Lesson 2
Public-key cryptosystems

A. Sorokin (Input | Output)

Self paced course

July 31, 2025

Cryptosystem revisited

Recall that a **cryptosystem** consists of two algorithms called

encryption and **decryption**

such that the decryption of an encrypted message coincides with the original message.

A communication in a cryptosystem diagramatically takes form

$$\begin{tikzcd}
\boxed{\text{Sender}} \arrow[d, "\text{message}"] & & \boxed{\text{Receiver}} \\
\boxed{\text{Encryption}} \arrow[r, "\text{ciphered}", "\text{message}"'] & \boxed{\text{Channel}} \arrow[r, "\text{ciphered}", "\text{message}"'] & \boxed{\text{Decryption}} \arrow[u, "\text{message}"]
\end{tikzcd}$$

Standard terminology

The following notions are usually used in cryptography:

- Alice – a party who sends a message;
- Bob – a party who receives a message;
- Eve – a non-malicious party who eavesdrops a message from Alice to Bob;
- Mallory – a malicious party who attacks the communication and can impersonate other parties;
- plaintext – Alice's original message before encryption;
- ciphertext – Alice's original message after encryption.

Communication in cryptosystem

Using the above standard terminology we get a diagram

$$
\begin{tikzcd}
\boxed{\text{Alice}} \arrow[d, "\text{plaintext}"] & \boxed{\text{Eve}} & \boxed{\text{Bob}} \\
\boxed{\text{Encryption}} \arrow[r, "\text{ciphertext}"] & \boxed{\text{Channel}} \arrow[u, "\text{ciphertext}"'] \arrow[r, "\text{ciphertext}"] \arrow[d, shift right=2ex, "\text{ciphertext}"'] & \boxed{\text{Decryption}} \arrow[u, "\text{plaintext}"] \\
 & \boxed{\text{Mallory}} \arrow[u, shift right=2ex, "\text{altered ciphertext}"'] &
\end{tikzcd}
$$

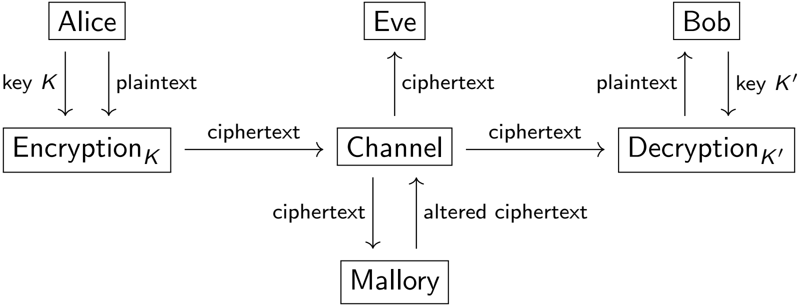
Keys

Often both encryption and decryption depend on a certain data which is essential for them. Such data is called a **key**. Actually we have a pair of keys

(i) K – a key for encryption,
(ii) K' – a key for decryption.

A knowledge of a decryption key K' by Eve or Mallory breakes the security of communication.

Keys in cryptosystem



(A)symmetric cryptosystems

Depending on a difference of encryption and decryption keys, cryptosystems are described as being

(i) **symmetric** – the information required to encrypt a plaintext is the same as the one required to decrypt a ciphertext:

$$K = K'$$

(ii) **asymmetric** – the information required to encrypt a plaintext differs from the one required to decrypt a ciphertext:

$$K \neq K'$$

Symmetric cryptosystems

Symmetric cryptosystems are in general stronger, faster, require less memory, and are easier to implement than asymmetric ones .

However, any *symmetric cryptosystem* immediately faces a key exchange problem:

how to exchange a key securely between comunicating parties to establish a connection?

Moreover, for any pair of users should be a unique key that they are agreed upon.

These issues make symmetric cryptosystems not suitable for a secure communication without any additional assumptions, comparing with asymmetric ones.

Public-key cryptosystem

Asymmetric cryptosystems satisfy the following properties:

(i) everyone can encrypt a plaintext,

(ii) only the receiver can decrypt a ciphertext.

So encryption and decryption algorithms of asymmetric cryptosystem are sufficiently different.

For this reason there is no reason to keep the encryption key K secret, and another name for a asymmetric cryptosystem is a

public-key cryptosystem

Public and private keys

In a public-key cryptosystem an encryption key K is called a **public key**, while a decryption key K' is called a **private key**.

Together a pair (K, K') is called a **public-private key pair**.

Public-key cryptosystem setup

In order to define a public-key cryptosystem one needs:

(i) an effective algorithm of generating public-private key pairs (K, K');

(ii) an effective encryption algorithm $Encr_K$ dependent on a public key K;

(iii) an effective decryption algorithm $Decr_{K'}$ dependent on a private key K';

(iv) for any plaintext M the following equality holds:

$$Decr_{K'}(Encr_K(M)) = M$$

(v) there are no effective algorithms how to find a plaintext M given a ciphertext $C = Encr_K(M)$, without knowledge about the private key K'.

Communication in public-key cryptosystem

Alice and Bob use the following protocol to communicate in a public-key cryptosystem.

(i) Bob generates a public-private key pair

$$(K_B, K_B')$$

and publishes his public key K_B.

(ii) Alice learns Bob's public key K_B, encrypts her plaintext M as

$$C = Enrc_{K_B}(M)$$

and sends Bob a ciphertext C.

(iii) Bob receives C and decrypts it as

$$M = Decr_{K_B'}(C)$$

using his private key K_B'.

Communication in public-key cryptosystem

$$
\begin{tikzcd}
 & \boxed{K_B} \arrow[d] & & \boxed{K_B'} \arrow[d] & \\
\boxed{\text{Alice}} \arrow[r, "M"] & \boxed{\text{Encryption}_{K_B}} \arrow[rr, "C"] & & \boxed{\text{Decryption}_{K_B'}} \arrow[r, "M"] & \boxed{\text{Bob}}
\end{tikzcd}
$$

Public-key cryptosystem features

In any public-key cryptosystem

(i) any user can encrypt and send their plaintexts using receiver's public key,

(ii) only a private key holder can decrypt a ciphertext,

(iii) for a secure communication between a group of users it is sufficient to a public-private key pair for each user.

Besides a possibility to create authentication and being scalable public-key cryptosystems are slower, harder to implement, and require more memory comparing to symmetric ones.

Symmetric vs asymmetric cryptosystems

Attribute	**Symmetric**	**Asymmetric**
Key size	Short (128-256 bits)	Long ($\geq$ 1024 bits)
Speed	Faster	Slower
Security	Strong	Strong
Implementation	Easy	Hard
Key exchange	Problematic	Easy
Authentication	Problematic	Easy
Scalablility	Problematic: a key for each pair of users	Easy: one key per user

Combining symmetric and asymmetric cryptosystems

In practice it is good to merge symmetric and asymmetric cryptosystems:

(i) use an asymmetric cryptosystem for identity verification and to exchange keys for symmetric encryption;

(ii) use symmetric cryptosystem for data encryption and sharing.

Examples

Symmetric cryptosystems:

AES, ChaCha20, Serpent, Blowfish

Asymmetric (public-key) cryptosystems:

RSA, ElGamal, ECC, Rabin