Computer Security

Cunsheng DING, HKUST

COMP4631

Lecture 9: Introduction to Public-Key Cryptography

Objectives of this Lecture

1. Introduce the idea of public-key cryptography.
2. Present the history of public-key cryptography.
3. Outline three applications of public-key ciphers.

A Disadvantage of One-Key Block Ciphers

A Disadvantage of One-Key Block Ciphers

One-key block ciphers: $(\mathcal{M}, \mathcal{C}, \mathcal{K}, E_k, D_k)$, where the encryption and decryption keys are the same.

- The sender and receiver must share the same secret key. Key distribution is a must.
- If 10000 people want to communicate (two and two, in all possible ways), each must keep 9999 secret keys, and the system requires a total of

$$9999 \cdot 10000/2 = 4995000$$

secret keys. This makes key management difficult.

The Idea of Public-Key Cryptography

Two-key Ciphers

A six-tuple $(\mathcal{M}, \mathcal{C}, \mathcal{K}_e, \mathcal{K}_d, E_{k_e}, D_{k_d})$, where

- $\mathcal{M}$, $\mathcal{C}$, $\mathcal{K}_e, \mathcal{K}_d$ are respectively the plaintext space, ciphertext space, encryption key space, and decryption key space;
- $k_e \in \mathcal{K}_e$ and $k_d \in \mathcal{K}_d$ are corresponding encryption and decryption keys respectively;
- E_{k_e} and D_{k_d} are the encryption and decryption transformations, and

  $$D_{k_d}(E_{k_e}(m)) = m,$$

  for all $m \in \mathcal{M}$ (unique and correct decryption).

The Idea of Public-Key Cryptography

Suppose that I have a two-key cipher $(\mathcal{M}, \mathcal{C}, \mathcal{K}_e, \mathcal{K}_d, E_{k_e}, D_{k_d})$.
I generate my encryption and decryption pair (k_e, k_d), and then publicize k_e together with the encryption algorithm in the public domain, in order for anybody else to encrypt a message and send it to me. Such a two-key cipher is called a **public-key cipher**.

Comment: The encryption key k_e is called the **public key**, and the decryption key k_d is called the **private key**.

The Security of Public-Key Ciphers

A public-key cipher $(\mathcal{M}, \mathcal{C}, \mathcal{K}_e, \mathcal{K}_d, E_{k_e}, D_{k_d})$ is **computationally secure** if and only if the following two conditions are satisfied:

C1: it is "computationally infeasible" to derive the decryption key k_d from the given encryption key k_e; and

C2: it is "computationally infeasible" to derive the plaintext m if the corresponding ciphertext c is known.

Comments:

- C1 and C2 are not rigorously defined in the mathematical sense.
- If one of the two conditions is not satisfied, the public-key cipher is insecure.

A Public-key Cipher not Satisfying C1 & C2

Matrix: An $n \times m$ matrix $A = [a[i, j]]$ over $\{0, 1\}$ is a 2-dimensional array

$$A = \begin{bmatrix} a[1,1] & a[1,2] & \cdots & a[1,m-1] & a[1,m] \\ a[2,1] & a[2,2] & \cdots & a[2,m-1] & a[2,m] \\ \vdots & \vdots & & \vdots & \vdots \\ a[n,1] & a[n,2] & \cdots & a[n,m-1] & a[n,m] \end{bmatrix},$$

which has n rows and m columns, and each $a[i, j] \in \{0, 1\}$.

A Public-key Cipher not Satisfying C1 & C2

Given an $n \times m$ matrix A and an $m \times l$ matrix B, the multiplication $C = AB$ over F_2 is an $n \times l$ matrix given by

$$c[i,j] = \sum_{k=1}^{m} a[i,k]b[k,j]$$

for $1 \leq i \leq n$ and $1 \leq j \leq l$, where operations in the sum are mudulo-2 additions and mudulo-2 multiplications.

A Public-key Cipher not Satisfying C1 & C2

$$A = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \qquad B = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix},$$

then

$$C = AB = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$$

A Public-key Cipher not Satisfying C1 & C2

Definition: Let A be an $n \times n$ matrix over F_2. If there exists an $n \times n$ matrix $B \in F_2$ such that $AB = I_n$, i.e., the $n \times n$ identity matrix, then A is said **invertible**, and B is the **inverse matrix** of A.

Example: A is the inverse of itself:

$$A = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

A Public-key Cipher not Satisfying C1 & C2

Let $\mathcal{M} = \mathcal{C} = \{0,1\}^*$, all the finite binary strings, and let $\mathcal{K}$ be the set of all invertible 512×512 matrices k over $F_2 = \{0,1\}$ with $k \neq k^{-1}$. Each message is broken into blocks of length 512 bits. The encryption and decryption algorithms work on blocks.

Encryption and decryption: For a 512-bit plaintext block x and ciphertext block y,

$$E_k(x) = kx, \quad D_{k^{-1}}(y) = k^{-1}y,$$

where all the arithmetic operations involved in computing kx are modulo-2, and $(k_e, k_d) = (k, k^{-1})$

Comment: C1 and C2 are not statisfied. Why?

Design Requirements for Public-Key Ciphers

The C1 and C2 described before plus the following efficiency requirements:

1. It is "computationally easy" for a party B to generate a pair $\left(k_e^{(B)}, k_d^{(B)}\right)$.
2. It is "computationally easy" for a sender A, knowing the public key and the message to be encrypted, m, to generate the corresponding ciphertext $c = E_{k_e^{(B)}}(m)$.
3. It is "computationally easy" for the receiver B to recover the message $m = D_{k_d^{(B)}}(c)$.

Existence and Construction Problems

Question: Is there any public-key cipher meeting the five requirements described in the previous page?

Answer: Several public-key ciphers in the literature are believed to meet these requirements. But there is no proof.

How to construct a public-key cipher?
Use a problem that is believed to be hard to solve, e.g., the discrete logarithm problem.

Advantages and Disadvantages

- With a public-key cipher, a user does not need to share many keys with others. This is an advantage of public-key ciphers over private-key ciphers.
- The **disadvantage** of public-key ciphers is their performance in hardware and software, as no **efficient** and **secure** public-key cipher is known.

History of Public-Key Cryptography

History of Public-Key Cryptography (I)

- The idea of public-key cryptography was published by W. Diffie and M. Hellman, and independently by R. Merkle in 1976. It is regarded as a REVOLUTION in the history of cryptography!
- Admiral Bobby Inman, while director of the NSA, claimed that public-key cryptography had been discovered at NSA in the mid-1960s.
- The first (???) *documented* introduction of these concepts was given in 1970 by the Communications-Electronics Security Group, Britain's counterpart of NSA, in a classified report by James Ellis.

History of Public-Key Cryptography (II)

- The Knapsack public-key cipher was developed by Ralph Merkle and Martin Hellman in 1978, but was broken in 1982 by Shamir and Zipple.
- In the same year (1978), another public-key block cipher was invented by Ron **R**ivest, Adi **S**hamir, and Leonard **A**dleman. It is known as RSA. It is easy to understand and to implement, and is one of a few that are still regarded as secure. It is widely used in real-world security systems.
- Many other public-key ciphers have been proposed. Most of them have been broken.

Three Applications of Public-Key Ciphers

Application in Encryption

Given a public-key cipher $(\mathcal{M}, \mathcal{C}, \mathcal{K}_e, \mathcal{K}_d, E_{k_e}, D_{k_d})$:

- Alice generates a key pair $\left(k_e^{(A)}, k_d^{(A)}\right)$, keeps the decryption key $k_d^{(A)}$ confidential, and publishes the encryption key $k_e^{(A)}$ and the encryption algorithm in a public directory.
- If Bob wants to send a message m to Alice, he finds Alice's encryption key $k_e^{(A)}$ and the encryption algorithm in the public directory, encrypts the message to get $c = E_{k_e^{(A)}}(m)$, and sends c to A.
- After receiving c, Alice uses her decryption key and computes

$$D_{k_d^{(A)}}(c) = D_{k_d^{(A)}}(E_{k_e^{(A)}}(m)) = m.$$

Application in Key Distribution

Session key: Two parties want to communicate using a one-key cipher for encryption. They need a session key for each session of communication.

Session key distribution with a public-key cipher

- Alice generates a session key k and then sends $E_{k_e^{(B)}}(k)$ to Bob.
- Bob uses his private key $k_e^{(B)}$ to decrypt $E_{k_d^{(B)}}(k)$ and recovers k.

Remark: The $E_{k_e^{(B)}}(k)$ is called a **digital envelope**.

Application in Digital Signature

Suppose that we have a public-key cipher $(\mathcal{M}, \mathcal{C}, \mathcal{K}_e, \mathcal{K}_d, E_{k_e}, D_{k_d})$ with $\mathcal{M} = \mathcal{C}$.

Then we can use such a system to **sign** messages.

- To sign a message m, the sender applies a public hash function f to m obtaining $f(m)$, which is called the **message digest**.
- He then uses his private key to sign on the message digest, obtaining $D_{k_d}(f(m))$. Then he sends the data $m||D_{k_d}(f(m))$ to the receiver.

Question: Why do we need a hash function here?

Application in Digital Signature – Continued

Checking the validity of signature

- To check the validity of the sender's signature, the receiver breaks the received message c into two parts $m'||c_2$, where c_2 has a fixed length (i.e., the length of the signature). Then he uses the sender's public key to obtain $E_{k_e}(c_2)$.
- He computes $f(m)$, (the hash function is public).
- Finally, he compares $f(m)$ with $E_{k_e}(c_2)$. If they match, he accepts $m'||c_2$ as a valid message with signature from the sender. Otherwise he rejects it.

Applications of Public-Key Cryptography

Three types of applications:

Encryption, digital signature, key distribution.

Comments: Some public-key ciphers can be used for all the three applications, while others can be used only for two of these applications.

This will be made clear later when we cover specific public-key ciphers.