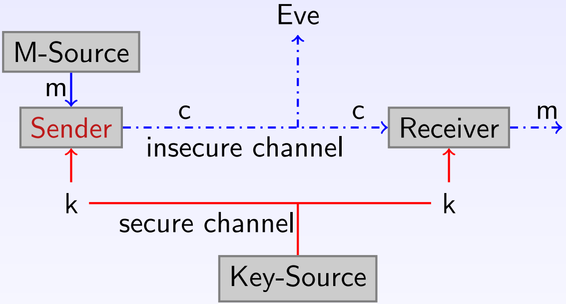
Syrvey on block ciphers

Anna Rimoldi

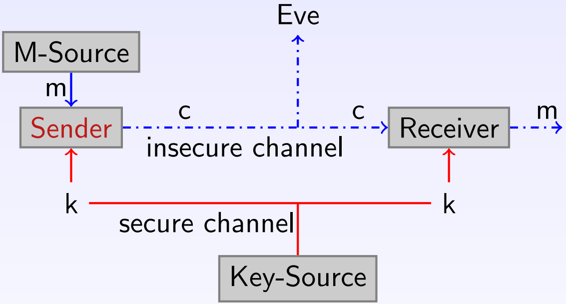
Department of Mathematics - University of Trento

BunnyTn 2012

Symmetric Key Cryptosystem

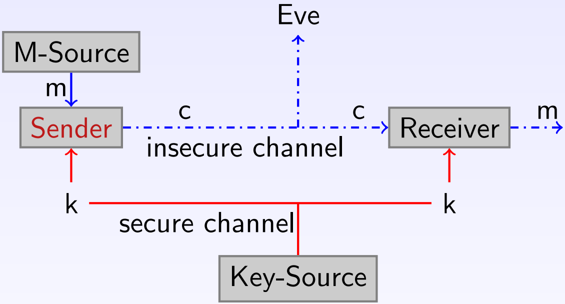


Symmetric Key Cryptosystem



Everything is known to an attacker except for the value of the secret key.

Symmetric Key Cryptosystem



Everything is known to an attacker except for the value of the secret key.

Possible attack scenarios:

- Known plaintext
- Chosen plaintext/ ciphertext

Symmetric key Cryptosystem

Following the most used structure in modern ciphers, we assume that the plaintext space coincides with the ciphertext space ($\mathcal{P} = \mathcal{C} = \mathcal{M}$)

Symmetric key Cryptosystem

Following the most used structure in modern ciphers, we assume that the plaintext space coincides with the ciphertext space ($\mathcal{P} = \mathcal{C} = \mathcal{M}$)

Definition

A cryptosystem is a pair $(\mathcal{M}, \mathcal{K})$, where:

- $\mathcal{M}$ is a finite set of possible messages (plaintexts, ciphertexts);
- $\mathcal{K}$, the key-space, is a finite set of possible keys;

Symmetric key Cryptosystem

Following the most used structure in modern ciphers, we assume that the plaintext space coincides with the ciphertext space ($\mathcal{P} = \mathcal{C} = \mathcal{M}$)

Definition

A cryptosystem is a pair $(\mathcal{M}, \mathcal{K})$, where:

- $\mathcal{M}$ is a finite set of possible messages (plaintexts, ciphertexts);
- $\mathcal{K}$, the key-space, is a finite set of possible keys;
- we have encryption and decryption functions for any key $k \in \mathcal{K}$:

$$\phi_k : \mathcal{M} \to \mathcal{M}, \quad \psi_k : \mathcal{M} \to \mathcal{M},$$

  such that

$$\psi_k = (\phi_k)^{-1}.$$

1. Let $\mathcal{M} = (\mathbb{F}_2)^n$ and $\mathcal{K} = (\mathbb{F}_2)^\ell$, with n and ℓ positive integers.
2. Same key k for encryption and decryption.
3. There are two main types of symmetric key algorithm:
    - block ciphers: these are algorithms that encrypt and decrypt blocks of data (with fixed length) according to the shared secret key.
    - stream ciphers.

1. Let $\mathcal{M} = (\mathbb{F}_2)^n$ and $\mathcal{K} = (\mathbb{F}_2)^\ell$, with n and ℓ positive integers.
2. Same key k for encryption and decryption.
3. There are two main types of symmetric key algorithm:
    - block ciphers: these are algorithms that encrypt and decrypt blocks of data (with fixed length) according to the shared secret key.
    - stream ciphers.

Possible Applications

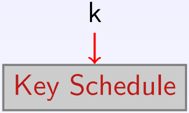
- block encryption (symmetric)
- pseudo random number generator
- stram ciphers
- building block in hash functions
- one-way functions

Structure of a block cipher

The block cipher is divided in two distinct parts:

Structure of a block cipher

The block cipher is divided in two distinct parts:

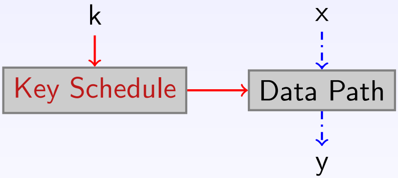


Key Schedule Algorithm

Public algorithm that elaborates the secret key and costructs $N + 1$ subkeys.

Structure of a block cipher

The block cipher is divided in two distinct parts:



Key Schedule Algorithm

Public algorithm that elaborates the secret key and costructs $N + 1$ subkeys.

Encryption function

A commonly used design is that of an iterated cipher:

1. Encryption of a plaintext proceeds through N similar rounds;
2. Round Function;

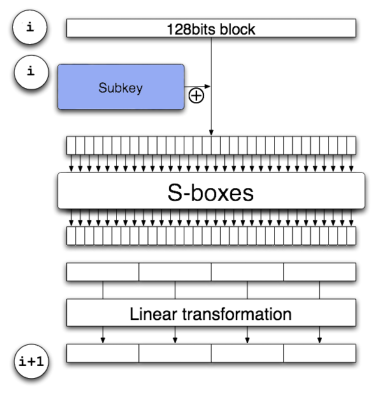
Usual parameters

Block cipher	n	ℓ	N
AES	128	128, 192, 256	10, 12, 14
SERPENT	128	128	32
PRESENT	64	80	31

Round function

In any *round* we have:

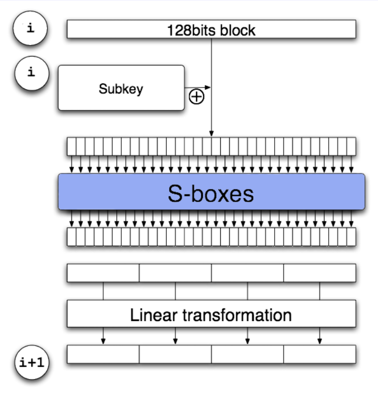
- add round key:
  the i-th round key k_i is added (XORed) k_i to the intermediate vector;
- a *non-linear* operation within groups of bits [S-box];
- a *linear* (or affine) transformation of the whole intermediate vector.



Round function

In any *round* we have:

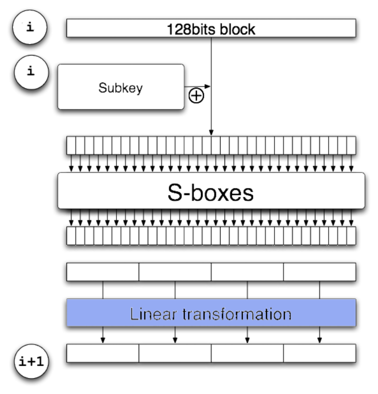
- add round key:
  the i-th round key k_i is added (XORed) k_i to the intermediate vector;
- a *non-linear* operation within groups of bits [S-box];
- a *linear* (or affine) transformation of the whole intermediate vector.



Round function

In any *round* we have:

- add round key:
  the i-th round key k_i is added (XORed) k_i to the intermediate vector;
- a *non-linear* operation within groups of bits [S-box];
- a *linear* (or affine) transformation of the whole intermediate vector.



Is a block cipher secure?

- Consider the key space and the block size; Is brute force feasible?
- Consider Mathematical attacks
- Consider implementation attacks.

Is a block cipher secure?

- Consider the key space and the block size; Is brute force feasible?
- Consider Mathematical attacks → analyze mathematical structure
- Consider implementation attacks.

The S-boxes

- play a fundamental role for the security of nearly all modern block ciphers;

The S-boxes

- play a fundamental role for the security of nearly all modern block ciphers;
- form the only non-linear part of a block cipher;

The S-boxes

- play a fundamental role for the security of nearly all modern block ciphers;
- form the only non-linear part of a block cipher;
- have to chosen carefully to make the cipher resistant all kinds of attacks.

The S-boxes

- play a fundamental role for the security of nearly all modern block ciphers;
- form the only non-linear part of a block cipher;
- have to chosen carefully to make the cipher resistant all kinds of attacks.

There are well studied criteria that a good block cipher has to fulfill to make it resistant against differential and linear cryptanalysis.

S-boxes

- There are mainly two way of generation good S-boxes

S-boxes

- There are mainly two way of generation good S-boxes
  1. picking a random large S-box;

S-boxes

- There are mainly two way of generation good S-boxes
    1. picking a random large S-box;
    2. generating small S-boxes with good linear and differential properties.

S-boxes

- There are mainly two way of generation good S-boxes
  1. picking a random large S-box;
  2. generating small S-boxes with good linear and differential properties.
- Most modern block ciphers uses 4 or 8 S-boxes (AES uses 8 bit , SERPENT uses 4 bit, PRESENT uses 4-bit).

S-boxes

- There are mainly two way of generation good S-boxes
  1. picking a random large S-box;
  2. generating small S-boxes with good linear and differential properties.
- Most modern block ciphers uses 4 or 8 S-boxes (AES uses 8 bit , SERPENT uses 4 bit, PRESENT uses 4-bit).
- The problem to find optimal S-boxes is very hard:

> the number of permutations mapping m bits to m bits is huge even for very small value of m.

S-boxes

Design issues

1. The sboxLayer has to maximize the nonlinearity
2. It has to be cheap.
3. The sboxLayer consists of a number of S-boxes executed in parallel $\gamma_i : (\mathbb{F}_2)^b \rightarrow (\mathbb{F}_2)^b$.
4. In hardware realized as Boolean functions; the bigger the S-box the more expensive it is in hardware.
5. A serialized implementation becomes smaller if all S-boxes are the same

S-boxes

Design issues

1. The sboxLayer has to maximize the nonlinearity → classification
2. It has to be cheap.
3. The sboxLayer consists of a number of S-boxes executed in parallel $\gamma_i : (\mathbb{F}_2)^b \rightarrow (\mathbb{F}_2)^b$.
4. In hardware realized as Boolean functions; the bigger the S-box the more expensive it is in hardware. $\rightarrow b = 4$
5. A serialized implementation becomes smaller if all S-boxes are the same → only one S-box

Mixing Layer

Design issues

1. The MixingLayer has to maximize the diffusion.
2. It has to be cheap.
3. Many modern block ciphers use MDS codes (good diffusion).
4. Bit permutation (no cost).

$\Downarrow$

Use less diffusion per round
Use more round.

CRYPTANALYSIS

Statistical tests

- When a statistical test on data from a cryptographic algorithm is performed, we wish to test whether the data seem random or not.

Statistical tests

- When a statistical test on data from a cryptographic algorithm is performed, we wish to test whether the data seem random or not.
- There are many different properties of randomness and non-randomness and it is possible to design tests for these specific properties.

Statistical tests

- When a statistical test on data from a cryptographic algorithm is performed, we wish to test whether the data seem random or not.
- There are many different properties of randomness and non-randomness and it is possible to design tests for these specific properties.
- Although there are many tests for disproving the randomness of a sequence, no specific finite set of tests is deemed "complete."

Statistical tests

- When a statistical test on data from a cryptographic algorithm is performed, we wish to test whether the data seem random or not.
- There are many different properties of randomness and non-randomness and it is possible to design tests for these specific properties.
- Although there are many tests for disproving the randomness of a sequence, no specific finite set of tests is deemed "complete."

Example

NIST Test Suite on the AES candidate algorithms is a statistical package consisting of 16 tests that focus on a variety of different types of non-randomness that could exist in a sequence.

Random properties

Typically the random properties of binary sequences to be tested are the following:

Random properties

Typically the random properties of binary sequences to be tested are the following:

- *Uniformity*: at any point in the generation of a sequence of bits, the occurrence of a zero or one is equally likely. The expected number of zeros (or ones) is $n/2$, where n is the sequence length.

Random properties

Typically the random properties of binary sequences to be tested are the following:

- *Uniformity*: at any point in the generation of a sequence of bits, the occurrence of a zero or one is equally likely. The expected number of zeros (or ones) is $n/2$, where n is the sequence length.
- *Scalability*: any test applicable to a sequence can also be applied to subsequences extracted at random. If a sequence is random, then any such extracted subsequence should also be random.

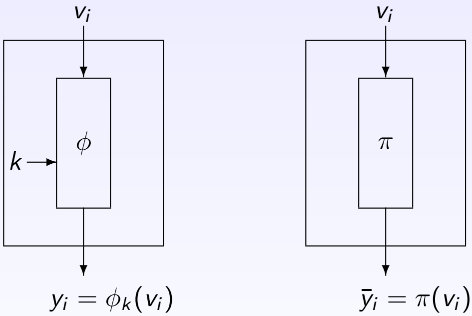
Random properties

Typically the random properties of binary sequences to be tested are the following:

- *Uniformity*: at any point in the generation of a sequence of bits, the occurrence of a zero or one is equally likely. The expected number of zeros (or ones) is $n/2$, where n is the sequence length.
- *Scalability*: any test applicable to a sequence can also be applied to subsequences extracted at random. If a sequence is random, then any such extracted subsequence should also be random.
- *Consistency*: the behavior of a generator must be consistent across starting values (seeds). It is inadequate to test a pseudo-random number generator based on the output from a single seed.

Single-key (known-key) distinguisher

Let $v_1, \ldots, v_\rho$ be some related plaintexts. Let k be a fixed key.



A distinguishing attack on $\mathcal{C}$ is any algorithm able to distinguish the ciphertexts $\{y_i\}_{1 \le i \le \rho}$ from the random ciphertexts $\{\bar{y}_i\}_{1 \le i \le \rho}$.

Outcome of an attack

The type of information recovered during an attack can be classified as

Key Recovery (Total break): Eve reconstructs the key K.

Global deduction: Eve finds an algorithm functionally equivalent to ϕ_K or ψ_K whit out knowing K.

Partial Key Recovery: Eve gets some information on the keys (relations,bits,etc..).

Distinguishing attack: Eve is able to tell whether the block cipher is a random permutation (chosen uniformly at random from the set of all permutations) or one of the permutations $\{\phi_K\}_{K\in\mathcal{K}}$.

Small scale variants

Small scale variants

- For most methods of cryptanalysis it is quite straightforward to perform experiments on reduced version of the cipher to understand how the attack might perform.

Small scale variants

- For most methods of cryptanalysis it is quite straightforward to perform experiments on reduced version of the cipher to understand how the attack might perform.

- It is difficult to design small versions that can replicate the main cryptographic and algebraic properties of the cipher.

Small scale variants

- For most methods of cryptanalysis it is quite straightforward to perform experiments on reduced version of the cipher to understand how the attack might perform.

- It is difficult to design small versions that can replicate the main cryptographic and algebraic properties of the cipher.

- The hope is that experiments on small versions can be give an idea about the behavior of cryptanalysis on block ciphers.

Perfect Secrecy

The concept of *perfect secrecy* has been formalized by Shannon.
The *perfect ciphers* are ciphers with a very strong model because one assumes that Eve's computational power is infinite.
They are impractical for a real use, as they require at least as many key bits as the message length.
Shannon gave a characterization of perfect secrecy

Theorem

Suppose that $|\mathcal{P}| = |\mathcal{C}| = |\mathcal{K}|$. *A cryptosystem provides perfect secrecy iff every key is used with equal probability* $1/|\mathcal{K}|$ *and, for every* $x \in \mathcal{P}$ *and* $y \in \mathcal{C}$, *there is a unique key* $\bar{k}$ *such that* $\phi_{\bar{k}} = y$.

Remark

Suppose that $|\mathcal{P}| = |\mathcal{C}| = |\mathcal{K}|$. *A cryptosystem provides perfect secrecy iff every key is used with equal probability* $1/|\mathcal{K}|$ *and the action of* $\{\phi_{\bar{k}}\}_{\bar{k} \in \mathcal{K}}$ *on* $\mathcal{P} = \mathcal{C}$ *is a regular action.*

Thank you for your attention!