Cryptography

Sources:

- *Applied Cryptography* by B. Schneier (primary source)
- *Making, Breaking Codes: An Introduction to Cryptology* by P. Garrett
- *Cryptography Decrypted* by H. Mel and D. Baker
- *Security in Computing* by C.P. Pfleeger
- *Cryptography and Network Security* by W. Stallings

Terminology

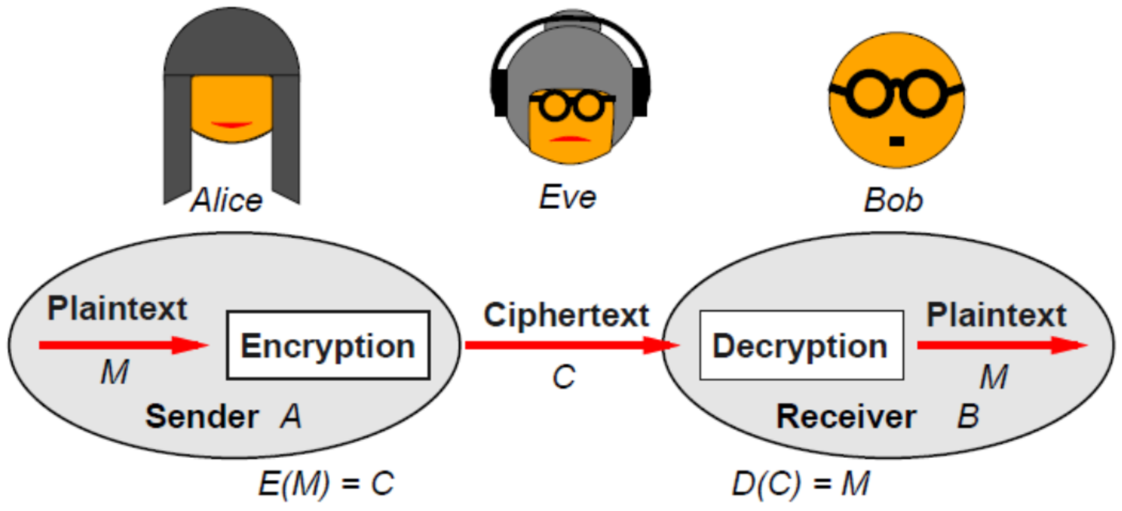
Cleartext: intelligible message, input to encryption

Plaintext: input to encryption that may or may not be intelligible

Encryption: process of disguising message

Ciphertext: encrypted, unintelligible message

Decryption: normal process of turning ciphertext back into cleartext



[wwwhome.cs.utwente.nl/~sape/sse/schneier01.pdf]

Cryptography: art and science of keeping messages secure

Cryptanalysis: art and science of breaking ciphertext

Cryptology: branch of mathematics encompassing cryptography and cryptanalysis

The encryption function E operates on a message M to produce a ciphertext C:

$$E(M) = C$$

The decryption function D operates on C to produce M:

$$D(C) = M$$

Hence,

$$D(E(M)) = M$$

Cryptography is used for other purposes besides ensuring confidentiality:

- *Authentication* enables the receiver of a message to ascertain who sent it.

- *Integrity* means that an intruder cannot substitute a false message for a legitimate one without the receiver being able to detect this.

- *Nonrepudiation* means that the sender of a message cannot falsely deny having sent it.

Algorithms and Keys

A *cryptographic algorithm* defines the mathematical functions used for encryption and decryption.

A *restricted* cryptographic algorithm is one whose security requires keeping the algorithm secret.

Restricted algorithms are impractical with a large or changing group of users.

They also don't permit quality control or standardization.

Modern cryptographic algorithms are not restricted; they may even be published.

Their security derives from the use of one or more *keys*.

A key K is typically an element of a large number space, called the *keyspace*.

Both encryption and decryption operations use a key.

There are two types of key-based algorithms: symmetric and public-key.

Cryptosystems

Cryptosystem: encryption and decryption algorithms plus formats for messages and keys.

Shannon characterized a desirable cryptosystem as follows:

- The ciphertext message space is the space of all possible messages.
- The cleartext message space is a *sparse region* inside the message space, in which messages have a fairly simple statistical structure.
- A (good) encryption algorithm is a *mixing transformation* that distributes cleartext messages fairly *uniformly* over the entire message space.

Semantic security of an encryption algorithm means that ciphertext has a distribution in the message space that is indistinguishable from the uniform distribution on the same space.

Symmetric Algorithms

With *symmetric algorithms*, the decryption key can be calculated from the encryption key.

With most symmetric algorithms, the same key is used for encryption and decryption.

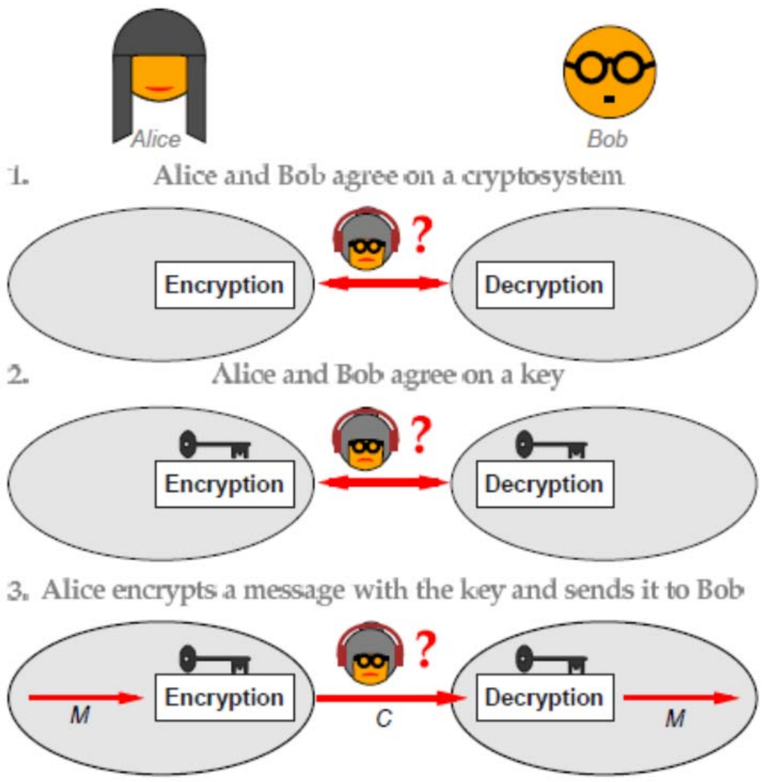
The sender and receiver must agree on a key before they can communicate secretly.

The key must be kept secret from others.

Encryption and decryption with symmetric algorithm are denoted as follows:

$$E_K(M) = C$$

$$D_K(C) = M$$


Alice
Bob
1. Alice and Bob agree on a cryptosystem
Encryption
Decryption
?
2. Alice and Bob agree on a key
Encryption
Decryption
?
3. Alice encrypts a message with the key and sends it to Bob
M
Encryption
C
Decryption
M
?

Symmetric Cryptography

[wwwhome.cs.utwente.nl/~sape/sse/schneier02.pdf]

There are two types of symmetric algorithms:

- *Stream algorithms* or *stream ciphers* operate on the plaintext one bit (or byte) at a time.

- *Block algorithms* or *block ciphers* operate on groups of bits called *blocks*.

A typical block size is 128 bits.

Public-Key Algorithms

With *public-key* or *asymmetric* algorithms, different keys are used for encryption and decryption.

Moreover, it is not feasible to compute the decryption key from the encryption key.

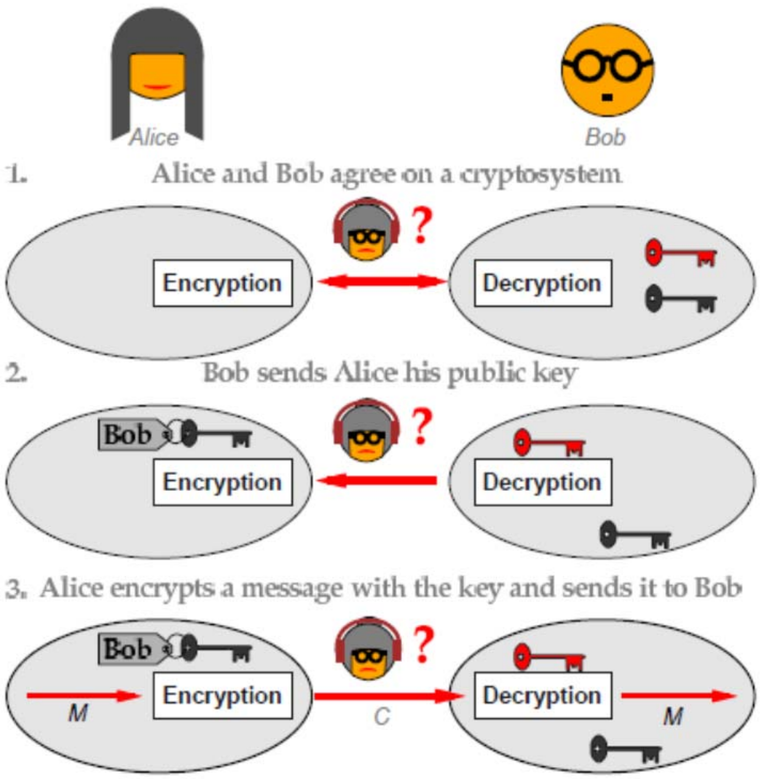
This means the encryption key can be made public.

Hence, it is often called the *public key*, and the decryption key is called the *private key*.

Public key encryption and decryption can be denoted as follows:

$$E_{K1}(M) = C$$

$$D_{K2}(C) = M$$


Alice
Bob
1. Alice and Bob agree on a cryptosystem
Encryption
Decryption
?
2. Bob sends Alice his public key
Bob
Encryption
Decryption
?
3. Alice encrypts a message with the key and sends it to Bob
Bob
Encryption
Decryption
M
C
M
?

Public-Key Cryptography

[wwwhome.cs.utwente.nl/~sape/sse/schneier02.pdf]

Cryptanalysis

Cryptanalysis is the science of recovering the plaintext of a message without access to the key(s).

An attempted cryptanalysis is called an *attack*.

Kerchoff's Principle: assume that the cryptanalyst knows the cryptographic algorithm and key size

It should also be assumed that eavesdroppers have access to communications between the sender and receiver.

Types of cryptanalytic attacks [Schneier]:

- *Ciphertext-only attack*: The cryptanalyst has the ciphertext of several messages to work with and attempts to deduce the corresponding plaintexts or the key(s).

- *Known-plaintext attack*: The cryptanalyst has access to both the ciphertext and the plaintext of several messages and attempts to deduce the key(s) or an algorithm to infer the plaintexts.

- *Chosen-plaintext attack*: The cryptanalyst not only has access to the ciphertext and plaintext of several messages, but he chooses the plaintext that gets encrypted. He attempts to deduce the key(s) or an algorithm to infer the plaintexts.

- *Adaptive-chosen-plaintext attack*: A chosen-plaintext attack in which the cryptanalyst can choose the plaintext to be encrypted based on the results of previous encryption.

- *Chosen-ciphertext attack*: The cryptanalyst can choose different ciphertexts to be decrypted and has access to the decrypted plaintext. He attempts to deduce the key(s).

Note that the cryptanalyst can also use theft, bribery, extortion, etc. to obtain the key(s).

Security of Algorithms

A cryptographic algorithm is *unconditionally secure* if it is impossible for a cryptanalyst to recover the plaintext, regardless of how much ciphertext they have.

Only a “one-time pad” is unbreakable in this sense.

Every other cryptosystem is breakable by trying every possible key one by one.

This is called a *brute-force* attack.

An algorithm is *computationally secure* if it cannot be broken with available resources.

The complexity of an attack is generally taken as the maximum of these three factors:

- The amount of data needed as input to the attack
- The time needed to perform the attack
- The amount of storage needed for the attack

You are safe if:

- The cost required to break an algorithm is greater than the value of the encrypted data.

- The time required to break an algorithm is longer than the time the encrypted data must remain secret.

- The amount of data encrypted with a single key is less than the amount of data needed to break the algorithm.

One-Time Pads

A classical *one-time pad* is a large set of truly random key letters, written on sheets of paper and glued together in a pad.

The sender uses each key letter on the pad to encrypt exactly one plaintext character.

Encryption is by addition modulo 26 of the plaintext character and the one-time pad key character.

Each key letter is used exactly once, for only one message.

The sender destroys used parts of the pad.

The receiver has an identical pad and uses it to decrypt the ciphertext.

Decryption is by subtraction modulo 26 of each key letter from the corresponding ciphertext letter.

Suppose the message consists of n characters $x = (x_1, x_2, ..., x_n)$.

We choose a key consisting of n random characters $k = (k_1, k_2, ..., k_n)$.

Each character is viewed as an integer in the range 0..25.

The one-time pad encryption function is defined by:

$$E_k(x) = ((x_1 + k_1) \% 26, (x_2 + k_2) \% 26, ..., (x_n + k_n) \% 26)$$

The decryption function is defined by:

$$D_k(y) = ((y_1 - k_1) \% 26, (y_2 - k_2) \% 26, ..., (y_n - k_n) \% 26)$$

Example [Garrett]: Suppose the plaintext is "IMPOSSIBLE" coded by integers:

$x = (8, 12, 15, 14, 18, 18, 8, 1, 11, 4)$

Let the key be

$k = (8, 13, 24, 19, 9, 1, 0, 7, 20, 3)$

Then

$E_k(x) = ((8 + 8) \% 26, (12 + 13) \% 26, ..., ((4 + 3) \% 26)$

$= (16, 25, 13, 7, 1, 19, 8, 8, 5, 7)$

= "QZNHBTIIFH"

With a one-time pad, an adversary has no information with which to cryptanalyze the ciphertext, since every key sequence is equally likely.

Key letters have to truly random – a pseudorandom generator cannot be used.

Note that you cannot reuse the key sequence again – ever.

The biggest obstacles to the use of one-time pads are secure key distribution and storage.

They are applicable primarily for ultra-secure, low bandwidth channels.

One time pads are often implemented by taking the *bitwise XOR* of a message and a key.

This operation is widely used in the design of modern secret key encryption algorithms, because it can be implemented by a circuit.