature verification can be done by a verifier.

The Digital Signature Scheme: Signing Process

Alice can use her private key $k_d^{(A)}$ to **sign** messages as follows.

- To sign a message m, Alice computes $h(m)$, which is called the **message digest**.
- She then uses her private key to sign on the message digest, obtaining $D_{k_d^{(A)}}(h(m))$, i.e., *her digital signature* on m. Then she sends the data $m||D_{k_d^{(A)}}(h(m))$ to the receiver Bob.

  — This explains why the range of h must be a subset of the domain of D_{k_d}.

Question: How does Bob verify the message and the digital signature?

Properties of the Digital Signature

The text from Alice to Bob: $m||D_{k_d^{(A)}}(h(m))$

Property 1: The digital signature $D_{k_d^{(A)}}(h(m))$ has a fixed length for all messages m.

Property 2: The message m and the digital signature $D_{k_d^{(A)}}(h(m))$ have the following relationship:

$$h(m) = E_{k_e^{(A)}}\left(D_{k_d^{(A)}}(h(m))\right).$$

Thus, if the received text c by Bob was indeed created by Alice, and is partitioned into $c = c_1||c_2$, where c_2 has the same length as the digital signature, then

$$h(c_1) = E_{k_e^{(A)}}(c_2).$$

This relation is the basis of the digital signature verification process.

The Digital Signature Scheme: Signature Verification

- Bob partitions the received text c into two parts $c_1||c_2$, where c_2 has the same length as the digital signature.
- Then he uses Alice's public key $k_e^{(A)}$ to compute $E_{k_e^{(A)}}(c_2)$.
- Then he computes $h(c_1)$ (the hash function is publicized).
- Finally, he compares $h(c_1)$ with $E_{k_e^{(A)}}(c_2)$.

  If $h(c_1) = E_{k_e^{(A)}}(c_2)$, he accepts $c_1||c_2$ as a valid message with signature from Alice.
  - In this case, $c_1||c_2$ may be a modified or forged one. But the probability of this event should be very small if the public-key cipher and h are "**well designed**" (to be explained later).

  If $h(c_1) \neq E_{k_e^{(A)}}(c_2)$, he is sure that c_2 is not the digital signature on c_1 created by Alice.

Security Requirements of the Two Building Blocks

Question: How should we design the public-key cipher and the hash function so that the success probability of forging the signer's digital signature is very small?

Answer: The answer depends on specific feasible forgery attacks on the digital signature scheme.

Security Requirements of the Two Building Blocks

First Forgery Attack on the Digital Signature Scheme

The Attack: After receiving the text $m||D_{k_d^{(A)}}(h(m))$, Bob or someone else uses the m and h in the public domain to find a different message m' such that $h(m') = h(m)$. Such a pair (m, m') is called a **weak collision**.

If this is computationally feasible, the attacker can claim that $D_{k_d^{(A)}}(h(m))$ is Alice's signature on m'. In this case, the forgery is successful.

Design requirement w.r.t. the first forgery attack: Given a message m, it should be computationally infeasible to find a different message m' such that $h(m) = h(m')$. This is called the **weak collision resistance property.**

Strong collisions: A pair (m, m') of different messages such $h(m) = h(m')$ is called a **strong collision**., where m and m' are of **free choice**.

Second Forgery Attack on the Digital Signature Scheme

The Attack: You find out a pair of messages m_1 and m_2 such that

1. m_2 has the same length as the digital signature; and
2. $h(m_1) = E_{k_e^{(A)}}(m_2)$.

If this is computationally feasible, you can claim that m_2 is Alice's digital signature on m_1.

There are two different ways to find such a pair (m_1, m_2).

(A) You may first choose m_2 freely and then find m_1 such that $h(m_1) = E_{k_e^{(A)}}(m_2)$.

(B) Another way is to choose m_1 freely first, then determine m_2 using the equation $h(m_1) = E_{k_e^{(A)}}(m_2)$.

Second Forgery Attack on the Digital Signature Scheme

Design requirements w.r.t. the second forgery attack:

(A) Given h and v, it should be computationally infeasible to find a message m such that $h(m) = v$.

- Such an h is called a *one-way hash function*. This is referred to as the *one-way property*.

(B) Given a piece of ciphertext c , it should be computationally infeasible to recover a message m such that $c = E_{k_e^{(A)}}(m)$.

- This is the Condition C2 that a public-key cipher should meet (see Lecture 10).

Third Forgery Attack on the Digital Signature Scheme

The third forgery attack: You try to compute the private key $k_d^{(A)}$ from Alice's public key $k_e^{(A)}$. If this is computationally feasible, you can forge Alice's digital signature on any message m.

Security requirement w.r.t. to this attack:

- It should be computationally infeasible to derive the private key $k_d^{(A)}$ from Alice's public key $k_e^{(A)}$.
- This is the Condition C1 that a public-key cipher should meet (see Lecture 10).

Security Requirements of the Two Building Blocks

Security requirements: Summarizing our analyses before, we have the following:

- The public-key cipher should be computationally secure. i.e., Conditions C1 and C2 should be met.
- The hash function h should have the one-way property.
- The hash function h should have the weak collision resistance property.

Remark: Specific hash functions used in real-world security systems will be introduced in the next lecture.

Remark: One-way hash functions are used in some operating systems for protecting the passwords of users.

Designing an h for the RSA in the Digital Signature Scheme

- Let $n = pq$ be the modulus of the RSA public-key cipher and $k = \lfloor \log_2 n \rfloor$ be the plaintext block size, where p and q are large primes.
- Let h be a hash function whose hash values are binary strings of length $\ell \leq k$ (ℓ should be as close to k as possible). This guarantees that the range of h is a subset of the $\mathbf{Z}_n$ (the domain of the RSA decryption function).

  The hash function h should have the one-way property and weak-collision resistance property.
- Some of the hash functions available in the public domain are believed to be such hash functions. They will be introduced in the next lecture.

Such a hash function h and the RSA public key cipher can be plugged into the digital signature scheme.