

# Systems' Security | Segurança de Sistemas

## Introduction – Computer Security Concepts

---

Miguel Frade



## Overview

---

Introduction

The OSI security Architecture

Security attack

Security Services

Security Mechanisms

Attack Surfaces

## Introduction

---

### The OSI security Architecture

- **Security attack** – any action that compromises the security of information owned by an organization;

### The OSI security Architecture

- **Security attack** – any action that compromises the security of information owned by an organization;
- **Security mechanism** – a process (or a device incorporating such a process) that is designed to detect, prevent, or recover from a security attack;

### The OSI security Architecture

- **Security attack** – any action that compromises the security of information owned by an organization;
- **Security mechanism** – a process (or a device incorporating such a process) that is designed to detect, prevent, or recover from a security attack;
- **Security service** – a processing or communication service that enhances the security of the data processing systems and the information transfers of an organization. The services are intended to counter security attacks, and they make use of one or more security mechanisms to provide the service;

Types of Security attacks:

- **Passive attacks** – The goal of the opponent is to obtain information that is being transmitted and is in the nature of eavesdropping on, or monitoring of, transmissions. There are two types:
  - **release of message contents**
  - **traffic analysis** – observe the pattern of these messages. The opponent could determine the location and identity of communicating hosts and could observe the frequency and length of messages being exchanged. This information might be useful in guessing the nature of the communication that was taking place.

### Types of Security attacks (continuation):

- **Active Attacks** – involve some modification of the data stream or the creation of a false stream and can be subdivided into four categories:
  - **masquerade** – when one entity pretends to be a different entity and usually includes one of the other forms of active attack;
  - **replay** – passive capture of a data unit and its subsequent retransmission to produce an unauthorized effect;
  - **modification of messages** – some portion of a legitimate message is altered, or that messages are delayed or reordered, to produce an unauthorized effect
  - **denial of service** – prevents or inhibits the normal use or management of communications facilities;

## Security Services

---

### Security service

a processing or communication service that is provided by a system to give a specific kind of protection to system resources; security services implement security policies and are implemented by security mechanisms.

### Security Services:

1. **Availability** – it's a service that protects a system to ensure its availability and assure that systems work promptly and service is not denied to authorized users;

### Security Services:

1. **Availability** – it's a service that protects a system to ensure its availability and assure that systems work promptly and service is not denied to authorized users;
2. **Data Confidentiality** – it's a service to assure that private or confidential information is made available or disclosed **only** to authorized entities (person, organization, or computer):
  - **Connection Confidentiality** – the protection of all user data on a connection;
  - **Connectionless Confidentiality** – the protection of all user data in a single data block;
  - **Selective-Field Confidentiality** – the confidentiality of selected fields within the user data on a connection or in a single data block;
  - **Traffic-Flow Confidentiality** – the protection of the information that might be derived from observation of traffic flows;

### Security Services (continuation):

3. **Data Integrity** – it's a service to assure that data received is exactly as sent by an authorized entity:
  - 3.1 **Connection Integrity with Recovery** – provides for the integrity of all user data on a connection and detects any modification, insertion, deletion, or replay of any data, with recovery attempted;
  - 3.2 **Connection Integrity without Recovery** – as above, but provides only detection without recovery;
  - 3.3 **Selective-Field Connection Integrity** – provides for the integrity of selected fields within the user data of a data block transferred over a connection and detects whether the selected fields have been modified, inserted, deleted, or replayed;
  - 3.4 **Connectionless Integrity** – provides for the integrity of a single connectionless data block and may take the form of detection of data modification.
  - 3.5 **Selective-Field Connectionless Integrity** – provides for the integrity of selected fields within a single connectionless data block; takes the form of determination of whether the selected fields have been modified;

### Security Services (continuation)

4. **Authentication** – it's a service to assure that an entity (person, organization, or computer) is the one she claims to be;
  - **Peer Entity Authentication** – used in association with a logical connection to provide confidence in the identity of the entities connected;
  - **Data-Origin Authentication** – in a connectionless transfer, provides assurance that the source of received data is as claimed;

### Security Services (continuation)

4. **Authentication** – it's a service to assure that an entity (person, organization, or computer) is the one she claims to be;
  - **Peer Entity Authentication** – used in association with a logical connection to provide confidence in the identity of the entities connected;
  - **Data-Origin Authentication** – in a connectionless transfer, provides assurance that the source of received data is as claimed;
5. **Access Control** – it's a service to prevent the unauthorized use of a **resource**;

### Security Services (continuation)

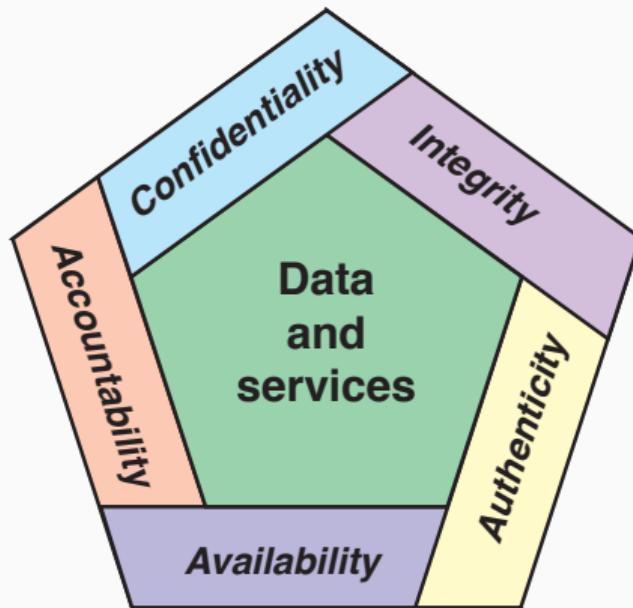
4. **Authentication** – it's a service to assure that an entity (person, organization, or computer) is the one she claims to be;
  - **Peer Entity Authentication** – used in association with a logical connection to provide confidence in the identity of the entities connected;
  - **Data-Origin Authentication** – in a connectionless transfer, provides assurance that the source of received data is as claimed;
5. **Access Control** – it's a service to prevent the unauthorized use of a **resource**;
6. **Non-repudiation** – it's a service to assure that an entity can not deny its participation in all or part of a communication:
  - **Source non-repudiation** – Proof that the message was sent by the specified party;
  - **Destination non-repudiation** – Proof that the message was received by the specified party;

Not part of the X.800 standard, but important:

7. **Accountability** – it's a service to assure that generates the requirement for actions of an entity to be traced uniquely to that entity:

- this supports non-repudiation, deterrence, fault isolation, intrusion detection and prevention, and after-action recovery and legal action;
- because truly secure systems are not yet an achievable goal, we must be able to trace a security breach to a responsible party;
- systems must keep records of their activities to permit later forensic analysis to trace security breaches or to aid in transaction disputes;

## Essential Network and Computer Security Requirements



## Security Mechanisms

---

### Security Mechanism

Security mechanisms are technical tools and techniques that are used **to implement security services**. A mechanism might operate by itself, or with others, to provide a particular service.

Security Mechanisms:

- **Specific security mechanisms** – may be incorporated into the appropriate protocol layer in order to provide some of the security services;

Security Mechanisms:

- **Specific security mechanisms** – may be incorporated into the appropriate protocol layer in order to provide some of the security services;
- **Pervasive security mechanisms** – mechanisms that are not specific to any particular security service or protocol layer;

### List of Specific Security Mechanisms:

- **Encipherment** – use of encryption algorithms;

### List of Specific Security Mechanisms:

- **Encipherment** – use of encryption algorithms;
- **Digital Signature** – data appended to, or a cryptographic transformation of, a data unit that allows a recipient of the data unit to prove the source and integrity of the data unit and protect against forgery (e.g., by the recipient);

### List of Specific Security Mechanisms:

- **Encipherment** – use of encryption algorithms;
- **Digital Signature** – data appended to, or a cryptographic transformation of, a data unit that allows a recipient of the data unit to prove the source and integrity of the data unit and protect against forgery (e.g., by the recipient);
- **Access Control** – a variety of mechanisms that enforce access rights to resources (*i. e.* firewalls);

### List of Specific Security Mechanisms:

- **Encipherment** – use of encryption algorithms;
- **Digital Signature** – data appended to, or a cryptographic transformation of, a data unit that allows a recipient of the data unit to prove the source and integrity of the data unit and protect against forgery (e.g., by the recipient);
- **Access Control** – a variety of mechanisms that enforce access rights to resources (*i. e.* firewalls);
- **Data Integrity** – a variety of mechanisms used to assure the integrity of a data unit or stream of data units;

List of Specific Security Mechanisms (continuation):

- **Authentication Exchange** – a mechanism intended to ensure the identity of an entity by means of information exchange;

List of Specific Security Mechanisms (continuation):

- **Authentication Exchange** – a mechanism intended to ensure the identity of an entity by means of information exchange;
- **Traffic Padding** – The insertion of bits into gaps in a data stream to frustrate traffic analysis attempts;

List of Specific Security Mechanisms (continuation):

- **Authentication Exchange** – a mechanism intended to ensure the identity of an entity by means of information exchange;
- **Traffic Padding** – The insertion of bits into gaps in a data stream to frustrate traffic analysis attempts;
- **Routing Control** – enables selection of particular physically secure routes for certain data and allows routing changes, especially when a breach of security is suspected;

List of Specific Security Mechanisms (continuation):

- **Authentication Exchange** – a mechanism intended to ensure the identity of an entity by means of information exchange;
- **Traffic Padding** – The insertion of bits into gaps in a data stream to frustrate traffic analysis attempts;
- **Routing Control** – enables selection of particular physically secure routes for certain data and allows routing changes, especially when a breach of security is suspected;
- **Notarization** – The use of a trusted third party to assure certain properties of a data exchange;

### List of Pervasive Security Mechanisms:

- Trusted Functionality – that which is perceived to be correct with respect to some criteria (e.g., as established by a security policy);

### List of Pervasive Security Mechanisms:

- **Trusted Functionality** – that which is perceived to be correct with respect to some criteria (e.g., as established by a security policy);
- **Security Label** – the marking bound to a resource (which may be a data unit) that names or designates the security attributes of that resource;

### List of Pervasive Security Mechanisms:

- **Trusted Functionality** – that which is perceived to be correct with respect to some criteria (e.g., as established by a security policy);
- **Security Label** – the marking bound to a resource (which may be a data unit) that names or designates the security attributes of that resource;
- **Event Detection** – detection of security-relevant events;

### List of Pervasive Security Mechanisms:

- **Trusted Functionality** – that which is perceived to be correct with respect to some criteria (e.g., as established by a security policy);
- **Security Label** – the marking bound to a resource (which may be a data unit) that names or designates the security attributes of that resource;
- **Event Detection** – detection of security-relevant events;
- **Security Audit Trail** – data collected and potentially used to facilitate a security audit, which is an independent review and examination of system records and activities;

### List of Pervasive Security Mechanisms:

- **Trusted Functionality** – that which is perceived to be correct with respect to some criteria (e.g., as established by a security policy);
- **Security Label** – the marking bound to a resource (which may be a data unit) that names or designates the security attributes of that resource;
- **Event Detection** – detection of security-relevant events;
- **Security Audit Trail** – data collected and potentially used to facilitate a security audit, which is an independent review and examination of system records and activities;
- **Security Recovery** – deals with requests from mechanisms, such as event handling and management functions, and takes recovery actions;

## Relationship Between Security Services and Mechanisms

SERVICE	MECHANISM							
	Encipherment	Digital signature	Access control	Data integrity	Authentication exchange	Traffic padding	Routing control	Notarization
Peer entity authentication	Y	Y		Y				
Data origin authentication	Y	Y						
Access control			Y					
Confidentiality	Y						Y	
Traffic flow confidentiality	Y					Y	Y	
Data integrity	Y	Y		Y				
Nonrepudiation		Y		Y				Y
Availability				Y	Y			

## Attack Surfaces

---

### Attack Surfaces

An attack surface consists of the reachable and exploitable vulnerabilities in a system.

### Attack Surfaces

An attack surface consists of the reachable and exploitable vulnerabilities in a system.

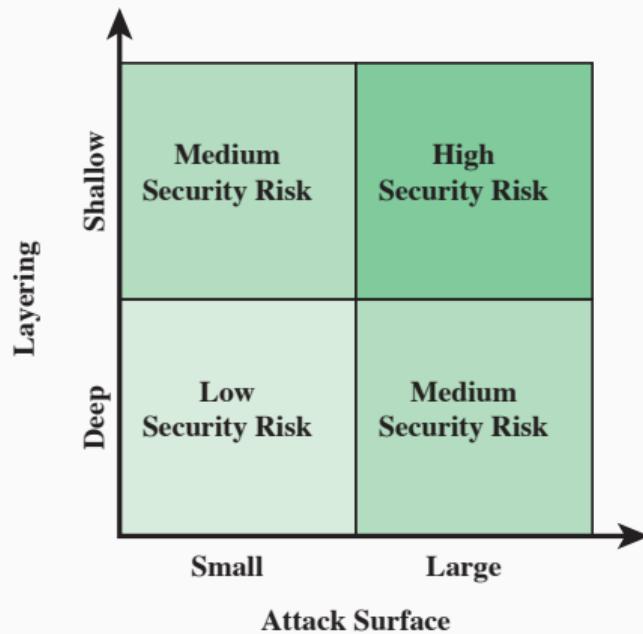
Attack surfaces can be categorized as follows:

- **Network attack surface** – vulnerabilities over an enterprise network, wide-area network, or the Internet.
- **Software attack surface** – vulnerabilities in application, utility, or operating system code.
- **Human attack surface** – vulnerabilities created by personnel or outsiders, such as social engineering, human error, and trusted insiders

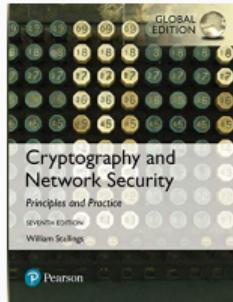
Examples of attack surfaces:

- Open ports on outward facing Web and other servers, and code listening on those ports
- Services available on the inside of a firewall
- Code that processes incoming data, email, XML, office documents, and industry specific custom data exchange formats
- Interfaces, SQL, and Web forms
- An employee with access to sensitive information vulnerable to a social engineering attack

Defense in depth versus attack surface



# Questions?



## Chapter 1 of

*William Stallings, Cryptography and Network Security: Principles and Practice, Global Edition, 2016*

# Systems' Security | Segurança de Sistemas

## Introduction to Cryptography

---

Miguel Frade



## Overview

---

### Learning Objectives

Introduction

Concepts

Symmetric Encryption

Asymmetric Ciphers

Hash Algorithms | *Algoritmos de Síntese*

## Learning Objectives

---

After this chapter, you should be able to:

1. Explain the difference between cryptographic algorithms categories
2. Present an overview of the main concepts of symmetric cryptography
3. Explain the difference between cryptanalysis and brute-force attack

## Introduction

---

### Cryptography

is the study of techniques for secure communication in the presence of third parties called adversaries

### Cryptography

is the study of techniques for secure communication in the presence of third parties called adversaries

Cryptography aims to develop techniques and algorithms to provide these security services:

- confidentiality
- data integrity
- authentication
- non-repudiation

Cryptographic algorithms can be characterized into **three independent dimensions**:

1. type of operations used for transforming plaintext to ciphertext:
  - substitution
  - transposition
  - both substitution and transposition

Cryptographic algorithms can be characterized into **three independent dimensions**:

1. type of operations used for transforming plaintext to ciphertext:

- substitution
- transposition
- both substitution and transposition

2. number of keys used:

- one key: symmetric, single-key, secret-key, or conventional encryption
- two keys: asymmetric, two-key, or public-key encryption

Cryptographic algorithms can be characterized into **three independent dimensions**:

1. type of operations used for transforming plaintext to ciphertext:

- substitution
- transposition
- both substitution and transposition

2. number of keys used:

- one key: symmetric, single-key, secret-key, or conventional encryption
- two keys: asymmetric, two-key, or public-key encryption

3. way in which the plaintext is processed:

- block cipher
- stream cipher

Cryptographic algorithms can be divided into three families

- Symmetric algorithms
- Asymmetric (or public key) algorithms
- Hash algorithms | *algoritmos de síntese*

## Symmetric Encryption

---

Symmetric encryption:

- also referred to as conventional encryption or single-key encryption
- the only type of encryption until the 1970s (before the development of public-key encryption)

Symmetric encryption ingredients:

1. **plaintext** – the original intelligible message or data that is fed into the algorithm as input

Symmetric encryption ingredients:

1. **plaintext** – the original intelligible message or data that is fed into the algorithm as input
2. **encryption algorithm** – algorithm that performs various substitutions and transformations on the plaintext

Symmetric encryption ingredients:

1. **plaintext** – the original intelligible message or data that is fed into the algorithm as input
2. **encryption algorithm** – algorithm that performs various substitutions and transformations on the plaintext
3. **secret key** – another input to the encryption algorithm. The key is a value independent of the plaintext and of the algorithm. The algorithm will produce a different output depending on the specific key being used at the time.

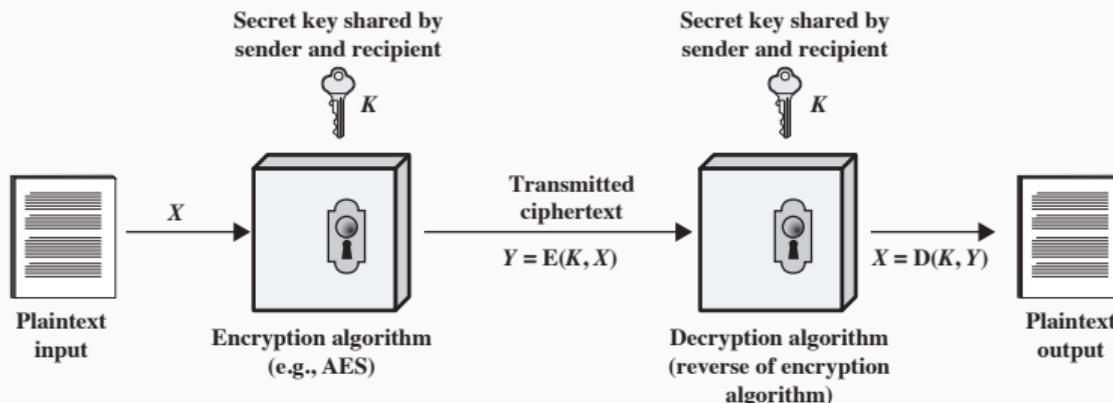
Symmetric encryption ingredients:

1. **plaintext** – the original intelligible message or data that is fed into the algorithm as input
2. **encryption algorithm** – algorithm that performs various substitutions and transformations on the plaintext
3. **secret key** – another input to the encryption algorithm. The key is a value independent of the plaintext and of the algorithm. The algorithm will produce a different output depending on the specific key being used at the time.
4. **ciphertext** – the scrambled message produced as output that depends on the plaintext and the secret key

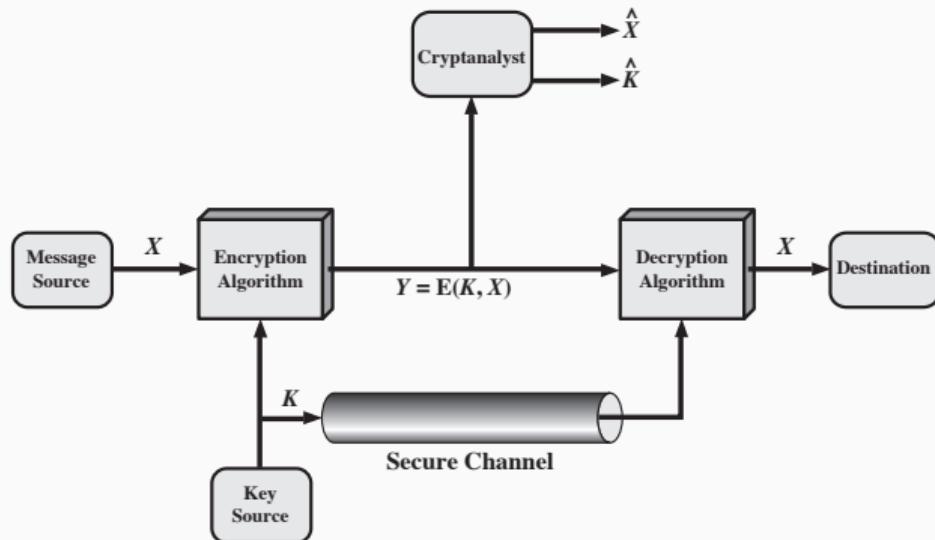
Symmetric encryption ingredients:

1. **plaintext** – the original intelligible message or data that is fed into the algorithm as input
2. **encryption algorithm** – algorithm that performs various substitutions and transformations on the plaintext
3. **secret key** – another input to the encryption algorithm. The key is a value independent of the plaintext and of the algorithm. The algorithm will produce a different output depending on the specific key being used at the time.
4. **ciphertext** – the scrambled message produced as output that depends on the plaintext and the secret key
5. **decryption algorithm** – takes the ciphertext and the secret key and produces the original plaintext.

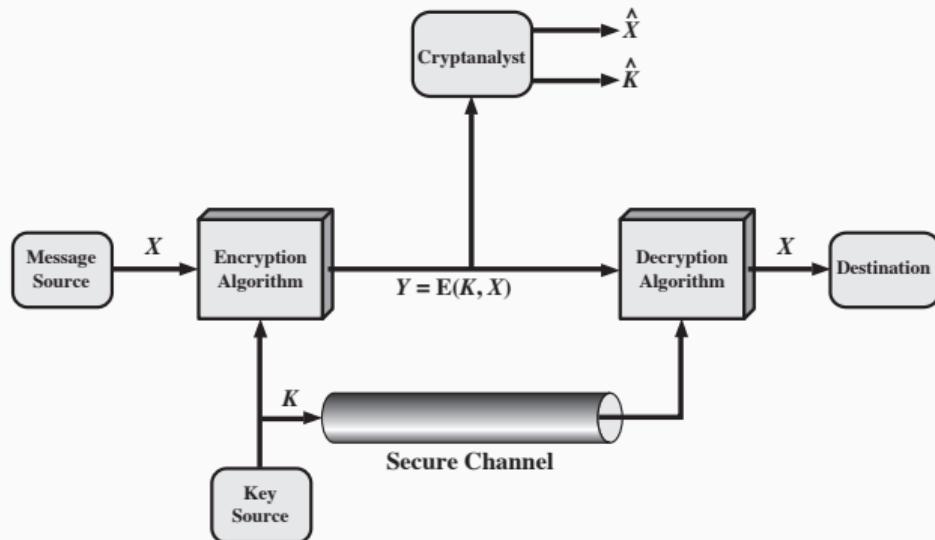
## Simplified model of symmetric encryption



Model of symmetric encryption



## Model of symmetric encryption



Short representation

$$A \rightarrow B : E_{ks}(M)$$

There are two general approaches to attack a conventional encryption scheme:

1. **Cryptanalysis** – relies on the nature of the algorithm plus perhaps some knowledge of the general characteristics of the plaintext or even some sample plaintext–ciphertext pairs. This type of attack exploits the characteristics of the algorithm to attempt to deduce a specific plaintext or to deduce the key being used.

There are two general approaches to attack a conventional encryption scheme:

1. **Cryptanalysis** – relies on the nature of the algorithm plus perhaps some knowledge of the general characteristics of the plaintext or even some sample plaintext–ciphertext pairs. This type of attack exploits the characteristics of the algorithm to attempt to deduce a specific plaintext or to deduce the key being used.
2. **Brute-force attack** – The attacker tries every possible key on a piece of ciphertext until an intelligible translation into plaintext is obtained:
  - On average, half of all possible keys must be tried to achieve success
  - The analyst must be able to recognize plaintext as plaintext.

**open algorithms** – reverse the encryption process must rely only on the key value and not the fact of knowing the used algorithm

**open algorithms** – reverse the encryption process must rely only on the key value and not the fact of knowing the used algorithm

An encryption scheme is said to be computationally secure against brute-force attacks if

- the **cost** of breaking the cipher exceeds the **value** of the encrypted information
- the **time** required to break the cipher exceeds the useful **lifetime** of the information

**open algorithms** – reverse the encryption process must rely only on the key value and not the fact of knowing the used algorithm

An encryption scheme is said to be computationally secure against brute-force attacks if

- the **cost** of breaking the cipher exceeds the **value** of the encrypted information
- the **time** required to break the cipher exceeds the useful **lifetime** of the information

These principles also apply to other cryptographic algorithms.

### Advantages

- Computationally efficient

### Disadvantages

### Advantages

- Computationally efficient
- Easy to implement in hardware

### Disadvantages

### Advantages

- Computationally efficient
- Easy to implement in hardware

### Disadvantages

- Difficult to distribute keys safely

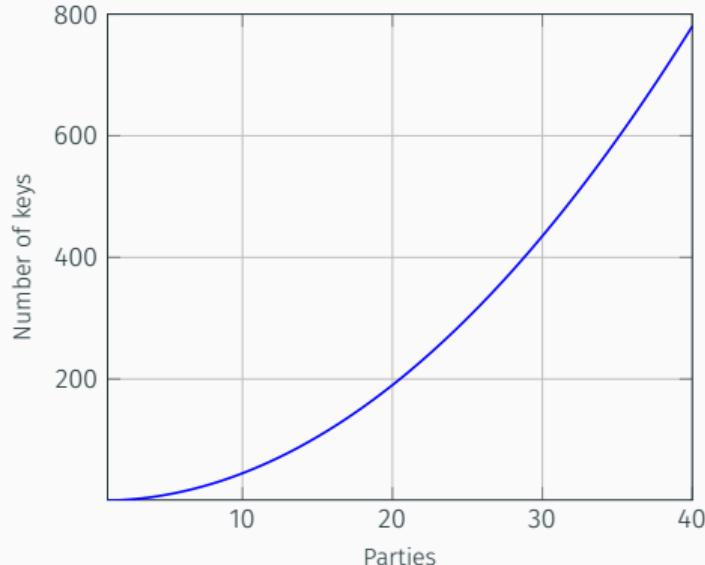
### Advantages

- Computationally efficient
- Easy to implement in hardware

### Disadvantages

- Difficult to distribute keys safely
- Number of keys required for independent communication between N parties

Number of keys required for independent communication between N parties



$$\text{Number of keys} = \frac{N(N-1)}{2}$$

## Asymmetric Ciphers

---

### Asymmetric algorithms

- use a key pair:
  - Private key ( $k_R$ )
  - Public key ( $k_U$ )

### Asymmetric algorithms

- use a key pair:
  - Private key ( $k_R$ )
  - Public key ( $k_U$ )
- the  $k_R$  and  $k_U$  keys are mathematically related
  - but it's computationally infeasible derive the private key from the public key

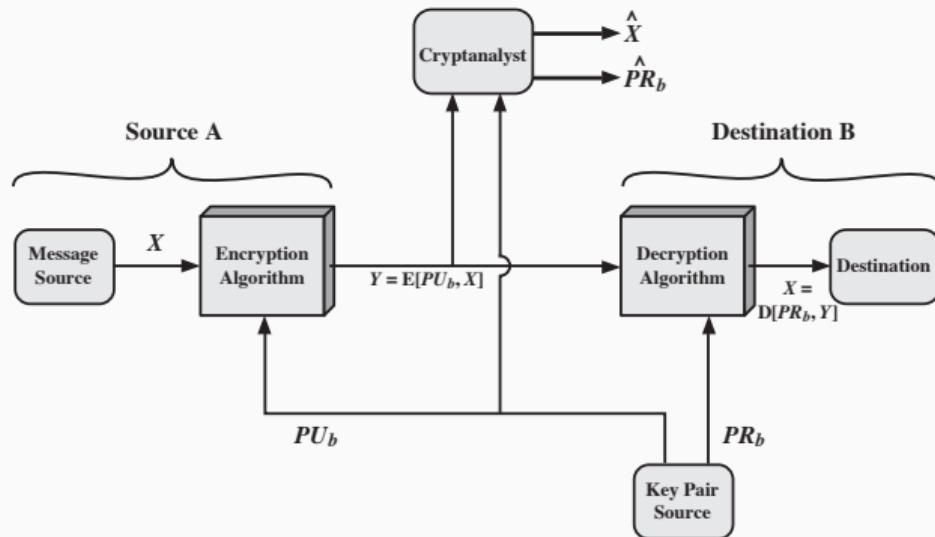
### Asymmetric algorithms

- use a key pair:
  - Private key ( $k_R$ )
  - Public key ( $k_U$ )
- the  $k_R$  and  $k_U$  keys are mathematically related
  - but it's computationally infeasible derive the private key from the public key
- these keys are used to perform complementary operations:
  - encryption / decryption
  - signature generation / signature verification

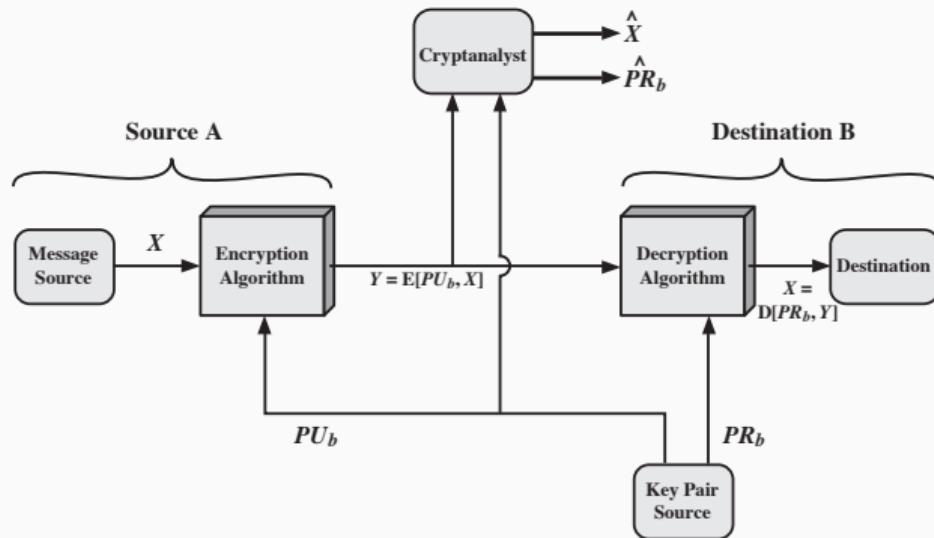
Asymmetric algorithms solve some problems found in symmetric algorithms:

- confidentiality is not required for key distribution of public keys
  - these algorithms are used to safely distribute symmetric keys
- perform digital signatures to ensure non-repudiation

## Model of asymmetric encryption – confidentiality



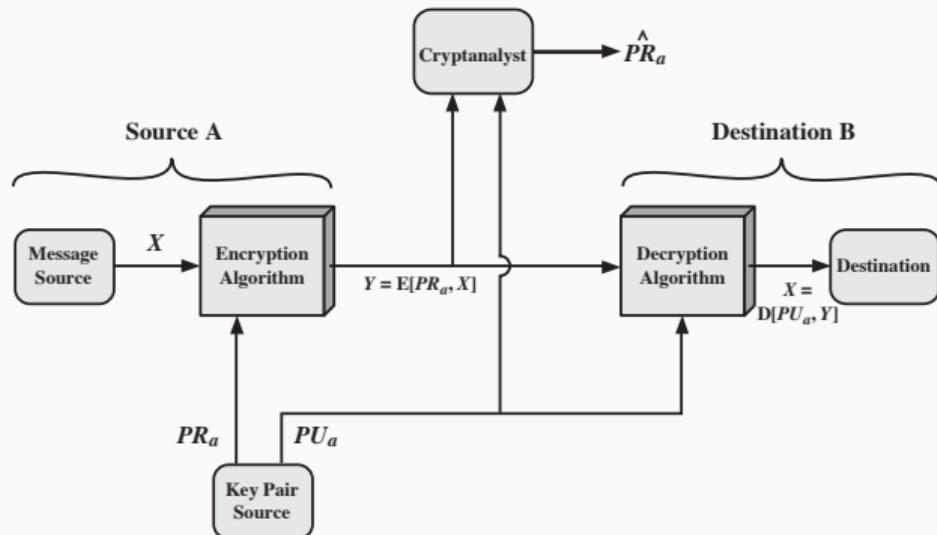
## Model of asymmetric encryption – confidentiality



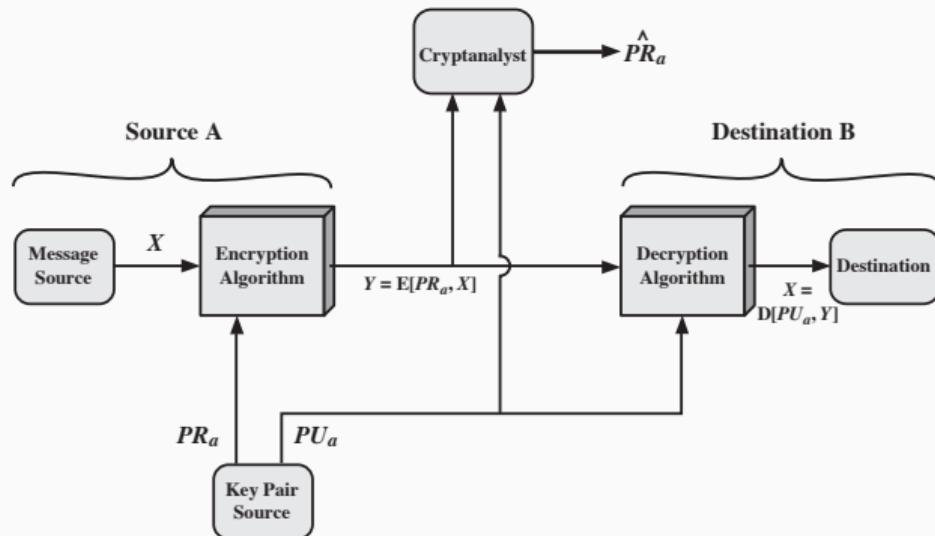
Short representation

$$A \rightarrow B : E_{k_{U_B}}(M)$$

## Model of asymmetric encryption – non-repudiation



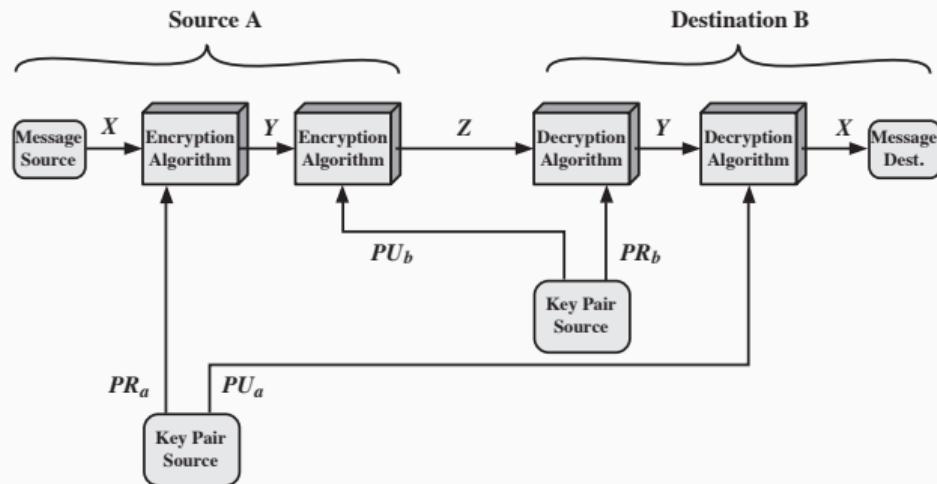
## Model of asymmetric encryption – non-repudiation



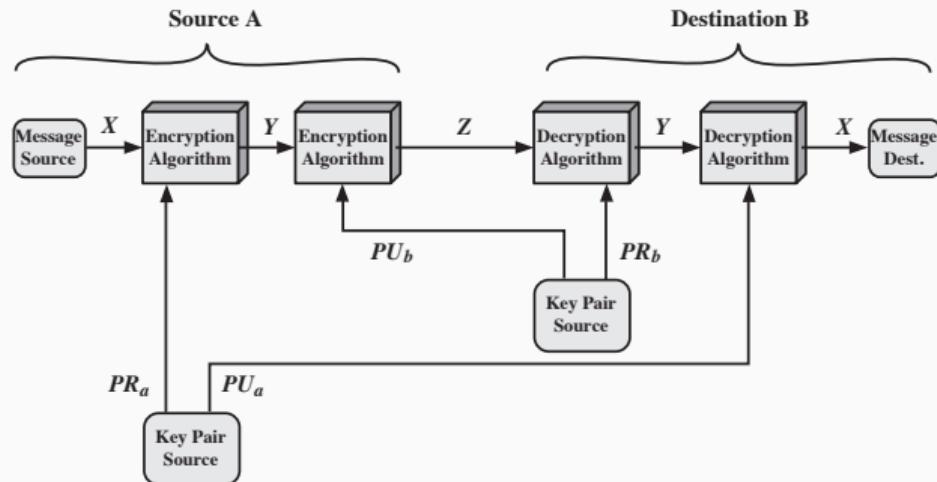
Short representation

$$A \rightarrow B : E_{k_{R_A}}(M)$$

## Model of asymmetric encryption – confidentiality and non-repudiation



## Model of asymmetric encryption – confidentiality and non-repudiation



Short representation

$$A \rightarrow B : E_{k_{U_B}}(E_{k_{R_A}}(M))$$

### Advantages

- public keys can be shared over insecure channels

### Disadvantages

### Advantages

- public keys can be shared over insecure channels
- ideal to protect secret keys of symmetric ciphers (key agreement)

### Disadvantages

### Advantages

- public keys can be shared over insecure channels
- ideal to protect secret keys of symmetric ciphers (key agreement)
- implementation of non-repudiation (digital signatures)

### Disadvantages

### Advantages

- public keys can be shared over insecure channels
- ideal to protect secret keys of symmetric ciphers (key agreement)
- implementation of non-repudiation (digital signatures)

### Disadvantages

- slower and more complex than symmetric ciphers

### Advantages

- public keys can be shared over insecure channels
- ideal to protect secret keys of symmetric ciphers (key agreement)
- implementation of non-repudiation (digital signatures)

### Disadvantages

- slower and more complex than symmetric ciphers
- infrastructure to certify owners of the public keys is complex

### Advantages

- public keys can be shared over insecure channels
- ideal to protect secret keys of symmetric ciphers (key agreement)
- implementation of non-repudiation (digital signatures)

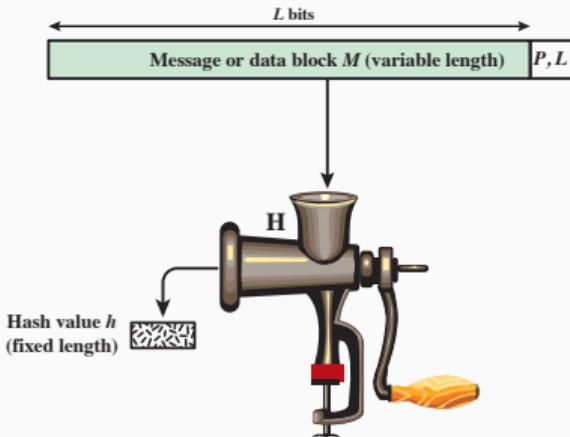
### Disadvantages

- slower and more complex than symmetric ciphers
- infrastructure to certify owners of the public keys is complex
- encryption is too slow, so key encapsulation is used to overcome this limitation

## Hash Algorithms | *Algoritmos de Síntese*

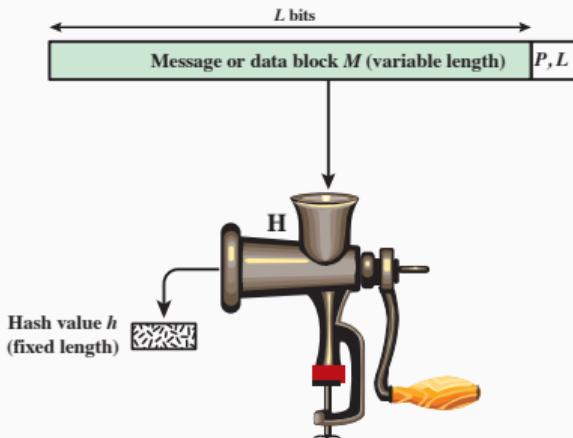
---

Hash algorithms, or hash functions



$P, L$  = padding plus length field

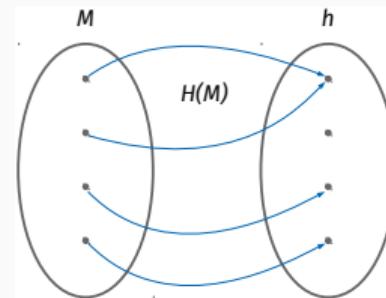
## Hash algorithms, or hash functions



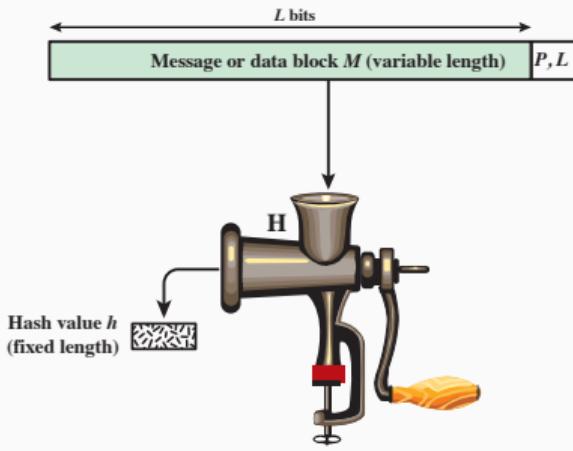
$P, L$  = padding plus length field

### Many-to-one function

- input – a message  $M$  of variable length
- output – a value  $h$  of fixed length, e.g. 256 bits



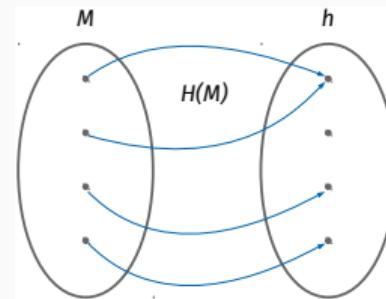
## Hash algorithms, or hash functions



$P, L$  = padding plus length field

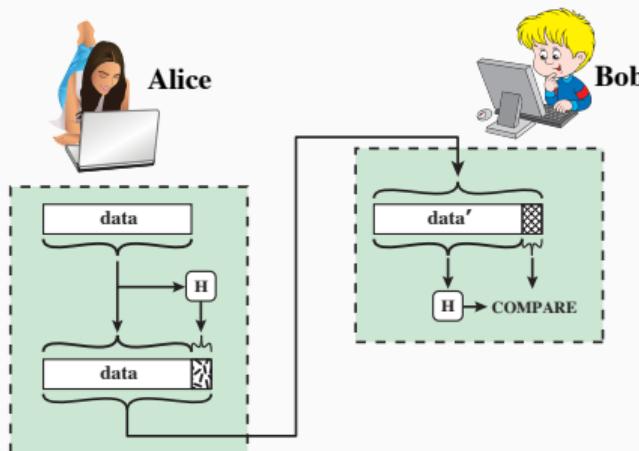
### Many-to-one function

- input – a message  $M$  of variable length
- output – a value  $h$  of fixed length, e.g. 256 bits



- size of the messages  $M$  universe =  $\infty$
- size of the hash values  $h$  universe =  $2^{n \text{ bits}}$

## Use of hash algorithms

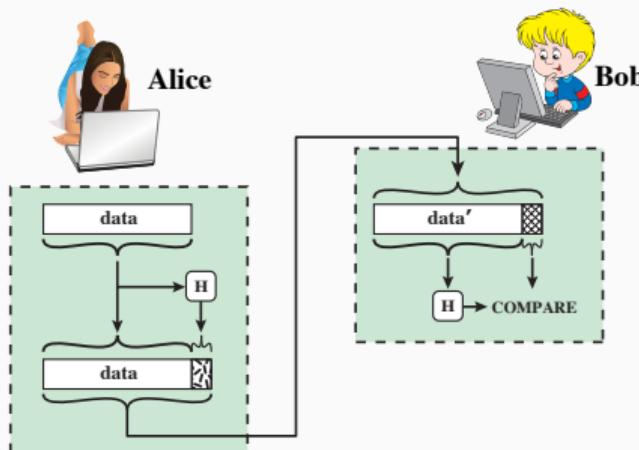


Short representation

$$A \rightarrow B : M \parallel H(M)$$

$$H(\text{Hello}) = 0x8b1a9953c4611296a827abf8c47804d7$$

## Use of hash algorithms



Short representation

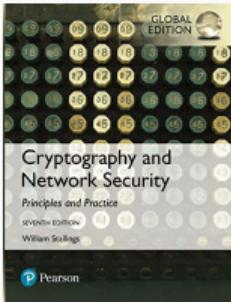
$$A \rightarrow B : M \parallel H(M)$$

Prone to attacks

1. the hash algorithm is not enough to guarantee data integrity
2. the hash value must be protected somehow

$$H(\text{Hello}) = 0x8b1a9953c4611296a827abf8c47804d7$$

# Questions?



Chapters 3.1, 9.1 and 11.1 of  
*William Stallings, Cryptography and Network Security: Principles and Practice, Global Edition, 2016*

# Systems' Security | Segurança de Sistemas

## Symmetric Cryptography – Classical Techniques

---

Miguel Frade



## Overview

---

### Learning Objectives

Introduction

Type of operations

### Substitution Techniques

Exercise

### Transposition Techniques

Exercise

### Rotor Machines

## Learning Objectives

---

After this chapter, you should be able to:

1. Understand the operation of a monoalphabetic substitution cipher
2. Understand the operation of a polyalphabetic cipher
3. Present an overview of the Vigenère cipher
4. Understand the operation of a transposition cipher
5. Describe the operation of a rotor machine

## Introduction

---

Cryptographic algorithms can be characterized into **three independent dimensions**:

1. type of operations used for transforming plaintext to ciphertext:
  - substitution
  - transposition
  - both substitution and transposition

Cryptographic algorithms can be characterized into **three independent dimensions**:

1. type of operations used for transforming plaintext to ciphertext:

- substitution
- transposition
- both substitution and transposition

2. number of keys used:

- one key: symmetric, single-key, secret-key, or conventional encryption
- two keys: asymmetric, two-key, or public-key encryption

Cryptographic algorithms can be characterized into **three independent dimensions**:

1. type of operations used for transforming plaintext to ciphertext:

- substitution
- transposition
- both substitution and transposition

2. number of keys used:

- one key: symmetric, single-key, secret-key, or conventional encryption
- two keys: asymmetric, two-key, or public-key encryption

3. way in which the plaintext is processed:

- block cipher
- stream cipher

### Substitution Operation

Each element in the plaintext (bit, letter, group of bits or letters) is mapped into another element, *e.g.* letters of plaintext are replaced by other letters or by numbers or symbols.

### Substitution Operation

Each element in the plaintext (bit, letter, group of bits or letters) is mapped into another element, e.g. letters of plaintext are replaced by other letters or by numbers or symbols.

### Transposition Operation

Each element in the plaintext (bit, letter, group of bits or letters) is rearranged

### Substitution Operation

Each element in the plaintext (bit, letter, group of bits or letters) is mapped into another element, e.g. letters of plaintext are replaced by other letters or by numbers or symbols.

### Transposition Operation

Each element in the plaintext (bit, letter, group of bits or letters) is rearranged

- information cannot be lost, i.e. all operations are reversible
- modern cryptographic systems involve multiple stages of **both** substitutions and transpositions

## Substitution Techniques

---

### Caeser Chiper | Cifra de César

- the earliest known, and the simplest, use of a substitution cipher
- created by the emperor Julius Caesar
- replacing each letter of the alphabet with the letter standing  $k$  places further down the alphabet, e.g. for  $k = 3$ :

```
plaintext:  meet me after the toga party  
ciphertext: PHHW PH DIWHU WKH WRJD SDUWB
```

### Caeser Chiper | Cifra de César

- the earliest known, and the simplest, use of a substitution cipher
- created by the emperor Julius Caesar
- replacing each letter of the alphabet with the letter standing  $k$  places further down the alphabet, e.g. for  $k = 3$ :

```
plaintext: meet me after the toga party  
ciphertext: PHHW PH DIWHU WKH WRJD SDUWB
```

### General description of the Caeser Chiper

value:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
letter:	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z

- encryption:  $C = E(k, p) = (p + k) \text{ mod } 26$
- decryption:  $p = E(k, C) = (C - k) \text{ mod } 26$

### Brute-force Caeser Chiper

- the encryption and decryption algorithms are known
- there are only 25 possible keys to try
- the language of the plaintext is known and easily recognizable

key	ciphertext:	PHHW PH DIWHU WKH WRJD SDUWB
1		oggv og chvgt vjg vqic rctva
2		nffu nf bgufs uif uphb qbsuz
3		meet me after the toga party <--
4		llds ld zesdq sgd snfz ozqsx
5		kccr kc ydrcp rfc rmey nyprw
...		

## Brute-force Caeser Chiper

- if the language of the plaintext is unknown, then plaintext output may not be recognizable
- the input may be compressed in some fashion, making recognition difficult
- for example a ZIP file:

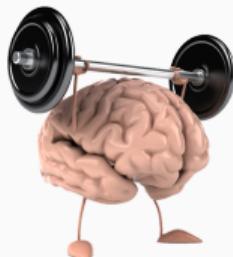
```

;; :&t[]|A9[]2=d7|6A|h4D$|;UR86~P[b|i|4N J||%$xn"Ln|)]]"kD|RS='>-D|up|C|R;Z|65|H'wes||-[J!]w:-Cj)/E|p|C|
+r3E^Ho{W|'TWQ_||(|B@z~||"J&4<,hK||RF|EhtF||W|n|5|}e@n'|C$|8|jFK,j@||>J+=v|I|[|||.o|2u`*|H
|RV@|;2%*|jd( ?hp|N%w\kW8j#Z|.Xd>o[E`Y|]K^T|dB9=XW|p'p1|:~ZG^D?S[|OC|q|_Or<:|H|w|bnx|W|xVb|||Vf|`i4.
$p4!|0N||"|||9|gj#r|<U|Z|6%|.|||q)rz^E_kYpV|9@ #.=|||U]*|Q|h)"o#o@:@d_HT|||Si9!|!%ki[|||`w
%=,..h|||{FD;|||&r|`EdcXr_|QHC_|6#ah@|f2|o )@k|||]0Fg|||R|||j=>|I|&=P~K貓G|||0|h/2|||`R} 6|tC|||Z1|QMvr|||7{|||Sz|||n
[|||?A35|||=|||rCwzuEMUvXrm/|eN0%rU|[#R|j-o$|d,|>+}|#WV;py|||y< T$|OJGVGwh|||l_{+|Hs|C|||$|||_WwY||| "S|||=1D|x|||
Й|||&|||"CC_||||L\K{t~Uw*=|||]pW^;0i&.|||F|N|2!2.xMzyT_|||<t:&73F9h|||*;v7DeW|Z'|_Yk;R|||V}|||g*H|||v|*|||-!M
%y|||JuCy|g|hd驶|I~|JW ~.<| m>b|*s8?U,>W|-s2|||x5|||R|||Iqv|:Mx|||/$lm|||c gN7|4>7Vt~w|||$|vY|||noP/9|vcK_Zm|?
^;BB|||W|Lb{wft|F|b^}<.|||y|||56|||~|||7c|42|z|>w<|||0b~AGIq|~%J|0E|U|?R|0|||{7;y|v|Q|2L^a|||R5|||pi?C|`w|||}|||8
|||Y|UK{|||}YzYo:]^z_}7g94g|||/|||rU|2} `Wo|||4IU|||yx|ahM=~^5a|51m|||lk6c;|x|||90|4|||Is|||w^0}*g7_|||}|||$|||gFPi|||
mijvP|||dj=NO|wOX|~Uy{|||}n|V|||%|J|?|||sl`o|0%0^|||^nsr9Q|||l^h]Q{|||o_{|||QZ9z|||;|||m9tN?|||4|N|ZL|v#.X7|b-b|||U|||聆
-9|^Tc|p|/>>8|+△-w|oAS1gt^H;/n|β|||j|||1|||.Jg蕨|s|||.||||qj|>m[91-SK|rg%,EE|t4?>6KA|||lp1|=|||44yGgx<q|||st}ple|||翫
%W+|||XsR|:g|N~gt1cm72u|Sz>m6~uZP|||nSF^p.|||=>oKjNk6^tA|||J+u|^k W~0A|||PU*7[pBF|||b8x|Jtu|||oi_|||k|w|S|n~w|||
X|||<U`|7>m|||_|||_|||X,c^A|||g4u+Hg<v=Bv|pLYchL|||~?7|||9|,|||yFdz|||9^=Wo|||i<^1=|||SL|||o毅|||I=N|io|gNi|||f-j}}|||r_|||3;t|||e|||s?E
='|||x|||ey9'|||<|||_|||2enY|*Bn_y{|||Y|ż|||7V]|||Aq施:|||3;[|||qa|U<d|CER|o|||n|||o|||3|||D|J+|||T|,|||TSR
|||;|||öHF|||~iS6>ε?'|||};pnHU-8;Nn{69:sukNw|||Y|_|||U}'|||?a|||#R|||}|||~Q$48|||f|||ivyM|||07nR|||v|wI/q|||U|7jk|||S>1~|||F-寂;|||_|||tF~
%[|||È|||()=j|||z|||Wn|||j|||~qdHP+-Onfb|9|j'x|4r>U|||yué|||K|||j+z3Sto=|||Å|||,=|||~9nj|||}bqF$|t|||_6_w|||5~wV^3t|m|||i+|||s|||K|||v|||
|||S|||u7g|||4o|||~M|||7Y|||rYM|||d|||a|||Q|||6~em|||kMZKH[e|||s{|||2U|||.|||bng|||Mr|||Vnkom|||4$34<j|||v|||3|||0~4|||9|||E>3_2v|||1Q|||?i|||0%?|

```

Exercise: Brute-force Caeser Chiper

Discover the message:



ciphertext: rd hfy mfx kqjfx

|

plaintext: m\_ \_ \_ \_ \_

What was the key value used in this example?

### Monoalphabetic Ciphers

the substitution could be any permutation of the 26 alphabetic characters

Advantages

### Monoalphabetic Ciphers

the substitution could be any permutation of the 26 alphabetic characters

#### Advantages

- there would be  $26!$  different keys → more than  $4 \times 10^{26}$  possible keys
  - this is 10 orders of magnitude greater than the key space for DES → more than  $7 \times 10^{16}$  possible keys
- it would eliminate brute-force techniques (too many keys to try)

### Monoalphabetic Ciphers

- however, there's another type of attack
- if the cryptanalyst knows the nature of the plaintext (*e.g.*, English text), then s/he can exploit the regularities of the language

#### Example

ciphertext:

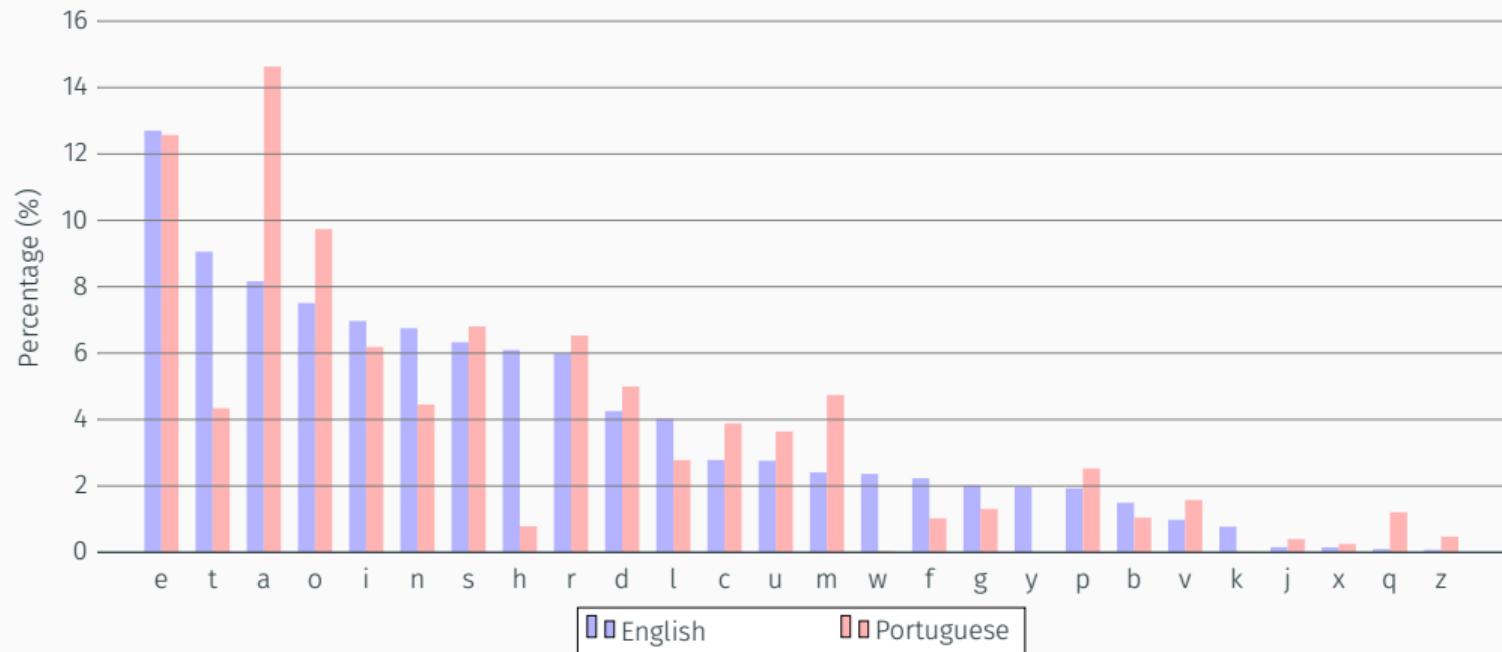
UZQSOVUOHXMOPVGPOZPEVSGZWSZOPFPESXUDBMETSXAIZ  
VUEPHZHMDZSHZOWSFPAPPDTSVPQUZWYMXUZUHSX  
EPYEPOPDZSZUFPOMBZWPFUPZHMDJUDTMOHMQ

relative frequency analysis:

P	13.33	H	5.83	F	3.33	B	1.67	C	0.00
Z	11.67	D	5.00	W	3.33	G	1.67	K	0.00
S	8.33	E	5.00	Q	2.50	Y	1.67	L	0.00
U	8.33	V	4.17	T	2.50	I	0.83	N	0.00
O	7.50	X	4.17	A	1.67	J	0.83	R	0.00
M	6.67								

## SUBSTITUTION TECHNIQUES

Language natural relative frequency



### Monoalphabetic Chipers

- exploiting the language natural relative frequency

UZQSOVUOHXMOPVGPOZPEVSGZWSZOPFPESXUDBMETSXAIZ  
t a e e te a that e e a a

VUEPHZHMDZSHZOWSFPAPPDTSPQUZWYMXUZUHSX  
e t ta t ha e ee a e th t a

EPYEPOPDZSZUFPOMBZWPFUPZHMDJUDTMOHMQ  
e e e tat e the t

### Monoalphabetic Chipers

- exploiting the language natural relative frequency

UZ QSO VUOHXMOPV GPOZPEVSG ZWSZ OPFPESX UDBMETS XAIZ  
it was disclosed yesterday that several informal but

VUEPHZ HMDZSHZO WSFP APPD TSVF QUZW YMXUZUHSX  
direct contacts have been made with political

EPYEPOPDZSZUFPO MB ZWP FUPZ HMDJ UD TMOHMQ  
representatives of the viet cong in moscow

### Playfair Cypher

The Playfair algorithm is based on the use of a  $5 \times 5$  matrix of letters constructed using a keyword

M	O	N	A	R
C	H	Y	B	D
E	F	G	I/J	K
L	P	Q	S	T
U	V	W	X	Z

### Playfair Cypher

The Playfair algorithm is based on the use of a  $5 \times 5$  matrix of letters constructed using a keyword

Plaintext is encrypted two letters at a time:

M	O	N	A	R
C	H	Y	B	D
E	F	G	I/J	K
L	P	Q	S	T
U	V	W	X	Z

1. repeating letters that are in the same pair are separated with a filler letter, such as *x*, e.g. *balloon* would be treated as *ba lx lo on*

### Playfair Cypher

The Playfair algorithm is based on the use of a  $5 \times 5$  matrix of letters constructed using a keyword

Plaintext is encrypted two letters at a time:

M	O	N	A	R
C	H	Y	B	D
E	F	G	I/J	K
L	P	Q	S	T
U	V	W	X	Z

1. repeating letters that are in the same pair are separated with a filler letter, such as *x*, e.g. *balloon* would be treated as *ba lx lo on*
2. two letters that fall in the same row are each replaced by the letter to the right, e.g. *ar*  $\rightarrow$  *RM*

### Playfair Cypher

The Playfair algorithm is based on the use of a  $5 \times 5$  matrix of letters constructed using a keyword

Plaintext is encrypted two letters at a time:

M	O	N	A	R
C	H	Y	B	D
E	F	G	I/J	K
L	P	Q	S	T
U	V	W	X	Z

1. repeating letters that are in the same pair are separated with a filler letter, such as *x*, e.g. *balloon* would be treated as *ba lx lo on*
2. two letters that fall in the same row are each replaced by the letter to the right, e.g. *ar*  $\rightarrow$  *RM*
3. two letters that fall in the same column are each replaced by the letter beneath, e.g. *mu*  $\rightarrow$  *CM*

### Playfair Cypher

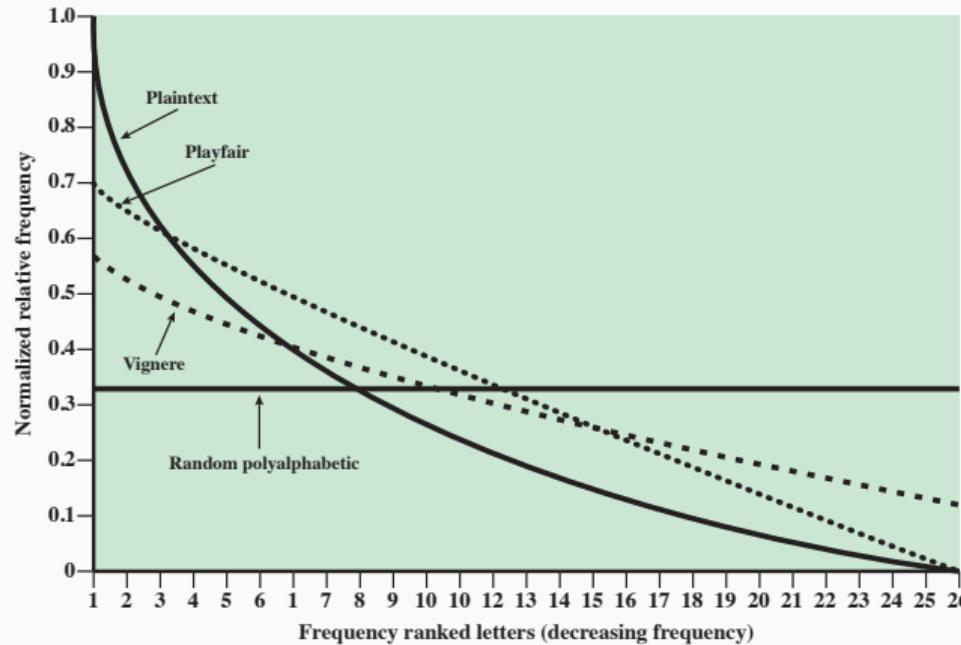
The Playfair algorithm is based on the use of a  $5 \times 5$  matrix of letters constructed using a keyword

M	O	N	A	R
C	H	Y	B	D
E	F	G	I/J	K
L	P	Q	S	T
U	V	W	X	Z

Plaintext is encrypted two letters at a time:

1. repeating letters that are in the same pair are separated with a filler letter, such as *x*, e.g. *balloon* would be treated as *ba lx lo on*
2. two letters that fall in the same row are each replaced by the letter to the right, e.g. *ar*  $\rightarrow$  *RM*
3. two letters that fall in the same column are each replaced by the letter beneath, e.g. *mu*  $\rightarrow$  *CM*
4. otherwise, each letter in a pair is replaced by the letter that lies in its own row and the column occupied by the other letter, e.g. *hs*  $\rightarrow$  *BP* and *ea*  $\rightarrow$  *IM*

Effectiveness of ciphers to hide the frequency distribution of the natural language



### Polyalphabetic Ciphers

uses different monoalphabetic substitutions as one proceeds through the plaintext message

Features:

- a set of related monoalphabetic substitution rules is used
- a key determines which particular rule is chosen for a given transformation

### Polyalphabetic Chipers

uses different monoalphabetic substitutions as one proceeds through the plaintext message

Features:

- a set of related monoalphabetic substitution rules is used
- a key determines which particular rule is chosen for a given transformation

One of the simplest, polyalphabetic ciphers is the Vigenère cipher

- uses a set of rules that consists of the 26 Caesar ciphers with shifts of 0 through 25
- encryption:  $C_i = (p_i + k_{i \text{ mod } m}) \text{ mod } 26$
- decryption:  $p_i = (C_i - k_{i \text{ mod } m}) \text{ mod } 26$

# SUBSTITUTION TECHNIQUES

Substitution table of the Vigenère chiper

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
B	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a
C	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b
D	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c
E	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d
F	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e
G	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f
H	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g
I	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h
J	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i
K	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j
L	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k
M	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l
N	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m
O	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n
P	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
Q	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p
R	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q
S	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r
T	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s
U	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t
V	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u
W	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v
X	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w
Y	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x
Z	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y

### Vigenère cipher example

key:	deceptive	deceptive	deceptive																									
plaintext:	w	e	a	r	e	d	i	s	c	o	n	r	z	g	v	t	w	a	v	z	h	c	q	y	g	l	m	j
ciphertext:	Z	I	C	V	T	W	Q	N	G	R	Z	G	V	T	W	A	V	Z	H	C	Q	Y	G	L	M	J		

### Vigenère chiper example

key:	deceptive	deceptive	deceptive
plaintext:	wearediscovered	saveyourself	
ciphertext:	ZICVTWQN	GRZGVTWAVZHCQYGLMGJ	

### Vernam chiper

- similar to the Vigenère chiper, but the key never repeats itself
- the key must be of the same size as the plaintext, a key stream generator is used
- encryption:  $c_i = p_i \oplus k_i$
- decryption:  $p_i = c_i \oplus k_i$



1. Using the Playfair on slide 14 encrypt the following message:  
**Systems Security**
  
2. Create a simple program to generate the Caesar cipher
  - choose your favorite tool, e.g. Excel, Libreoffice Calc, Perl, Python, ...

## Transposition Techniques

---

### Transposition

- the techniques examined so far involve the substitution of a plaintext symbol for a ciphertext symbol
- the transposition technique performs some sort of permutation on the plaintext letters

### Transposition

- the techniques examined so far involve the substitution of a plaintext symbol for a ciphertext symbol
- the transposition technique performs some sort of permutation on the plaintext letters

Rail fence example with depth = 2



plaintext: meet me after the toga party

construction: m e m a t r h t g p r y  
e t e f e t e o a a t

ciphertext: MEMATRHTGPRYETEFETEOAAT

Columnar transposition example

plaintext: attack postponed until tomorrow

key: 4 3 1 2 5 6 7

construction: a t t a c k p  
o s t p o n e  
d u n t i l t  
o m o r r o w

ciphertext: TTNOAPTRTSUMAODOOCOIRKNLOPETW

Columnar transposition example

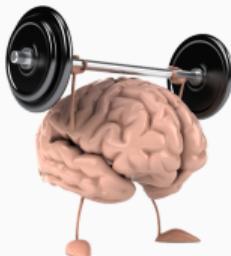
plaintext: attack postponed until tomorrow

key: 4 3 1 2 5 6 7

construction: a t t a c k p  
o s t p o n e  
d u n t i l t  
o m o r r o w

ciphertext: TTNOAPTRTSUMAODOOCOIRKNLOPETW

- the transposition cipher can be made significantly more secure by performing more than one stage of transposition



1. Use the columnar transposition
  - to the word **segurança**
  - with the key **3 1 2**
  - apply the transformation twice

## Rotor Machines

---

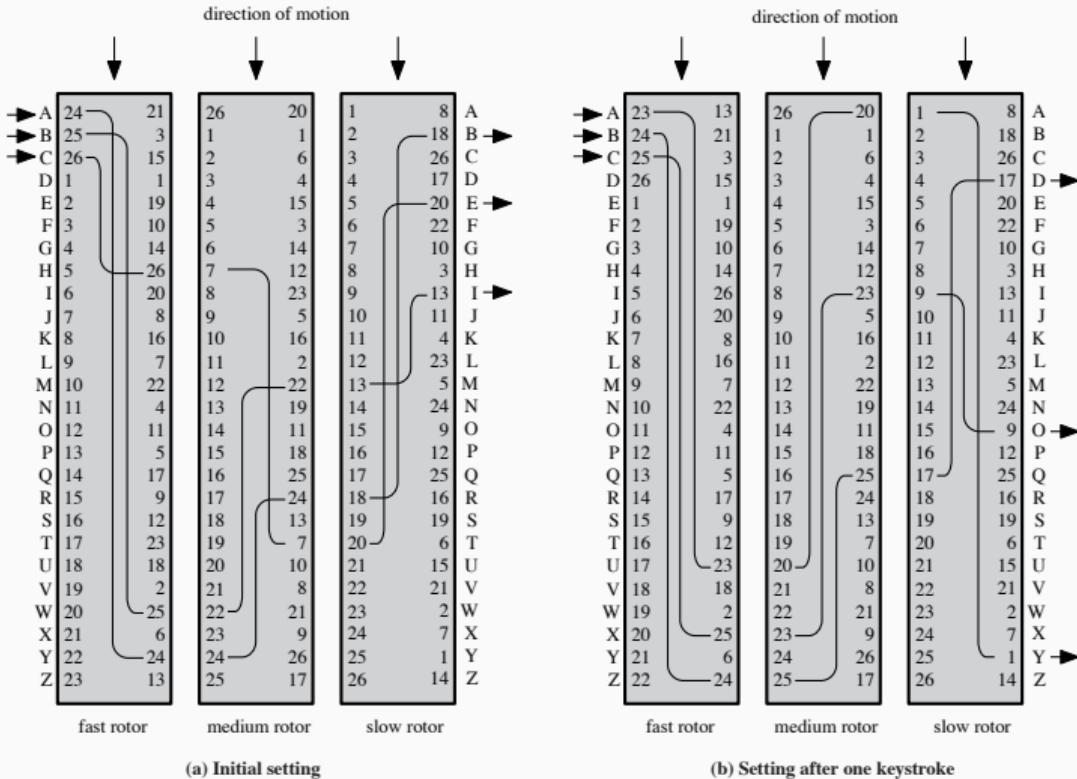
### Rotor Machines

- were used by both Germany (Enigma) and Japan (Purple) in World War II
- the machine consists of a set of independently rotating cylinders
- each cylinder defines a monoalphabetic substitution
- after each input key is depressed, the cylinder rotates one position
- after 26 letters of plaintext, the cylinder would be back to the initial position and the second cylinder shifts one position
- with  $n$  cylinders we get  $26^n$  different substitution alphabets
  - 3 cylinders → 17 576
  - 4 cylinders → 456 976
  - 5 cylinders → 11 881 376

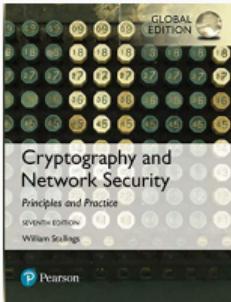
## MULTIPLE STAGES OF ENCRYPTION

## Rotor Machines

- Numberphile explains the Enigma machine
  - The Imitation Game  
Alan Turing cracks the Enigma code with help from fellow mathematicians
  - set the way to modern cryptography, of which DES is the most prominent



# Questions?



Chapters 3 of  
*William Stallings, Cryptography and Network Security: Principles and Practice, Global Edition, 2016*

# Systems' Security | Segurança de Sistemas

## Symmetric Cryptography – Modern Techniques

---

Miguel Frade



## Overview

---

Learning Objectives

Introduction

Feistel Cipher

Data Encryption Standard

Exercise

Block Cipher Modes of Operation

Stream Ciphers

Symmetric Algorithms

## Learning Objectives

---

After this chapter, you should be able to:

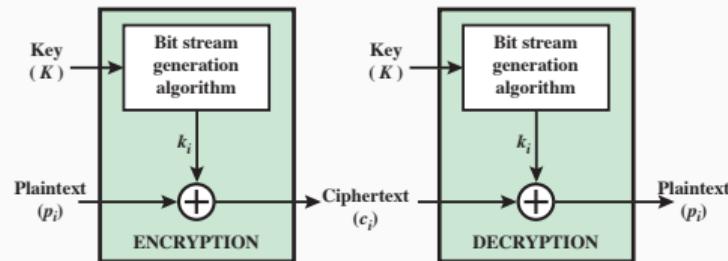
1. Understand the distinction between stream ciphers and block ciphers
2. Present an overview of the Feistel cipher
3. Present an overview of Data Encryption Standard (DES)
4. Analyze the security of multiple encryption schemes
5. Compare and contrast ECB, CBC, CFB, OFB, and counter modes of operation

## Introduction

---

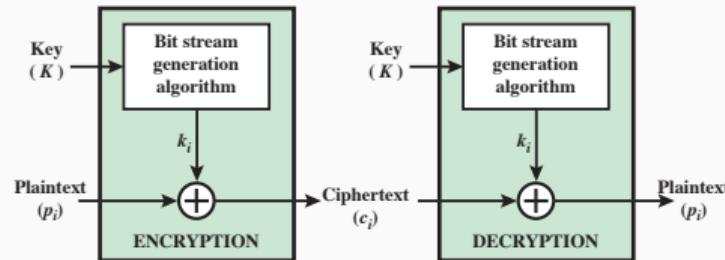
## Methods of plaintext processing

**stream cipher** – encrypts a digital data stream  
one bit or one byte at a time

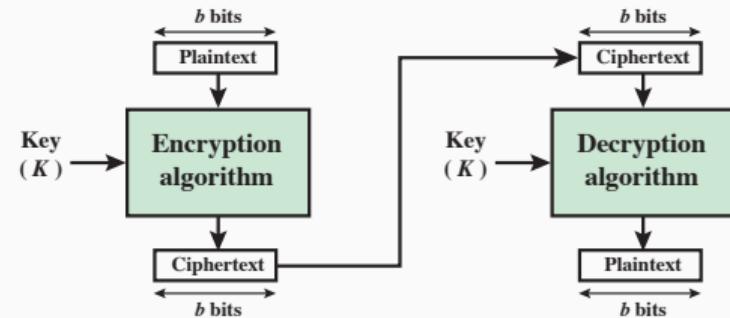


## Methods of plaintext processing

**stream cipher** – encrypts a digital data stream one bit or one byte at a time



**block cipher** – a block of plaintext is treated as a whole and used to produce a ciphertext block of equal length. Typically, a block size of 64 or 128 bits



### Nonsingular transformation

A block cipher operates on a plaintext block of  $n$  bits to produce a ciphertext block of  $n$  bits. A transformation is called reversible, or nonsingular, if there is a **unique** ciphertext block for each of the  $2^n$  possible different plaintext, *i.e.* for decryption to be possible.

### Nonsingular transformation

A block cipher operates on a plaintext block of  $n$  bits to produce a ciphertext block of  $n$  bits. A transformation is called reversible, or nonsingular, if there is a **unique** ciphertext block for each of the  $2^n$  possible different plaintext, *i.e.* for decryption to be possible.

### Ideal block cipher

For each  $2^n$  possible different plaintext blocks there is an arbitrary substitution map, *i.e.* the mapping itself is the key for encryption.

- for  $n = 4$  the key size is:  $4 \text{ bits} \times 2^4 \text{ possible combinations} = 64 \text{ bits}$
- for  $n = 64$  the key size is:  $64 \text{ bits} \times 2^{64} = 2^{70} \text{ bits} \rightarrow 1\,073\,741\,824 \text{ Tera Bytes}$

## Feistel Cipher

---

### Feistel Cipher Motivations

- a small block size, such as  $n = 4$ , then the system is equivalent to a classical substitution cipher → vulnerable to a statistical analysis
- a large block size is able to hide the statistical characteristics of the source plaintext → an arbitrary reversible substitution cipher for a large block size is not practical
- an approximation to the ideal block cipher is needed

## Product Cipher

the execution of two or more simple ciphers in sequence that the final result, or product, is cryptographically stronger than any of the component ciphers.

## Product Cipher

the execution of two or more simple ciphers in sequence that the final result, or product, is cryptographically stronger than any of the component ciphers.

## Feistel Cipher Structure

- product cipher → approximation to the ideal block cipher
- proposes the alternate usage of **substitutions** and **transpositions** functions
- applies the concepts of **diffusion** and **confusion**

To thwart cryptanalysis

### Diffusion

Diffusion seeks to make the statistical relationship between **the plaintext and ciphertext** as complex as possible:

- the statistical structure of the plaintext is dissipated into long-range statistics of the ciphertext.
- each plaintext digit affect the value of many ciphertext digits
- achieved by repeatedly performing some permutation on the data followed by applying a function to that permutation;

To thwart cryptanalysis

### Diffusion

Diffusion seeks to make the statistical relationship between **the plaintext and ciphertext** as complex as possible:

- the statistical structure of the plaintext is dissipated into long-range statistics of the ciphertext.
- each plaintext digit affect the value of many ciphertext digits
- achieved by repeatedly performing some permutation on the data followed by applying a function to that permutation;

### Confusion

Confusion seeks to make the relationship between the statistics of **the ciphertext and the value of the encryption key** as complex as possible

- achieved by the use of a complex substitution algorithm

Feistel Cipher depends on the choice of the following parameters

- **block size** – larger block sizes mean greater security, but reduced operation speed for a given algorithm, common size was 64 bits, now is 128 bits

Feistel Cipher depends on the choice of the following parameters

- **block size** – larger block sizes mean greater security, but reduced operation speed for a given algorithm, common size was 64 bits, now is 128 bits
- **key size** – larger key size means greater security, but may decrease performance, common size nowadays is 128 bits

Feistel Cipher depends on the choice of the following parameters

- **block size** – larger block sizes mean greater security, but reduced operation speed for a given algorithm, common size was 64 bits, now is 128 bits
- **key size** – larger key size means greater security, but may decrease performance, common size nowadays is 128 bits
- **number of rounds** – multiple rounds offer increasing security, a typical value is 16 rounds

Feistel Cipher depends on the choice of the following parameters

- **block size** – larger block sizes mean greater security, but reduced operation speed for a given algorithm, common size was 64 bits, now is 128 bits
- **key size** – larger key size means greater security, but may decrease performance, common size nowadays is 128 bits
- **number of rounds** – multiple rounds offer increasing security, a typical value is 16 rounds
- **subkey generation algorithm** – greater complexity should lead to greater difficulty of cryptanalysis

Feistel Cipher depends on the choice of the following parameters

- **block size** – larger block sizes mean greater security, but reduced operation speed for a given algorithm, common size was 64 bits, now is 128 bits
- **key size** – larger key size means greater security, but may decrease performance, common size nowadays is 128 bits
- **number of rounds** – multiple rounds offer increasing security, a typical value is 16 rounds
- **subkey generation algorithm** – greater complexity should lead to greater difficulty of cryptanalysis
- **round function F** – greater complexity generally means greater resistance to cryptanalysis

Feistel Cipher depends on the choice of the following parameters

- **block size** – larger block sizes mean greater security, but reduced operation speed for a given algorithm, common size was 64 bits, now is 128 bits
- **key size** – larger key size means greater security, but may decrease performance, common size nowadays is 128 bits
- **number of rounds** – multiple rounds offer increasing security, a typical value is 16 rounds
- **subkey generation algorithm** – greater complexity should lead to greater difficulty of cryptanalysis
- **round function F** – greater complexity generally means greater resistance to cryptanalysis
- **fast encryption/decryption** – speed of execution of the algorithm might be a concern in embedded applications

Feistel Cipher depends on the choice of the following parameters

- **block size** – larger block sizes mean greater security, but reduced operation speed for a given algorithm, common size was 64 bits, now is 128 bits
- **key size** – larger key size means greater security, but may decrease performance, common size nowadays is 128 bits
- **number of rounds** – multiple rounds offer increasing security, a typical value is 16 rounds
- **subkey generation algorithm** – greater complexity should lead to greater difficulty of cryptanalysis
- **round function F** – greater complexity generally means greater resistance to cryptanalysis
- **fast encryption/decryption** – speed of execution of the algorithm might be a concern in embedded applications
- **ease of analysis** – if the algorithm can be concisely and clearly explained, it is easier to analyze for cryptanalytic vulnerabilities and therefore develop a higher level of assurance of its strength

## Data Encryption Standard

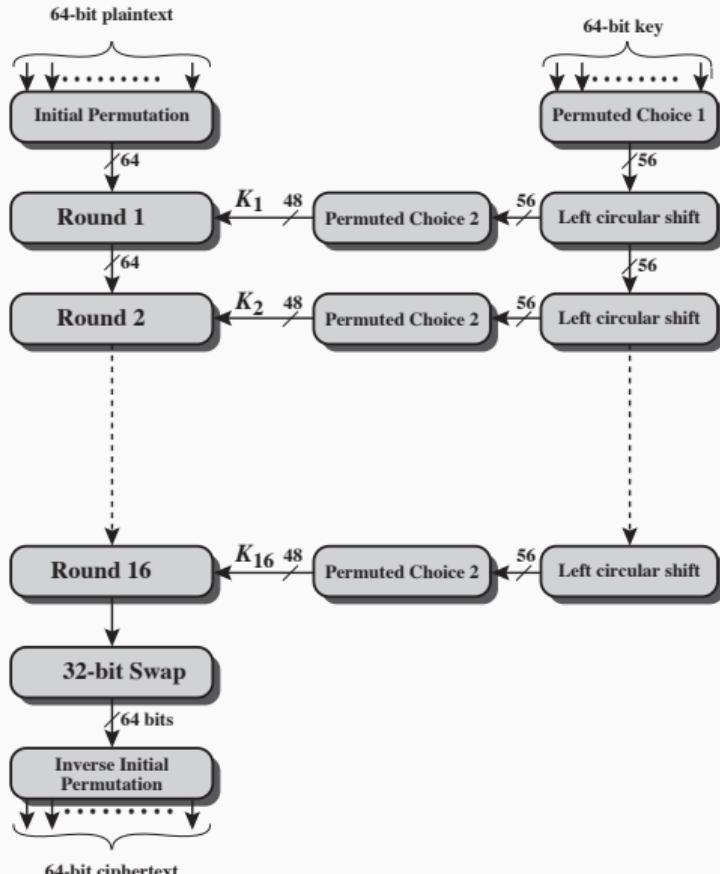
---

### History of DES

- DES was issued in 1977 by the National Bureau of Standards as **FIPS PUB 46**
- the algorithm itself is referred to as the Data Encryption Algorithm (DEA)
- data are encrypted in 64-bit blocks using a 56-bit key
- In 1994, NIST reaffirmed DES for federal use for another 5 years for the protection of non-classified information
- In 1999, NIST issued a new version of its standard **FIPS PUB 46-3** that DES should be used only for legacy systems and 3DES should be used instead
- until 2001 (when AES was introduced) was the most widely used encryption algorithm

## Internal scheme of DES Encryption Algorithm

- the algorithm expects a 64-bit key as input, but only 56 of these are ever used, the other 8 bits can be used as parity bits or simply set arbitrarily



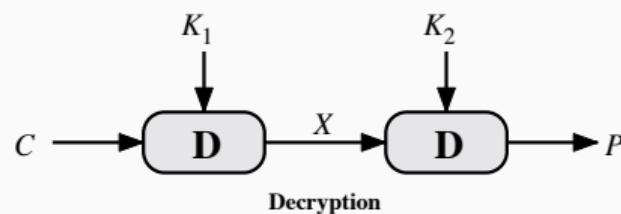
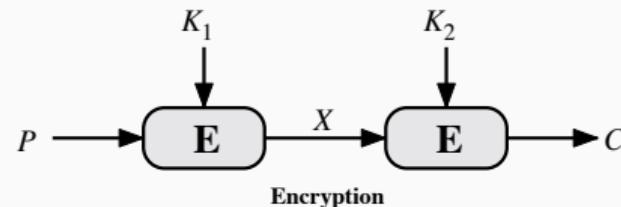
### Multiple encryption

- due to its short key and the increased computational power, DES can now be brute-forced
  - in 1997 the first DES Challenge was solved in 96 days by the DESCHALL Project
  - in 1998 the DES Challenge II-1 was solved in 39 days by distributed.net
  - still in 1998 Deep Crack decrypted a message after only 56 hours
  - in 1999 the DES Challenge III was solved in 22h15m by Deep Crack

### Multiple encryption

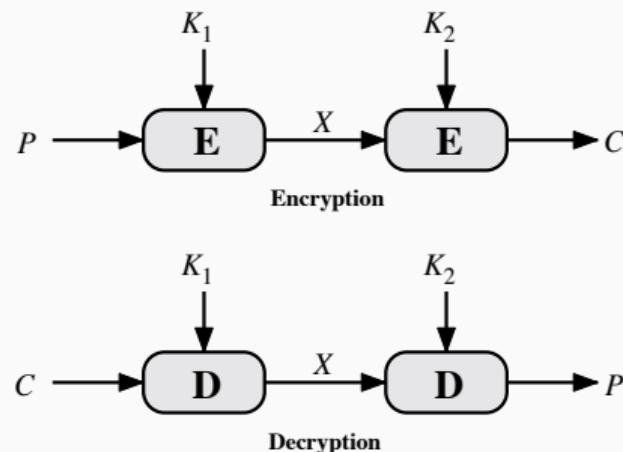
- due to its short key and the increased computational power, DES can now be brute-forced
  - in 1997 the first DES Challenge was solved in 96 days by the DESCHALL Project
  - in 1998 the DES Challenge II-1 was solved in 39 days by distributed.net
  - still in 1998 Deep Crack decrypted a message after only 56 hours
  - in 1999 the DES Challenge III was solved in 22h15m by Deep Crack
- to increase security and preserve the investment in software and equipment it is possible to use multiple encryption with DES and multiple keys
  - Double DES was created

Multiple encryption – Double DES



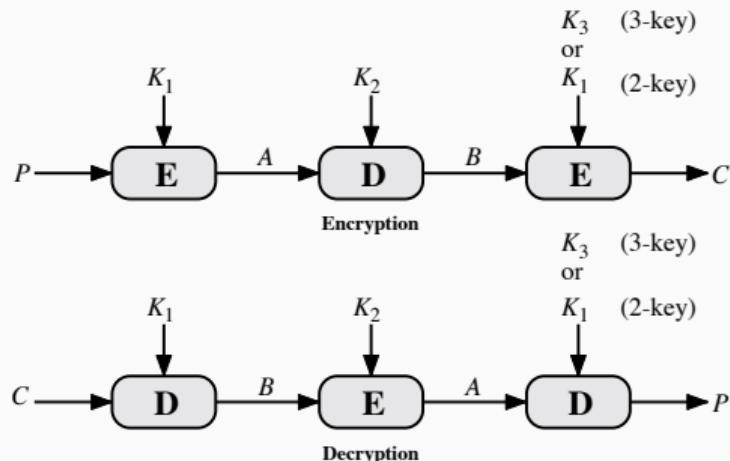
- apparently has a key strength of  $2 \times 56 = 112$  bits

### Multiple encryption – Double DES



- apparently has a key strength of  $2 \times 56 = 112$  bits
- meet-in-the-middle attack
  - if  $C = E_{k2}(E_{k1}(P))$ , then  $X = E_{k1}(P) = D_{k2}(C)$
  - the computational effort to crack 2DES is  $2^{56}$
  - therefore, this type of double encryption is not useful

## Triple DES (3DES)



- three stages of encryption solve the meet-in-the-middle attack
- Triple DES (3DES) was created with 2 versions
  - two-key ( $k_1, k_2$ ) triple encryption, 112 bits key compatible with single DES (1DES) if  $k_1 = k_2$
  - three-key ( $k_1, k_2, k_3$ ) triple encryption, 168 bits key



1. What is the difference between **stream** and **block** ciphers?
2. What is the problem of the **Ideal Block Cipher**?
3. What is the **product cipher**?
4. What are the parameters that Feistel took into account when designing a cipher?
5. Does multiple encryption always increase the robustness of the ciphertext?
6. Write the short representation of the 3DES with 168 bits key

$A \rightarrow B :$

## Block Cipher Modes of Operation

---

### Block Ciphers

- take a fixed-length block of text of length  $b$  bits and a key
- produces a  $b$  bits block of ciphertext
- the amount of plaintext to be encrypted can be greater than  $b$  bits
  - the plaintext must be broken into  $b$  bits blocks

### Block Ciphers

- take a fixed-length block of text of length  $b$  bits and a key
- produces a  $b$  bits block of ciphertext
- the amount of plaintext to be encrypted can be greater than  $b$  bits
  - the plaintext must be broken into  $b$  bits blocks
- multiple blocks of plaintext are encrypted using the same key
  - several security issues might arise
  - several modes of operation have been created to address those issues

Modes of Operation for any symmetric block cipher

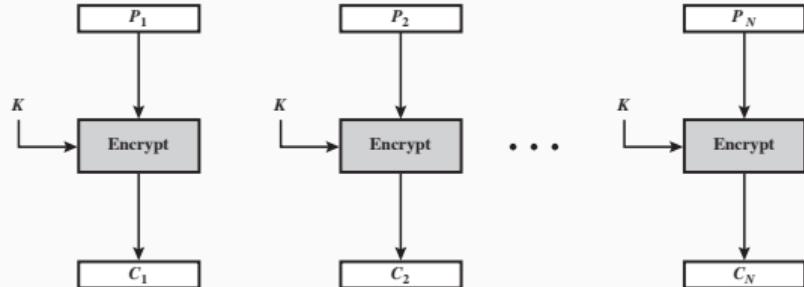
- Electronic Codebook (**ECB**)
- Cipher Block Chaining (**CBC**)
- Cipher Feedback (**CFB**)
- Output Feedback (**OFB**)
- Counter (**CTR**)

## BLOCK CIPHER MODES OF OPERATION

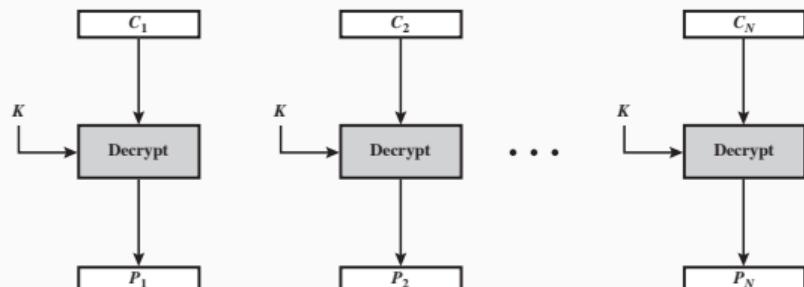
MODE	DESCRIPTION	TYPICAL APPLICATION
ECB	Each block of plaintext bits is encoded independently using the same key.	Secure transmission of single values, e.g. an encryption key
CBC	The input to the encryption algorithm is the XOR of the next block of plaintext and the preceding block of ciphertext.	General-purpose block-oriented transmission. Authentication.
CFB	Input is processed s bits at a time. Preceding ciphertext is used as input to the encryption algorithm to produce pseudorandom output, which is XORed with plaintext to produce next unit of ciphertext.	General-purpose stream-oriented transmission. Authentication.
OFB	Similar to CFB, except that the input to the encryption algorithm is the preceding encryption output, and full blocks are used.	Stream-oriented transmission over noisy channel e.g. satellite communication
CTR	Each block of plaintext is XORed with an encrypted counter. The counter is incremented for each subsequent block.	General-purpose block-oriented transmission. Useful for high-speed requirements

### Electronic Codebook (ECB)

- is the simplest mode
- plaintext is handled one block at a time
- each block of plaintext is encrypted using the same key
- for the same key, 2 equal plaintext blocks result in the same ciphertext
- should be used only to secure messages shorter than a single block
- ECB mode is not secure for lengthy message



(a) Encryption



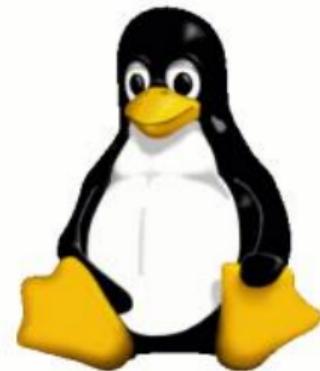
(b) Decryption

### Electronic Codebook (ECB)

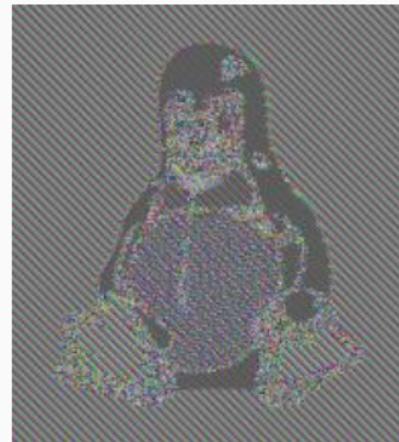
- example of pattern repetition in lengthy messages

source [https://en.wikipedia.org/wiki/Block\\_cipher\\_mode\\_of\\_operation](https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation)

Original image



ECB mode

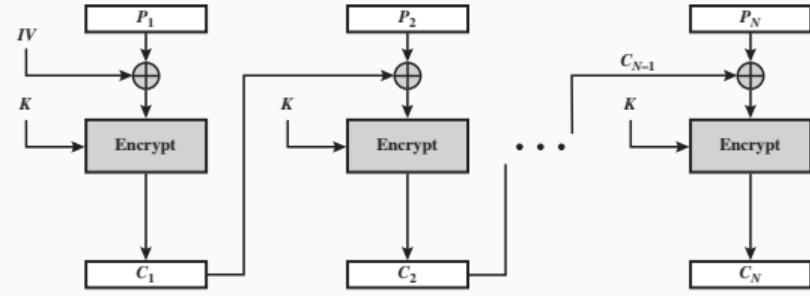


Other mode

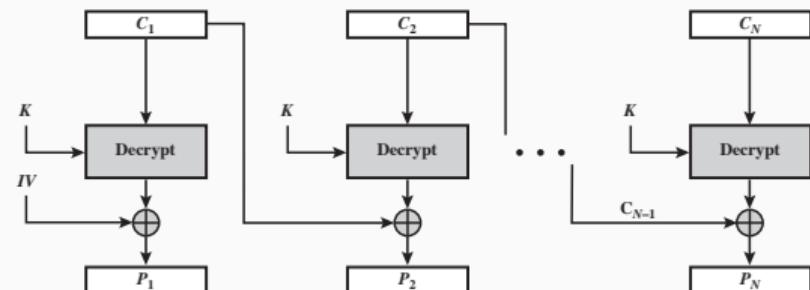


## Cipher Block Chaining (CBC)

- for the same key, 2 equal plaintext blocks result in different ciphertext
- repeating patterns of the plaintext blocks are not exposed on the ciphertext
- an initialization vector (IV) is XORed with the first block of plaintext
- the IV must be known to both the sender and receiver, but be unpredictable by a third party
- the IV should be protected against unauthorized changes, e.g. using ECB encryption
- propagates eventual transmission errors



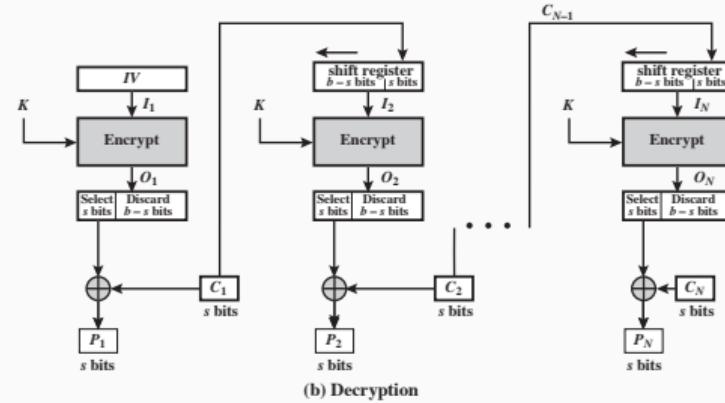
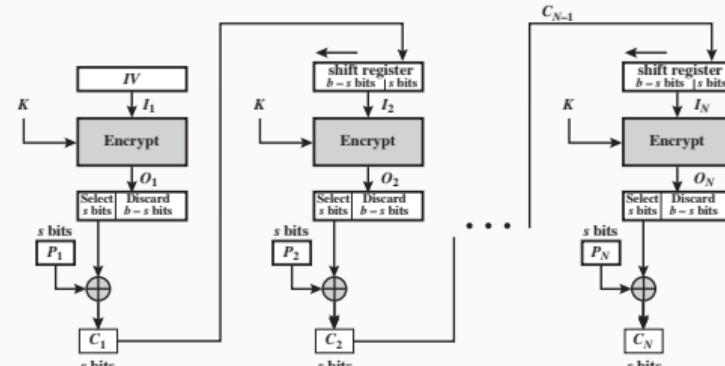
(a) Encryption



(b) Decryption

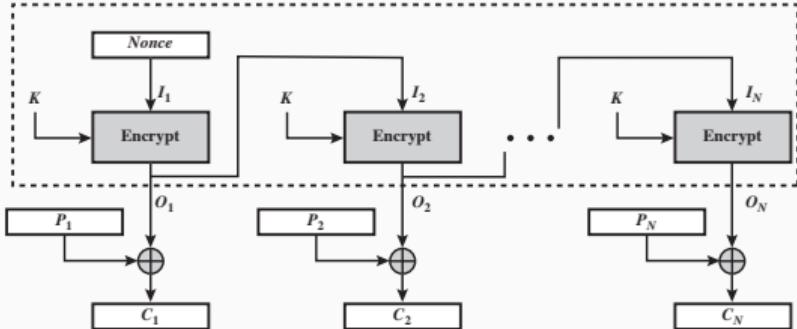
## Cipher Feedback (CFB)

- converts a block cipher into a stream cipher
- no need to pad a message in the last block
- can operate in real time
- the ciphertext has the same length of the plaintext
- an initialization vector (IV) is needed
- ideal to encrypt character streams
- propagates eventual transmission errors
- doesn't require decryption algorithm

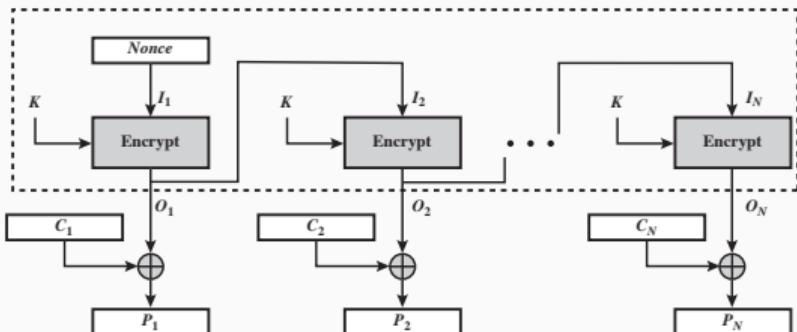


## Output Feedback (OFB)

- converts a block cipher into a stream cipher
- similar structure to CFB
- the output of the encryption function is fed back as the input for the next block
- eventual errors in transmission do not propagate
- more vulnerable to a message stream modification attack than CFB
- doesn't require decryption algorithm



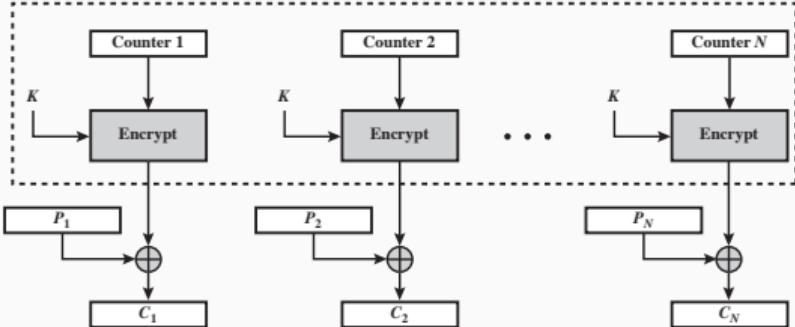
(a) Encryption



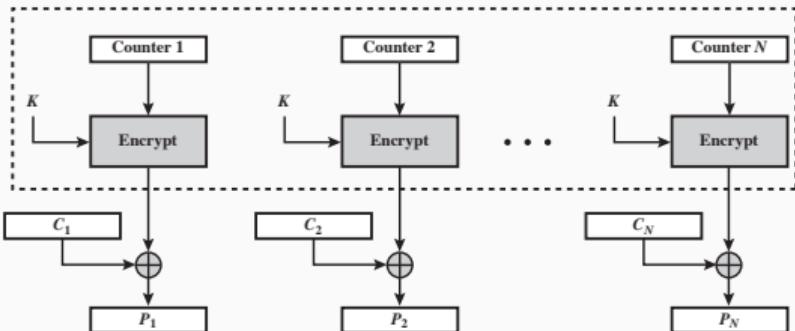
(b) Decryption

## Counter (CTR)

- uses a counter equal to the plaintext block size (similar to IV)
- the counter randomly is initialized and then incremented by 1 for each subsequent block
- simpler than CBC, CFB and OFB, but at least as secure
- doesn't require decryption algorithm
- no feed back
  - does not require padding
  - does not propagate transmission errors
  - is very efficient, encryption and decryption can be done in parallel



(a) Encryption



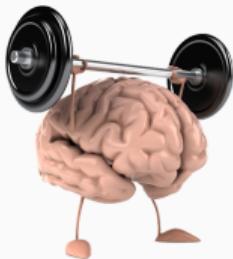
(b) Decryption

There are many other modes of operation

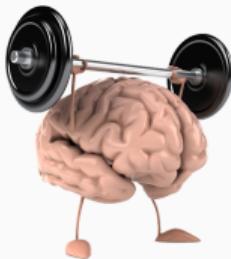
- but are out of the scope of this course
- many of them were specifically developed with encryption of large amounts of data in mind,  
e.g. storage devices
- some examples are XTS, LRW, XEX, CMC, EME
- for more information visit  
[https://en.wikipedia.org/wiki/Block\\_cipher\\_mode\\_of\\_operation](https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation)



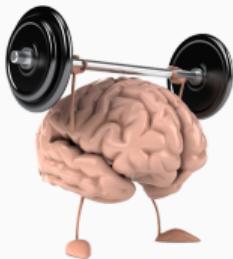
1. Why do some block cipher modes of operation only use encryption while others use both encryption and decryption?



1. Why do some block cipher modes of operation only use encryption while others use both encryption and decryption?
2. In the CBC mode transmission errors propagate. How many blocks can be affected by a single transmission error?



1. Why do some block cipher modes of operation only use encryption while others use both encryption and decryption?
2. In the CBC mode transmission errors propagate. How many blocks can be affected by a single transmission error?
3. Is it possible to perform encryption operations in parallel on multiple blocks of plaintext in CBC mode? How about decryption?



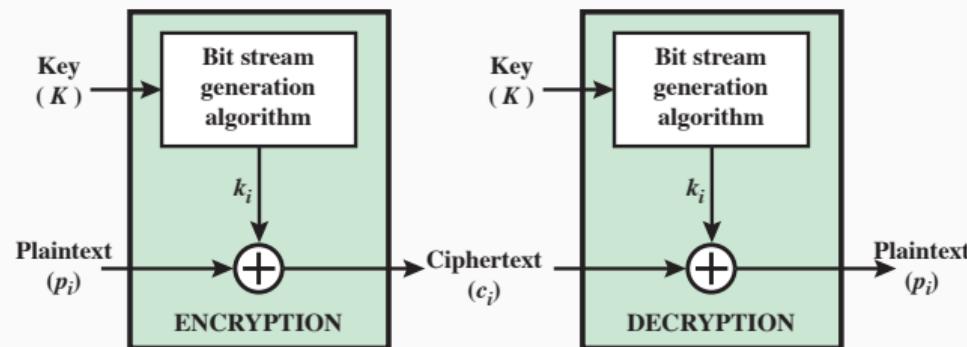
1. Why do some block cipher modes of operation only use encryption while others use both encryption and decryption?
2. In the CBC mode transmission errors propagate. How many blocks can be affected by a single transmission error?
3. Is it possible to perform encryption operations in parallel on multiple blocks of plaintext in CBC mode? How about decryption?
4. If a bit error occurs in the transmission of a ciphertext character in 8-bit CFB mode, how far does the error propagate?

## Stream Ciphers

---

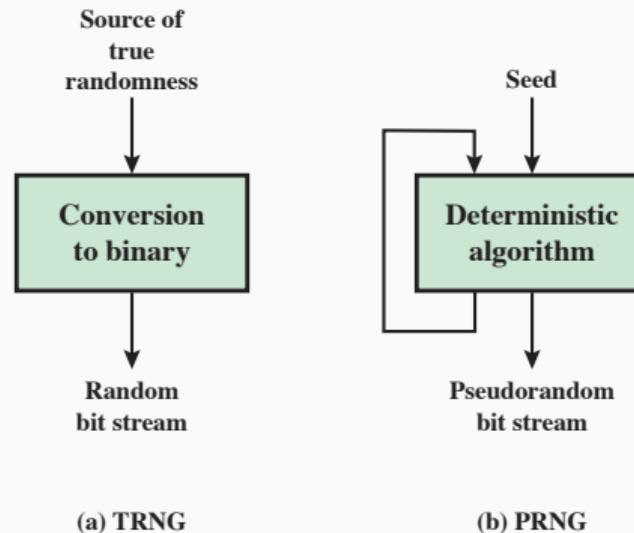
## How stream ciphers work

- a key is input to a pseudorandom bit generator that produces a stream of 8-bit numbers that are apparently random
- the output of the generator, is combined with the plaintext stream using the bitwise exclusive-OR (XOR) operation



### Random bits streams

- are used in key generation and encryption
- there are two classes of random bits generators
  1. **True Random Number Generators (TRNGs)** – produce bits non-deterministically using some physical source that produces random output
  2. **Pseudorandom Number Generators (PRNGs)** – computes bits deterministically using an algorithm



True Random Number Generators (TRNGs) are best fit to

- generate nonces (numbers used only once)
- generate keys (session keys and long term keys, such as RSA)

### True Random Number Generators (TRNGs) are best fit to

- generate nonces (numbers used only once)
- generate keys (session keys and long term keys, such as RSA)

RFC 4086 lists the following possible sources of randomness:

- sound/video input – the “input” from a sound digitizer with no source plugged in or from a camera with the lens cap on is essentially thermal noise and can provide reasonably high quality random bits
- disk drives – disk drives have small random fluctuations in their rotational speed due to chaotic air turbulence, low-level disk seek-time instructions produces a series of measurements with randomness
- to prevent bias, more than one source should be used

### Pseudorandom Number Generators (PRNGs)

- are best fit to generate a predictable bit stream for symmetric stream encryption
- but the seed must be unpredictable

Entropy source

True random number generator (TRNG)

Seed

Pseudorandom number generator (PRNG)

Pseudorandom bit stream

### Comparison of PRNGs and TRNGs

	PRNGs	TRNGs
Efficiency	Very efficient	Generally inefficient
Determinism	Deterministic	Nondeterministic
Periodicity	Periodic	aperiodic

### Stream ciphers versus block ciphers

- a stream cipher can be as secure as a block cipher of comparable key length
- stream ciphers are typically faster and use far less code than do block ciphers
- on block cipher keys can be reused, but not in a stream cipher
  - two plaintexts encrypted with the same key are easy to attack through cryptanalysis

## Symmetric Algorithms

---

1. Modern **symmetric algorithms** can be divided into:

### 1.1 Stream ciphers:

- can encrypt one bit at a time
- good for communications (HTTPs, SSH, etc)

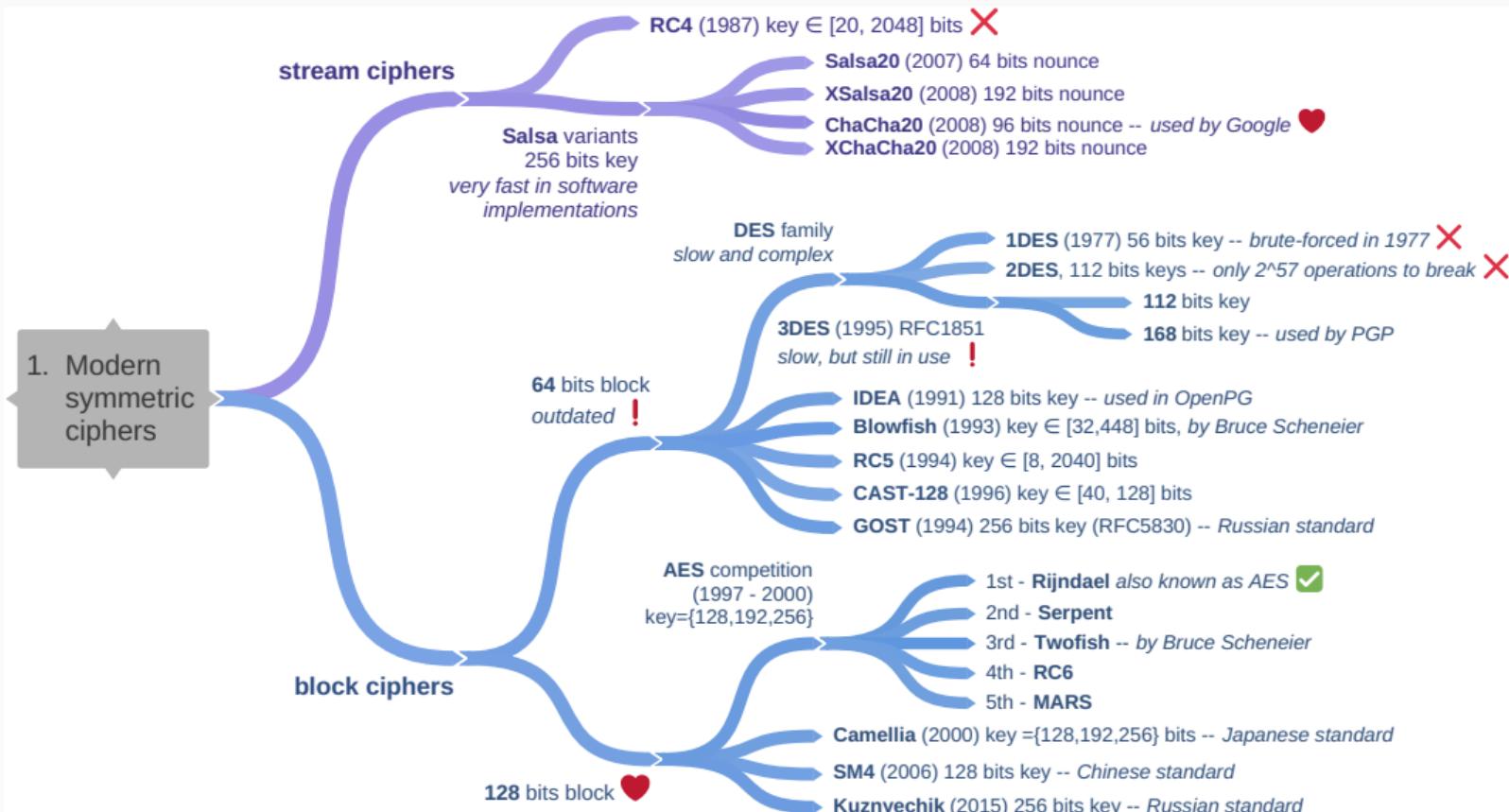
1. Modern **symmetric algorithms** can be divided into:

### 1.1 Stream ciphers:

- can encrypt one bit at a time
- good for communications (HTTPs, SSH, etc)

### 1.2 Block ciphers:

- encrypt a block of bits at a time
  - nowadays 128 bits block is the standard
- good to encrypt large amounts of data, such as files
- can be adapted to work as stream ciphers, but will be less efficient



Some tools that use symmetric encryption algorithms:

- disk-based encryption:
  - Veracrypt (Windows, MacOS, Linux)
  - BitLocker (Windows)
  - cryptfs (Linux)
  - dm-crypt (Linux, Android 7–9)
- file-based encryption:
  - EFS (Windows)
  - 7-zip (Windows, MacOS, Linux)
  - dm-crypt (Android 10–?)
- password managers:
  - keepass (Windows, MacOS, Linux, Android, iOS)

**VeraCrypt - Algorithms Benchmark**

Benchmark: Encryption Algorithm ▾

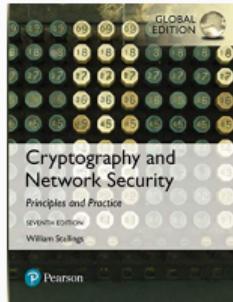
Buffer Size: 10,0 MB ▾

Algorithm	Encryption	Decryption	Mean	Benchmark
AES	1,6 GB/s	2,4 GB/s	2,0 GB/s	
Camellia	740 MB/s	679 MB/s	710 MB/s	
Twofish	485 MB/s	508 MB/s	496 MB/s	
AES(Twofish)	420 MB/s	427 MB/s	423 MB/s	Speed is affected by CPU load and storage device characteristics.  These tests take place in RAM.
Serpent	431 MB/s	409 MB/s	420 MB/s	
Serpent(AES)	400 MB/s	392 MB/s	396 MB/s	
Kuznyechik	413 MB/s	327 MB/s	370 MB/s	
Kuznyechik(AES)	322 MB/s	272 MB/s	297 MB/s	
Camellia(Serpent)	267 MB/s	272 MB/s	270 MB/s	
Camellia(Kuznyechik)	252 MB/s	223 MB/s	237 MB/s	
Twofish(Serpent)	232 MB/s	234 MB/s	233 MB/s	
AES(Twofish(Serpent))	214 MB/s	232 MB/s	223 MB/s	
Serpent(Twofish(AES))	209 MB/s	223 MB/s	216 MB/s	
Kuznyechik(Twofish)	220 MB/s	186 MB/s	203 MB/s	
Kuznyechik(Serpent(Camellia))	170 MB/s	145 MB/s	158 MB/s	

## Veracrypt benchmark

hardware acceleration for AES has a big impact on performance

# Questions?



Chapters 4, 7 and 8 of  
*William Stallings, Cryptography and Network Security: Principles and Practice, Global Edition, 2016*

# Systems' Security | Segurança de Sistemas

## Asymmetric Cryptography

---

Miguel Frade



## Overview

---

### Learning Objectives

Introduction

Public-key Algorithms

Diffie-Hellman Algorithm

Terminology

Exercises

## Learning Objectives

---

After this chapter, you should be able to:

1. Present an overview of the basic principles of public-key cryptosystems
2. Explain the two distinct uses of public-key cryptosystems
3. List and explain the requirements for a public-key cryptosystem.

## Introduction

---

### Misconceptions about public-key encryption

- public-key encryption is more secure than is symmetric encryption from cryptanalysis



### Misconceptions about public-key encryption

- public-key encryption is more secure than is symmetric encryption from cryptanalysis ← **wrong**
  - the security of any encryption scheme depends on the length of the key and the computational work involved in breaking a cipher



### Misconceptions about public-key encryption

- public-key encryption is more secure than is symmetric encryption from cryptanalysis ← **wrong**
  - the security of any encryption scheme depends on the length of the key and the computational work involved in breaking a cipher
- public-key encryption is a general-purpose technique that has made symmetric encryption obsolete



### Misconceptions about public-key encryption

- public-key encryption is more secure than is symmetric encryption from cryptanalysis ← **wrong**
  - the security of any encryption scheme depends on the length of the key and the computational work involved in breaking a cipher
- public-key encryption is a general-purpose technique that has made symmetric encryption obsolete ← **wrong**
  - symmetric encryption is faster and public-key cryptography are best suited for key management and digital signatures



### Misconceptions about public-key encryption

- public-key encryption is more secure than is symmetric encryption from cryptanalysis ← **wrong**
  - the security of any encryption scheme depends on the length of the key and the computational work involved in breaking a cipher
- public-key encryption is a general-purpose technique that has made symmetric encryption obsolete ← **wrong**
  - symmetric encryption is faster and public-key cryptography are best suited for key management and digital signatures
- public key distribution is trivial



### Misconceptions about public-key encryption

- public-key encryption is more secure than is symmetric encryption from cryptanalysis ← **wrong**
  - the security of any encryption scheme depends on the length of the key and the computational work involved in breaking a cipher
- public-key encryption is a general-purpose technique that has made symmetric encryption obsolete ← **wrong**
  - symmetric encryption is faster and public-key cryptography are best suited for key management and digital signatures
- public key distribution is trivial ← **wrong**
  - distribution of public keys does not require confidentiality, however to assure the authenticity of the public keys some form of protocol is needed and it is complex



### Genesis of asymmetric cryptography

Address two of the most difficult problems associated with symmetric encryption:

1. key distribution

- why develop impenetrable cryptosystems, if their users were forced to share their keys with a centralized system that could be compromised

### Genesis of asymmetric cryptography

Address two of the most difficult problems associated with symmetric encryption:

1. key distribution

- why develop impenetrable cryptosystems, if their users were forced to share their keys with a centralized system that could be compromised

2. digital signatures

- for widespread use on commercial and private purposes, electronic messages and documents would need the equivalent of signatures used in paper documents

### Genesis of asymmetric cryptography



In 1976, Whitfield Diffie and Martin Hellman created the first public-key cryptographic algorithm

- Diffie-Hellman – for symmetric key exchange

Requirements that public-key algorithm must fulfill

1. it is computationally easy for a party  $B$  to generate a key pair (public key  $PU_b$ , private key  $PR_b$ )

Requirements that public-key algorithm must fulfill

1. it is computationally easy for a party  $B$  to generate a key pair (public key  $PU_b$ , private key  $PR_b$ )
2. it is computationally easy for a sender  $A$  to encrypt a message  $M$ :  $C = E_{PU_b}(M)$

### Requirements that public-key algorithm must fulfill

1. it is computationally easy for a party  $B$  to generate a key pair (public key  $PU_b$ , private key  $PR_b$ )
2. it is computationally easy for a sender  $A$  to encrypt a message  $M$ :  $C = E_{PU_b}(M)$
3. it is computationally easy for the receiver  $B$  to decrypt the ciphertext using the private key to recover the original message:  $M = D_{PR_b}(C) \Leftrightarrow M = D_{PR_b}(E_{PU_b}(M))$

### Requirements that public-key algorithm must fulfill

1. it is computationally easy for a party  $B$  to generate a key pair (public key  $PU_b$ , private key  $PR_b$ )
2. it is computationally easy for a sender  $A$  to encrypt a message  $M$ :  $C = E_{PU_b}(M)$
3. it is computationally easy for the receiver  $B$  to decrypt the ciphertext using the private key to recover the original message:  $M = D_{PR_b}(C) \Leftrightarrow M = D_{PR_b}(E_{PU_b}(M))$
4. it is computationally infeasible to derive the private key  $PR_b$  from the public key  $PU_b$

### Requirements that public-key algorithm must fulfill

1. it is computationally easy for a party  $B$  to generate a key pair (public key  $PU_b$ , private key  $PR_b$ )
2. it is computationally easy for a sender  $A$  to encrypt a message  $M$ :  $C = E_{PU_b}(M)$
3. it is computationally easy for the receiver  $B$  to decrypt the ciphertext using the private key to recover the original message:  $M = D_{PR_b}(C) \Leftrightarrow M = D_{PR_b}(E_{PU_b}(M))$
4. it is computationally infeasible to derive the private key  $PR_b$  from the public key  $PU_b$
5. it is computationally infeasible to recover the original message  $M$  knowing the public key  $PU_b$ , and a ciphertext  $C$

### Public-Key Cryptanalysis

- public-key encryption algorithms are also vulnerable to a brute-force attack
  - the countermeasure is increase key size
  - but these algorithms depend on an invertible mathematical function whose complexity increases exponentially with the size of the key
  - the key size must be large enough to make brute-force attack impractical but small enough for practical encryption and decryption

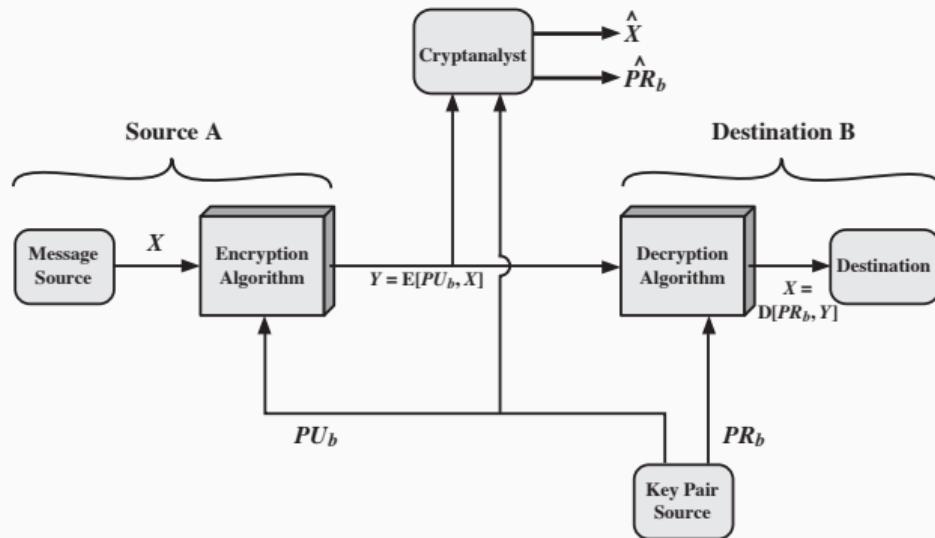
### Public-Key Cryptanalysis

- public-key encryption algorithms are also vulnerable to a brute-force attack
  - the countermeasure is increase key size
  - but these algorithms depend on an invertible mathematical function whose complexity increases exponentially with the size of the key
  - the key size must be large enough to make brute-force attack impractical but small enough for practical encryption and decryption
- find a way to compute the private key from the public key
  - to date, it has not been mathematically proven that this form of attack is infeasible
  - the history of cryptanalysis shows that, given the time, a problem that seems insoluble can be found to have a solution

**Security Services** that can be achieved with public-key algorithms

- confidentiality
- non-repudiation
  - includes integrity and authenticity
- both confidentiality and non-repudiation

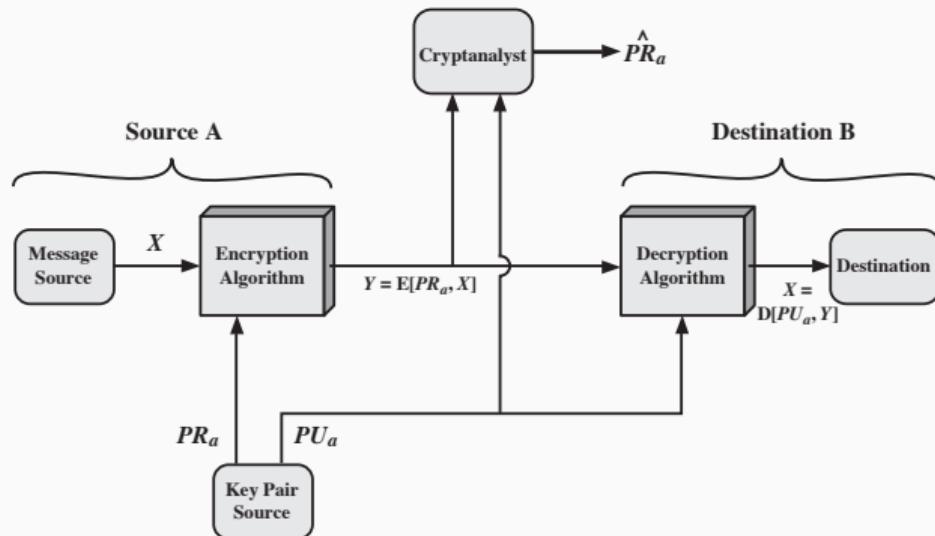
## Model of asymmetric encryption – confidentiality



Short representation

$$A \rightarrow B : E_{PU_B}(M)$$

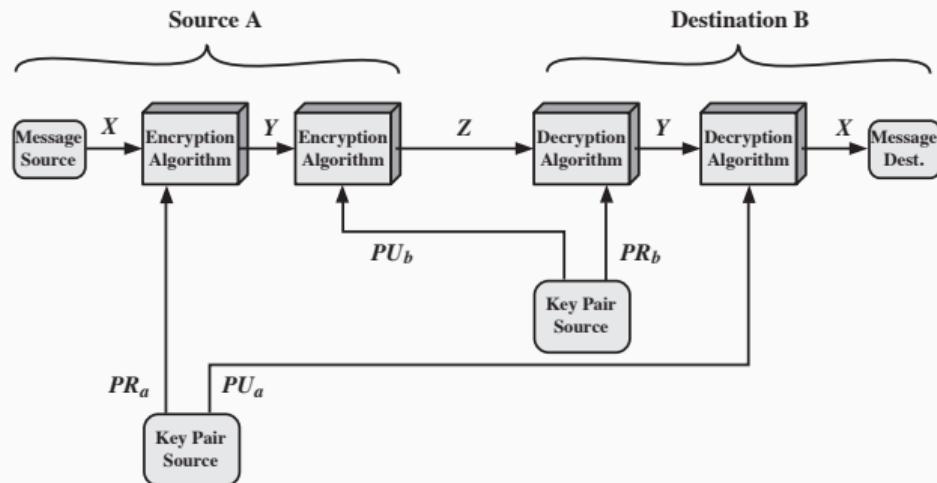
## Model of asymmetric encryption – non-repudiation



Short representation

$$A \rightarrow B : E_{PR_A}(M)$$

## Model of asymmetric encryption – confidentiality and non-repudiation



Short representation

$$A \rightarrow B : E_{PU_B}(E_{PR_A}(M))$$

## Public-key Algorithms

---

Public-key algorithms can be used for:

**Key exchange** two entities cooperate to exchange a session key (secret key for symmetric encryption generated for a session and valid for a short period of time)

**Digital signature** the sender “signs” a message with its private key. Signing is achieved by a cryptographic algorithm applied to the message or to a small block of data that is a function of the message

**Encryption/decryption** the sender encrypts a message with the recipient’s public-key, and the recipient decrypts the message with the recipient’s private-key.

## Applications for Public-Key Algorithms

ALGORITHM	ENCRYPTION	SIGNATURE	KEY EXCHANGE
Diffie–Hellman	—	—	Yes
DSA	—	Yes	—
RSA	Yes	Yes	Yes
ElGamal	Yes	Yes	Yes
Elliptic Curve	Yes	Yes	Yes

2. Asymmetric algorithms can be divided into:

### 2.1 Prime factorization

- large key sizes are required, minimum recommended is 2048 bits

2. Asymmetric algorithms can be divided into:

### 2.1 Prime factorization

- large key sizes are required, minimum recommended is 2048 bits

### 2.2 Discrete logarithm

- Modular Arithmetic
  - large key sizes are required, minimum recommended is 2048 bits
- Elliptic Curve
  - shorter key lengths, minimum recommended is 256 bits
  - faster than Modular Arithmetic and Prime factorization, except to verify digital signatures

2. Asymmetric algorithms can be divided into:

### 2.1 Prime factorization

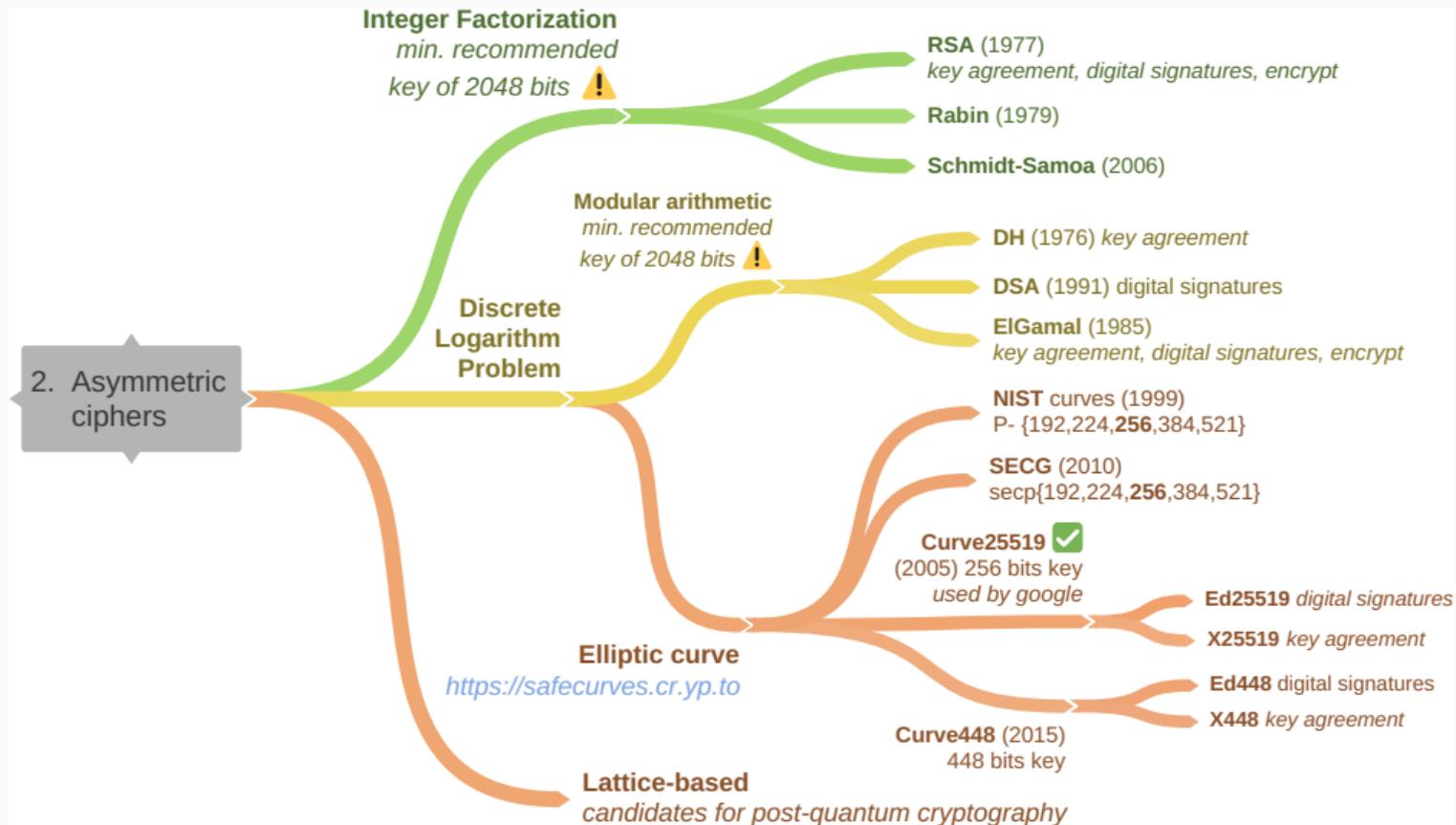
- large key sizes are required, minimum recommended is 2048 bits

### 2.2 Discrete logarithm

- Modular Arithmetic
  - large key sizes are required, minimum recommended is 2048 bits
- Elliptic Curve
  - shorter key lengths, minimum recommended is 256 bits
  - faster than Modular Arithmetic and Prime factorization, except to verify digital signatures

### 2.3 Lattice-based

- candidates for post-quantum cryptography



### Tools that use asymmetric algorithms

- digital signatures:
  - Portuguese eID software, aCCinaPDF, Acrobat Reader DC, ...
  - email clients, *i. e.* Thunderbird, Outlook, ...
- symmetric key distribution:
  - TLS, IPSec, SSH, ...
- symmetric key encapsulation (for encryption):
  - minilock.org, email clients with encryption (S/MIME, OpenPG), ...

### Comparing different public-key algorithms

- never use key size to compare different algorithms
- computational effort to brute-force an algorithm
- measured in Millions of Instructions Per Second over a year (MIPS-Year)
  - $1 \text{ MIPS-year} = 10^6 \text{ instructions/second} \times 86\,400 \text{ seconds/day} \times 365 \text{ days/year} = 3.15 \times 10^{13} \text{ instructions}$

## Comparing different public-key algorithms

- never use key size to compare different algorithms
- computational effort to brute-force an algorithm
- measured in Millions of Instructions Per Second over a year (MIPS-Year)
  - 1 MIPS-year =  $10^6$  instructions/second  $\times$  86 400 seconds/day  $\times$  365 days/year =  $3.15 \times 10^{13}$  instructions

Time to break (MIPS-Year)	RSA Key size (bits)	ECC key size (bits)
$10^4$	512	106
$10^8$	768	132
$10^{11}$	1 024	160
$10^{20}$	2 048	210
$10^{78}$	21 000	600

### Comparable strengths to resist a brute-force attack

Bits	Symmetric	RSA/DH/DSA	ECC
80	2DES	1 024	160 – 223
112	3DES	2 048	224 – 255
128	AES-128	3 072	256 – 383
192	AES-192	7 680	384 – 511
256	AES-256	15 360	$\geqslant$ 512

Source: NIST SP 800-57 Pt. 1 Rev. 4 (<https://csrc.nist.gov/publications/detail/sp/800-57-part-1/rev-4/final>)

## Diffie-Hellman Algorithm

---

## Key exchange



▶ See Youtube Video

### Key exchange

- the parties agree on two numbers, a prime number  $p$  and a primitive root of that prime  $a$
- public values in blue, and secret/private values in red
- for example  $p = 23$  and  $a = 5$

## Key exchange

- the parties agree on two numbers, a prime number  $p$  and a primitive root of that prime  $a$
- public values in blue, and secret/private values in red
- for example  $p = 23$  and  $a = 5$

Alice (A)

- private key is a random integer  $0 \leq PR_A < p$
- public key is  $PU_A = a^{PR_A} \bmod p$ 
  - $PR_A = 4$
  - $PU_A = 5^4 \bmod 23 = 4$

Bob (B)

- private key is a random integer  $0 \leq PR_B < p$
- public key is  $PU_B = a^{PR_B} \bmod p$ 
  - $PR_B = 3$
  - $PU_B = 5^3 \bmod 23 = 10$

## Key exchange

- the parties agree on two numbers, a prime number  $p$  and a primitive root of that prime  $a$
- public values in blue, and secret/private values in red
- for example  $p = 23$  and  $a = 5$

Alice (A)

- private key is a random integer  $0 \leq PR_A < p$
- public key is  $PU_A = a^{PR_A} \bmod p$ 
  - $PR_A = 4$
  - $PU_A = 5^4 \bmod 23 = 4$
- secret key
  - $k = PU_B^{PR_A} \bmod p$
  - $k = 10^4 \bmod 23 = 18$

Bob (B)

- private key is a random integer  $0 \leq PR_B < p$
- public key is  $PU_B = a^{PR_B} \bmod p$ 
  - $PR_B = 3$
  - $PU_B = 5^3 \bmod 23 = 10$
- secret key
  - $k = PU_A^{PR_B} \bmod p$
  - $k = 4^3 \bmod 23 = 18$

## Key exchange attack

- an adversary has access to  $p$ ,  $a$ ,  $PU_A$ , and  $PU_B$
- in order to get the key  $k$  he has to calculate the private key of A or B
  - $PR_A = dlog_{a,p}(PU_A)$ , or
  - $PR_B = dlog_{a,p}(PU_B)$
- and then calculate the  $k$  like the parties A, or B do

## Key exchange attack

- an adversary has access to  $p$ ,  $a$ ,  $PU_A$ , and  $PU_B$
- in order to get the key  $k$  he has to calculate the private key of A or B
  - $PR_A = dlog_{a,p}(PU_A)$ , or
  - $PR_B = dlog_{a,p}(PU_B)$
- and then calculate the  $k$  like the parties A, or B do

### Discrete Logarithm ( $dlog$ )

The discrete logarithm ( $dlog$ ) is computationally infeasible to calculate for very large numbers

## Terminology

---

**Asymmetric Keys** two related keys, a public key and a private key, that are used to perform complementary operations, such as encryption and decryption or signature generation and signature verification.

**Asymmetric Keys** two related keys, a public key and a private key, that are used to perform complementary operations, such as encryption and decryption or signature generation and signature verification.

**Public Key Certificate** A digital document issued and digitally signed by the private key of a Certification Authority that binds the name of a subscriber to a public key.

**Asymmetric Keys** two related keys, a public key and a private key, that are used to perform complementary operations, such as encryption and decryption or signature generation and signature verification.

**Public Key Certificate** A digital document issued and digitally signed by the private key of a Certification Authority that binds the name of a subscriber to a public key.

**Asymmetric Cryptographic Algorithm** A cryptographic algorithm that uses two related keys, a public key and a private key. Deriving the private key from the public key is computationally infeasible.

**Asymmetric Keys** two related keys, a public key and a private key, that are used to perform complementary operations, such as encryption and decryption or signature generation and signature verification.

**Public Key Certificate** A digital document issued and digitally signed by the private key of a Certification Authority that binds the name of a subscriber to a public key.

**Asymmetric Cryptographic Algorithm** A cryptographic algorithm that uses two related keys, a public key and a private key. Deriving the private key from the public key is computationally infeasible.

**Public Key Infrastructure (PKI)** A set of policies, processes, server platforms, software and workstations used for the purpose of administering certificates and public-private key pairs, including the ability to issue, maintain, and revoke public key certificates.

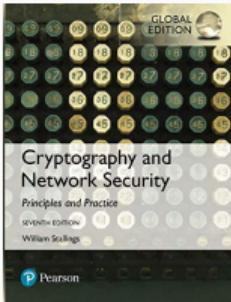
## Exercises

---



1. Consider the following expression:  $A \rightarrow B : E_{PR_B}(E_{PU_A}(M)) = C$   
Is it possible to compute  $C$ ? Justify.
  
2. Consider the following DH parameters:  $p = 353$  and  $a = 3$ .  
A chooses  $PR_A = 97$  as his private key and B chooses  $PR_B = 233$ .  
Calculate the secret key  $k$  using the DH algorithm.  
Tip: use <https://www.wolframalpha.com/> for the calculations.

# Questions?



Chapters 9 of  
*William Stallings, Cryptography and Network Security: Principles and Practice, Global Edition, 2016*

# Systems' Security | Segurança de Sistemas

## Cryptographic Hash Algorithms

---

Miguel Frade



## Overview

---

Learning Objectives

Introduction

Applications

Requirements

Authentication Algorithms

Exercises

## Learning Objectives

---

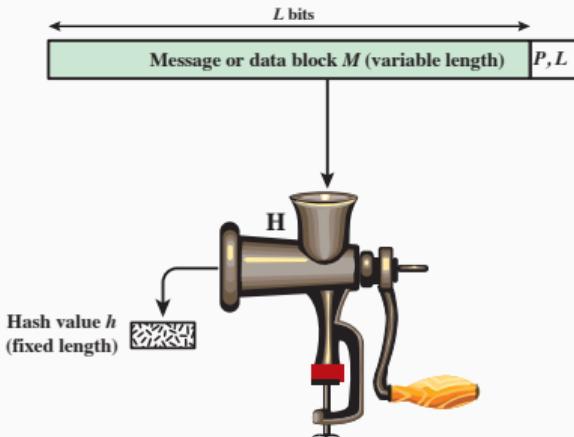
After this chapter, you should be able to:

1. Summarize the applications of cryptographic hash functions
2. Explain why a hash function used for message authentication needs to be secured
3. Understand the differences among preimage resistant, second preimage resistant, and collision resistant properties
4. Present an overview of the basic structure of cryptographic hash functions
5. Understand the birthday paradox and present an overview of the birthday attack

## Introduction

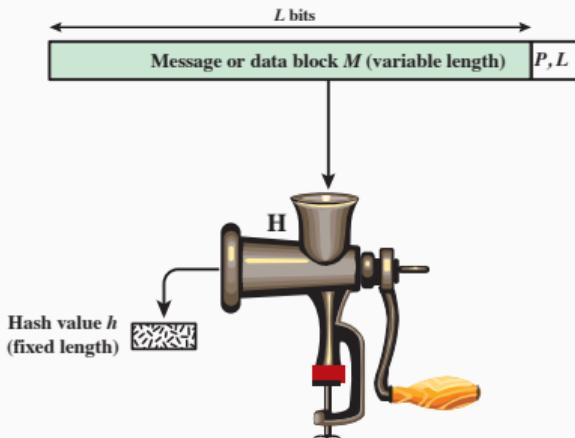
---

## Hash algorithms, or hash functions



$P, L$  = padding plus length field

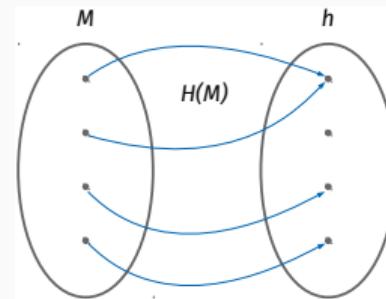
## Hash algorithms, or hash functions



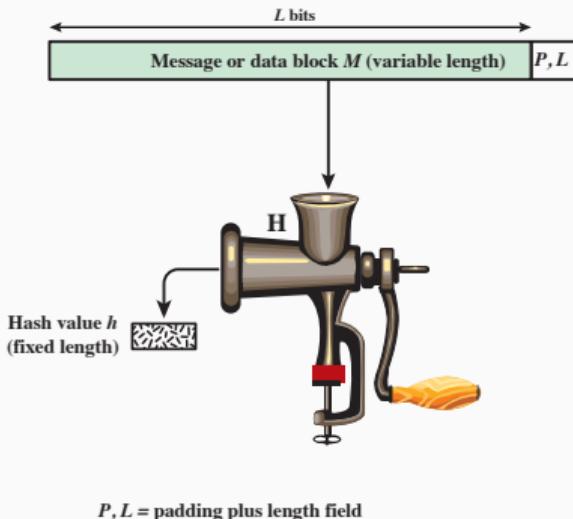
$P, L$  = padding plus length field

### Many-to-one function

- input – a message  $M$  of variable length
- output – a value  $h$  of fixed length, e.g. 256 bits

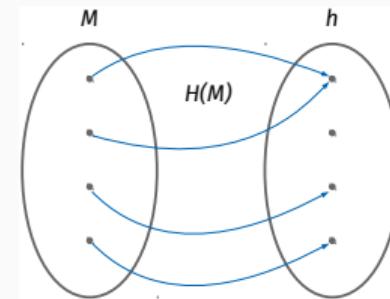


## Hash algorithms, or hash functions



Many-to-one function

- input – a message  $M$  of variable length
- output – a value  $h$  of fixed length, e.g. 256 bits



- size of the messages  $M$  universe =  $\infty$
- size of the hash values  $h$  universe =  $2^{n \text{ bits}}$

## Definitions

- hash function
  - accepts a variable-length block of data  $M$  as input and produces a fixed-size hash value  $h = H(M)$
  - if applied to a large set of inputs the output should be evenly distributed and apparently random
  - a change to any bit or bits in  $M$  results, with high probability, in a change to the hash value
  - are used to determine whether or not data has changed, that is, to verify data integrity

### Definitions

- hash function
  - accepts a variable-length block of data  $M$  as input and produces a fixed-size hash value  $h = H(M)$
  - if applied to a large set of inputs the output should be evenly distributed and apparently random
  - a change to any bit or bits in  $M$  results, with high probability, in a change to the hash value
  - are used to determine whether or not data has changed, that is, to verify data integrity

#### Cryptographic hash function

It must be computationally infeasible to find either:

- a data object that maps to a pre-specified hash result (the one-way property)
- two data objects that map to the same hash result (the collision-free property)

## Applications

---

Cryptographic Hash Functions are very versatile and can be used for:

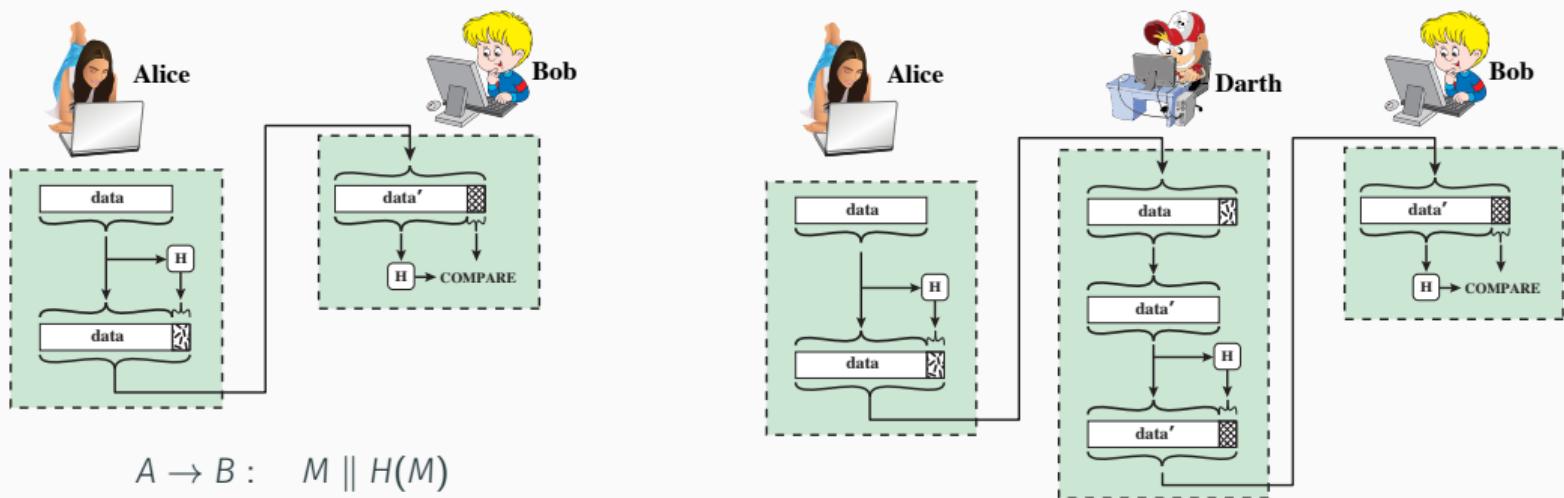
- data authentication
- digital signatures
- non-reversal password storage
- intrusion and virus detection
- pseudorandom number generator (PRNG)

### Data Authentication

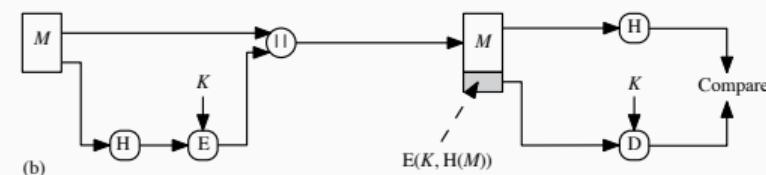
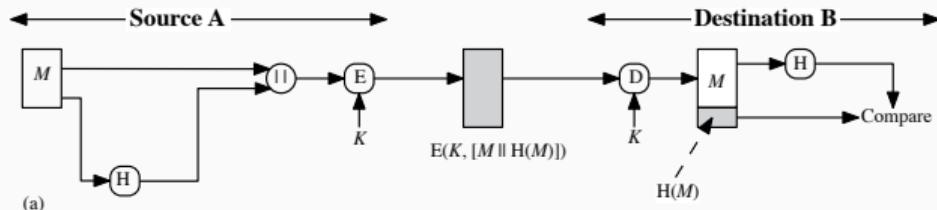
- to verify the integrity of a message assuring that data received are exactly as sent
  - there is no modification, insertion, deletion, or replay
- to assure the identity of the sender
- the hash value must be securely transmitted

## Data Authentication

- to verify the integrity of a message assuring that data received are exactly as sent
  - there is no modification, insertion, deletion, or replay
- to assure the identity of the sender
- the hash value must be securely transmitted



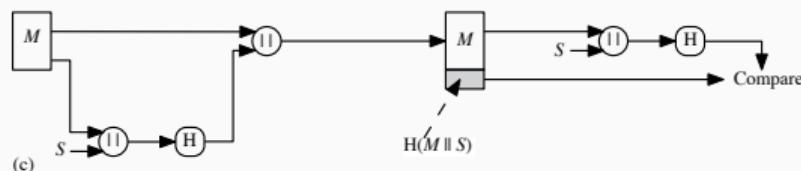
## Data Authentication – hash value protected with encryption



(a)  $A \rightarrow B : E_k(M \parallel H(M))$   
provides confidentiality of  $M$

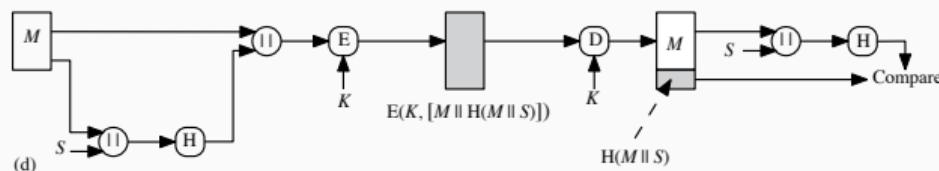
(b)  $A \rightarrow B : M \parallel E_k(H(M))$   
 $M$  can be read by anyone

## Data Authentication – hash value protected without encryption



(c)  $A \rightarrow B : M \parallel H(M \parallel S)$

$M$  can be read by anyone



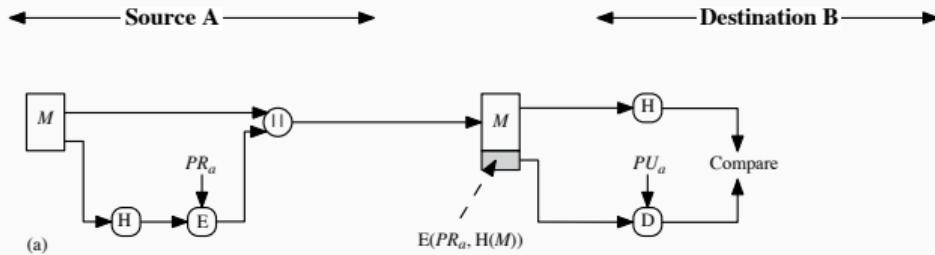
(d)  $A \rightarrow B : E_k(M \parallel H(M \parallel S))$

provides confidentiality of  $M$

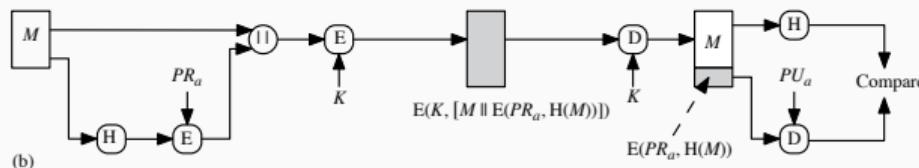
### Data Authentication – Message Authentication Code (MAC)

- other way to authenticate without encryption
- also known as keyed hash functions
- MAC will be addressed on another section

## Digital Signatures



(a)  $A \rightarrow B : M \parallel E_{PR_A}(H(M))$   
 $M$  can be read by anyone



(b)  $A \rightarrow B : E_K[M \parallel E_{PR_A}(H(M))]$   
provides confidentiality of  $M$

## Requirements

---

### Practical requirements

- Variable input size –  $H$  can be applied to a block of data of any size

### Practical requirements

- Variable input size –  $H$  can be applied to a block of data of any size
- Fixed output size –  $H$  produces a fixed-length output

### Practical requirements

- Variable input size –  $H$  can be applied to a block of data of any size
- Fixed output size –  $H$  produces a fixed-length output
- Efficiency –  $H(M)$  is relatively easy to compute for any given  $M$ , making both hardware and software implementations practical

### Definitions

- Preimage – For a hash value  $h = H(M)$ , we say that  $M$  is the preimage of  $h$

### Definitions

- Preimage – For a hash value  $h = H(M)$ , we say that  $M$  is the preimage of  $h$
- Collision
  - A collision occurs if we have  $M \neq N$  and  $H(M) = H(N)$
  - Collisions are undesirable for data integrity

### Cryptographic requirements

#### Preimage resistant

For any given hash value  $h$ , it is computationally infeasible to find  $N$  such that  $H(N) = h$ , i.e. the hash function is not reversible.

### Cryptographic requirements

#### Preimage resistant

For any given hash value  $h$ , it is computationally infeasible to find  $N$  such that  $H(N) = h$ , i.e. the hash function is not reversible.

#### Second preimage resistant (weak collision resistant)

For any given block  $M$ , it is computationally infeasible to find  $N \neq M$  with  $H(N) = H(M)$ .

### Cryptographic requirements

#### Preimage resistant

For any given hash value  $h$ , it is computationally infeasible to find  $N$  such that  $H(N) = h$ , i.e. the hash function is not reversible.

#### Second preimage resistant (weak collision resistant)

For any given block  $M$ , it is computationally infeasible to find  $N \neq M$  with  $H(N) = H(M)$ .

#### Collision resistant (strong collision resistant)

It is computationally infeasible to find any pair  $(M, N)$  with  $M \neq N$ , such that  $H(M) = H(N)$ .

### Cryptographic requirements

#### Preimage resistant

For any given hash value  $h$ , it is computationally infeasible to find  $N$  such that  $H(N) = h$ , i.e. the hash function is not reversible.

#### Second preimage resistant (weak collision resistant)

For any given block  $M$ , it is computationally infeasible to find  $N \neq M$  with  $H(N) = H(M)$ .

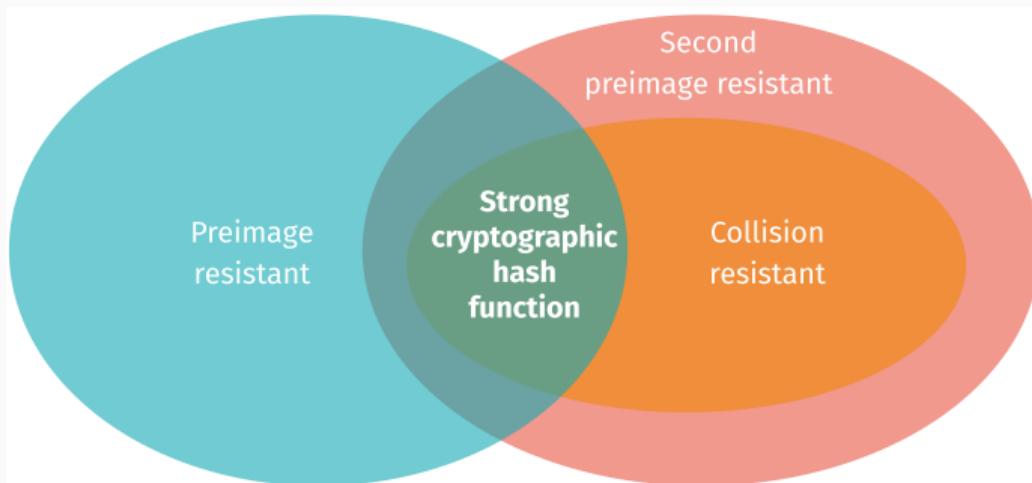
#### Collision resistant (strong collision resistant)

It is computationally infeasible to find any pair  $(M, N)$  with  $M \neq N$ , such that  $H(M) = H(N)$ .

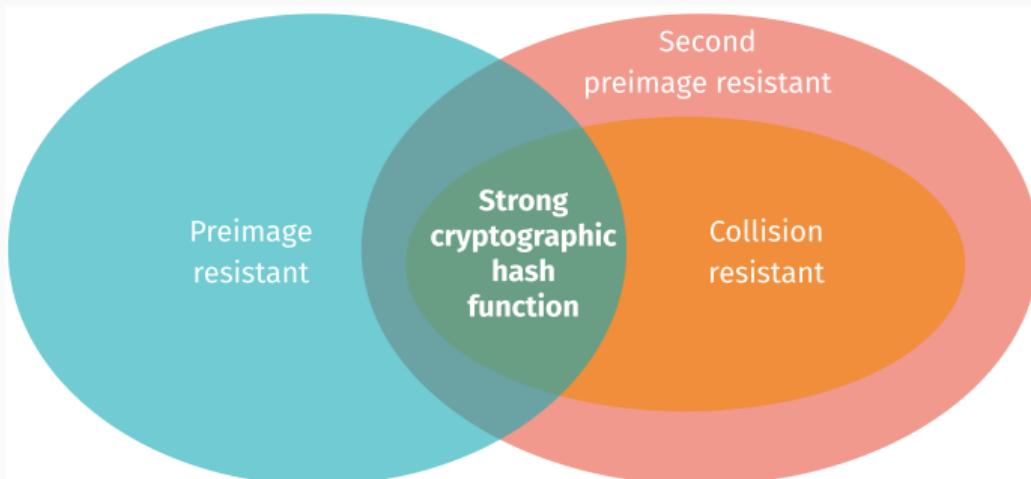
#### Pseudorandomness

Output of  $H$  meets standard tests for pseudorandomness.

### Cryptographic requirements



## Cryptographic requirements



Effort to attack a hash of  $m$  bits

RESISTANCE	OPERATIONS
preimage	$2^m$
$2^{nd}$ preimage	$2^m$
collision	$2^{\frac{m}{2}}$

## Cryptographic requirements for specific applications

APPLICATION	Type of resistance		
	Preimage	$2^{nd}$ preimage	Collision
digital signatures + hash	yes	yes	yes *
MAC	yes	yes	yes *
password storage	yes	—	—
IDS and virus	—	yes	—
encryption + hash	—	—	—

\* to protect against a chosen message attack

## Chosen message attack

As { the -- } Dean of Blakewell College, I have { had the pleasure of knowing known } Cherise Rosetti for the { last past } four years. She { has been was } { a tremendous an outstanding } { asset to role model in } { our the } school. I { would like to take this opportunity to wholeheartedly } recommend Cherise for your { school's -- } graduate program. I { am feel } { confident certain } { that -- } { she Cherise } will { continue to -- } succeed in her studies. { She Cherise } is a dedicated student and { thus far her grades her grades thus far } { have been are } { exemplary excellent }. In class, { she Cherise } { has proven to be has been } a take-charge { person who is individual } { who is -- } able to successfully develop plans and implement them.

{ She Cherise } has also assisted { us -- } in our admissions office. { She Cherise } has { successfully -- } demonstrated leadership ability by counseling new and prospective students. { Her Cherise's } advice has been { a great of considerable } help to these students, many of whom have { taken time to share shared } their comments with me regarding her pleasant and { encouraging reassuring } attitude. { For these reasons It is for these reasons that } I { highly recommend offer high recommendations for } Cherise { without reservation unreservedly }. Her { ambition drive } and { abilities potential } will { truly surely } be an { asset to plus for } your { establishment school }.

Letter with  $2^{38}$  variations

## Authentication Algorithms

---

### 3. Hash functions main characteristics

- are one-way functions → cannot be reversed
- have fixed length output, regardless of the input size
- the hash value must be protected

### 3. Hash functions main characteristics

- are one-way functions → cannot be reversed
- have fixed length output, regardless of the input size
- the hash value must be protected

Are used to:

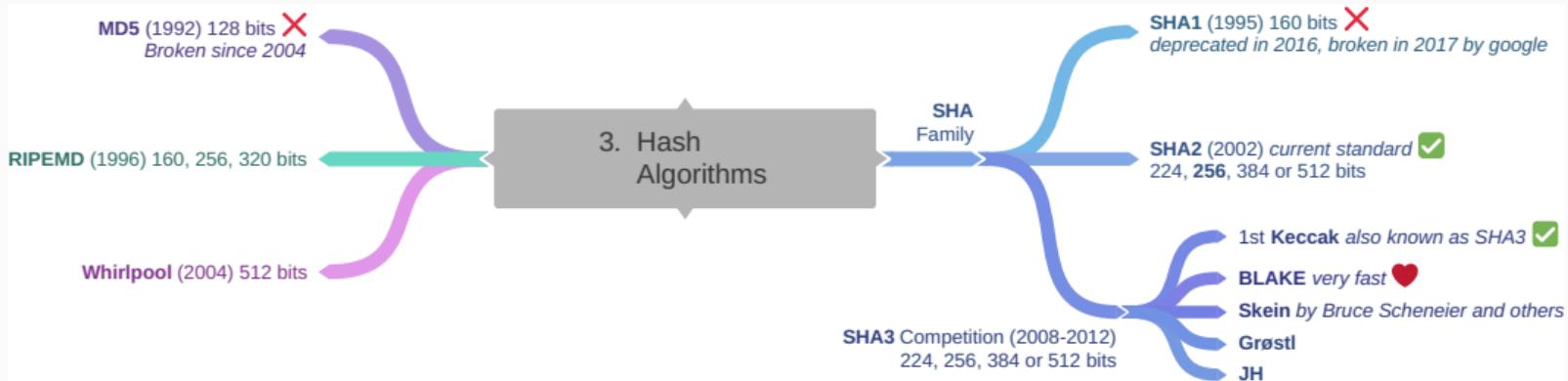
- data integrity verification
- digital signatures schemes
- crypto-currency
- file control version (*e.g.* git)
- find duplicate files
- intrusion detection systems and anti-virus

### 3. Hash functions main characteristics

- are one-way functions → cannot be reversed
- have fixed length output, regardless of the input size
- the hash value must be protected

Are used to:

- data integrity verification
- digital signatures schemes
- crypto-currency
- file control version (*e.g.* git)
- find duplicate files
- intrusion detection systems and anti-virus
- cryptographic schemes
  - Message Authentication Codes (MAC), also known as keyed hash
  - Key Derivation Functions (KDF)
  - One-Time-Password (OTP)



**Table 1:** Comparable strengths to resist a brute-force attack

Bits	Symmetric	Hash	ECC	RSA/DH/DSA
80	2DES	SHA1 (160)	160 – 223	1 024
112	3DES	SHA2-224	224 – 255	2 048
128	AES-128	SHA2-256	256 – 383	3 072
192	AES-192	SHA2-384	384 – 511	7 680
256	AES-256	SHA2-512	≥512	15 360

Source: NIST SP 800-57 Pt. 1 Rev. 4 (<https://csrc.nist.gov/publications/detail/sp/800-57-part-1/rev-4/final>)

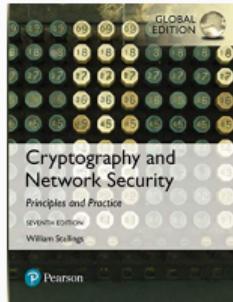
## Exercises

---



1. What are the 6 characteristics needed in a secure hash function?
2. What is the difference between weak and strong collision resistance?
3. Why it is required to protect the hash value?
4. In what ways can a hash value be secured so as to provide message authentication?

# Questions?



Chapters 11 of  
*William Stallings, Cryptography and Network Security: Principles and Practice, Global Edition, 2016*

## Systems' Security | Segurança de Sistemas

### Message Authentication Codes

---

Miguel Frade



## Overview

---

Learning Objectives

Introduction

Message Encryption as an Authenticator

Message Authentication Code as an Authenticator

Message Authentication Code

MAC Algorithms

Authenticated Encryption

Exercises

Cryptographic Schemes

Key Derivation Functions

One-Time Password

## Learning Objectives

---

After this chapter, you should be able to:

1. List and explain the possible attacks that are relevant to message authentication.
2. Define the term message authentication code.
3. List and explain the requirements for a message authentication code.
4. Explain the concept of authenticated encryption.

## Introduction

---

### Message Authentication

- one of the most complex areas of cryptography
- one approach is the usage of **Message Authentication Code (MAC)**
  - can be built from cryptographic hash functions
  - or built using a block cipher mode of operation
  - or a new approach known as authenticated encryption

### Types of attacks in the context of communications across a network:

- attacks to the confidentiality
  - **Disclosure** – release of message contents to any person or process not possessing the appropriate authorization (by means of a cryptographic key)
  - **Traffic analysis** – discovery of the pattern of traffic between parties (determine the frequency and duration of connections, the number and length of messages between parties)

### Types of attacks in the context of communications across a network:

- attacks to the confidentiality
  - **Disclosure** – release of message contents to any person or process not possessing the appropriate authorization (by means of a cryptographic key)
  - **Traffic analysis** – discovery of the pattern of traffic between parties (determine the frequency and duration of connections, the number and length of messages between parties)
- attacks to the non-repudiation
  - **Source repudiation** – denial of transmission of message by source
  - **Destination repudiation** – Denial of receipt of message by destination

### Types of attacks in the context of communications across a network (continuation):

- attacks to the authentication
  - **Masquerade** – insertion of messages into the network from a fraudulent source (also known as impersonification)
  - **Content modification** – changes to the contents of a message, including insertion, deletion, transposition, and modification
  - **Sequence modification** – any modification to a sequence of messages between parties, including insertion, deletion, and reordering
  - **Timing modification** – delay or replay of messages

### Types of attacks in the context of communications across a network (continuation):

- attacks to the authentication
  - **Masquerade** – insertion of messages into the network from a fraudulent source (also known as impersonification)
  - **Content modification** – changes to the contents of a message, including insertion, deletion, transposition, and modification
  - **Sequence modification** – any modification to a sequence of messages between parties, including insertion, deletion, and reordering
  - **Timing modification** – delay or replay of messages

#### Message Authentication

- is a procedure to verify that received messages come from the alleged source and have not been altered and may also verify sequencing and timeliness.
- a digital signature is an authentication technique that also includes measures to counter repudiation by the source.

### Message authentication strategies

1. **hash value** – the hash value serves as an authenticator, but must me protected (as stated on the previous chapter)

### Message authentication strategies

1. **hash value** – the hash value serves as an authenticator, but must me protected (as stated on the previous chapter)
2. **message encryption** – the ciphertext of the entire message serves as its authenticator

### Message authentication strategies

1. **hash value** – the hash value serves as an authenticator, but must me protected (as stated on the previous chapter)
2. **message encryption** – the ciphertext of the entire message serves as its authenticator
3. **message authentication code (MAC)** – a function of the message and a secret key that produces a fixed-length value that serves as the authenticator

## Message Encryption as an Authenticator

---

### Symmetric encryption

- $A \rightarrow B : E_k(M)$
- provides confidentiality
- if  $A$  and  $B$  are the only ones that know the key  $k$ , then can we say that the message could only be sent by  $A$ ?

### Symmetric encryption

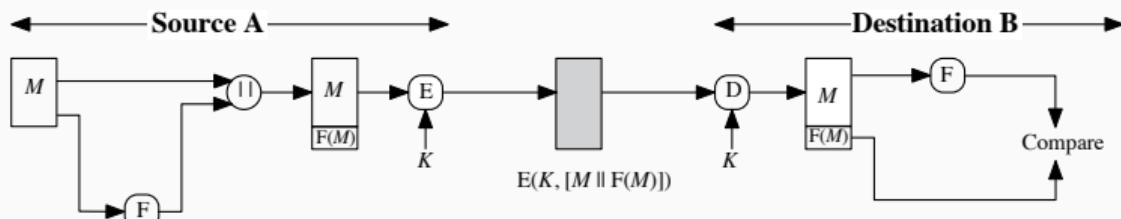
- $A \rightarrow B : E_k(M)$
- provides confidentiality
- if  $A$  and  $B$  are the only ones that know the key  $k$ , then can we say that the message could only be sent by  $A$ ?
  - no!
  - the decryption algorithms are blind, and can “decrypt” any pattern of bits
  - this means that an attacker could perform a DoS by sending random messages that once decrypted don’t have any meaning

### Symmetric encryption

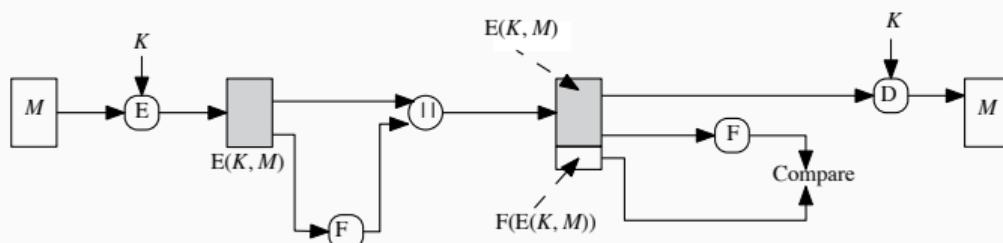
- $A \rightarrow B : E_k(M)$
- provides confidentiality
- if  $A$  and  $B$  are the only ones that know the key  $k$ , then can we say that the message could only be sent by  $A$ ?
  - no!
  - the decryption algorithms are blind, and can “decrypt” any pattern of bits
  - this means that an attacker could perform a DoS by sending random messages that once decrypted don’t have any meaning
- solution: add some form of error control

## Symmetric encryption + error control

- internal error control provides authentication
- external error control, may, or may not, provide authentication depending on the used function



(a) Internal error control



(b) External error control

### Public key encryption

- $A \rightarrow B : E_{PU_B}(M)$ 
  - provides confidentiality, but not authentication

### Public key encryption

- $A \rightarrow B : E_{PU_B}(M)$ 
  - provides confidentiality, but not authentication
- $A \rightarrow B : E_{PR_A}(M)$ 
  - provides authentication and non-repudiation, but not confidentiality
  - for the authentication some form of error control must be added

### Public key encryption

- $A \rightarrow B : E_{PU_B}(M)$ 
  - provides confidentiality, but not authentication
- $A \rightarrow B : E_{PR_A}(M)$ 
  - provides authentication and non-repudiation, but not confidentiality
  - for the authentication some form of error control must be added
- $A \rightarrow B : E_{PU_B}[E_{PR_A}(M)]$ 
  - provides confidentiality, authentication and non-repudiation
  - this approach is very slow

## Message Authentication Code as an Authenticator

---

### Message Authentication Code (MAC)

- use of a secret key to generate a small fixed-size block of data
- known as a cryptographic checksum or MAC
- this value is appended to the message
- does not provide a digital signature, because both sender and receiver share the same key

$$mac = C_k(M)$$

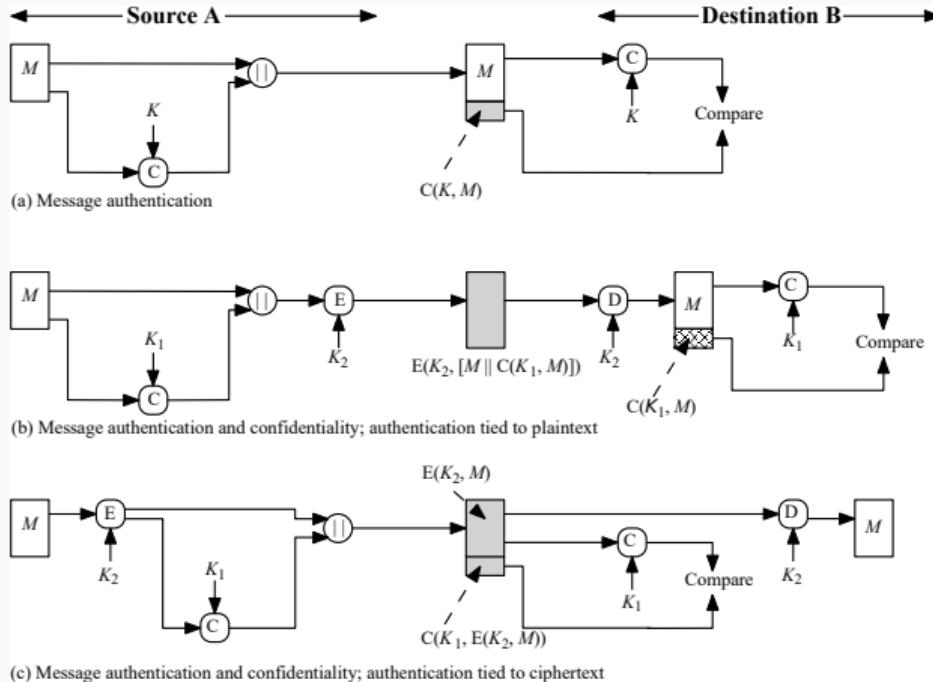
M input message

C MAC function

k shared secret key

mac message authentication code value  
also known as tag

## Message Authentication Code (MAC) – basic uses



Short representations

(a)  $A \rightarrow B : M \parallel C_k(M)$   
without confidentiality

(b)  $A \rightarrow B : E_{k_2}[M \parallel C_{k_1}(M)]$   
internal control

(c)  $A \rightarrow B : E_{k_2}(M) \parallel C_{k_1}[E_{k_2}(M)]$   
external control

## Message Authentication Code

---

### Cryptographic requirements

- for any given  $M$ , it is computationally infeasible to find  $N \neq M$  with  $C_k(N) = C_k(M)$   
an attacker should not be able to construct a new message to match a given tag without knowing the key

### Cryptographic requirements

- for any given  $M$ , it is computationally infeasible to find  $N \neq M$  with  $C_k(N) = C_k(M)$   
an attacker should not be able to construct a new message to match a given tag without knowing the key
- it is computationally infeasible, without knowing the key, to find any pair  $(M, N)$  with  $M \neq N$ , such that  $C_k(M) = C_k(N)$  – to prevent brute-force attacks based on chosen plaintext

### Cryptographic requirements

- for any given  $M$ , it is computationally infeasible to find  $N \neq M$  with  $C_k(N) = C_k(M)$   
an attacker should not be able to construct a new message to match a given tag without knowing the key
- it is computationally infeasible, without knowing the key, to find any pair  $(M, N)$  with  $M \neq N$ , such that  $C_k(M) = C_k(N)$  – to prevent brute-force attacks based on chosen plaintext
- the authentication algorithm should not be weaker with respect to certain parts of the message than others. If this were not the case, then an opponent who had  $M$  and  $C_k(M)$  could attempt variations on  $M$  at the known “weak spots” with a likelihood of early success at producing a new message that matched the old tags

### Security of MACs

- resistance to brute-force attacks
  - the assessment of strength is similar to that for symmetric encryption algorithms
  - the level of effort for brute-force attack on a MAC algorithm can be expressed as  $\min(2^k, 2^n)$ ,  
 $k$  = number of bits of the key and  $n$  = number of bits of the tag
  - the key length and tag length should be:  $\min(k, n) \geq 128$  bits

### Security of MACs

- resistance to brute-force attacks
  - the assessment of strength is similar to that for symmetric encryption algorithms
  - the level of effort for brute-force attack on a MAC algorithm can be expressed as  $\min(2^k, 2^n)$ ,  
 $k$  = number of bits of the key and  $n$  = number of bits of the tag
  - the key length and tag length should be:  $\min(k, n) \geq 128$  bits
- resistance to cryptanalysis
  - there is much more variety in the structure of MACs, so it is not possible to generalize about the cryptanalysis of MACs
  - there are far less works done on studying this type of attacks when compared with hash functions

## MAC Algorithms

---

There are two types of MACs:

- **block cipher-based** – can use any symmetric encryption algorithm
  - traditionally been the most common approach to constructing a MAC

There are two types of MACs:

- **block cipher-based** – can use any symmetric encryption algorithm
  - traditionally been the most common approach to constructing a MAC
- **hash based** – can use any hash algorithm (also known as keyed hash)
  - generally execute faster in software than symmetric block cipher-based
  - became more popular after the mandatory use of HMAC in IPsec

### Hash based MAC algorithms

- hash function such as SHA was not designed for use as a MAC because it does not rely on a secret key
- there are many ways to incorporate a secret key into an existing hash algorithm
- the approach that has received the most support is HMAC

### Hash based MAC algorithms

- hash function such as SHA was not designed for use as a MAC because it does not rely on a secret key
- there are many ways to incorporate a secret key into an existing hash algorithm
- the approach that has received the most support is HMAC

### HMAC

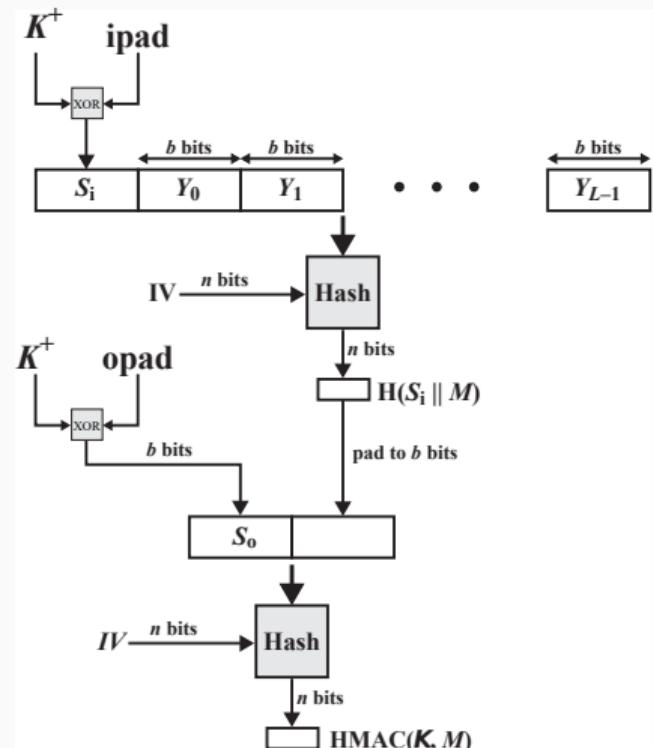
- is the mandatory-to-implement MAC for IP security
- is used in other Internet protocols, such as SSL/TLS
- is published as RFC 2104 and NIST standard (FIPS 198)

## Hash based MAC algorithms

### HMAC structure

1.  $K^+ = \text{append zeros to the left end of } k \text{ to create a } b\text{-bit string (e.g., if } K \text{ length} = 160 \text{ bits and } b = 512, \text{ then } K \text{ will be appended with } (512 - 160) = 352 \text{ bits with zeroes)}$
2.  $S_i = K^+ \oplus \text{ipad}$ , to produce the  $b$ -bit block ( $\text{ipad} = 00110110$  repeated  $b/8$  times)
3. append  $M$  to  $S_i$
4. apply hash to the result of step 3:  $H(S_i \parallel M)$
5.  $S_o = K^+ \oplus \text{opad}$ , to produce the  $b$ -bit block ( $\text{opad} = 01011100$  repeated  $b/8$  times)
6. Append the hash result from step 4 to  $S_o$
7. Apply  $H$  to the stream generated in step 6 and output the result

$$\text{HMAC}_k(M) = H[K^+ \oplus \text{opad} \parallel H(K^+ \oplus \text{ipad}) \parallel M]$$



### Hash based MAC algorithms

#### Security of HMAC

- depends on the cryptographic strength of the hash function
- if the security of the embedded hash function were compromised, the security of HMAC could be retained simply by replacing the embedded hash function with a more secure one

### Hash based MAC algorithms

#### Security of HMAC

- depends on the cryptographic strength of the hash function
- if the security of the embedded hash function were compromised, the security of HMAC could be retained simply by replacing the embedded hash function with a more secure one
- attacking HMAC is harder than attacking a hash function
  - the attacker cannot generate message/code pairs because he does not know  $k$
  - therefore, the attacker must observe a sequence of messages generated by HMAC under the same key
  - for a hash of 128 bits, this requires  $2^{64}$  observed blocks ( $2^{72}$  bits) with the same key
  - on a 1 Gbps link, one would need to observe a continuous stream of messages with no change in key for about 150 000 years in order to succeed
  - for that reason the use of MD5 is acceptable inside an HMAC

### Block-cipher based MAC algorithms

- use a symmetric cipher algorithm as its base
- symmetric cipher algorithms already have a key as input
- used to inside authenticated encryption schemes
- 2 proposals:
  - Data Authentication Algorithm (DAA) – based on DES and no longer safe to use
  - Cipher-based Message Authentication Code (CMAC) – a refinement of DAA, can be used with 3DES and AES

## Block-cipher based MAC algorithms

## CMAC structure

$$C_1 = E_k(M_1)$$

$$C_2 = E_k(M_2 \oplus C_1)$$

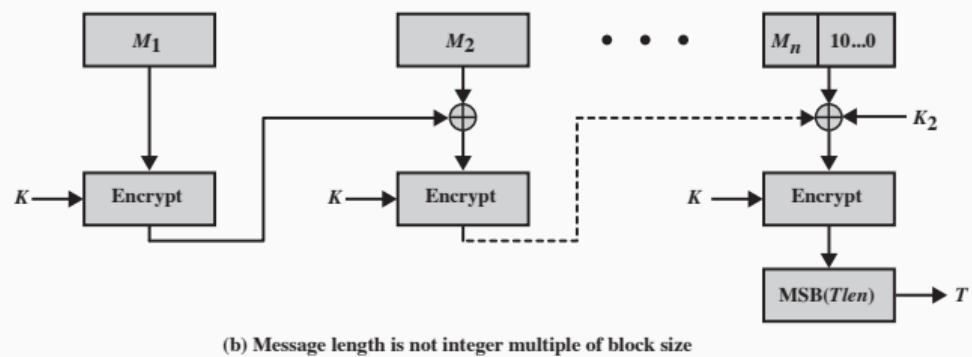
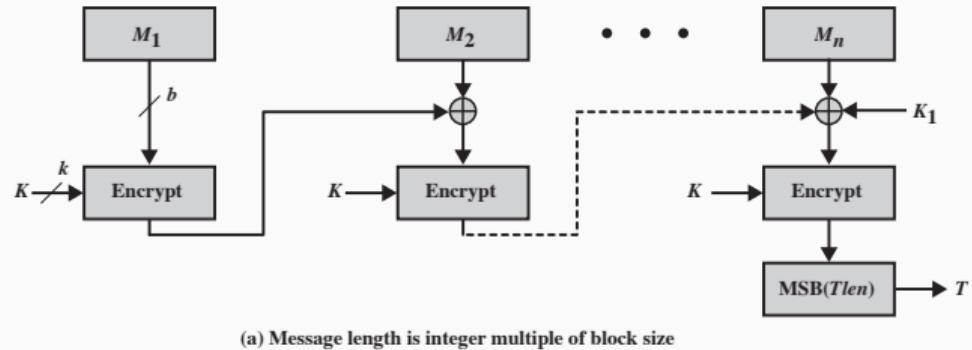
$$C_3 = E_k(M_3 \oplus C_2)$$

...

$$C_n = E_k(M_n \oplus C_{n-1} \oplus k_1)$$

$$T = \text{MSB}_{T_{\text{len}}}(C_n)$$

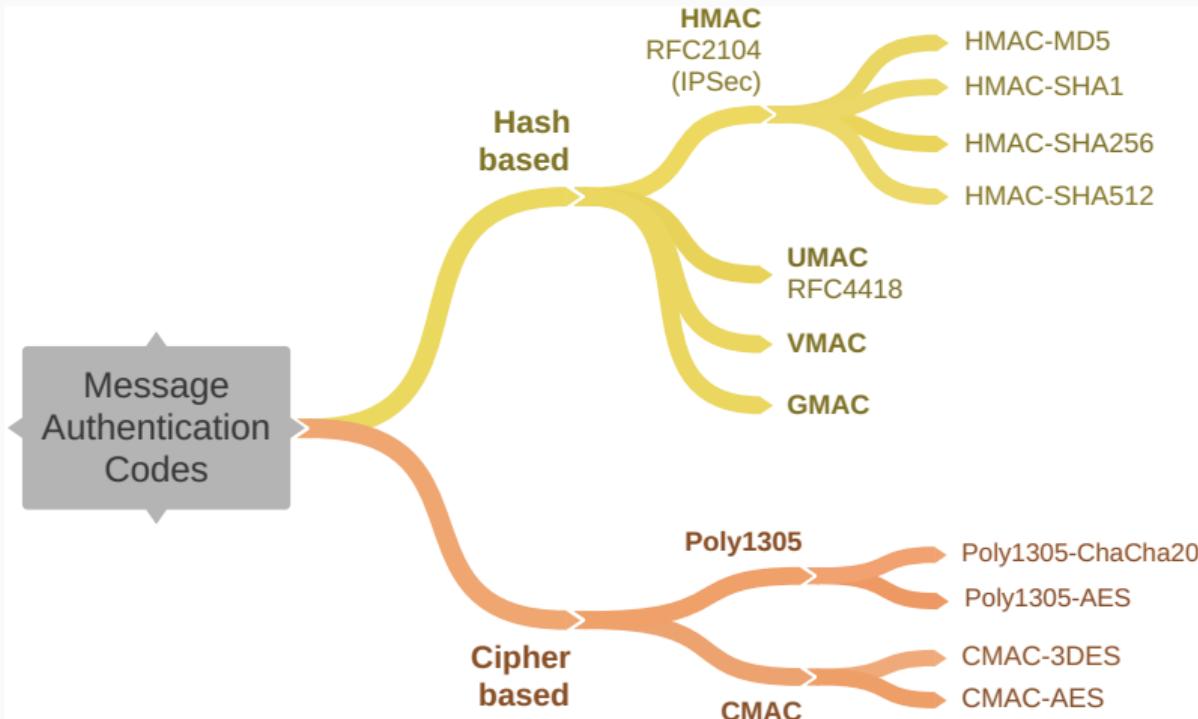
$k_1$  and  $k_2$  are derived from  $k$



### Block-cipher based MAC algorithms

#### Poly1305

- created by Daniel J. Bernstein
- the algorithm name is based on the fact that it evaluates the *modulus* of the prime number  $2^{130} - 5 = 1361129\,467\,683\,753\,853\,853\,498\,429\,727\,072\,845\,819$  (40 decimal digits)
- there are several variants:
  - Poly1305-AES
  - Poly1305-Salsa20
  - Poly1305-ChaCha20 – adopted by google for TLS communications (standardized in RFC 7905), can also be used in SSH



## Authenticated Encryption

---

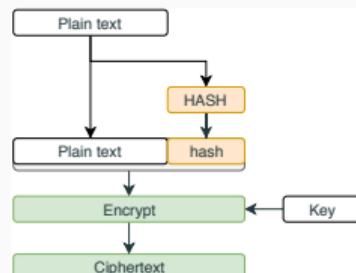
### Authenticated encryption (AE)

- is a term used to describe encryption systems that simultaneously protect confidentiality and authenticity (integrity)
- until recently the two services have been designed separately
  - Hashing followed by encryption
  - Independently encrypt and authenticate
  - Authentication followed by encryption
  - Encryption followed by authentication

## Traditional approaches

Hash-then-Encrypt

$$E_k[M \parallel H(M)]$$

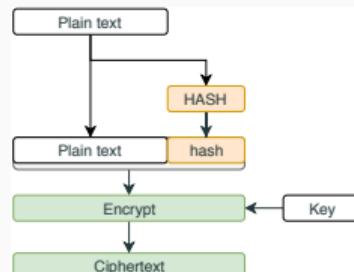


(used on WiFi WEP)

## Traditional approaches

Hash-then-Encrypt

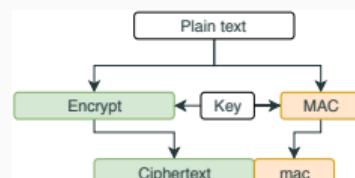
$$E_k[M \parallel H(M)]$$



(used on WiFi WEP)

Encrypt-and-MAC

$$E_k(M) \parallel C_k(M)$$

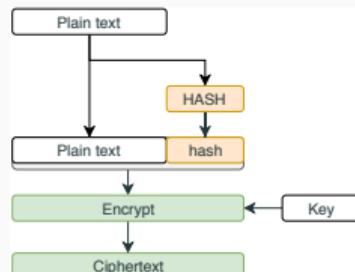


(used on SSH)

## Traditional approaches

Hash-then-Encrypt

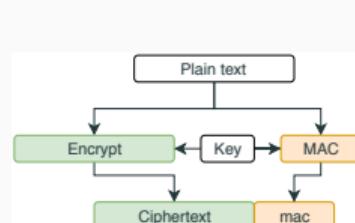
$$E_k[M \parallel H(M)]$$



(used on WiFi WEP)

Encrypt-and-MAC

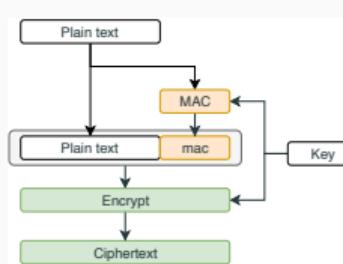
$$E_k(M) \parallel C_k(M)$$



(used on SSH)

MAC-then-Encrypt

$$E_k[M \parallel C_k(M)]$$

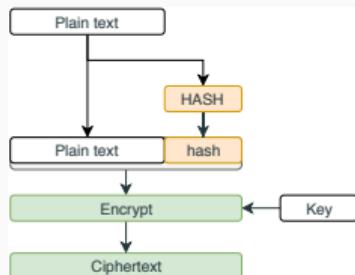


(used on TLS)

## Traditional approaches

Hash-then-Encrypt

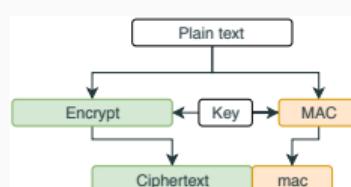
$$E_k[M \parallel H(M)]$$



(used on WiFi WEP)

Encrypt-and-MAC

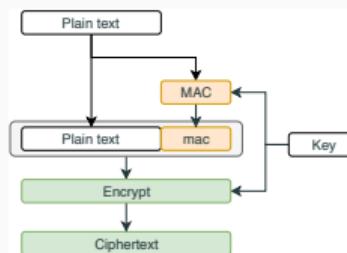
$$E_k(M) \parallel C_k(M)$$



(used on SSH)

MAC-then-Encrypt

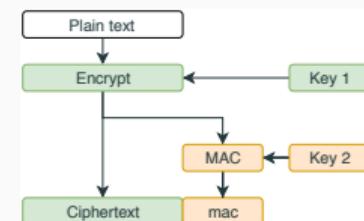
$$E_k[M \parallel C_k(M)]$$



(used on TLS)

Encrypt-then-MAC

$$E_{k2}(M) \parallel C_{k1}[E_{k2}(M)]$$



(used on IPsec)

### Traditional approaches

- these approaches require two passes through the data being protected (time consuming)
- there are security vulnerabilities with all of them
- nevertheless, with proper design, any of these approaches can provide a high level of security

### Traditional approaches

- these approaches require two passes through the data being protected (time consuming)
- there are security vulnerabilities with all of them
- nevertheless, with proper design, any of these approaches can provide a high level of security

### Authenticated encryption (AE)

- requires only a single pass through the data being protected (more efficient)
- address the security vulnerabilities of the traditional approaches
- it is possible to authenticate data that is not encrypted:
  - Authenticated Encryption with Associated Data (AEAD)

### Authenticated Encryption with Associated Data (AEAD):

- allows a recipient to check the integrity of both the **encrypted** and **unencrypted** information in a message
- associated data – is a part of the message that does not require confidentiality, but must be authentic, *e.g.* network protocol headers

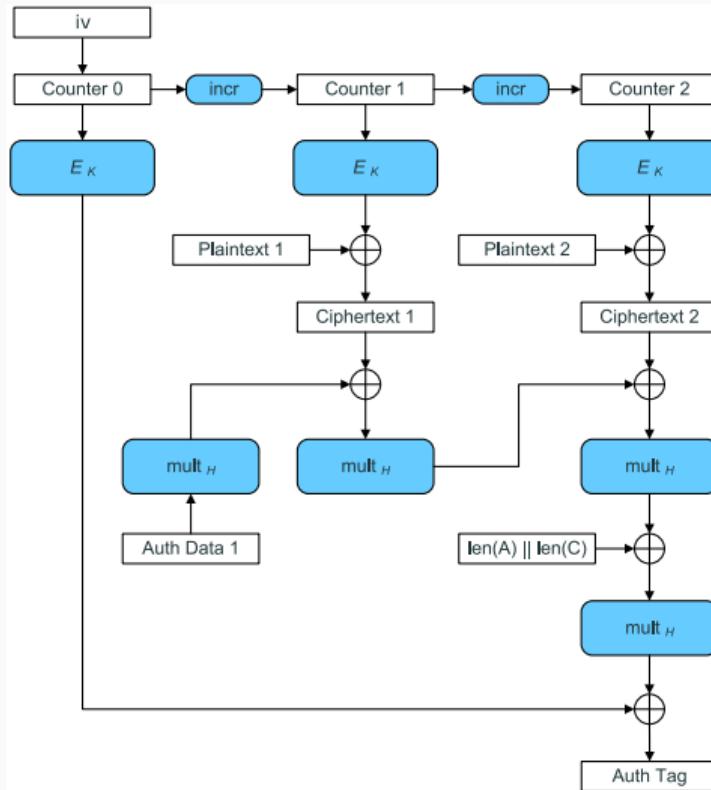
### Authenticated Encryption with Associated Data (AEAD):

- allows a recipient to check the integrity of both the **encrypted** and **unencrypted** information in a message
- associated data – is a part of the message that does not require confidentiality, but must be authentic, *e.g.* network protocol headers

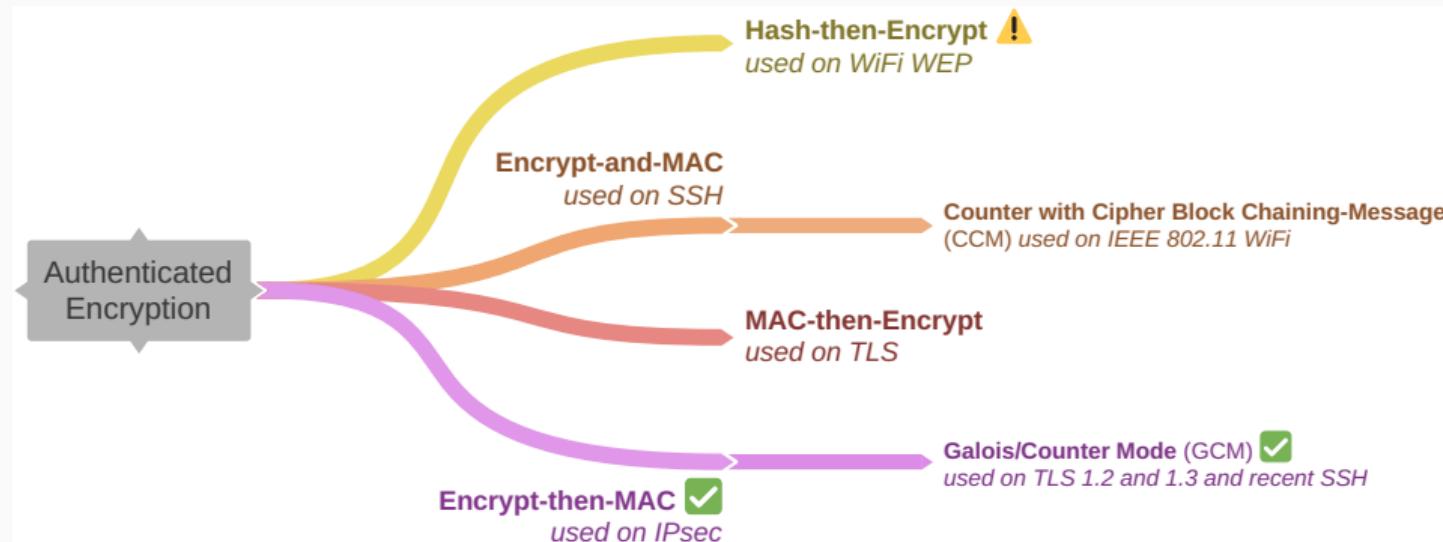
### Some Variants:

- **Counter with Cipher Block Chaining-Message** (CCM) – used by IEEE 802.11 WiFi
- **Galois/Counter Mode** (GCM) – mode of operation for symmetric-key cryptographic block ciphers that has been widely adopted because of its performance
  - processes with a single pass over the data
  - supported in the newer versions of IPsec, SSH, TLS 1.2 and TLS 1.3, ...

## AUTHENTICATED ENCRYPTION WITH ASSOCIATED DATA – GALOIS/COUNTER MODE (GCM)



## AUTHENTICATED ENCRYPTION



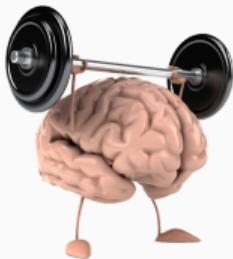
**Table 1:** Limitations for an untruncated 128-bit authentication tag

<i>Construction</i>	Max bytes for a single (key,nonce)	Max bytes for a single key
AES256-GCM <sup>1</sup>	64 GB <sup>2</sup>	≈ 350 GB (for ≈16 KB long messages)
ChaCha20-Poly1305 <sup>3</sup>	$2^{64}$ bytes (no practical limit)	Up to $2^{64}$ messages (no practical limit)
ChaCha20-Poly1305-IETF <sup>3</sup>	256 GB <sup>2</sup>	Up to $2^{64}$ messages (no practical limit)
XChaCha20-Poly1305-IETF <sup>3,4</sup>	$2^{64}$ bytes (no practical limit)	Up to $2^{64}$ messages (no practical limit)

<sup>1</sup> fastest algorithm due to hardware acceleration introduced by Intel in the Westmere processors (in 2010) and newer<sup>2</sup> it is possible to overcome the limitation by rekeying: using a new (key,nonce) pair<sup>3</sup> faster than AES in software only implementations (without AES specific hardware acceleration)<sup>4</sup> is the safest choice (key = 256 bits, block = 512 bits, nonce = 192 bits), but not wide spread usage

## Exercises

---



1. What types of attacks are addressed by message authentication?
2. What are some approaches to producing message authentication?
3. When a combination of symmetric encryption and an error control code is used for message authentication, in what order must the two functions be performed?
4. What is a message authentication code?
5. What is the difference between a message authentication code and a one-way hash function?

## Cryptographic Schemes

---

Use one, or more, cryptographic algorithm, to achieve new goals. There are four types of cryptographic schemes with hash:

1. Message Authentication Code (MAC) – already addressed
2. Authenticated Encryption – already addressed
3. Key Derivation Functions (KDF) – to store passwords in a non-reversible fashion
4. One-Time Password (OTP) – mainly used as second authentication factor

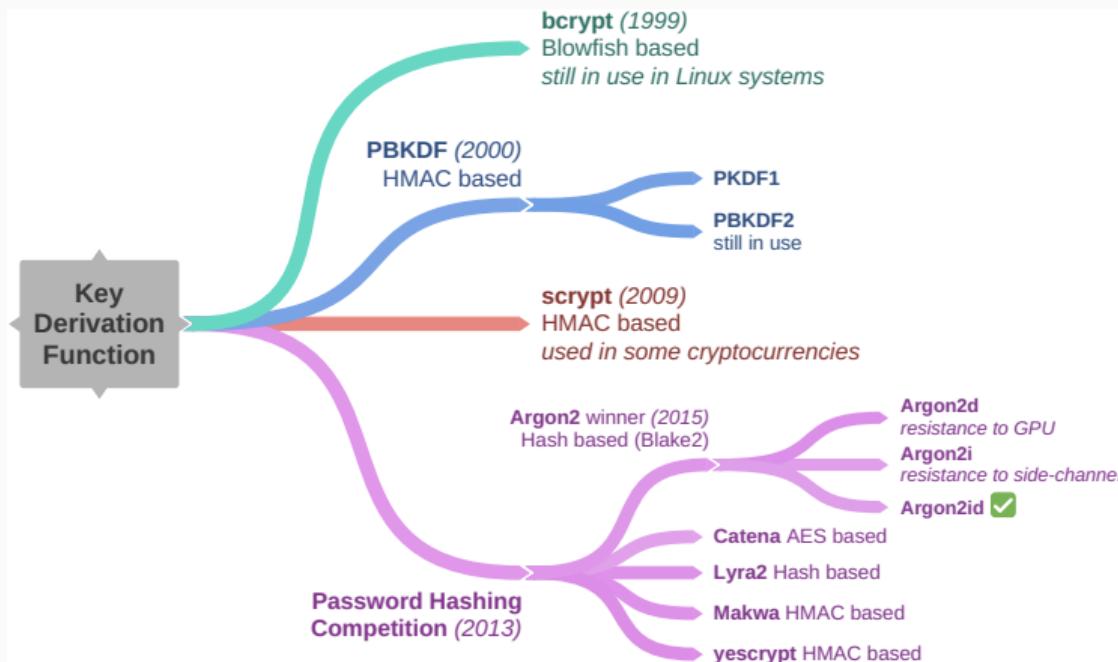
### Key Derivation Functions (KDF)

- designed to securely store passwords without encryption
- like a hash function:
  - the output has fixed length (256, 512, ...) bits
  - non-reversible
- slower than hash functions → harder to brute-force
  - can be parameterized to adjust the required computational effort
  - common parameters are: salt, number of iterations, derived key length

### Key Derivation Functions (KDF)

- designed to securely store passwords without encryption
- like a hash function:
  - the output has fixed length (256, 512, ...) bits
  - non-reversible
- slower than hash functions → harder to brute-force
  - can be parameterized to adjust the required computational effort
  - common parameters are: salt, number of iterations, derived key length
- Disadvantages:
  - it is still needed to keep a secret value on the server side
  - people tend to use weak passwords and reuse them (this is a problem with passwords in general)

## Key Derivation Functions



### Key Derivation Function

#### Argon2

- based on Blake2b hash
- three versions
  - Argon2d – is faster, suitable for cryptocurrencies, is resistant to GPU acceleration;
  - Argon2i – is slower, preferred for password hashing, is resistant to side-channel attacks;
  - Argon2id – is a hybrid version;

## Key Derivation Function

### Argon2

- based on Blake2b hash
- three versions
  - Argon2d – is faster, suitable for cryptocurrencies, is resistant to GPU acceleration;
  - Argon2i – is slower, preferred for password hashing, is resistant to side-channel attacks;
  - Argon2id – is a hybrid version;

Parameters:

- primary:
    - **password** – any length from 0 to  $2^{32} - 1$  bytes;
    - **number of iterations** – used to tune the running time independently of the memory size;
    - **memory required** – it is a memory-hard function;
    - **parallelism** – how many independent computational chains can be run;
  - secondary (default values maybe used):
    - salt – random value, 128 bits is recommended for password hashing;
    - output length – any integer number of bytes from 4 to  $2^{32} - 1$
    - secret value – serves as key if necessary, but by default none is used;
    - associated data – optional arbitrary extra data;
- algorithm type –  $d, i$ , or  $id$ ;

### Key Derivation Function

Argon2 example:

- input values:
  - password = Correct Horse Battery Staple;
  - memory = 1 GB
  - iterations = 4
  - parallelism = 1
- output:

```
$argon2id$v=19$m=1048576,t=4,p=1$1k8ic3+vS0kDB+4J7bcjQg$ok3jcq9yW7Rw7GuhZfJi0IdcoixCZYWgCcHxp8bny6M
```

- \$argon2id – the variant of Argon2 being used
- \$v=19 – the version of Argon2 being used
- \$m=1048576,t=4,p=1 – the memory (m), iterations (t) and parallelism (p)
- \$1k8ic3+vS0kDB+4J7bcjQg – the base64-encoded randomly selected salt
- \$ok3jcq9yW7Rw7GuhZfJi0IdcoixCZYWgCcHxp8bny6M – the base64-encoded derived key

### Key Derivation Function

Argon2 example:

- input values:

- password = Correct Horse Battery Staple;
- memory = 1 GB
- iterations = 4
- parallelism = 1

- output:

```
$argon2id$v=19$m=1048576,t=4,p=1$1k8ic3+vS0kDB+4J7bcjQg$ok3jcq9yW7Rw7GuhZfJi0IdcoixCZYWgCcHxp8bny6M
```

- \$argon2id – the variant of Argon2 being used
- \$v=19 – the version of Argon2 being used
- \$m=1048576,t=4,p=1 – the memory (m), iterations (t) and parallelism (p)
- \$1k8ic3+vS0kDB+4J7bcjQg – the base64-encoded randomly selected salt
- \$ok3jcq9yW7Rw7GuhZfJi0IdcoixCZYWgCcHxp8bny6M – the base64-encoded derived key

- time to calculate on my laptop ≈ 2,5 seconds

### Key Derivation Function

Argon2 guidelines for choosing the parameters:

1. set **memory** limit to the amount of memory you want to reserve for password hashing
2. then, set **iterations** to 3 and measure the time it takes to hash a password
3. if it is too long for your application, reduce **memory**, but keep **iterations** set to 3

### Key Derivation Function

Argon2 guidelines for choosing the parameters:

1. set **memory** limit to the amount of memory you want to reserve for password hashing
2. then, set **iterations** to 3 and measure the time it takes to hash a password
3. if it is too long for your application, reduce **memory**, but keep **iterations** set to 3

Use cases:

- for online use (e.g. login in on a website), a **1 second** computation is likely to be the acceptable maximum
- for interactive use (e.g. a desktop application), a **5 second** pause after having entered a password is acceptable if the password doesn't need to be entered more than once per session
- for non-interactive use and infrequent use (e.g. restoring an encrypted backup), an even slower computation can be an option

### Key Derivation Function

Argon2 guidelines for choosing the parameters:

1. set **memory** limit to the amount of memory you want to reserve for password hashing
2. then, set **iterations** to 3 and measure the time it takes to hash a password
3. if it is too long for your application, reduce **memory**, but keep **iterations** set to 3

Use cases:

- for online use (e.g. login in on a website), a **1 second** computation is likely to be the acceptable maximum
- for interactive use (e.g. a desktop application), a **5 second** pause after having entered a password is acceptable if the password doesn't need to be entered more than once per session
- for non-interactive use and infrequent use (e.g. restoring an encrypted backup), an even slower computation can be an option

#### Note

But the best defense against brute-force password cracking remains using **strong passwords**

### One-Time Password (OTP)

- also known as **one-time pin** or **dynamic password**
- is a password that is valid for **only one** login session or transaction

### One-Time Password (OTP)

- also known as **one-time pin** or **dynamic password**
- is a password that is valid for **only one** login session or transaction
- common as a two-factor authentication → requires access to something a person has
  - such as a small device with the OTP calculator built into it, or a smartcard, or specific cellphone, or software application
- main advantage: not vulnerable to **replay attacks**
- main disadvantage: vulnerable to man-in-the-middle attacks

### One-Time Password (OTP)

- also known as **one-time pin** or **dynamic password**
- is a password that is valid for **only one** login session or transaction
- common as a two-factor authentication → requires access to something a person has
  - such as a small device with the OTP calculator built into it, or a smartcard, or specific cellphone, or software application
- main advantage: not vulnerable to **replay attacks**
- main disadvantage: vulnerable to man-in-the-middle attacks
- there are two types:
  - based only on a seed (or counter)
  - based on time-synchronization

## HMAC-based One-time Password (HOTP)

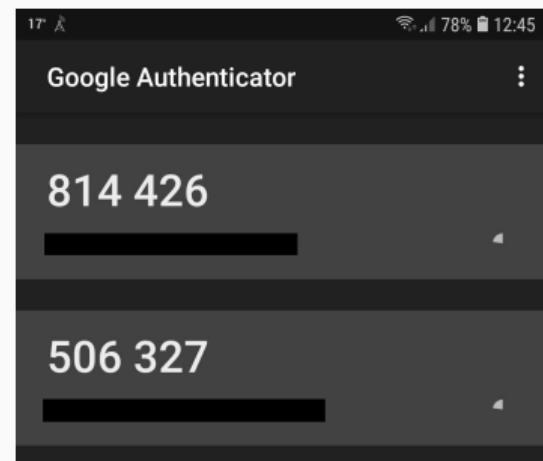
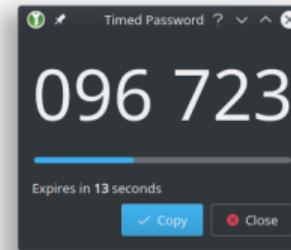
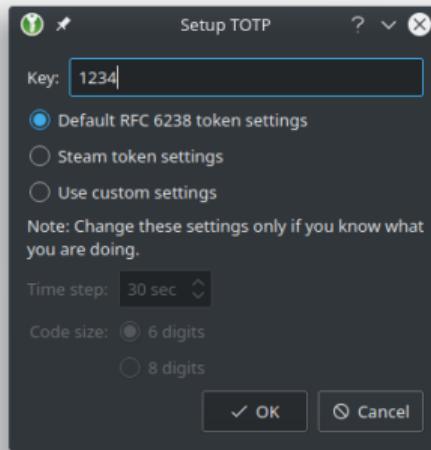
- based on an increasing counter (C) value and a static symmetric key (k)
- specified in RFC4226
  - uses HMAC-SHA-1
  - $\text{HOTP}(k, C) = \text{Truncate}(\text{HMAC\_SHA1}(k, C))$
  - `Truncate()` returns a number with a configurable amount of digits from 6 to 10
- used on SMS tokens sent by banks
  - vulnerable to SIM swap scam and fake cell phone towers

### Time-based One-Time Password (TOTP) RFC 6238

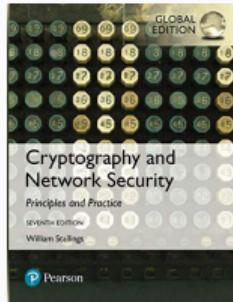
- is an extension of the HMAC-based One-time Password algorithm
- $TOTP = HTOP(k, T)$ , where  $T = (Current\_Unix\_time - T0)/X$  and  $X$  is the time step
- must validate over a range of time (typically 1 minute) due to latency, both network and human, and unsynchronised clocks

## Time-based One-Time Password (TOTP) RFC 6238

- is an extension of the HMAC-based One-time Password algorithm
- $TOTP = HTOP(k, T)$ , where  $T = (Current\_Unix\_time - T_0)/X$  and  $X$  is the time step
- must validate over a range of time (typically 1 minute) due to latency, both network and human, and unsynchronised clocks
- TOTP apps: KeePass, Google Authenticator, LastPass Authenticator, ...



# Questions?



Chapters 12 of  
*William Stallings, Cryptography and Network Security: Principles and Practice, Global Edition, 2016*

# Cifragem Simétrica

Distribuição de Chaves

# Distribuição de Chaves (1)

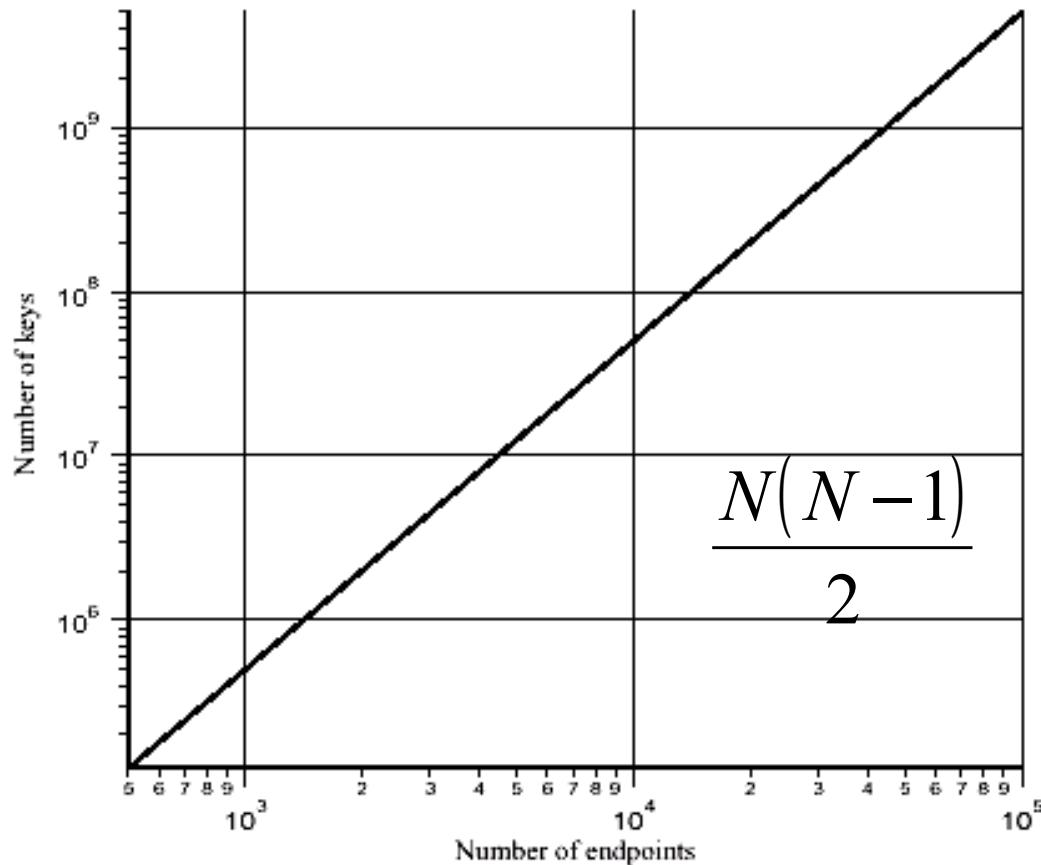
- Alternativas para a troca de chaves entre 2 intervenientes A e B:
  1. A escolhe a chave e entrega-a a B em mão
  2. Um 3º interveniente escolhe a chave e entrega-a em mão a A e a B
  3. Se A e B utilizaram recentemente uma chave comum, um deles pode escolher uma nova chave e enviá-la codificada com a anterior
  4. Se A e B têm uma ligação segura para um 3º interveniente C, C pode entregar uma chave a A e a B através dos canais seguros existentes

# Distribuição de Chaves (2)

1. A escolhe a chave e entrega-a a B em mão
2. Um 3º interveniente escolhe a chave e entrega-a em mão a A e a B
  - Opções razoáveis para a usar na cifragem do canal
  - Mas incompatível na cifragem ponto a ponto
    - Chaves diferentes para cada par comunicante (aplicações, equipamentos, utilizadores, ...)

# Distribuição de Chaves (3)

- Número de chaves para ligações arbitrárias



# Distribuição de Chaves (4)

1. Se A e B utilizaram recentemente uma chave comum, um deles pode escolher uma nova chave e enviá-la codificada com a anterior
  - Pode ser usado na cifragem do canal ou ponto-a-ponto
    - Mas se um atacante conseguir obter uma chave terá acesso a todas as chaves seguintes
    - O problema da distribuição de chaves mantém-se

# Distribuição de Chaves (5)

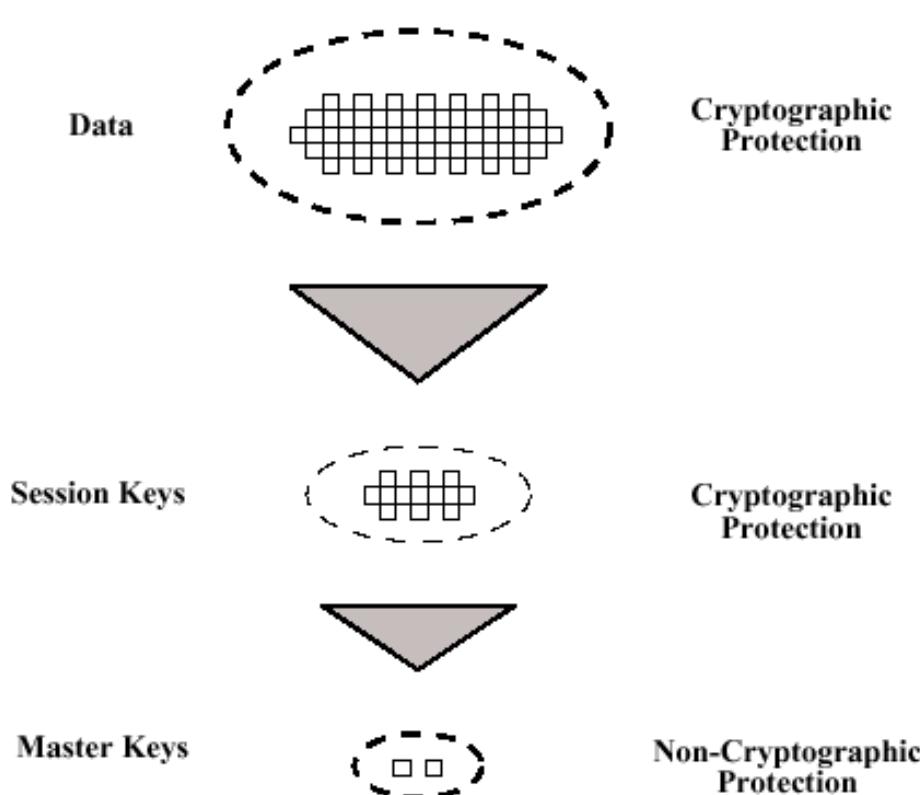
1. Se A e B têm uma ligação segura para um 3º interveniente C, C pode entregar uma chave a A e a B através dos canais seguros existentes
  - Esquema mais adoptado
  - O problema da distribuição das chaves é resolvido através de um centro de distribuição de chaves (3º interveniente)
  - Cada utilizador deve partilhar uma chave única com o centro de distribuição de chaves (KDC)

# KDC (1)

- *Key Distribution Center* (KDC)
- Utilização baseada numa hierarquia de chaves com um mínimo de 2 níveis
  - *Session Key*: chave temporária usada para a comunicação entre sistemas finais. São distribuídas pelo próprio canal cifradas com a *Master Key*
  - *Master Key*: é partilhada entre o KDC e o sistema final (utilizador ou computador). Deve ser distribuída fisicamente e não cifrada.

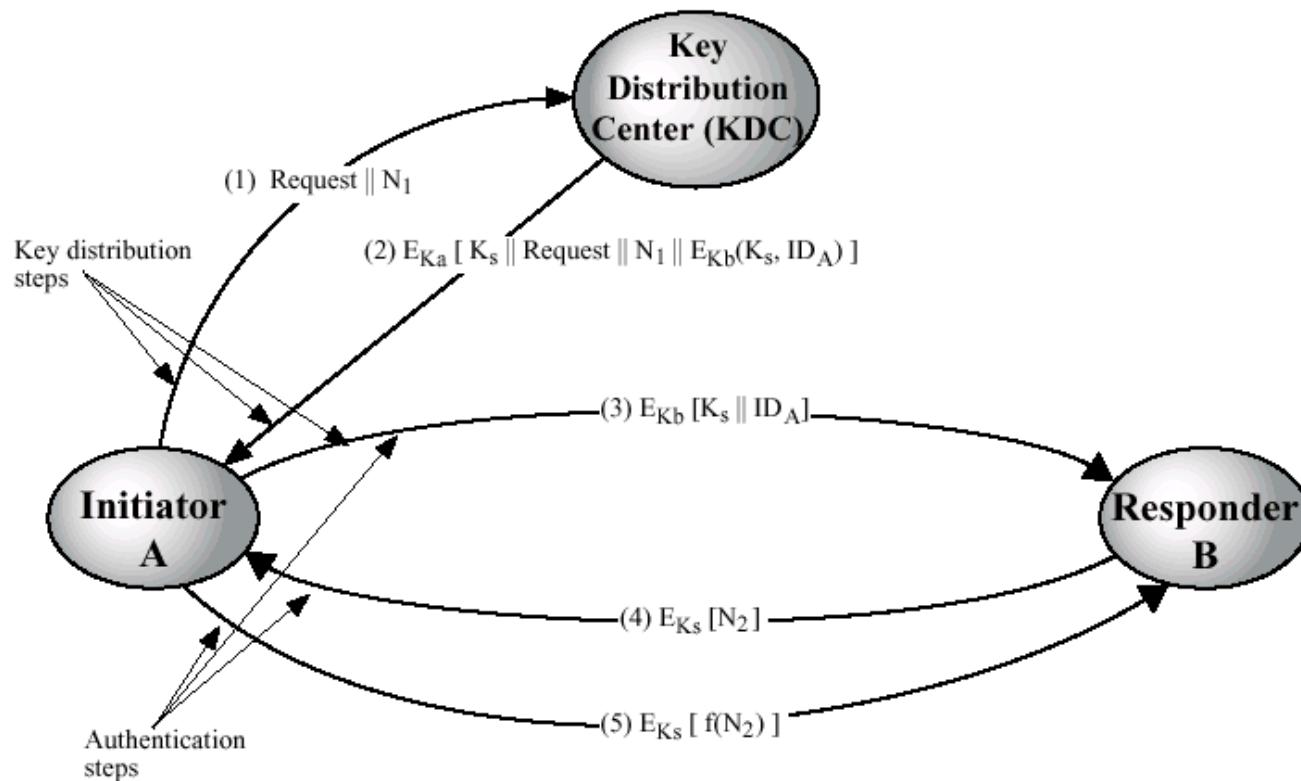
# KDC (2)

- Utilização de uma hierarquia de chaves



# KDC (3)

- Cenário típico de distribuição de chaves



# KDC (4)

- A possui uma *Master Key*  $K_a$
- B possui uma *Master Key*  $K_b$
- Passos para a comunicação:
  1. A pede uma *session key* ao KDC para comunicar com B. O pedido é identificado com  $N_1$  (Um *nounce* que pode ser um nº aleatório, um *timestamp* ou um contador, desde que seja difícil de prever)
  2. O KDC responde com uma mensagem cifrada com  $K_a$ . Essa mensagem contém o *nounce*, uma *session key*  $K_s$  e cifrado com  $K_b$  a  $K_s$  e  $ID_A$  para estabelecer a comunicação com B e provar a identidade de A

# KDC (5)

1. A guarda a  $K_s$  e envia a B a informação originada pelo KDC:  $E_{K_0}[K_s, ID_A]$ . Neste momento a chave de sessão  $K_s$  já foi distribuída a A e a B de forma segura.
2. Usando  $K_s$ , B envia um  $N_2$  a A
3. Usando  $K_s$ , A envia um valor  $f(N_2)$  a B (por exemplo incrementar um valor a  $N_2$ )

# KDC (6)

- Controle hierárquico de chaves
  - As funções de distribuição de chaves podem ser distribuídas por diversos KDC
  - Um KDC local pode ser responsável por uma LAN ou por 1 edifício
  - Quando se pretendem estabelecer comunicações não locais, os KDCs locais podem comunicar com um KDC global
  - Minimização do esforço de distribuição de chaves mestras

# KDC (7)

- Duração das chaves de sessão
  - Compromisso entre Segurança (chaves frequentes) e Desempenho (chaves pouco frequentes)
  - Para protocolos *connection-oriented*
    - A escolha mais óbvia será a duração da ligação
    - Se a ligação for muito longa será prudente que a chave seja mudada periodicamente
  - Para protocolos *connection-less*
    - Não há início ou fim da comunicação explícitos
    - Utilização da mesma chave durante um determinado período temporal ou durante um determinado número de transacções

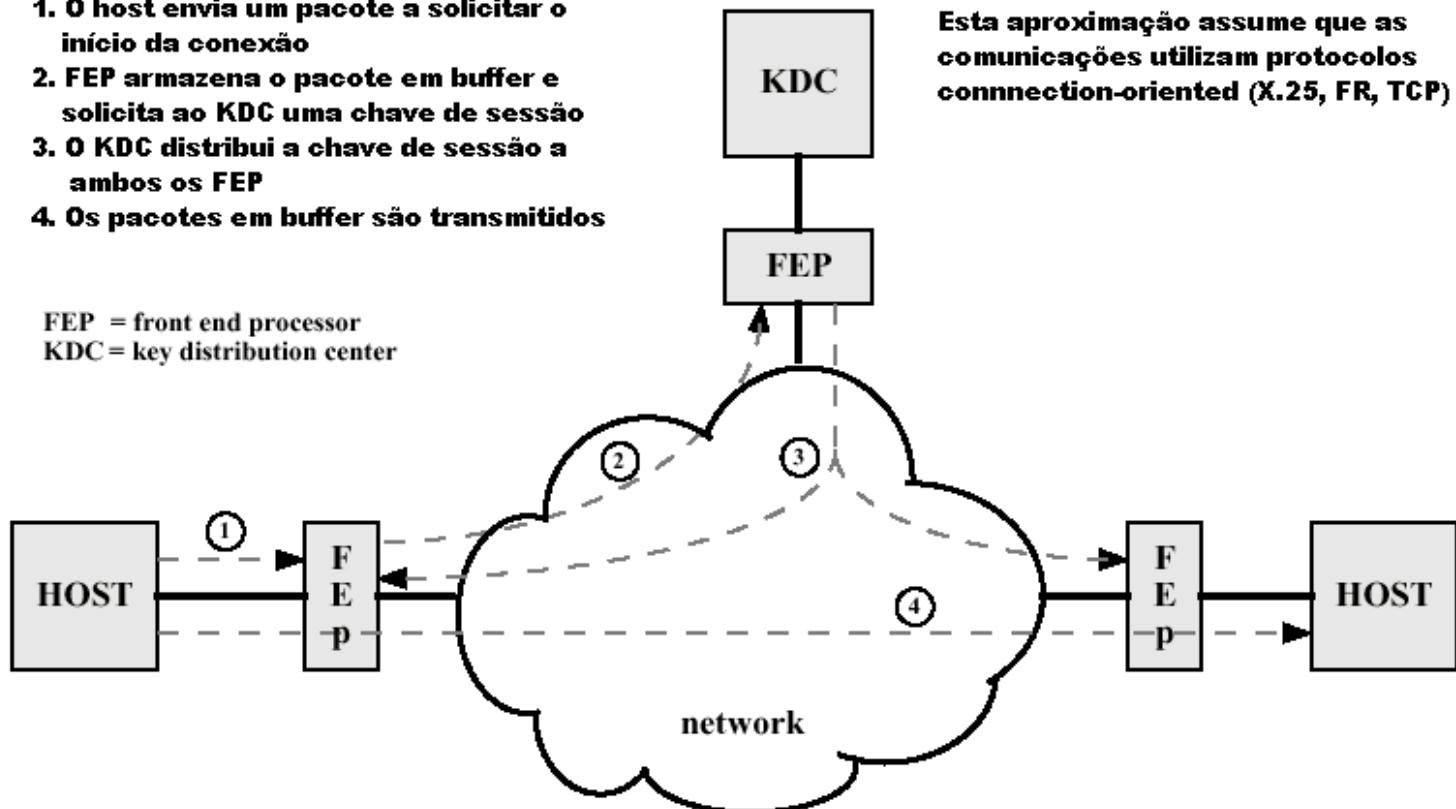
# KDC (8)

- Esquema de controlo de chaves transparente

1. O host envia um pacote a solicitar o início da conexão
2. FEP armazena o pacote em buffer e solicita ao KDC uma chave de sessão
3. O KDC distribui a chave de sessão a ambos os FEP
4. Os pacotes em buffer são transmitidos

FEP = front end processor  
KDC = key distribution center

Esta aproximação assume que as comunicações utilizam protocolos connection-oriented (X.25, FR, TCP)

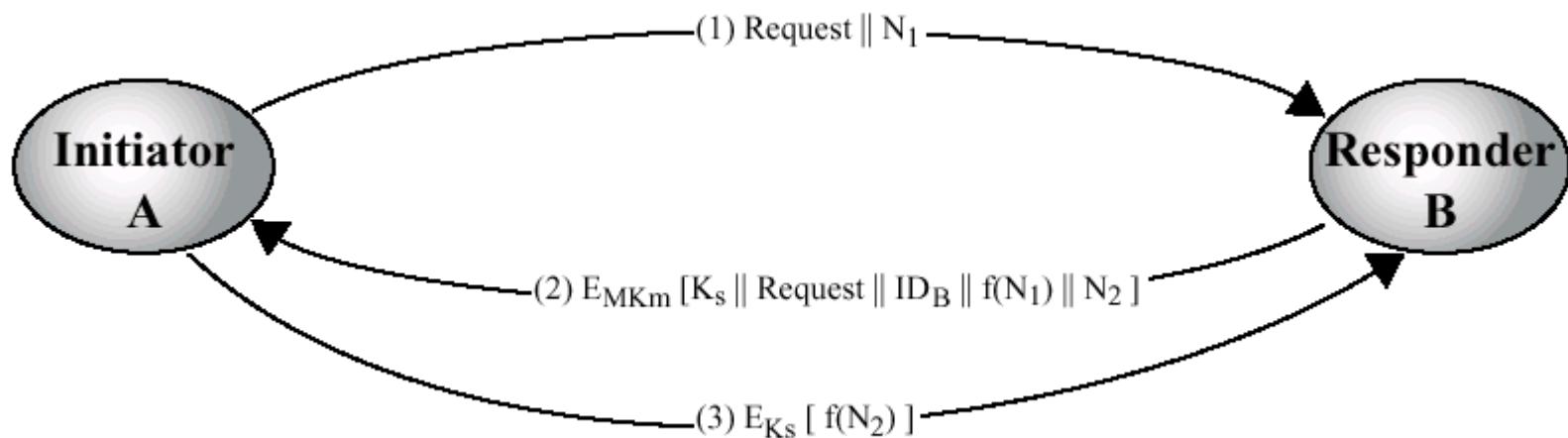


# KDC (9)

- Controle de chaves descentralizado
  - Cada interveniente possui uma *master key* para cada potencial receptor
  - O número de *master keys* é  $N(N-1)/2$ , o que limita este esquema a um contexto local

# KDC (10)

- Controle de chaves descentralizado



# Criptografia Assimétrica

Distribuição de Chaves

# Distribuição de Chaves Públicas (1)

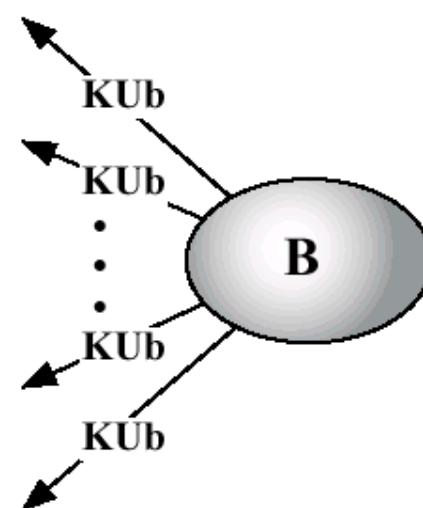
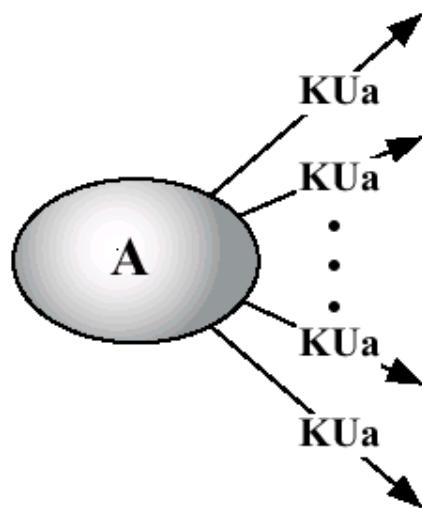
- Técnicas de distribuição de chaves públicas:
  - Anúncio público
  - Disponibilização de uma directoria pública
  - Autoridade de chave pública
  - Certificados de chave pública

# Distribuição de Chaves Públicas (2)

- Anúncio público
  - Chave-pública de conhecimento público
  - Utilização de um algoritmo como o RSA
  - Cada utilizador pode enviar a sua chave pública para outro participante ou realizar a sua difusão
  - Exemplo
    - PGP em que muitos utilizadores difundem a sua chave pública em anexo a mensagens
  - Um anúncio pode ser facilmente forjado

# Distribuição de Chaves Públicas (3)

- Anúncio público (cont.)

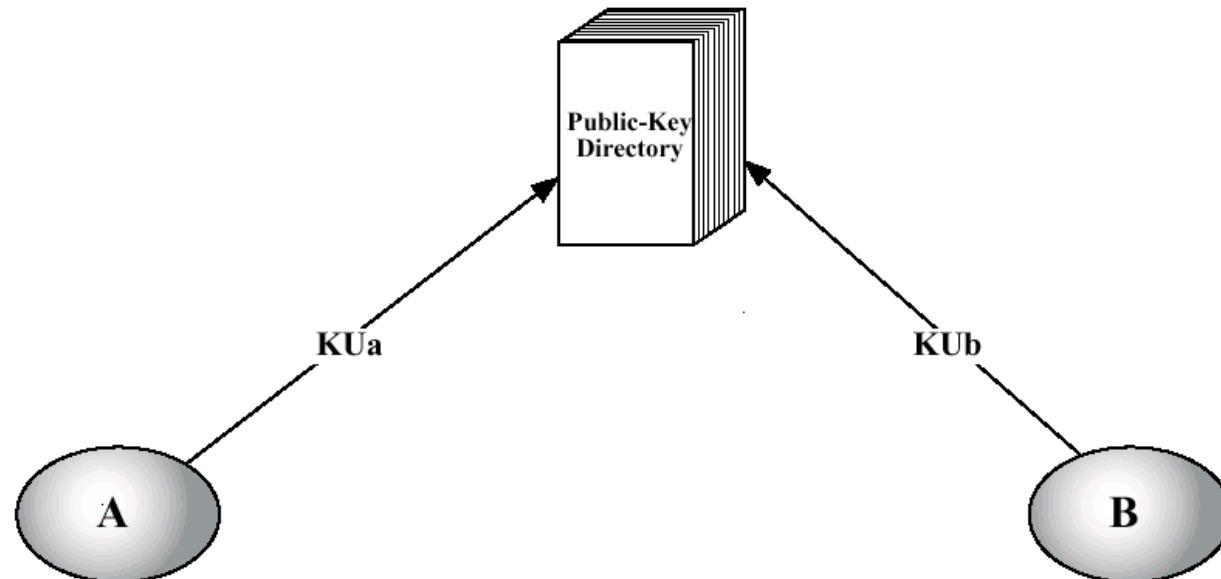


# Distribuição de Chaves Públicas (4)

- Disponibilização de uma directoria pública
  - Mais seguro que o anúncio público
  - Manutenção da responsabilidade de uma terceira entidade
  - Características:
    1. Uma entrada por cada participante com o nome e chave pública
    2. Cada participante regista a sua chave pessoalmente ou através de um canal seguro e autenticado
    3. Os participantes podem substituir a sua chave pública em qualquer altura
    4. A lista completa de chaves deve ser publicada periodicamente pela entidade responsável pela directoria
    5. Acesso à directoria por meios electrónicos

# Distribuição de Chaves Públicas (5)

- Disponibilização de uma directoria pública (cont.)
  - Se alguém conseguir obter a chave privada da entidade responsável pode forjar as chaves dos participantes e divulgá-las como válidas

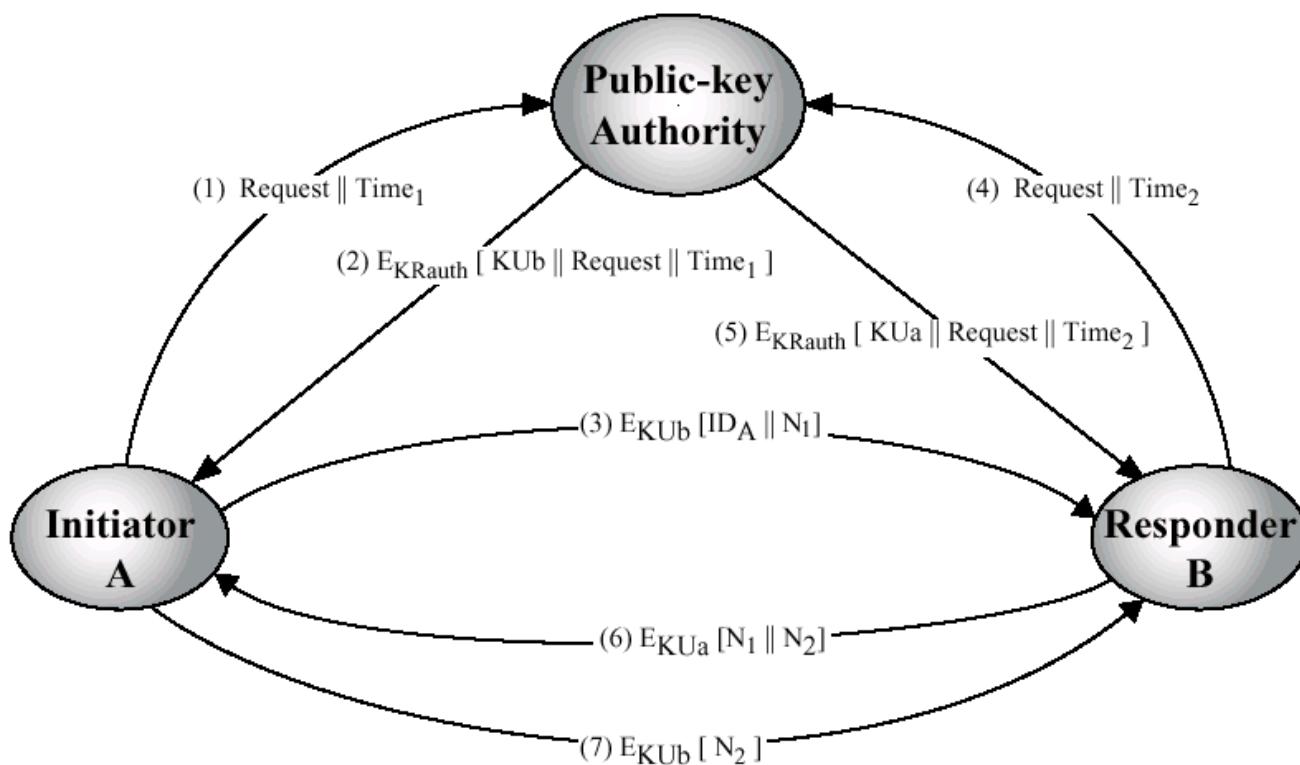


# Distribuição de Chaves Públicas (6)

- Autoridade de chave pública (PKA)
  1.  $A$  envia envia um pedido, com *timestamp*, à  $PKA$  para obter a chave pública de  $B$
  2. A  $PKA$  responde com uma mensagem cifrada com a sua chave privada  $KR_{auth}$ . A mensagem contém:
    - $KU_b$ , o pedido original (para verificação) e o *timestamp*.
  3.  $A$  envia uma mensagem a  $B$  cifrada com  $KU_b$  contendo  $ID_A$  e  $N_1$
  4. Igual ao passo 1, mas para  $B$
  5. Igual ao passo 2, mas para  $B$  onde obtém  $KU_a$
  6.  $B$  envia uma mensagem a  $A$  cifrada com  $KU_a$  contendo  $N_1$  e  $N_2$
  7.  $A$  envia uma mensagem a  $B$  cifrada com  $KU_b$  contendo  $N_2$

# Distribuição de Chaves Públicas (7)

- Autoridade de chave pública (cont.)



# Distribuição de Chaves Públicas (8)

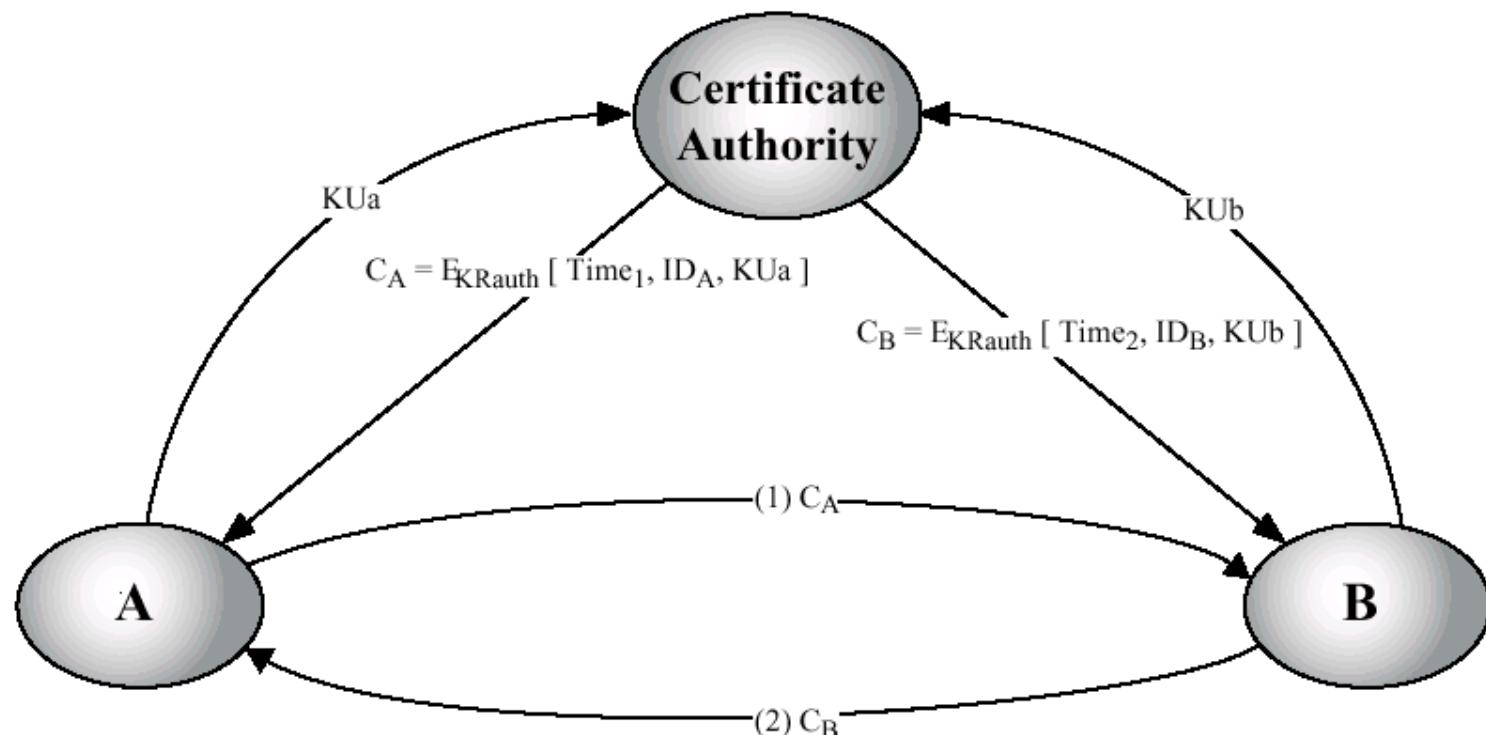
- Autoridade de chave pública (cont.)
  - Apesar de ser necessário 7 mensagens, as primeiras 4 ocorreram poucas vezes porque os intervenientes podem armazenar as chaves públicas
  - A *PKA* pode ser um ponto de engarrafamento
  - Se alguém conseguir obter a chave privada da *PKA*, esta fica comprometida.

# Distribuição de Chaves Públicas (9)

- Certificados de chave pública (CA)
  1. Cada participante pode ler um certificado para determinar o nome e a chave pública do dono do certificado
  2. Cada participante pode verificar que um certificado foi originado na CA e não foi alterado
  3. Só a CA pode criar e actualizar certificados
- Uma chave privada comprometida é semelhante à perca de um cartão de crédito: o dono cancela o cartão, mas está em risco enquanto todas as comunicações não tiverem conhecimento do seu cancelamento
- O protocolo X.509 usa este tipo de distribuição de chaves

# Distribuição de Chaves Públicas (10)

- Certificados de chave pública (cont.)



# Conclusões

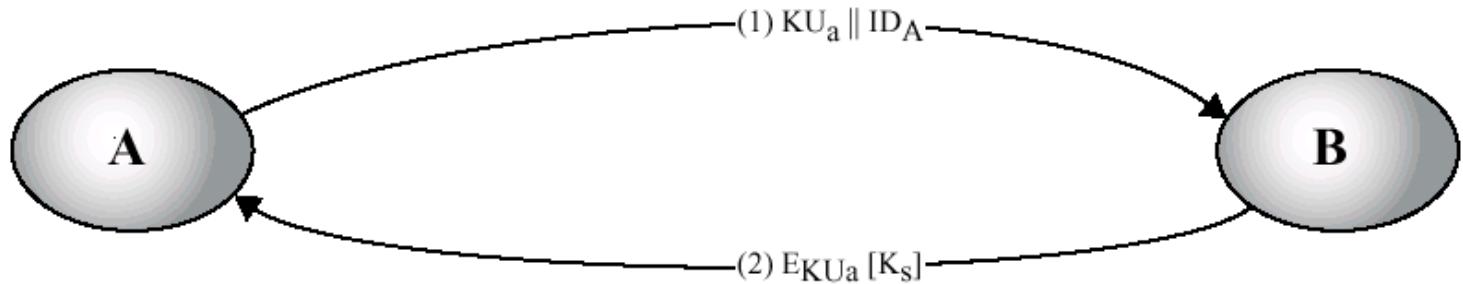
- Depois de distribuídas ou disponibilizadas as chaves, podem realizar-se comunicações seguras resistentes a ataques
- A utilização de técnicas de cifragem assimétrica são bastante pesadas, tornando pouco comum a sua utilização exclusiva
- As técnicas de cifragem assimétrica são vistas como um bom veículo de distribuição de chaves secretas utilizadas na criptografia convencional

# Distribuição de Chaves Secretas (1)

- Técnicas de distribuição de chaves secretas:
  - Distribuição simples
  - Distribuição com confidencialidade e autenticidade
  - Esquema híbrido

# Distribuição de Chaves Secretas (2)

- Distribuição simples
  1.  $A$  gera um par de chaves  $\{KU_a, KR_a\}$  e envia uma mensagem a  $B$  com  $KU_a$  e um identificador  $ID_A$
  2.  $B$  gera uma chave secreta  $K_s$  e envia a  $A$  cifrada com  $KU_a$ 
    - Este esquema é vulnerável a ataques *Man-in-the-middle*

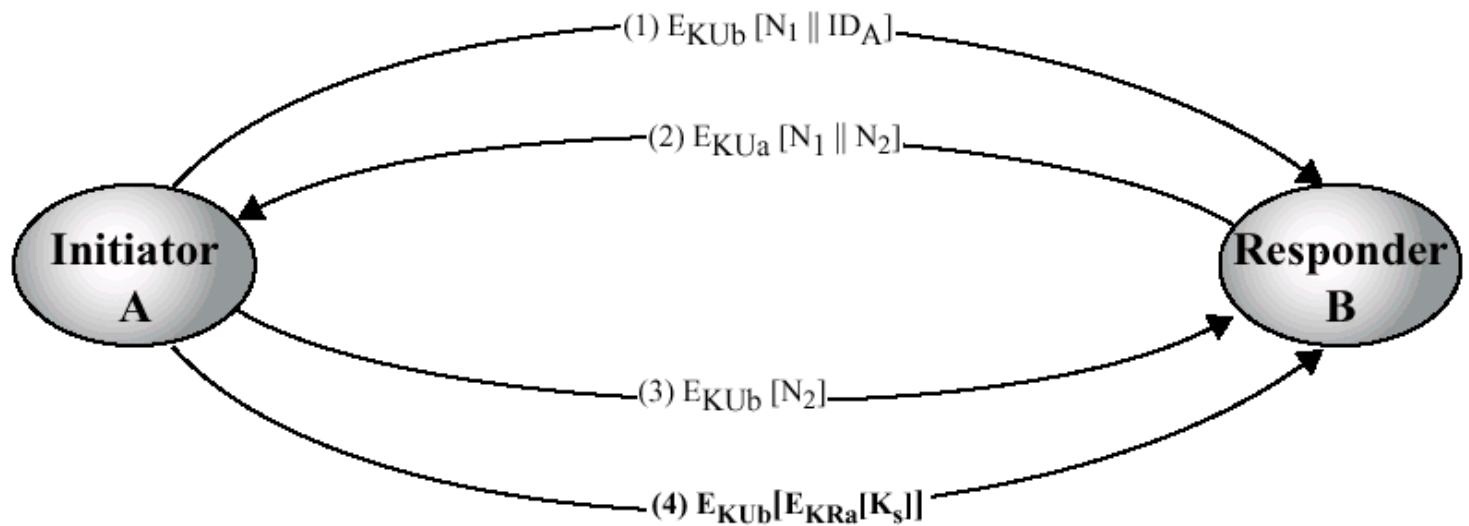


# Distribuição de Chaves Secretas (3)

- Distribuição com confidencialidade e autenticidade
  - Troca de chaves públicas através de um dos esquemas anteriores
    1.  $A$  cifra uma mensagem para  $B$  com  $KU_b$  contendo  $ID_A$  e  $N_1$
    2.  $B$  cifra uma mensagem para  $A$  com  $KU_a$  contendo  $N_1$  e  $N_2$
    3.  $A$  cifra uma mensagem para  $B$  com  $KU_b$  contendo  $N_2$
    4.  $A$  escolhe  $K_s$  e envia  $M = E_{KUb}[E_{KRa}[K_s]]$

# Distribuição de Chaves Secretas (4)

- Distribuição com confidencialidade e autenticidade (cont.)



# Distribuição de Chaves Secretas (5)

- Esquema Híbrido
  - Utilizado em mainframes da IBM
  - Mantém a utilização de 1 KDC
  - Partilha de chaves mestras com cifragem assimétrica
  - Melhorias alcançadas
    - Desempenho – em aplicações orientadas às transacções com trocas frequentes de chaves a distribuição de chaves através de cifras assimétricas torna-se pesada
    - Compatibilidade – garante a compatibilidade com sistemas de KDC existentes através de alguns ajustes nas aplicações



# *Public Key Infrastructure* (PKI)



# O que é o PKI?

***Conjunto de hardware, software, pessoas, políticas e procedimentos necessários para a criação, gestão armazenamento, distribuição e revogação de chaves públicas***

- ◆ Principal questão associada ao desenvolvimento destes modelos: Confiança



**Certificados Digitais**



# Características (1)

- ◆ É baseado na cifragem assimétrica
- ◆ Infraestrutura tão segura quanto o mecanismo utilizado na distribuição certificada de informação
- ◆ Integra 3 componentes principais
  - ❖ Certificados Digitais/Assinaturas
  - ❖ Criptografia de chave pública
  - ❖ Certificate Authorities (CA)



# Características (2)

- ◆ Protecção da informação de várias formas
  - Autenticação da identidade
  - Verificação da integridade
  - Privacidade
  - Autorização de acessos
  - Autorização de transações
  - Não repúdio



# Certificados Digitais

- ◆ Associam entidades a chaves públicas
- ◆ Contêm informação detalhada sobre a entidade
- ◆ A validade dum certificado é assegurada pela presença da assinatura digital de uma terceira entidade
- ◆ Vamos ver:
  - Certificados X.509



# Certificados X.509



# Descrição (1)

- ◆ Criados pelo ITU-T e pela OSI no âmbito do Serviço de Directoria (recomendações da série X.500 que definem os serviços de directoria)
- ◆ Utilização do Serviço de Directoria como repositório de chaves públicas das entidades
- ◆ O X.509 é importante porque a sua estrutura é usada em vários contextos:
  - S/MIME, IPsec, SSL, etc



# Descrição (2)

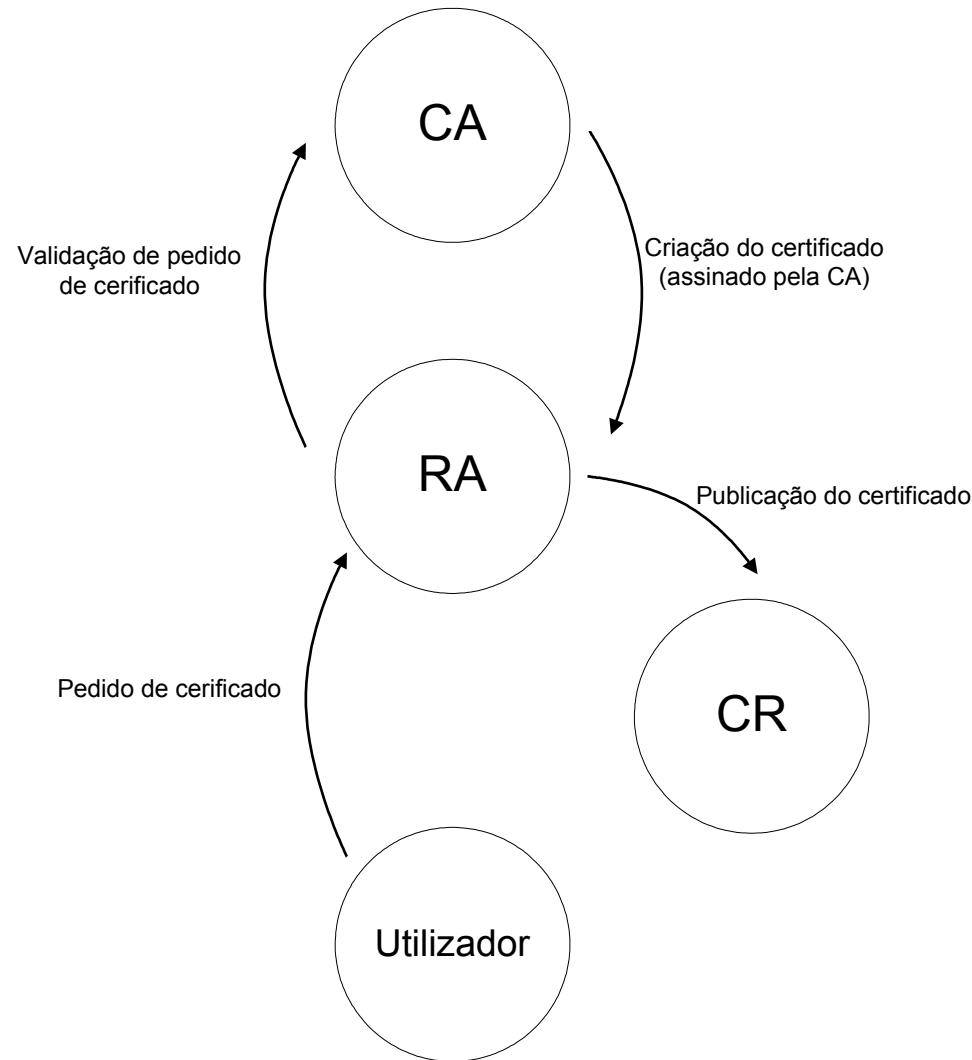
- ◆ Foi criada a norma PKIX (PKI X.509)
  - [www.ietf.org/html.charters/pkix-charter.html](http://www.ietf.org/html.charters/pkix-charter.html)
- ◆ Existem várias RFC sobre PKI, das quais se destacam:
  - **RFC 2510** *PKIX Certificate Management Protocols*
  - **RFC 2559, RFC 2585, RFC 2560** *Operational protocols*
  - **RFC 3647** *Certificate Policy and Certification Practices Framework* (Actualiza a RFC 2527)



# Modelo PKIX (1)

- ◆ Componentes
  - **CA** (*Certification Authorities*) – são as entidades que emitem os certificados digitais.
  - **RA** (*Registration Authorities*) – entidades que ajudam as CA's na identificação dos candidatos ou requerentes de criação, renovação ou revogação de certificados digitais
  - **CR** (*Certificate Repositories*) – entidades que disponibilizam um espaço para publicação, armazenamento e acesso aos certificados digitais e outra informação relacionada com a infra-estrutura de chave pública
  - **Titulares** (*Subscribers* ou *Users*) – são as pessoas singulares ou organizações colectivas detentoras de um par de chaves cuja chave pública está certificada por uma autoridade certificadora e inclusa no respectivo certificado digital
  - **Partes Interessadas** (*Relying Partys, RP*) – entidades receptoras da chave pública e/ou assinatura digital de titulares, que confiam na autenticidade da chave pública certificada.
  - **Entidade Credenciadora** – Não faz parte da PKI em si mas é quem autoriza e reconhece o funcionamento de uma CA do ponto de vista legal. Em Portugal, o organismo que exerce essa função é o Instituto das Tecnologias da Informação na Justiça (ITIJ)

# Modelo PKIX (2)





# Documentação de uma PKIX

- ◆ Política de certificados (*Certificate Policy, CP*): requisitos gerais que os intervenientes na PKI devem satisfazer. Descreve os usos que os certificados podem ter. Por razões de segurança, os requisitos podem ser mantidos confidenciais.
- ◆ Declaração das práticas de certificação (*Certification Practice Statement, CPS*): descreve as práticas assumidas pela CA para a exerção das suas funções. Relata os procedimentos de uma perspectiva legal, técnica e de negócio e é mais detalhada do que a CP. A CPS pode também ser a base para a certificação cruzada entre duas CA's.
- ◆ Acordo de utilizador (*Subscriber Agreement, SA*): é o acordo estabelecido entre o titular de um certificado e uma CA ou RA. Este documento descreve as responsabilidades e condições de utilização dos certificados que o seu titular deve respeitar.
- ◆ Acordo das Partes Interessadas (*Relying Party Agreement, RPA*) – é o acordo estabelecido entre determinada entidade interessada em confiar num certificado e a CA que o emitiu ou a RA que aprovou a sua criação. Normalmente determina que a parte interessada deve verificar o estado do certificado antes de confiar nele.



# CA (1)

- ◆ Serviços disponibilizados:
  - Gestão de chaves: inclui geração de pares de chaves, manipulação segura das chaves privadas da CA e distribuição das chaves públicas da CA
  - Criar um meio para que os requerentes de certificado possam submeter a sua requisição de certificado (por exemplo, uma página web)
  - Identificação e autenticação dos indivíduos ou entidades que requerem à CA a criação de certificados, renovação de certificados ou alteração de chaves
  - Aprovação ou rejeição de requisições de certificados submetidas à CA
  - Emissão de certificados referentes às requisições aprovadas
  - Publicação dos certificados num repositório para posterior usufruto das entidades interessadas
  - Revogação de certificados, quer a pedido do titular, quer por iniciativa própria
  - Publicação da lista de certificados revogados



# CA (2)

- ◆ Cada certificado contém a chave pública da entidade
- ◆ A chave pública da entidade é assinada (validada) com a chave privada da entidade emissora (Autoridade de Certificação)
- ◆ A verificação da identidade faz-se com a chave pública da Autoridade de Certificação



# CA (3)

- ◆ Políticas que deve estabelecer
  - Que tipos de certificados existem
  - Para um determinado tipo de certificado, quais são os passos necessários para a verificação
  - Armazenamento dos certificados e das chaves privadas
  - Expiração dos certificados
  - Registo dos certificados



# RA (1)

- ◆ Serviços disponibilizados
  - Criar um meio para que os requerentes de certificado possam submeter a sua requisição de certificado (por exemplo, uma página web)
  - Identificação e autenticação dos indivíduos ou entidades que requerem à CA a criação de certificados, renovação de certificados ou alteração de chaves
  - Aprovação ou rejeição de requisições de certificados submetidas à CA
  - Revogação de certificados, quer a pedido do titular, quer por iniciativa própria



# Ciclo de vida do certificado (1)

- ◆ Requisição
  - Pelo titular, ou por um procurador
  - Certificado pessoal, ou para um servidor, ...
- ◆ Aceitação
  - O titular deve verificar se existe alguma incorrecção
- ◆ Publicação
  - O certificado é assinado com a chave privada da CA
  - É colocado num repositório acessível a todas as partes interessadas
  - Envio do certificado ao titular (por e-mail, smart card, ...)



# Ciclo de vida do certificado (2)

## ♦ Revogação

- Por iniciativa do titular (perca da chave privada, a chave secreta está comprometida, etc)
- Por iniciativa da CA (incumprimento do acordo de titulares)
- adição do certificado à lista de certificados revogados
- O utilizador já não é certificado pela CA
- O certificado emitido pela CA está comprometido
- Criação de uma *Certificate Revocation List* (CRL) que deve ser mantida na directoria
- Utilização do *Online Certificate Status Protocol* (OCSP)

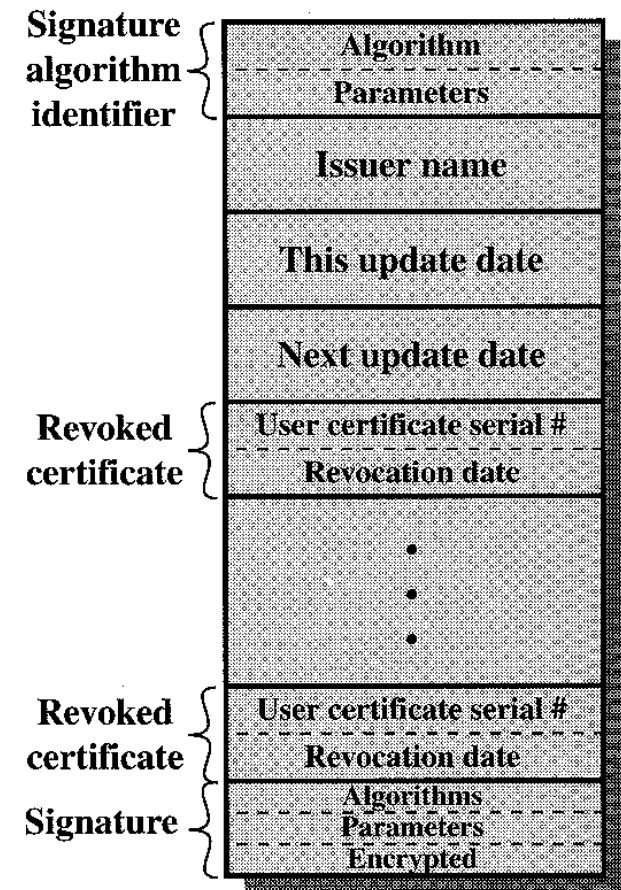


# Ciclo de vida do certificado (3)

## ♦ Revogação (cont.)

### — CRL

- Cada entrada contém o nº de série (que identifica o certificado dentro da CA) e a data de revogação
- Quem recebe um certificado tem que verificar se pertence à CRL





# Ciclo de vida do certificado (4)

## ♦ Revogação (cont.)

- Dois modelos para a entrega de CRLs
  - **Polling**: a CRL é pedida pelo utilizador quando este precisa da chave de um certificado digital
    - Problema: Variação de tempo entre a revogação e a publicação
  - **Pushing**: cada CRL é entregue pelo CA ao utilizador assim que ocorre uma nova revogação
    - Problema: armazenamento de CRLs mesmo que não sejam necessárias e perigo de intercepção/remoção



# Ciclo de vida do certificado (5)

## ◆ Revogação (cont.)

### – Limitações

- Os CRLs são emitidos/actualizados de forma periódica de acordo com a política da CA
- Possibilidade de se aceitar um certificado digital revogado, mas ainda não publicado na CRL



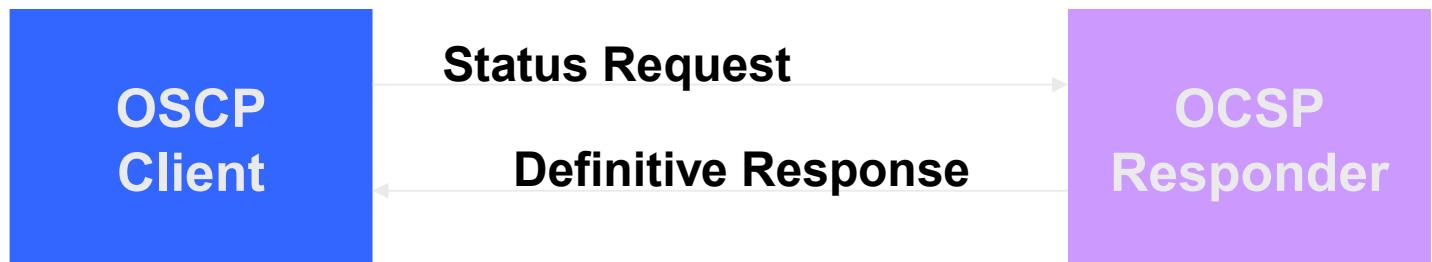
**Solução: OCSP**



# Ciclo de vida do certificado (6)

- ◆ Revogação (cont.)

- Permite às aplicações determinarem a revogação de um certificado de uma forma mais expedita
- Modelo cliente-servidor





# Ciclo de vida do certificado (7)

## ◆ Suspensão

- Certificado inválido por um determinado período (férias, greves, inactividade, etc)
- Serviço que não é disponibilizado por todas as PKI
- Pode ser usado para publicar um certificado que ainda não foi aceite pelo titular

## ◆ Renovação

- Os certificados tem uma validade a partir da qual deixam de ser válidos, por isso necessitam de ser renovados
- O certificado é assinado pela CA com nova data limite



# Ciclo de vida do certificado (8)

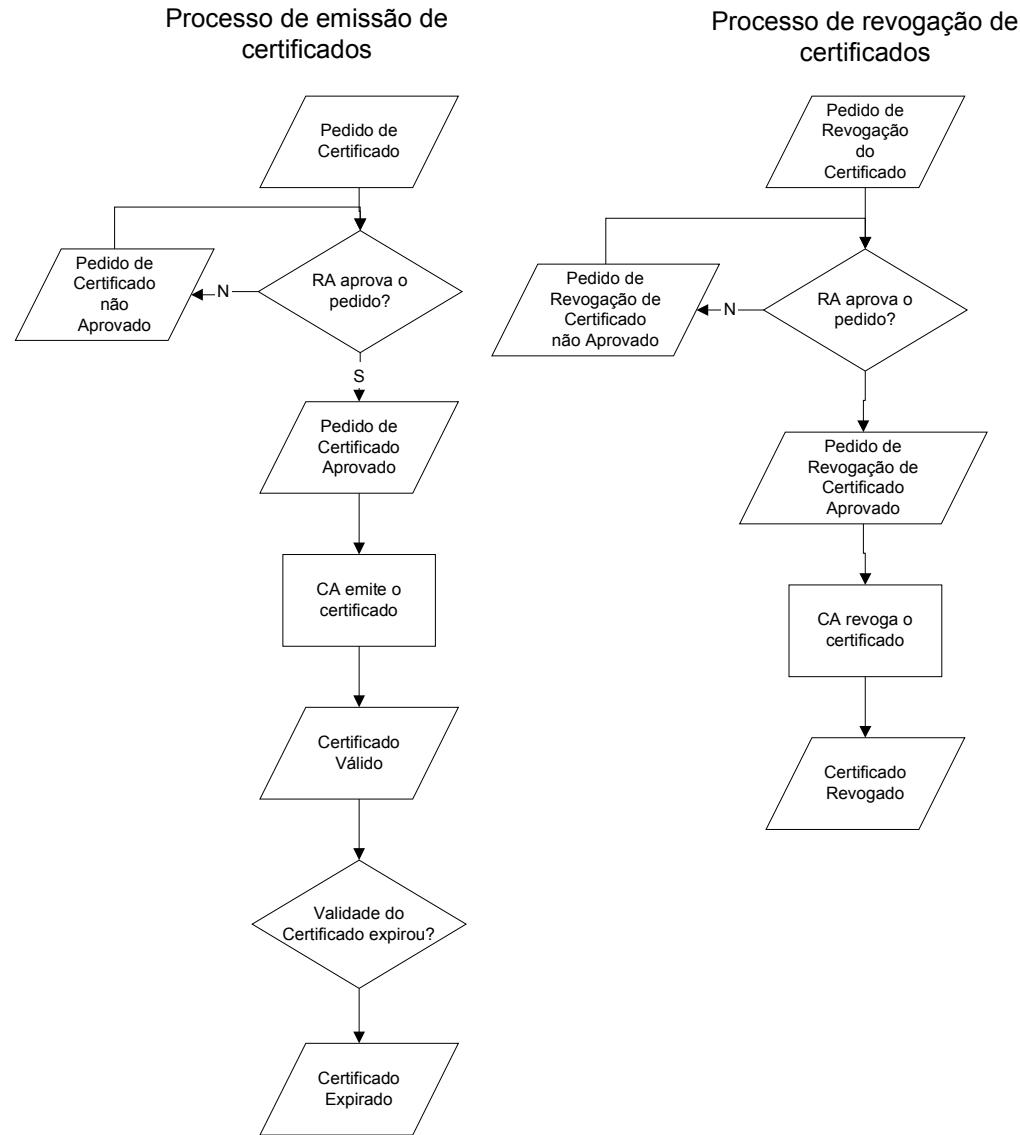
## ◆ Alteração

- Pode ser necessário alterar um certificado devido a: alteração do nome legal do titular, alteração de informação regional, alteração do e-mail, alteração das chaves, etc
- É emitido um novo certificado com a informação actualizada e o certificado antigo é revogado

## ◆ Verificação do estado

- Lista de certificados revogados (CRL - *Certificate Revocation Lists*)
- Verificação on-line (*On-line revocation/status checking*, e.g. OCSP – On-line Certificate Status Protocol)

# Ciclo de vida do certificado (9)





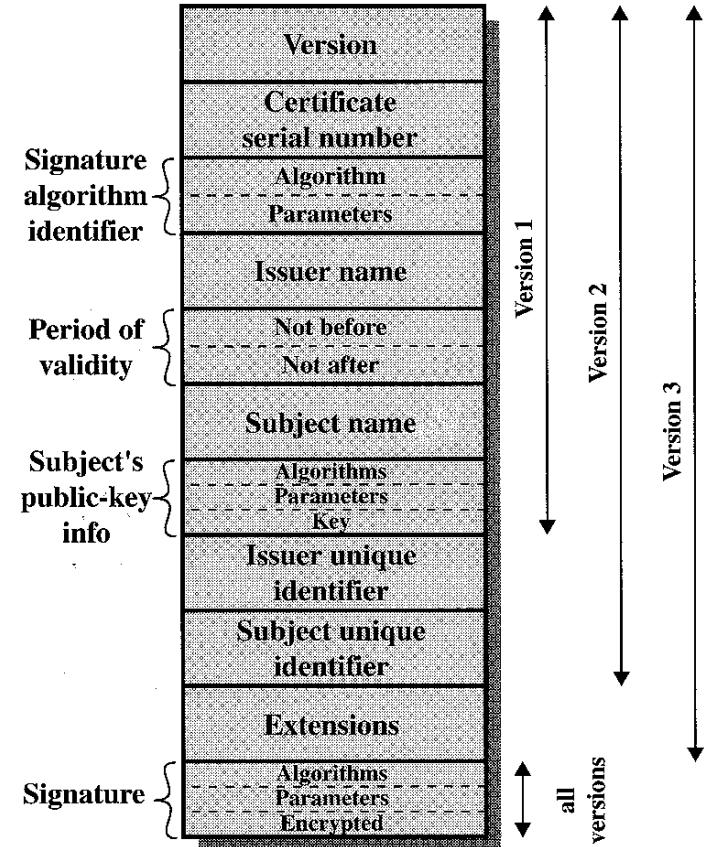
# Formato do Certificado (1)

- ◊ Versão: Determina qual a versão da norma X.509 que o certificado respeita.
- ◊ Número de série: É um identificador único para cada certificado atribuído pela entidade que o emitiu e que permite distinguir inequivocamente todos os certificados emitidos. Este número é utilizado, por exemplo, quando o certificado é revogado este número é colocado numa lista de certificados revogados.
- ◊ Identificador do algoritmo de assinatura: Determina qual o algoritmo usado pela CA para assinar o certificado.
- ◊ Emissor do certificado: Indica o nome da entidade que emitiu o certificado.
- ◊ Período de validade: Define o tempo de vida do certificado.
- ◊ Nome do titular: Indica o nome da entidade (por exemplo pessoa) a quem foi emitido certificado. Este campo respeita a norma X.500.
- ◊ Chave pública do titular: Aqui vai a chave pública do titular do certificado.
- ◊ Algoritmo da chave pública: Indica qual o algoritmo usado para criar o par de chaves do titular.
- ◊ Assinatura: contém um *hash* de todos os campos que depois é assinado com a chave provada da CA

# Formato do Certificado (2)

- ◆ Até agora foram publicadas 3 versões
  - A 1<sup>a</sup> versão foi publicada em 1988
  - Em 1993 foi efectuada uma revisão
  - A mais recente é X.509 V3, foi publicada em 1995 e revista em 2000
- ◆ Notação:

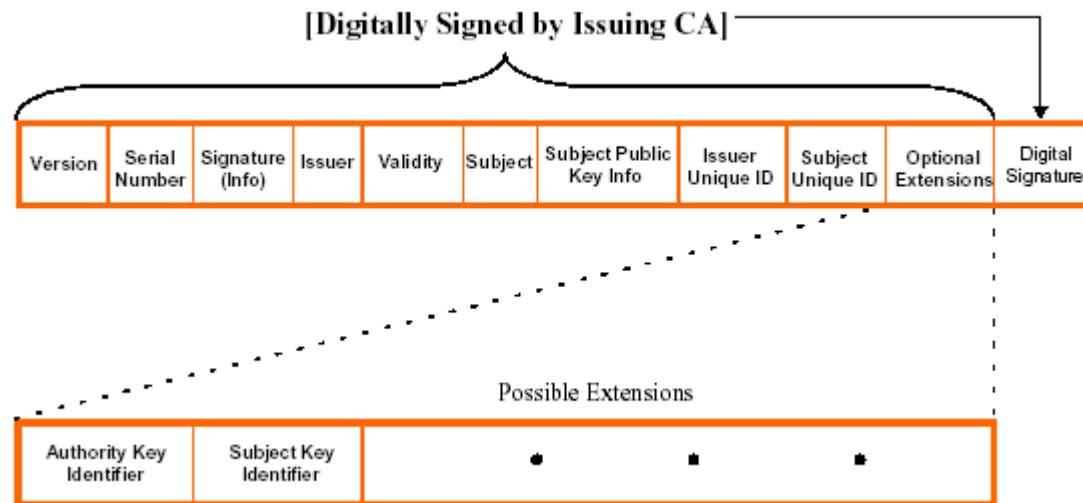
$$\text{CA} << \text{A} >> = \text{CA} \{ \text{V}, \text{SN}, \text{AI}, \text{CA}, \text{T}_\text{A}, \text{A}, \text{Ap} \}$$





# Formato do Certificado (3)

- ◆ Extensões do certificado X.509
  - permitem adicionar mais atributos ao certificado digital





# Formato do Certificado (4)

- ◆ Exemplo dos campos de um certificado

0	VERSION
1234567891011121314	SERIAL NUMBER
RSA+SHA1, 2048	SIGNATURE ALGORITHM
C=US, S=VA, O=GMU, OU=ISE	ISSUER NAME
1/1/03-3/3/03	VALIDITY
C=US, S=VA, O=GMU, OU=ISE, CN=FPP	SUBJECT
RSA, 1024, xxxxxxxxxxxxxxxxxxxxxxxxx	SUBJECT PUBLIC KEY INFO
SIGNATURE	SIGNATURE



# X.509 Versão 3 (1)

- ◊ Resolver limitações das versões anteriores
  - Campo *Subject* muito pequeno não permitia a inclusão de detalhes necessários (por exemplo *links* ou endereços de *email*)
  - Não há informação sobre a política de segurança (por exemplo para usar no IPSec)
  - Limitar os efeitos de um CA malicioso através de restrições.
  - Identificação das diferentes chaves utilizadas pelo mesmo utilizador em tempos distintos, para gerir o ciclo de vida dos certificados
- ◊ Utilização de extensões opcionais
  - Identificador da extensão
  - Indicador crítico (indica se pode ser ignorado ou não)
  - Valor
  - Extensões podem ser de três tipos:
    - Chave e informação sobre a política
    - Atributos do Issuer
    - Restrições do caminho de certificação



# X.509 Versão 3 (2)

- ◆ Chave e informação sobre a política
  - **Authority Key Identifier**: indica a chave pública a usar para verificar o certificado
  - **Subject Key Identifier**: identifica a chave pública certificada
  - **Key Usage**: indica restrições à utilização do certificado (assinatura digital, não repudiação, cifragem, etc)
  - **Private-key Usage Period**: a validade de uso da chave privada pode ser diferente da validade da chave pública (numa assinatura digital a chave privada normalmente tem validade inferior)
  - **Certificate Policies**: usada quando existem várias políticas
  - **Policy Mappings**: usado apenas nos certificados de um CA emitidos para outro CA



# X.509 Versão 3 (3)

- ◆ Sujeito e Atributos da CA
  - *Subject alternative name*
  - *Issuer alternative name*
  - *Subject directory attributes*: permite adicionar qualquer atributo da directoria X.500
- ◆ Restrições do caminho de certificação
  - *Basic constraints*: indica se o sujeito pode actuar como CA
  - *Name constraints*: indica um espaço de nomes (semelhante ao DNS)
  - *Policy constraints*



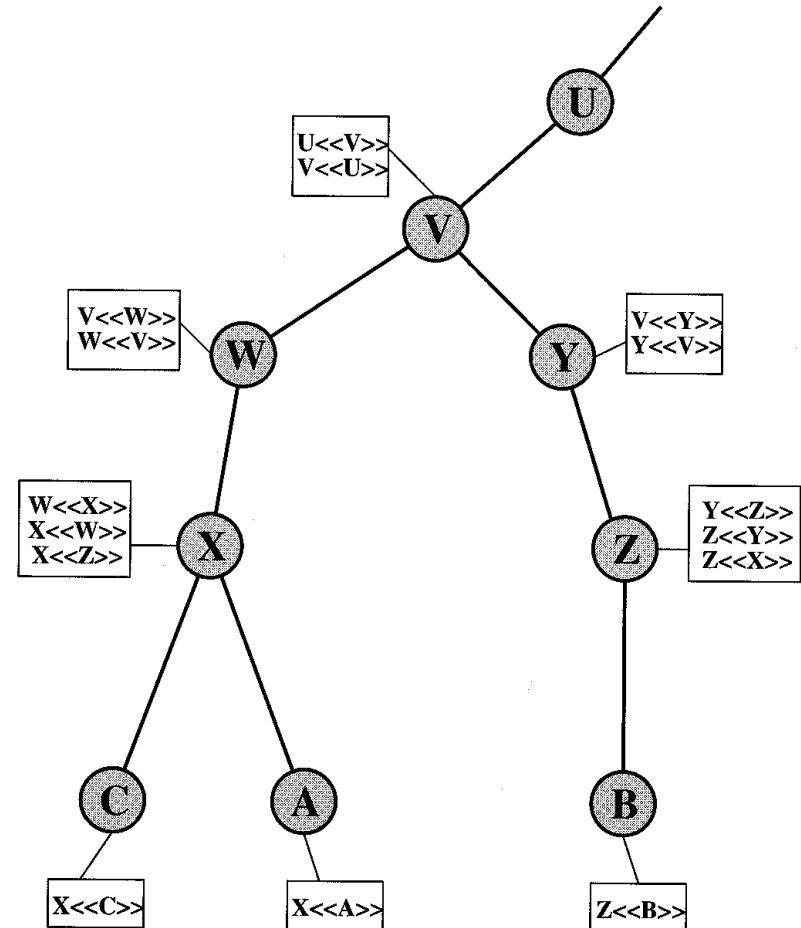
# Cadeias de Certificados (1)

- ◆ Qualquer utilizador com acesso à chave pública da CA pode recuperar a chave pública certificada
- ◆ Nenhuma outra parte, além da CA, pode modificar o certificado sem que seja detectado
- ◆ Quando há muitos utilizadores, eles são distribuídos por várias CAs
- ◆ A autenticação entre dois utilizadores de diferentes CA's:
  - X1<< X2 >> X2<< X3 >> ... XN<< B >>



# Cadeias de Certificados (2)

- ◆ *Forward certificates*
  - Certificados de X gerados por outros CAs
- ◆ *Reverse certificates*
  - Certificados gerados por X, que certificam outros CAs





# Procedimentos de autenticação (1)

- ◆ Existem várias forma de autenticação (assumindo que ambas as entidades conhecem as chaves públicas uma da outra)
  - Autenticação *one-way*
  - Autenticação *two-way*
  - Autenticação *three-way*

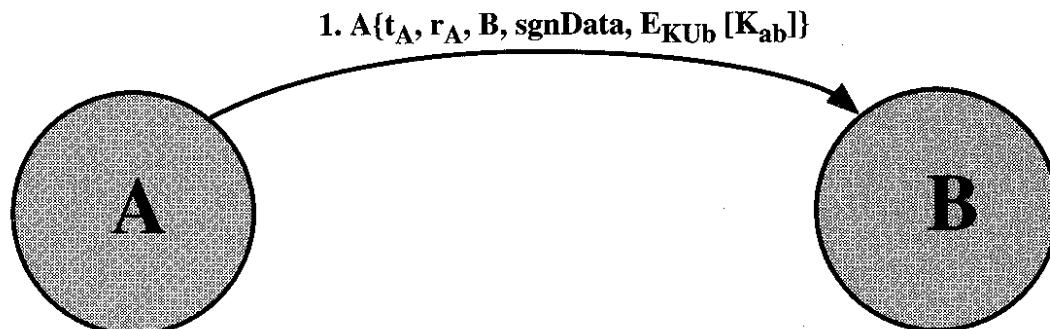


# Procedimentos de autenticação (2)

- ◆ Autenticação *one-way*

- Envio de apenas uma mensagem que garante:
    - A autenticidade de  $A$  e que a mensagem foi criada por  $A$
    - Que o destinatário da mensagem é  $B$
    - A integridade da mensagem e a originalidade (que não foi enviada mais do que uma vez)

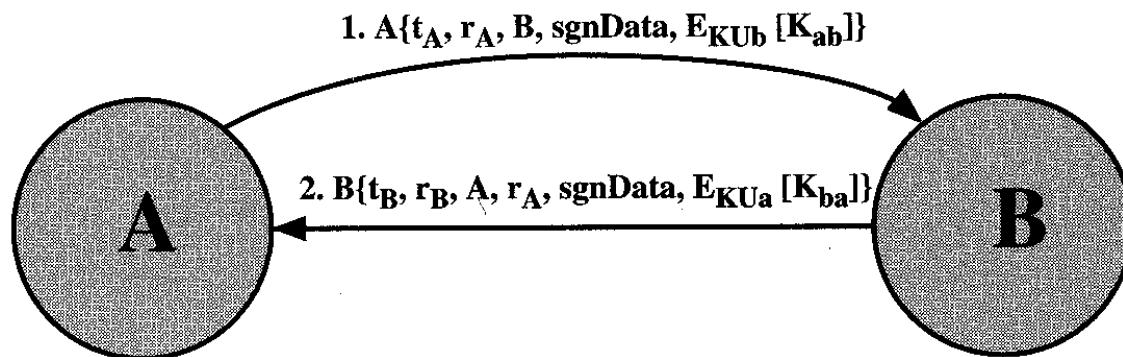
- ◆ Não é verificada a identidade de  $B$





# Procedimentos de autenticação (3)

- ◆ Autenticação *two-way*
  - Envio de duas mensagens, garante que:
    - A identidade de **B** e que a mensagem de resposta foi gerada por **B**
    - Que a mensagem se destina a **A**
    - A integridade e originalidade da resposta
- ◆ Permite a identificação de ambas as entidades

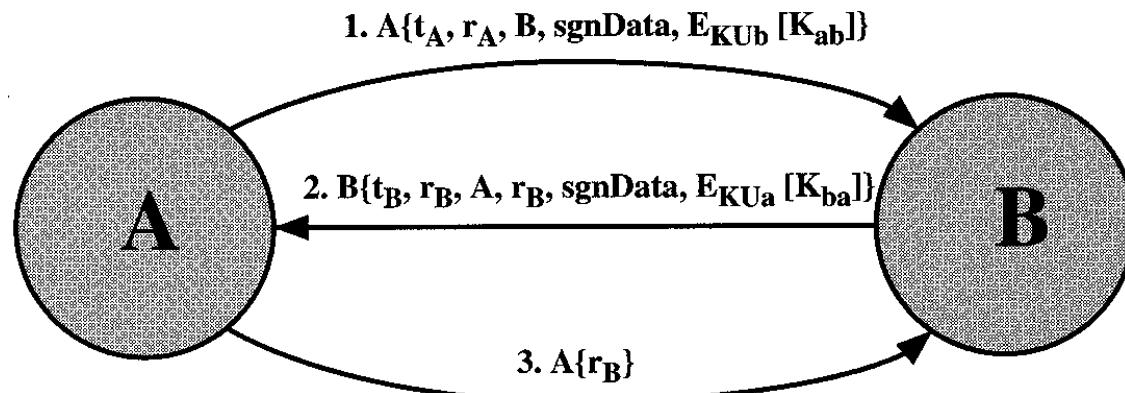




# Procedimentos de autenticação (4)

## ♦ Autenticação *three-way*

- Devolvida uma mensagem  $A \rightarrow B$  com o *nounce*  $r_B$  assinado
- Utilizado quando não há sincronização de relógios na transmissão



---

**Assinaturas digitais  
qualificadas com o  
Cartão de Cidadão**



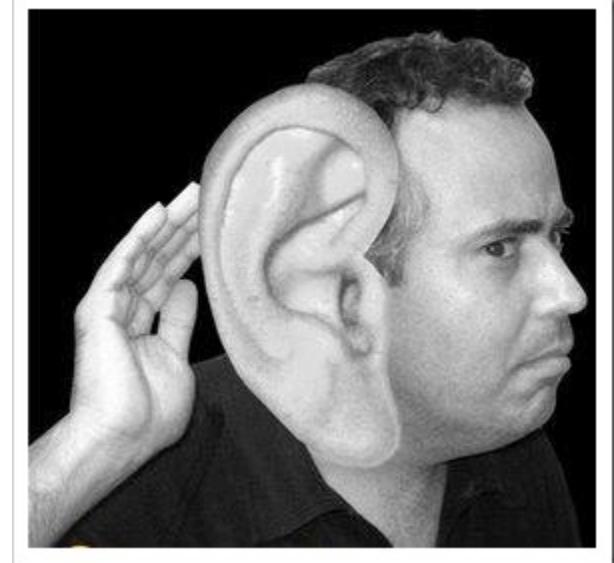
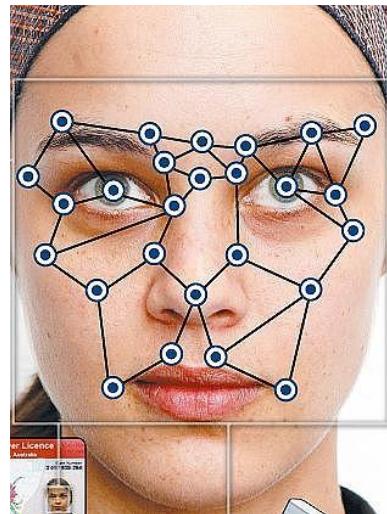


## Definições

- **Autenticação**
  - capacidade de garantir que uma entidade é quem afirma ser

# Assinaturas Digitais

## Autenticação – Mundo real



# Assinaturas Digitais

## Autenticação – Informática

- Segredo
  - Senha
- Algo que possuímos
  - Token criptográfico
- Algo que somos
  - Dados biométricos
    - Impressão digital
    - Íris, Voz, face



Chip criptográfico





## Definições

- **Não-repúdio**
  - capacidade de impedir que uma entidade negue a sua participação numa transação

## Não-repúdio – Mundo real

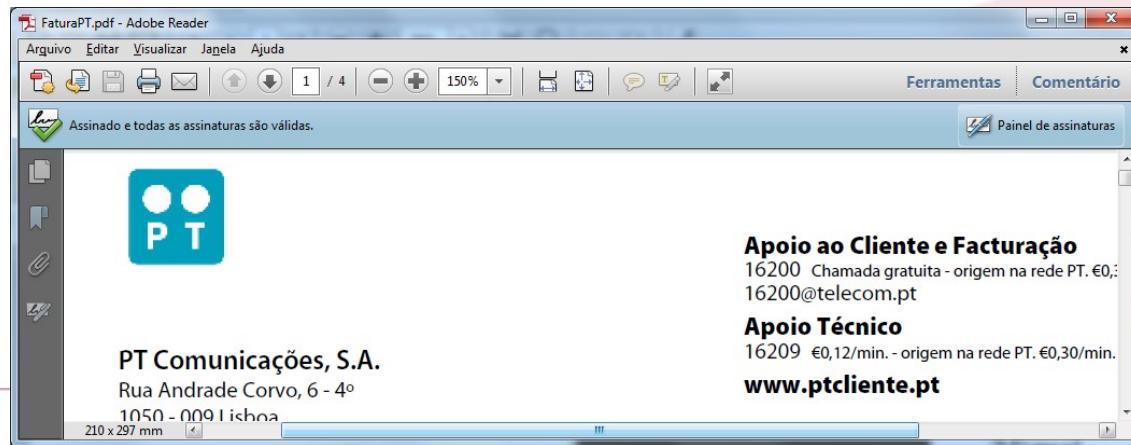
- Assinaturas (em alguns casos têm de ser presenciais)



A large, handwritten signature in black ink, reading "Trauitho Campos", is displayed prominently in the center of the slide. The signature is fluid and cursive, with a long horizontal stroke at the end.

## Não-repúdio – Informática

- Assinaturas electrónicas qualificadas
  - Pessoais: com cartão do cidadão
    - Documentos, emails, ...
  - Empresas: ex. facturas, ...



FaturaPT.pdf - Adobe Reader

Assinado e todas as assinaturas são válidas.

**PT Comunicações, S.A.**  
Rua Andrade Corvo, 6 - 4º  
1050 - 009 Lisboa

**Apoio ao Cliente e Facturação**  
16200 Chamada gratuita - origem na rede PT. €0,00;  
16200@telecom.pt

**Apoio Técnico**  
16209 €0,12/min. - origem na rede PT. €0,30/min.  
[www.ptcliente.pt](http://www.ptcliente.pt)

- Assinaturas digitais

- O que são?
- Como funcionam?
- Qual é a sua validade legal?
- Como se faz uma assinatura digital?

## Conceitos

- Assinaturas digitais
  - Mecanismo criptográfico que permite garantir o não-repúdio
- Não confundir:
  - Assinaturas digitalizadas
  - Assinaturas electrónicas
  - Assinaturas digitais
- Embora parecidos, têm significados e validade legais diferentes.



## Assinaturas digitalizadas

- Conversão de uma assinatura manuscrita em suporte físico (ex. papel) para uma imagem em suporte digital (ex. ficheiro)
  - Muito fácil de copiar
  - Sozinha não dá garantias de não-repúdio
  - Exemplo de utilização: Fax



## Assinaturas electrónicas

- Definição legal (Decreto-Lei 62/2003):
  - **Assinatura electrónica:** resultado de um processamento electrónico de dados susceptível de constituir objecto de direito individual e exclusivo e de ser utilizado para dar a conhecer a autoria de um documento electrónico;
  - **Exemplo:** simples escrita do nome completo para identificar o remetente de um documento electrónico (email, .doc, pdf, etc)

## Assinaturas electrónicas

- Definição legal (Decreto-Lei 62/2003):
  - **Assinatura electrónica avançada:** assinatura electrónica que preenche os seguintes requisitos:
    - Identifica de forma unívoca o titular como autor do documento;
    - A sua aposição ao documento depende apenas da vontade do titular;
    - É criada com meios que o titular pode manter sob seu controlo exclusivo;
    - A sua conexão com o documento permite detectar toda e qualquer alteração superveniente do conteúdo deste;
  - **Exemplo:** assinaturas arbitradas: [www.hellosign.com](http://www.hellosign.com) e [www.eavid.com](http://www.eavid.com)
    - Neste contexto as assinaturas digitalizadas são válidas

## Assinaturas electrónicas

- Definição legal (Decreto-Lei 62/2003):
  - **Assinatura digital:** modalidade de assinatura electrónica avançada baseada em **sistema criptográfico** assimétrico composto de um algoritmo ou série de algoritmos, mediante o qual é gerado um par de chaves assimétricas exclusivas e interdependentes, uma das quais privada e outra pública, e que permite ao titular usar a chave privada para declarar a autoria do documento electrónico ao qual a assinatura é apostada e concordância com o seu conteúdo e ao destinatário usar a chave pública para verificar se a assinatura foi criada mediante o uso da correspondente chave privada e **se o documento electrónico foi alterado depois de apostar a assinatura;**
  - **Exemplo:** email com certificados digitais em ficheiros

## Assinaturas electrónicas

- Definição legal (Decreto-Lei 62/2003):
  - **Assinatura electrónica qualificada:** assinatura digital ou outra modalidade de assinatura electrónica avançada que satisfaça exigências de segurança idênticas às da assinatura digital baseadas num certificado qualificado e criadas **através de um dispositivo seguro de criação de assinatura;**
    - Nem todas as assinaturas digitais são qualificadas
  - **Exemplo:** assinaturas digitais com cartão de cidadão



## Assinaturas electrónicas

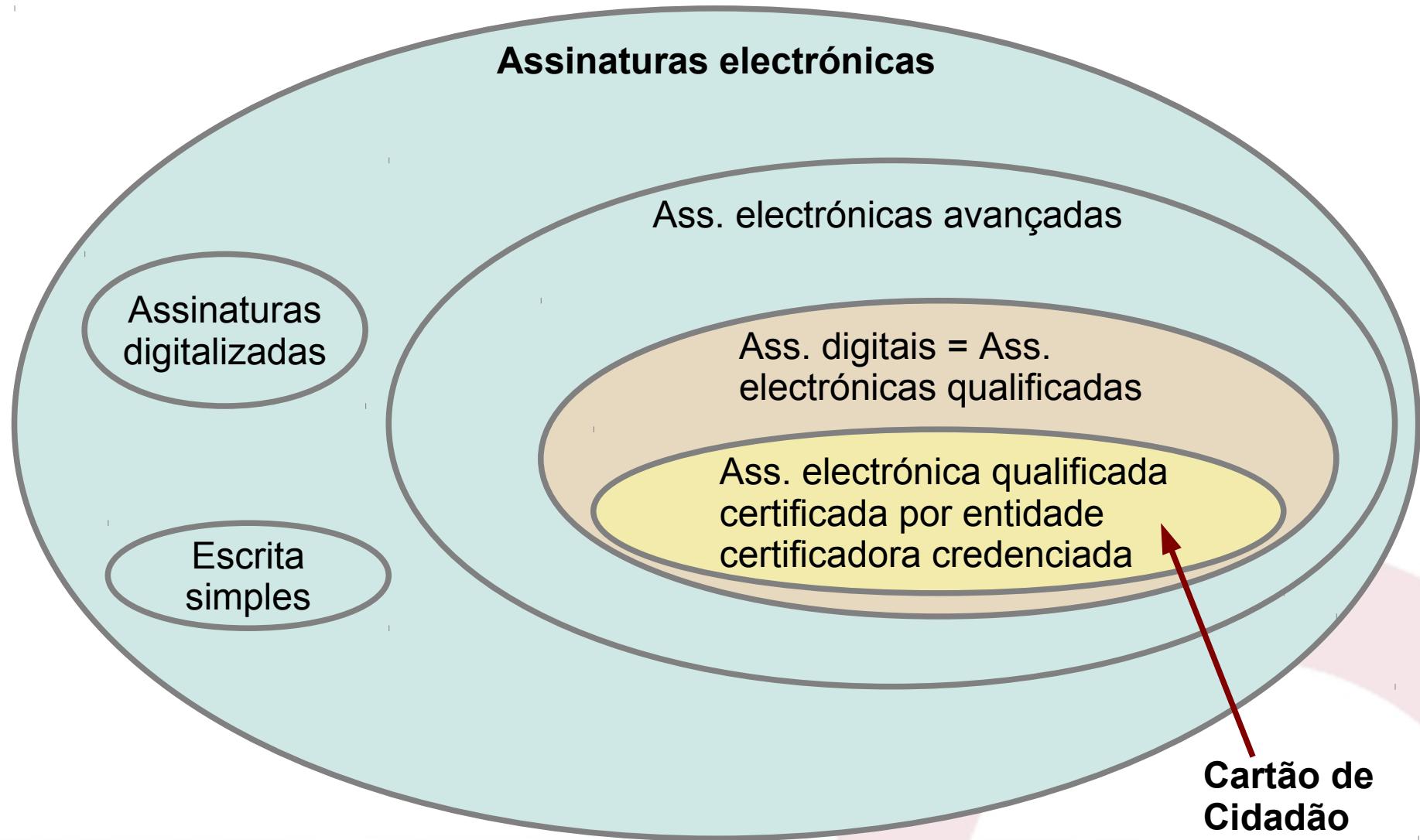
- Definição legal (Decreto-Lei 62/2003):
  - **Dispositivo seguro de criação de assinatura:** dispositivo de criação de assinatura que assegure, através de meios técnicos e processuais adequados, que:
    - Os **dados necessários à criação de uma assinatura** utilizados na geração de uma assinatura **só possam ocorrer uma única vez** e que **a confidencialidade desses dados se encontre assegurada**;
    - Os dados necessários à criação de uma assinatura utilizados na geração de uma assinatura não possam, com um grau razoável de segurança, ser deduzidos de outros dados e **que a assinatura esteja protegida contra falsificações realizadas através das tecnologias disponíveis**;



## Assinaturas electrónicas

- Definição legal (Decreto-Lei 62/2003):
  - **Dispositivo seguro de criação de assinatura:**  
(continuação)
    - Os **dados necessários à criação de uma assinatura** utilizados na geração de uma assinatura possam ser eficazmente **protetidos pelo titular contra a utilização ilegítima por terceiros**;
    - Os dados que careçam de assinatura não sejam modificados e possam ser apresentados ao titular antes do processo de assinatura;

# Assinaturas Digitais



Cartão de  
Cidadão



## Assinaturas com Cartão de Cidadão (DL 62/2003)

- **Assinatura electrónica qualificada certificada por entidade certificadora credenciada**
  - **Assinatura electrónica qualificada** → já falámos
  - **certificada por entidade certificadora** → uma 3<sup>a</sup> entidade (conhecida como entidade certificadora, ou CA) que certifica a identidade (certificados digitais, norma x.509)
  - **credenciada** → essa 3<sup>a</sup> entidade legalmente autorizada e reconhecida para emitir identidades digitais: **entidade credenciadora**
    - Em Portugal, o organismo que exerce essa função é a Autoridade Nacional de Segurança (ANS) DL 116-A/2006



## Porque é que o sistema das assinaturas com cartão de cidadão é tão complexo?

- Sistema baseada em criptografia assimétrica
- Tipos de cifra:
  - Simétrica – a mesma chave para cifrar e decifrar
    - Troca-se de chave regularmente (duração <= 1 dia)
  - Assimétrica – a chave que cifra é diferente da chave que decifra
    - As chaves identificam entidades, permanecem as mesmas por longos períodos (duração: 1 a 5 anos)

## Criptografia assimétrica e o DL 62/2003

- Terminologia tecnologicamente neutra
- Criptografia assimétrica:
  - Chave privada (não pode ser divulgada) → dados de criação de assinatura
  - Chave pública (deve ser divulgada) → dados de validação de assinatura
  - Assinatura digital → Ass. electrónicas qualificadas
- No DL não existe referência à cifra de dados
  - Porque o CC não o permite fazer (infelizmente)



## Certificados digitais

- As chaves públicas
  - São um conjunto grande de bits  $\geq 1024$  bits
  - Sozinhos não permitem identificar ninguém
  - É necessário adicionar informação da identidade (ID) do dono da chave pública
- Como adicionar o ID?
  - ID + Chave pública
    - Não é seguro, é facilmente forjado
  - Certificados digitais (x.509)
    - ID + Chave pública + T + assinatura da EC

## Como se validam assinaturas?

- Com o certificado digital (chave pública) de quem assinou
- E com o certificado digital da EC que assinou o certificado digital
- E quem assina o certificado da EC?
  - Outra EC (cadeia)
  - Ou ela própria (auto assinado)
    - Ela própria? Mas isso é de confiança?
      - » Se for credenciada sim.

## Como se validam assinaturas?

- Como é que obtemos os certificados das EC, nomeadamente os auto assinados?
  - A maioria dos credenciados já vêm pré-instalados nos SO e programas (ex. Browsers)
  - Os outros podem ser adicionados pelos utilizadores: perigo!



### This Connection is Untrusted

You have asked Firefox to connect securely to [www.cacert.org](http://www.cacert.org), but we can't confirm that your connection is secure.

Normally, when you try to connect securely, sites will present trusted identification to prove that you are going to the right place. However, this site's identity can't be verified.

### What Should I Do?

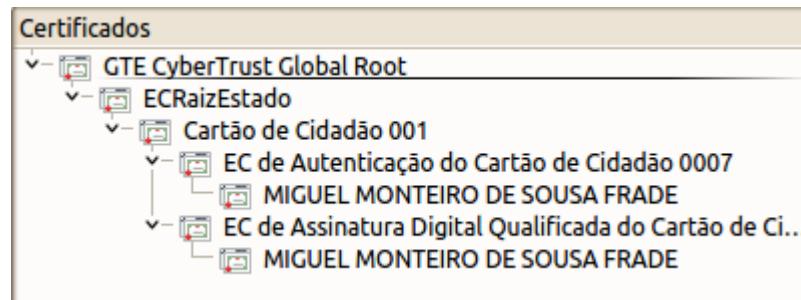
If you usually connect to this site without problems, this error could mean that someone is trying to impersonate the site, and you shouldn't continue.

[Get me out of here!](#)

- ▶ [Technical Details](#)
- ▶ [I Understand the Risks](#)

## Como se validam assinaturas?

- Cadeia de certificados no Cartão de Cidadão



- É preciso tê-los **TODOS** para validar uma assinatura
  - O GTE já vem pré-instalado
  - Os outros são instalados com o SW do cartão
  - Ou terão de ser instalados manualmente

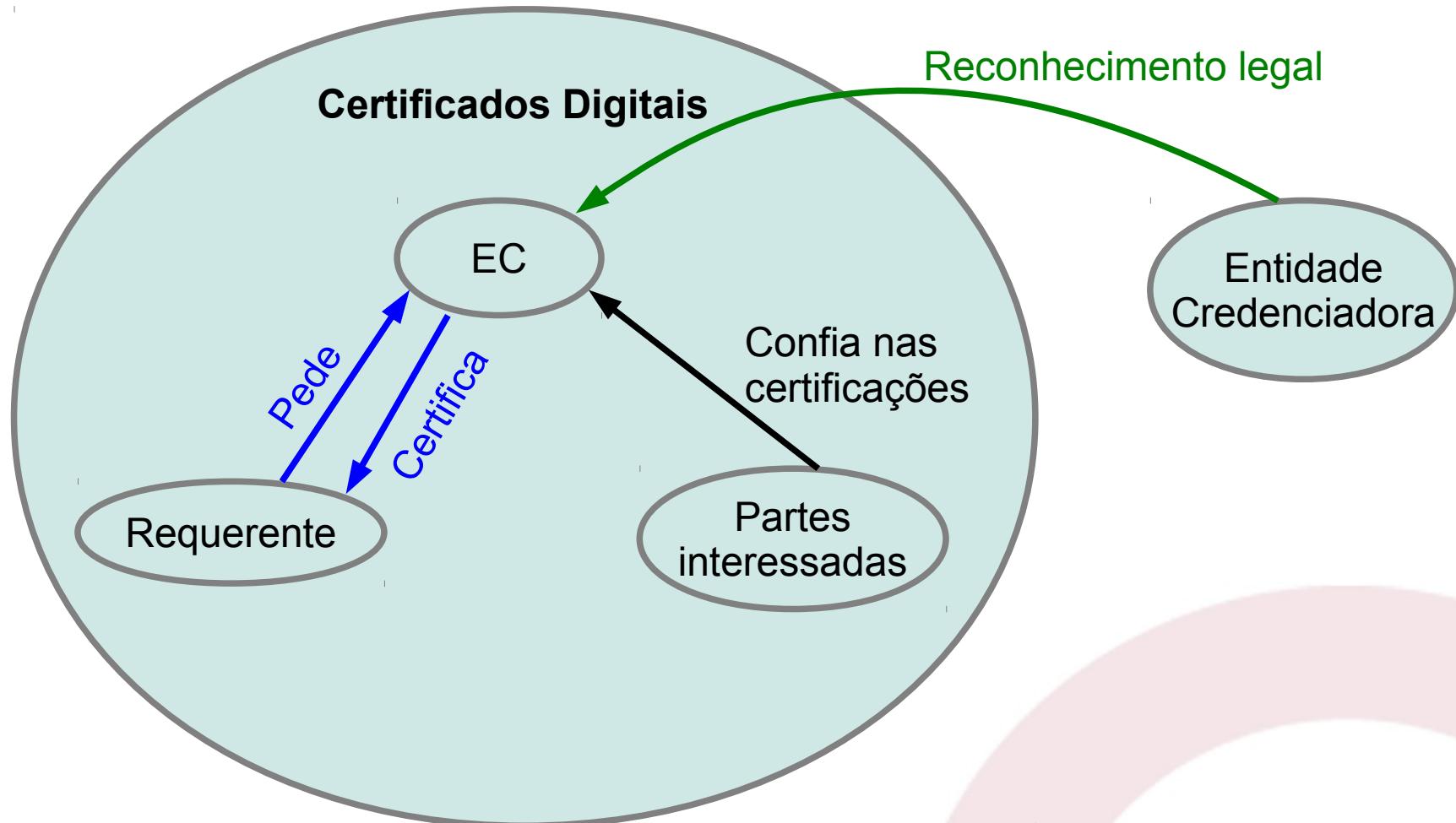
## Certificados das ECs (ou certificados de raíz)

- Quem decide quais são pré-instalados?
  - Os fabricantes dos SOs e dos programas
- Posso confiar nos fabricantes?
  - Normalmente baseiam-se em listas governamentais das ECs credenciadas
    - Portugal: [www.gns.gov.pt/gns/pt/tsl/](http://www.gns.gov.pt/gns/pt/tsl/)
    - É preciso pagar para ser credenciado
      - » E os certificados não são gratuitos!
  - Então, isto dos certificados é um negócio?
    - Sim, também é um negócio!
    - Embora existam exceções ([www.cacert.org](http://www.cacert.org))

## Certificados das ECs (ou certificados de raíz)

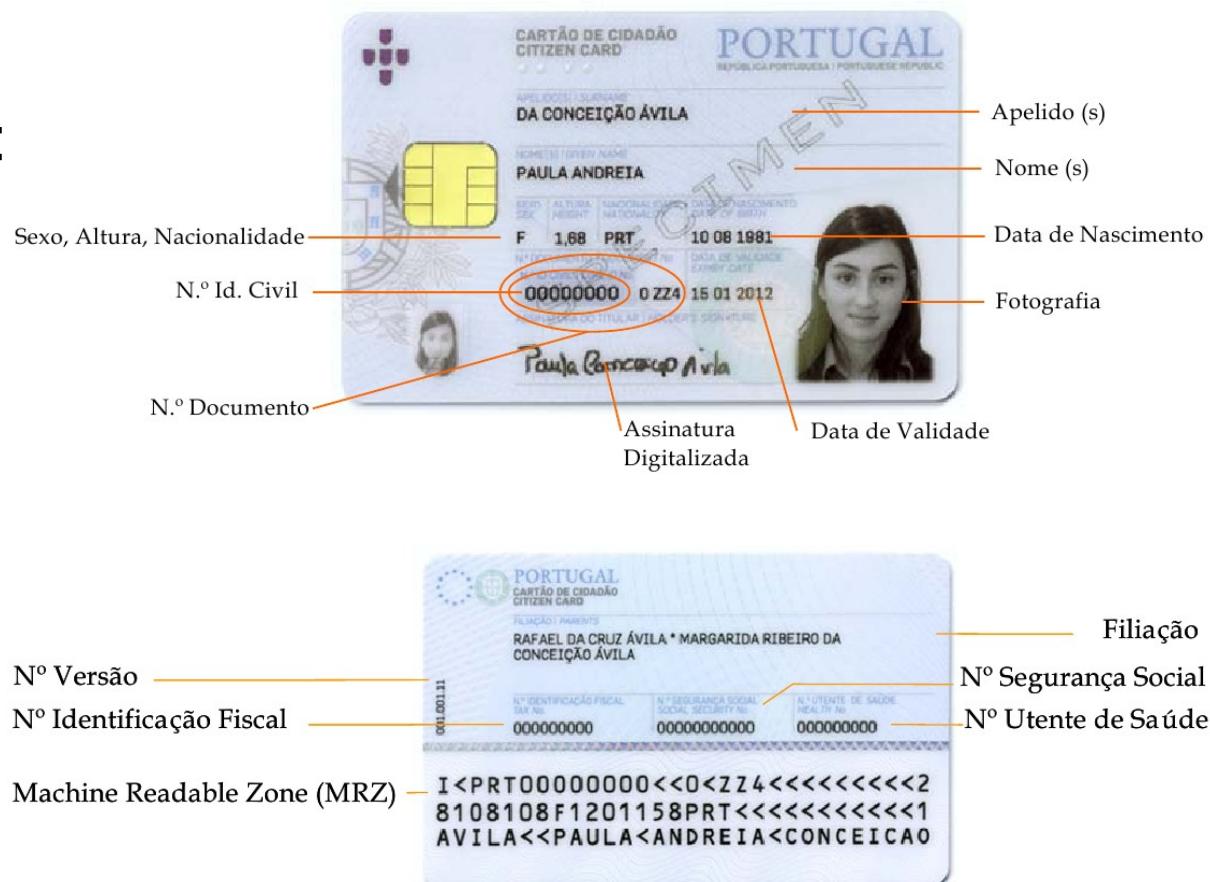
- Então \$\$\$ = segurança ?
  - Não! É preciso haver confiança no sistema.
  - Confiança
    - Não se compra
    - Não se resolve só com criptografia
    - Não se resolve só com meios técnicos
    - É preciso conquistá-la
      - » A entidade credenciadora dá credibilidade ao sistema

# Assinaturas Digitais



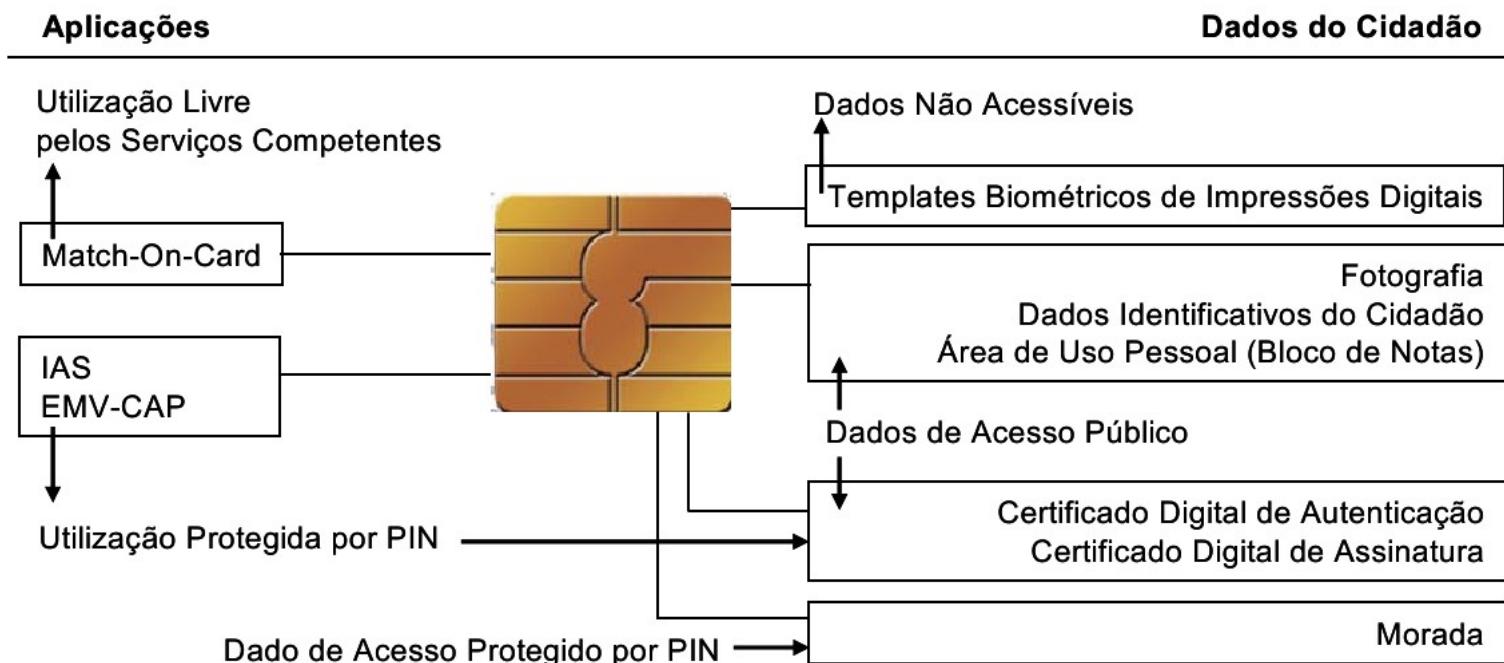
## Cartão de Cidadão

- Substitui 5 cartões:
  - Bilhete de identidade
  - segurança social
  - cartão de contribuinte;
  - cartão de utente
  - cartão de eleitor (??)



## Cartão de Cidadão

- Smart card (cartão inteligente): tem um microcomputador embebido



## Cartão de Cidadão

- Realiza operações criptográficas
- Tem 3 PINs com 4 algarismos cada:
  - Autorizar a indicação de morada
  - Autenticação do titular
  - Produzir uma assinatura digital
    - *Assinatura electrónica qualificada certificada por entidade certificadora credenciada*

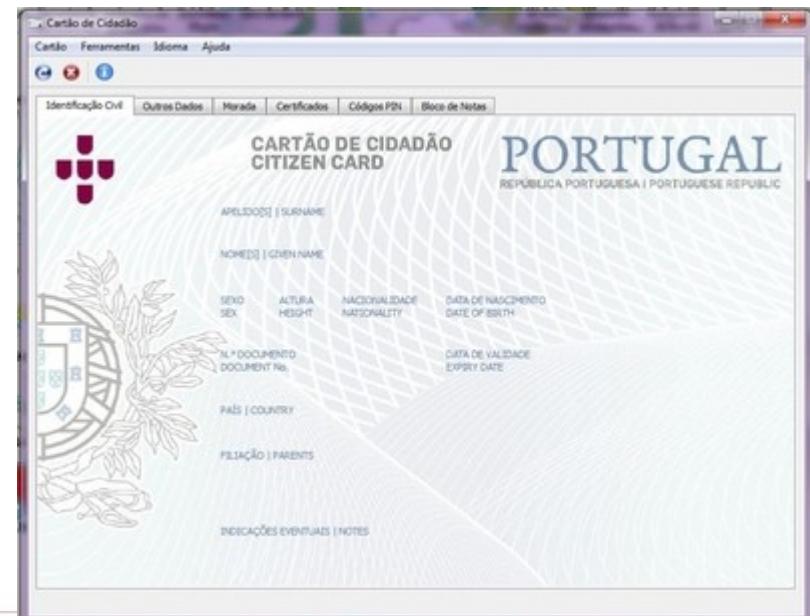
# Assinaturas Digitais

## Cartão de Cidadão – Requisitos

- Leitor de *smartcards*
- PINs
- Software <http://www.cartaodecidadao.pt/>

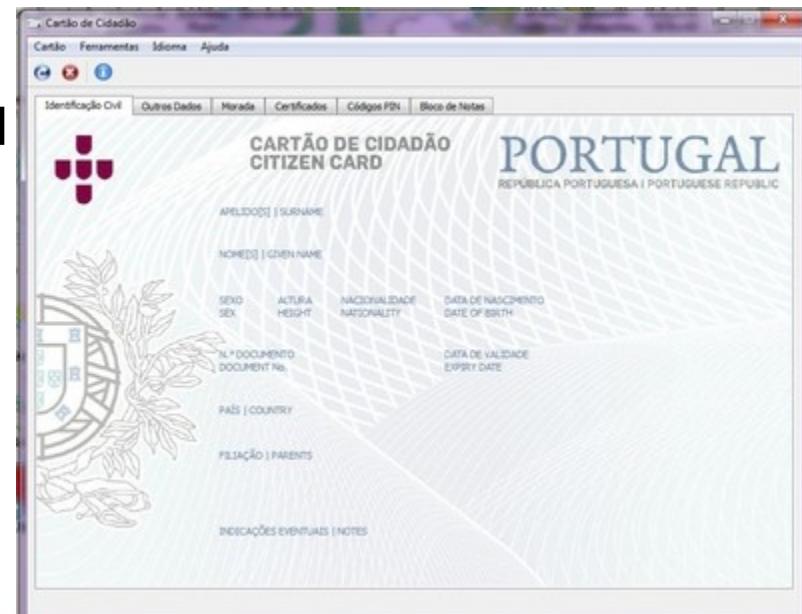


- PIN da Morada:**
- PIN da Autenticação:**
- PIN da Assinatura Digital:**



## Cartão de Cidadão

- Aplicação fornecida <http://www.cartaodecidadao.pt/>
  - Permite ver os dados visíveis no cartão físico
  - Ver a morada (com PIN)
  - Ver os certificados
  - Permite alterar os códigos PIN
  - Bloco de notas  
(1000 caracteres) ?????



## Cartão de Cidadão

- Exercício 1 – ler os dados gravados no cartão de cidadão
  - Introduzir o cartão no leitor
  - Abrir a aplicação do “cartão de cidadão”
  - Esperar alguns segundos pela leitura
    - Navegar pelas diferentes abas

## Cartão de Cidadão – Autenticação

- Alguns sítios já suportam autenticação com o CC
  - [www.portaldasfinancas.gov.pt](http://www.portaldasfinancas.gov.pt)
  - <http://queixaselectronicas.mai.gov.pt>
  - <http://www.portaldasescolas.pt> (matrícula electrónica)
  - ...
- Fora do âmbito desta atividade
  - É relativamente simples de fazer
  - Consultar manual do Cartão de Cidadão

## Cartão de Cidadão – Assinatura digital

- Requisitos
  - Além do que já foi referenciado (PIN, leitor, SW do CC)
  - Programas que permitem usar assinaturas digitais, exemplos:
    - Adobe Reader (0€) → reconhece e valida assinaturas digitais em **PDF**
    - Adobe Acrobat (700€) → reconhece, valida e faz assinaturas digitais em **PDF**
    - DigiSigner (0€ 14 dias, 35€) → reconhece, valida e faz assinaturas digitais em **PDF**
    - Thunderbird (0€), Outlook (100€) → reconhece, valida e faz assinaturas digitais nos **emails**

## Cartão de Cidadão – Assinatura digital

- Exercício 2 – validar um PDF assinado
  - Não é preciso ter leitor de smartcards
  - Configurar o Adobe Reader:
    - 1. clique em Editar → Preferências → Assinaturas → Verificação → Integração com o Windows
    - 2. No separador Integração com o Windows e coloque um visto na 1<sup>a</sup> opção
      - NOTA: Se o SW do CC não estiver instalado é necessário instalar TODOS os certificados primeiro
    - 3. Abrir o PDF assinado
      - Clicar em cima da assinatura

## Cartão de Cidadão – Assinatura digital

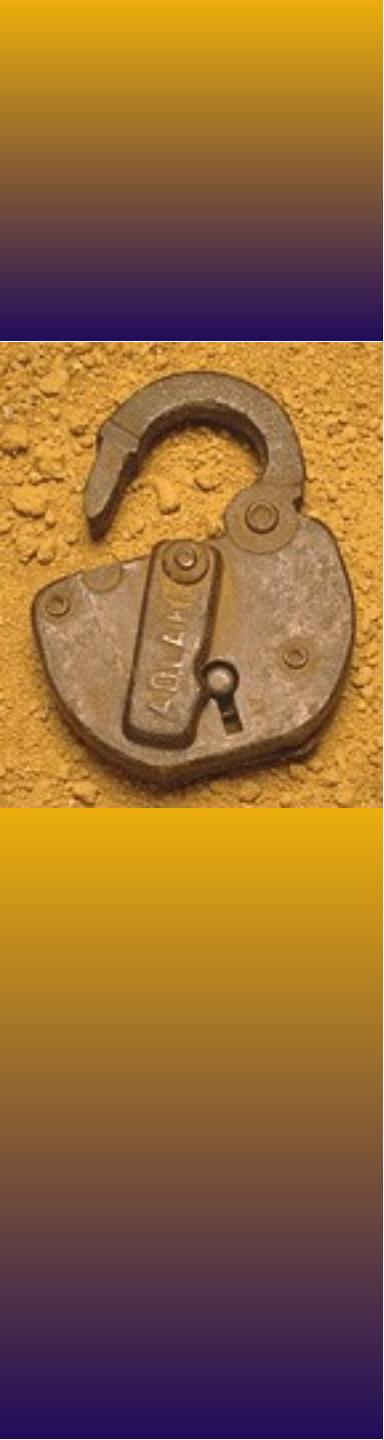
- Exercício 3 – Assinar um **PDF** com o CC
  - DigiSigner → Abrir PDF a assinar
    - Assinar documento → “Não são permitidas alterações”
    - Escolher o certificado “Assinatura Digital”
    - Avançado → URL do servidor (validação cronológica):
      - <http://ts.cartao-decidadao.pt/tsa/server>
    - Algoritmo de hash: SHA256 (mais seguro)
    - Assinar

## Cartão de Cidadão – Assinatura digital

- Exercício 4 – Assinar um email
  - Criar conta no Gmail só para este exercício
  - Configurar o Thunderbird e testar envio e receção de email
  - No Thunderbird:
    - Manual do cartão do cidadão: pag. 56
  - Trocar emails com os colegas
  - Validar email

## Bibliografia

- Decreto de Lei nº 290/1999
- Decreto de Lei nº 62/2003
- André Zuquete, Segurança em Redes Informáticas, FCA, 4<sup>a</sup> edição → Capítulo 3
- Manual do Software do cartão de cidadão ([https://www.portaldocidadao.pt/ccsoftware/Manual\\_Carta\\_o\\_de\\_Cidadao\\_v1.24.1.pdf](https://www.portaldocidadao.pt/ccsoftware/Manual_Carta_o_de_Cidadao_v1.24.1.pdf))

A photograph of a vintage-style metal padlock mounted on a yellow wall. The padlock is open, showing its internal mechanism and a small key hanging from the shackle. The background is a solid yellow color.

# Arquitectura IPSec



# Motivação

- ◆ Ainda existem muitas aplicações sem qualquer tipo de segurança
- ◆ Existem protocolos aplicacionais específicos de segurança, como o S/MIME, PGP, Kerberos, SSL
  - A sua utilização implica a alteração de software
  - Não é transparente para o utilizador
  - Implica educar os utilizadores
- ◆ Uma solução, implementar a segurança ao nível da camada de rede (IP):
  - Não é necessário alterar *software*
  - É transparente para o utilizador

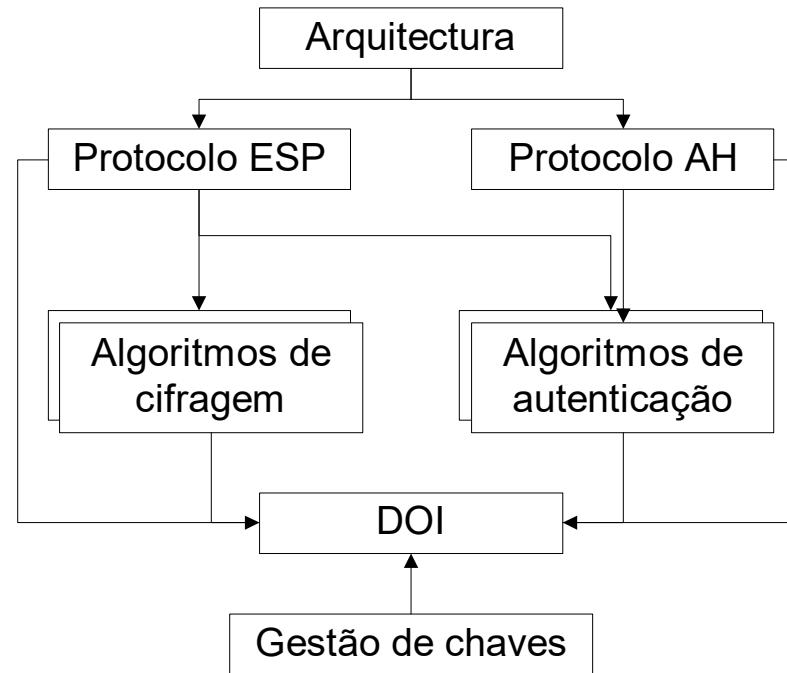


# Caracterização (1)

- ◆ Norma de segurança para a Internet, publicado em vários RFC's:
  - RFC 4301 – *Security Architecture for the Internet Protocol*
  - RFC 4302 – *IP Authentication Header*
  - RFC 4303 – *IP Encapsulating Security Payload (ESP)*
  - RFC 2411 – *IP Security Document Roadmap*
- ◆ Um dos autores da RFC 2401 escreveu um livro onde explica detalhadamente o funcionamento do IPSec:
  - N. Doraswamy, D. Harkins, “*IPSec – The New Security Standard for the Internet, Intranets and Virtual Private Networks*”, Prentice Hall, 1999
- ◆ A norma foi criada para o IPv6, mas foi assegurada a compatibilidade para o IPv4

# Caracterização (2)

## ◆ Roteiro da documentação IPSec





# Caracterização (3)

- ◆ O IPsec oferece vários serviços:
  - Autenticação
  - Integridade
  - Anti-replay
  - Confidencialidade
  - Confidencialidade limitada do fluxo
- ◆ Através da definição de dois novos cabeçalhos de extensão ao pacote IP
  - AH (Authentication Header)
  - ESP (Encapsulation Security Payload)

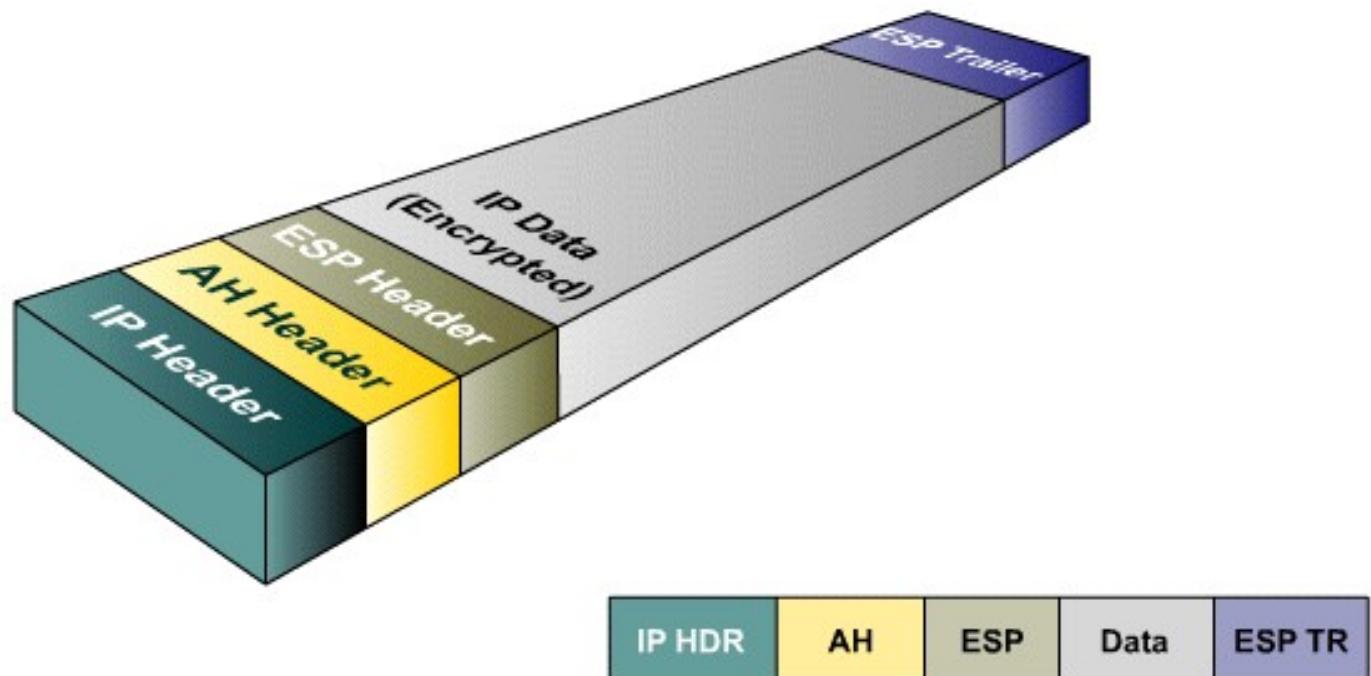


# Caracterização (4)

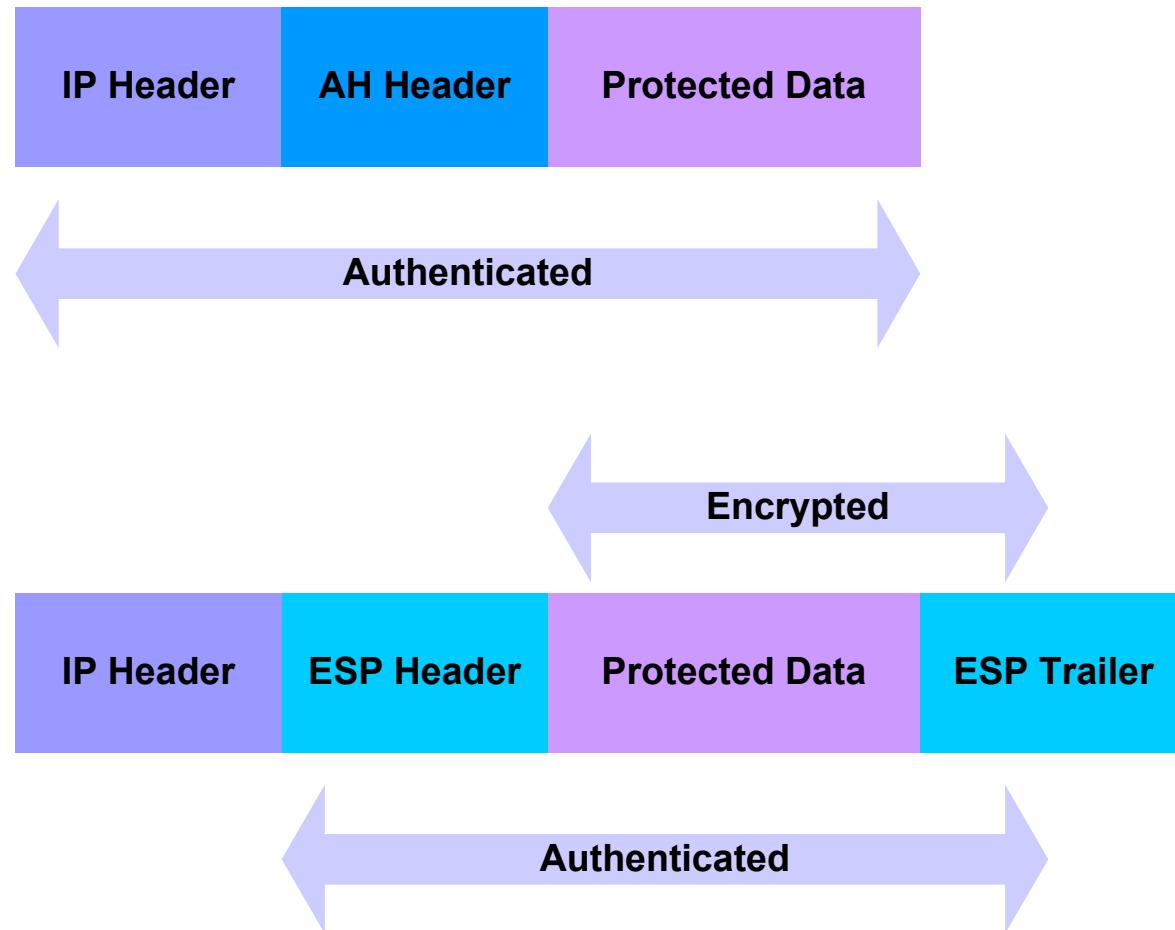
Serviço	AH	ESP	ESP com autenticação
<i>Autenticação</i>	o		o
<i>Integridade</i>	o		o
<i>Anti-replay</i>	o	o	o
<i>Confidencialidade</i>		o	o
<i>Confidencialidade limitada do fluxo</i>		o	o

# Caracterização (5)

- ◆ Estrutura de um pacote IP com os cabeçalhos de extensão



# Caracterização (6)





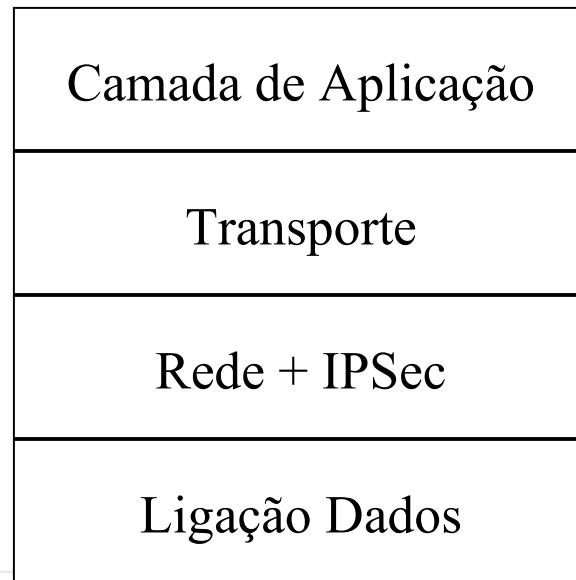
# Localização (1)

- ◆ Nos sistemas terminais
  - Segurança extremo-a-extremo
  - Segurança individual de cada fluxo
  - Possibilidade de ligar a SA ao contexto do utilizador
  - Impede a utilização de NAT e IPs privados
  - Duas alternativas
    - No sistema operativo
    - Bump In The Stack (BITS)



# Localização (2)

- ◆ No sistema operativo
  - Ao lado do IP
  - Mais eficiente
  - É mais fácil assegurar segurança fluxo-a-fluxo





# Localização (3)



- ◆ *Bump In The Stack* (BITS)
  - Não são necessárias alterações à pilha protocolar
  - Implica a duplicação de algumas funções IP
  - Menos eficiente e versátil

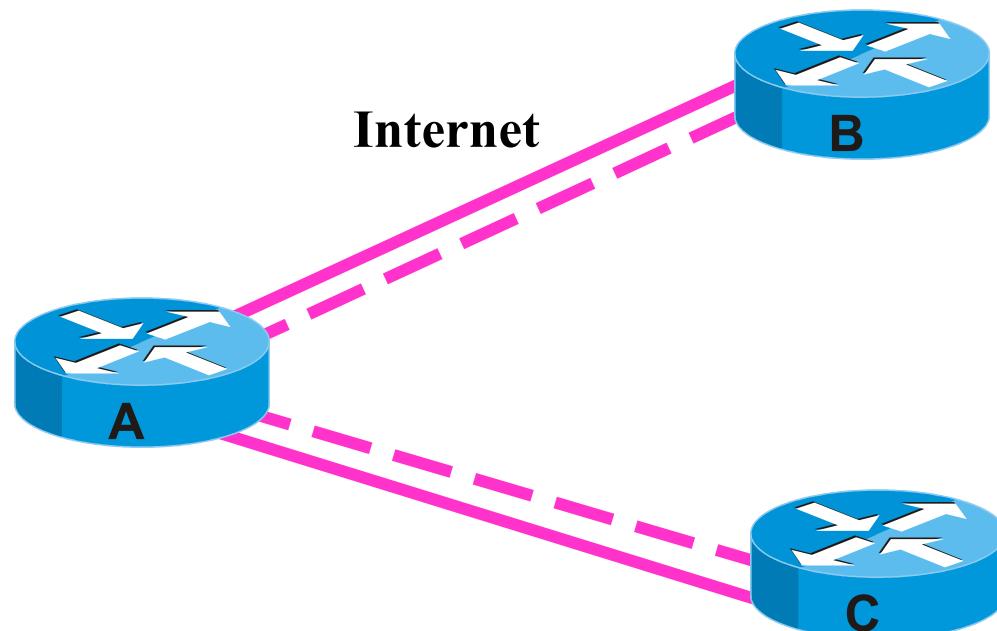


# Localização (4)

- ◆ Nos routers
  - Segurança entre LANs sobre a Internet
  - Transparente para os utilizadores finais
  - Facilita a autenticação de utilizadores à entrada das redes privadas
  - Funciona associado ao NAT e a IPs privados
  - Duas alternativas
    - Em modo nativo
    - Bump In The Wire (BITW)

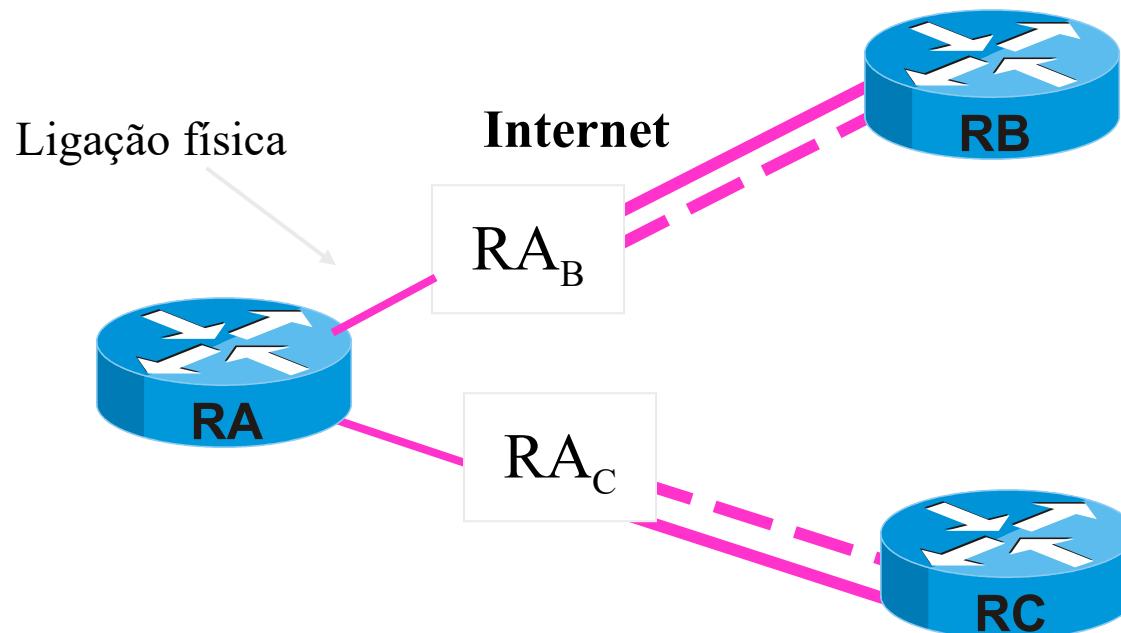
# Localização (5)

- ◆ Modo nativo
  - IPSec integrado na pilha protocolar do router



# Localização (6)

- ◆ Bump In The Wire (BITW)
  - Utilização de hardware adicional entre o router e a linha





# Associações de Segurança (1)

## ♦ Security Association (SA)

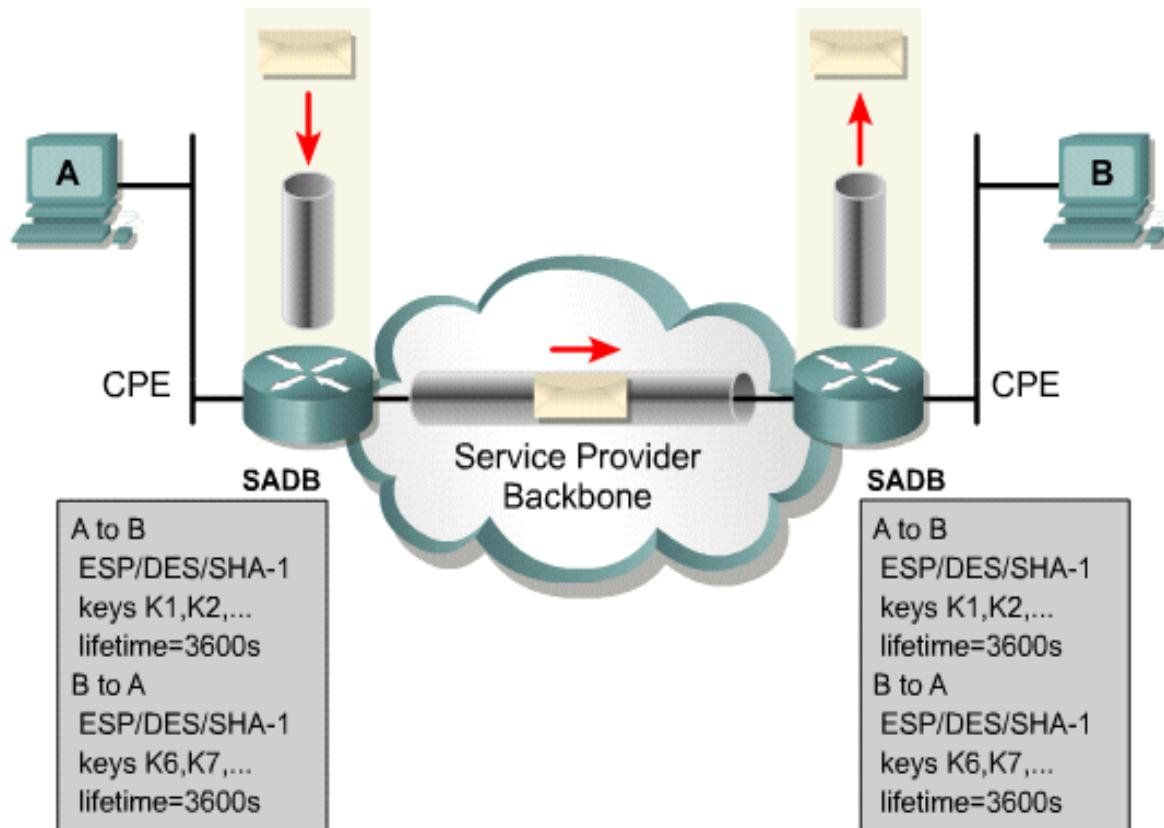
- Espécie de contracto entre o emissor e o receptor
- Definem todos os elementos necessários para se proceder à comunicação de forma segura
- É unidireccional (para uma comunicação de dados bidireccional são necessários dois SA's).
- São identificados de forma unívoca por três parâmetros:
  - o SPI (*Security Parameters Index*), que é um número de identificação local do SA
  - o endereço IP de destino
  - o *Security Protocol Identifier* que indica se o protocolo a usar no SA é o AH ou o ESP.
- Todos os SA's são guardados numa base de dados denominada SAD (*Security Association Database*)



# Associações de Segurança (2)

- ◆ Parâmetros guardados na SAD:
  - Contador de número de sequência
  - Sequence Counter Overflow
  - Anty-replay Window
  - Informação AH (algoritmos, chaves, validade, ...)
  - Informação ESP (chaves, valores de inicialização, algoritmos, ...)
  - Tempo de vida da SA
  - Modo IPSec (transporte ou túnel)
  - Path MTU (tamanho máximo sem fragmentação)

# Associações de Segurança (3)





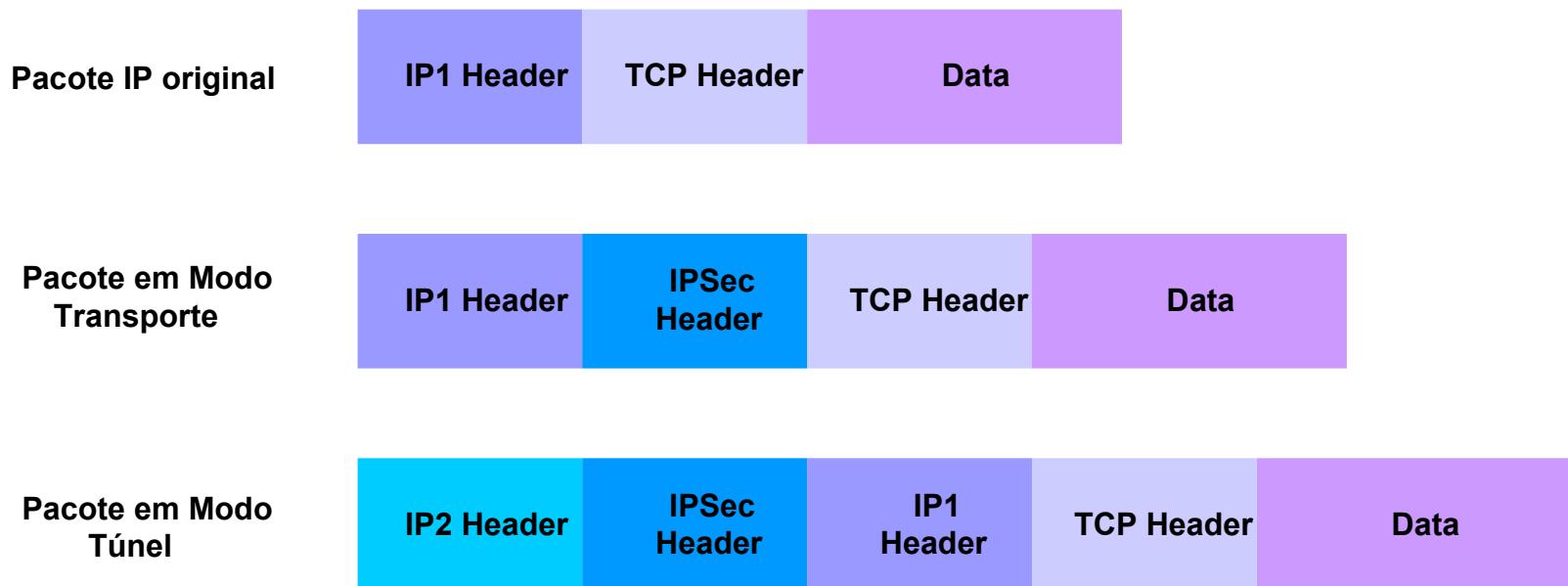
# Associações de Segurança (4)

- ◆ A discriminação do tráfego a proteger é feita na SPD (security policy database), baseada em selectores:
  - Endereço IP origem e destino
  - UserID (válido se o IPSec estiver no mesmo SO do utilizador)
  - Nível de segurança
  - Protocolo da camada de transporte
  - Protocolo IPSec (AH e/ou ESP)
  - Portos origem e destino
  - Tipo de serviço



# Modos de Operação (1)

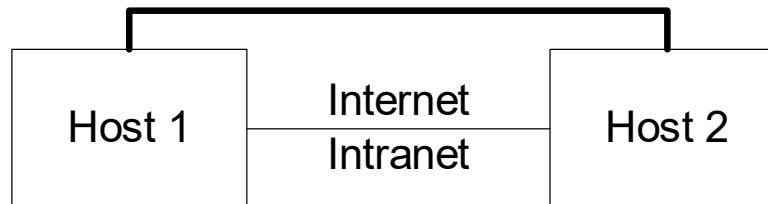
- ◆ Transporte
- ◆ Túnel





# Modos de Operação (2)

- ◆ Ponto a ponto



## Transporte

[IP1][AH][upper]

[IP1][ESP][upper]

[IP1][AH][ESP][upper]

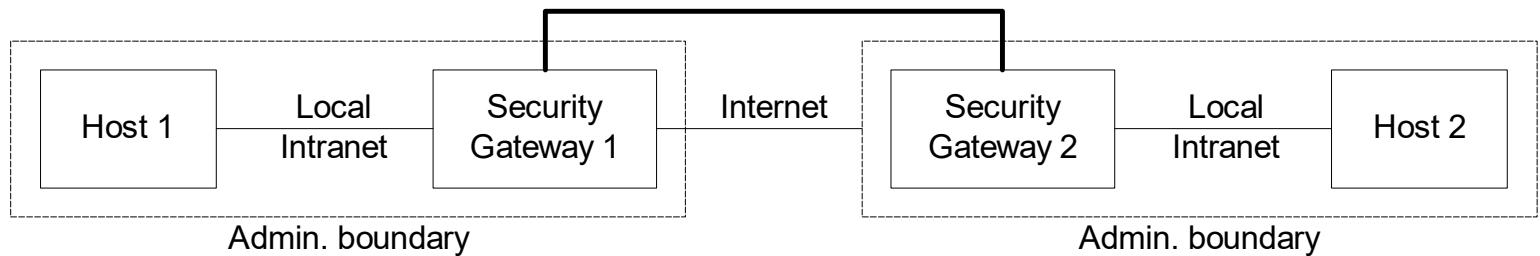
## Túnel

[IP2][AH][IP1][upper]

[IP2][ESP][IP1][upper]

# Modos de Operação (3)

## ♦ VPN



### Transporte

(não pode ser usado)

### Túnel

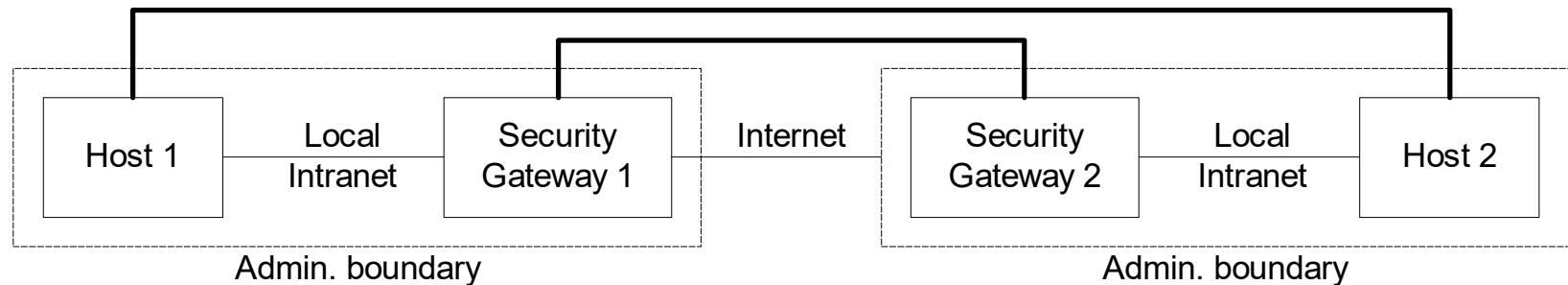
[IP2][AH][IP1][upper]

[IP2][ESP][IP1][upper]



# Modos de Operação (4)

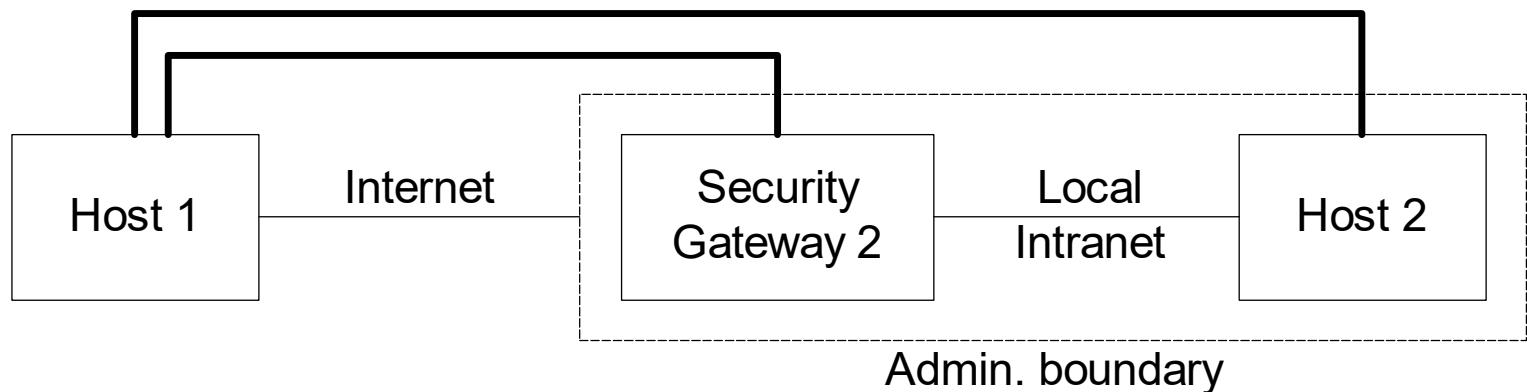
- ◆ Combinação do modo VPN com o modo ponto a ponto





# Modos de Operação (5)

- ◆ Ligação remota via Internet





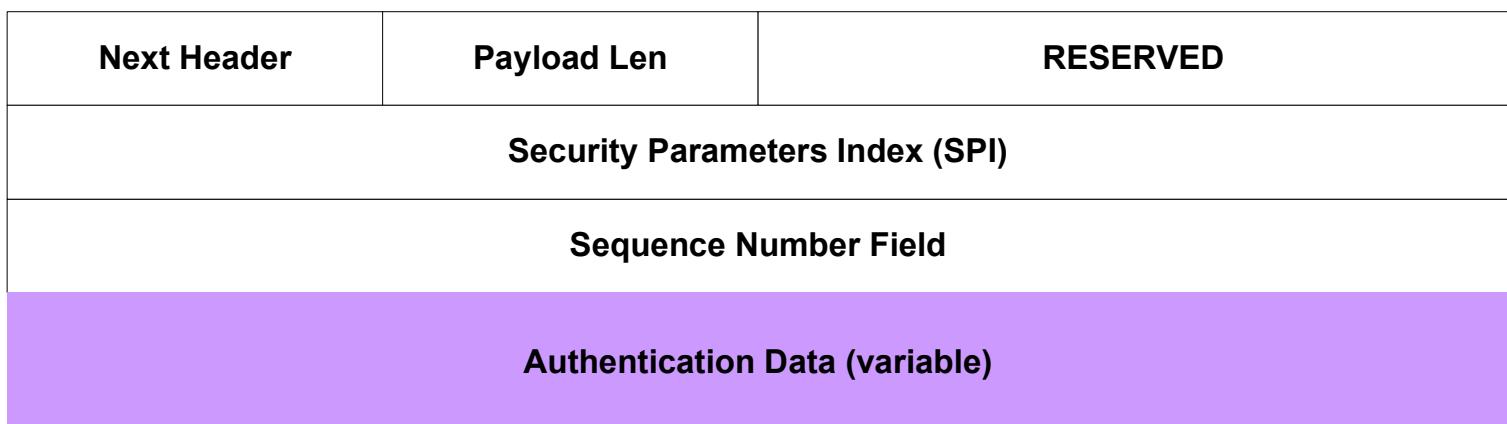
# Authentication Header (1)

- ◆ Definido no RFC 2402
- ◆ Garante a integridade e autenticação dos pacotes IP (dos campos imutáveis e dos mutáveis, mas previsíveis)
- ◆ Não permite a cifragem de pacotes IP
- ◆ Pode ser aplicado sozinho ou em combinação com o ESP
- ◆ Permite o uso de algoritmos existentes
  - HMAC-MD5 (por omissão)
  - HMAC-SHA-1



# Authentication Header (2)

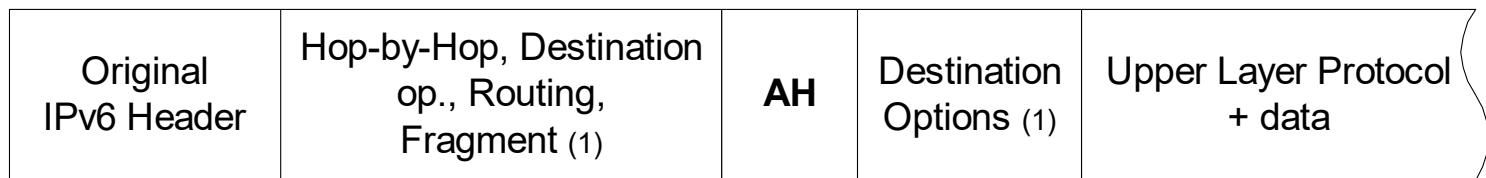
- ◆ Formato do cabeçalho
  - SPI - Security Parameters Index
  - Número de sequência
  - Parâmetros e dados para autenticação



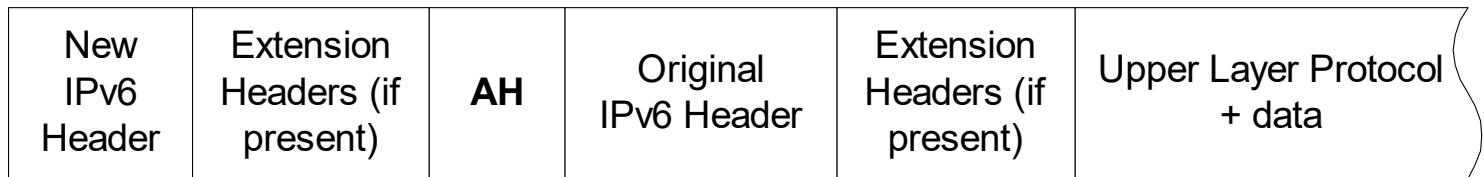


# Authentication Header (3)

- ◆ A cobertura da autenticação AH é superior à cobertura de autenticação do ESP



←→ Autenticado (excepto campos mutáveis)



←→ Autenticado (excepto campos mutáveis)



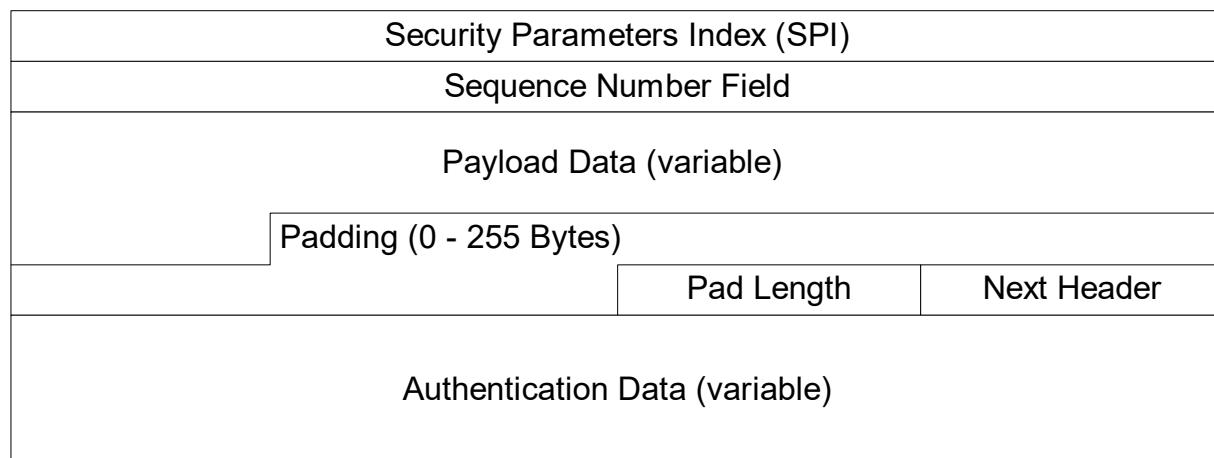
# Encapsulating Security Payload (1)

- ◆ Definido no RFC 2402
- ◆ Garante a integridade dos pacotes IP
- ◆ Permite a cifragem de pacotes IP
- ◆ Suporte de diversos algoritmos:
  - DES (no modo Cypher Block Chaining): por omissão
  - 3-DES
  - MD5 (por omissão) e SHA-1 para autenticação
  - Sem autenticação, ou sem confidencialidade



# Encapsulating Security Payload (2)

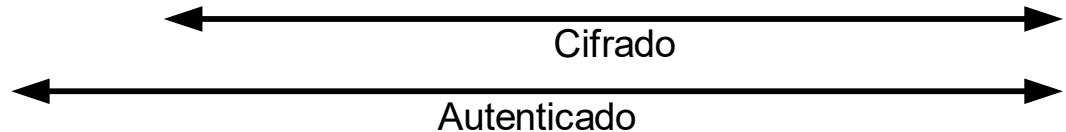
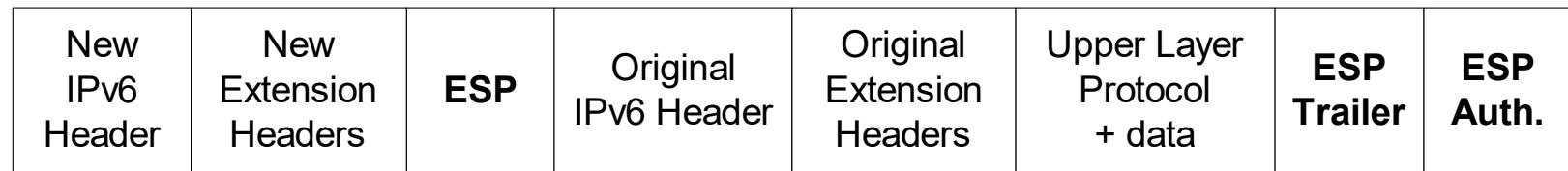
- ◆ Formato do cabeçalho
  - SPI - Security Parameters Index
  - Número de sequência
  - Parâmetros e dados para cifragem
  - Parâmetros e dados para autenticação





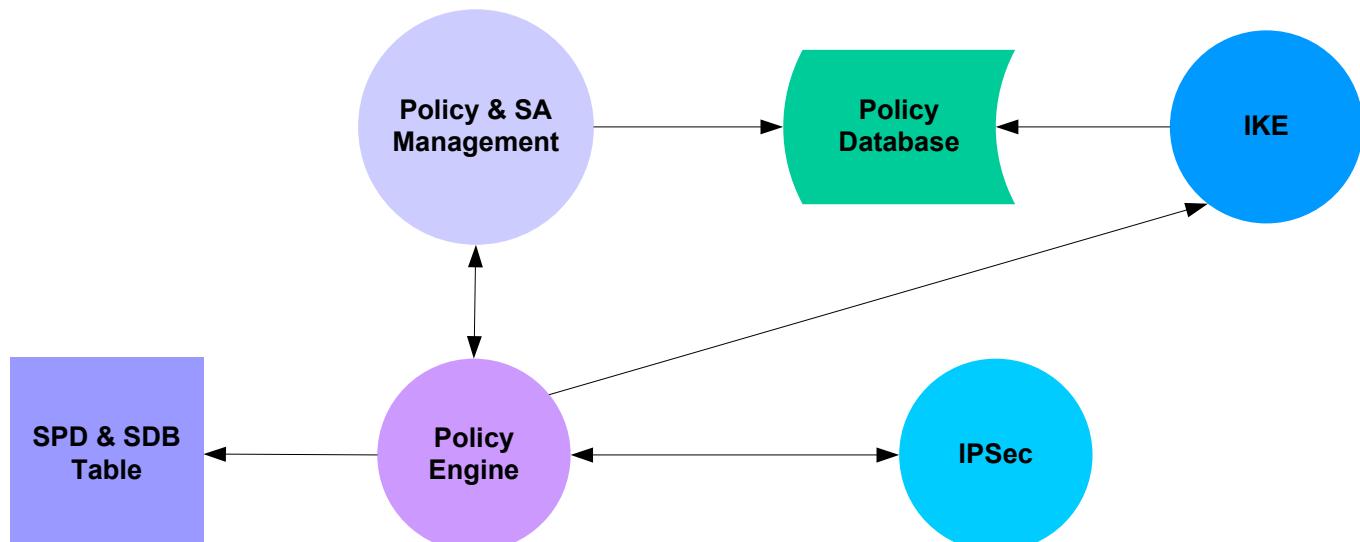
# Encapsulating Security Payload (3)

- ♦ ESP no modo de transporte e túnel



# Implementação do IPSec (1)

- ◆ Protocolos IPSec
- ◆ Security Policy Database (SPD)
- ◆ Security Association Database (SAD)
- ◆ Internet KEY Exchange (IKE)
- ◆ Gestão e implementação da política





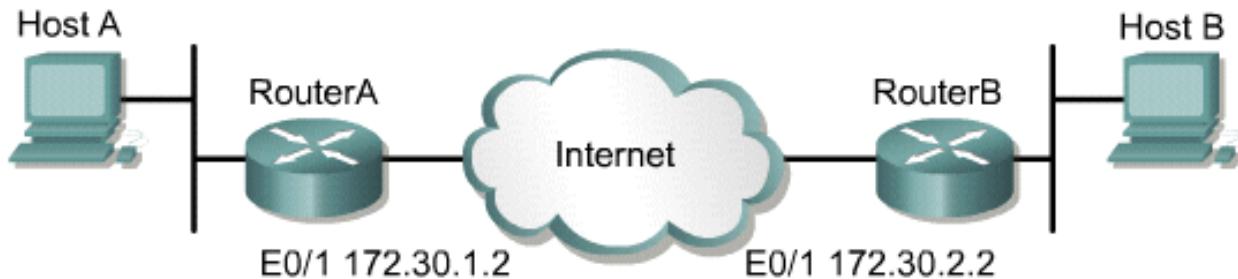
# Implementação do IPSec (2)

## IKE Policy Parameters

Parameter	Strong	Keyword	Default
Message encryption algorithm	DES 3-DES	des 3des	768-bit Diffie-Hellman
Message integrity has algorithm	SHA-1, HMAC variant MD5, HMAC variant	sha md5	86400 seconds, or one day
Peer authentication method	Pre-shared keys RSA encrypted nonces RSA signatures	pre-share rsa-encr rsa-sig	768-bit Diffie-Hellman
Key exchange parameters, Diffie-Hellman group identifier	768-bit Diffie-Hellman or 1024-bit Diffie-Hellman	1 2	768-bit Diffie-Hellman
ISAKMP-established security associations lifetime	Can specify any number of seconds	-	86400 seconds, or one day

# Implementação do IPSec (3)

## Exemplo de uma Política IPSec



Policy	Host A	Host B
Transform set	ESP-DES, Tunnel	ESP-DES, Tunnel
Peer hostname	RouterB	RouterA
Peer IP address	172.30.2.2	172.30.1.2
Hosts to be encrypted	10.0.1.3	10.0.2.3
Traffic (packet) type to be encrypted	TCP	TCP
SA establishment	ipsec-isakmp	ipsec-isakmp



# Gestão de Chaves (1)

- ◊ Manual
  - Utilizada na fase inicial da implementação
  - Introdução manual das chaves nos extremos das ligações IPSec
- ◊ Automática:
  - IKE (*Internet Key Exchange*)
    - RFC 2409
    - Usado para definir Associações de Segurança (SAs) entre entidades
    - Baseado no ISAKM (Internet Security Association and Key Management Protocol)/Oakley do IETF
    - Troca de parâmetros de segurança SPD (*Security Parameters Definition*)
    - Troca de chaves públicas (Diffie-Hellman)
    - Para além do IPSec pode ser utilizado outros domínios dependendo do DOI (*Domain of Interpretation*)
  - Outros
    - SNKI (Sun)
    - Photuris



# Gestão de Chaves (2)

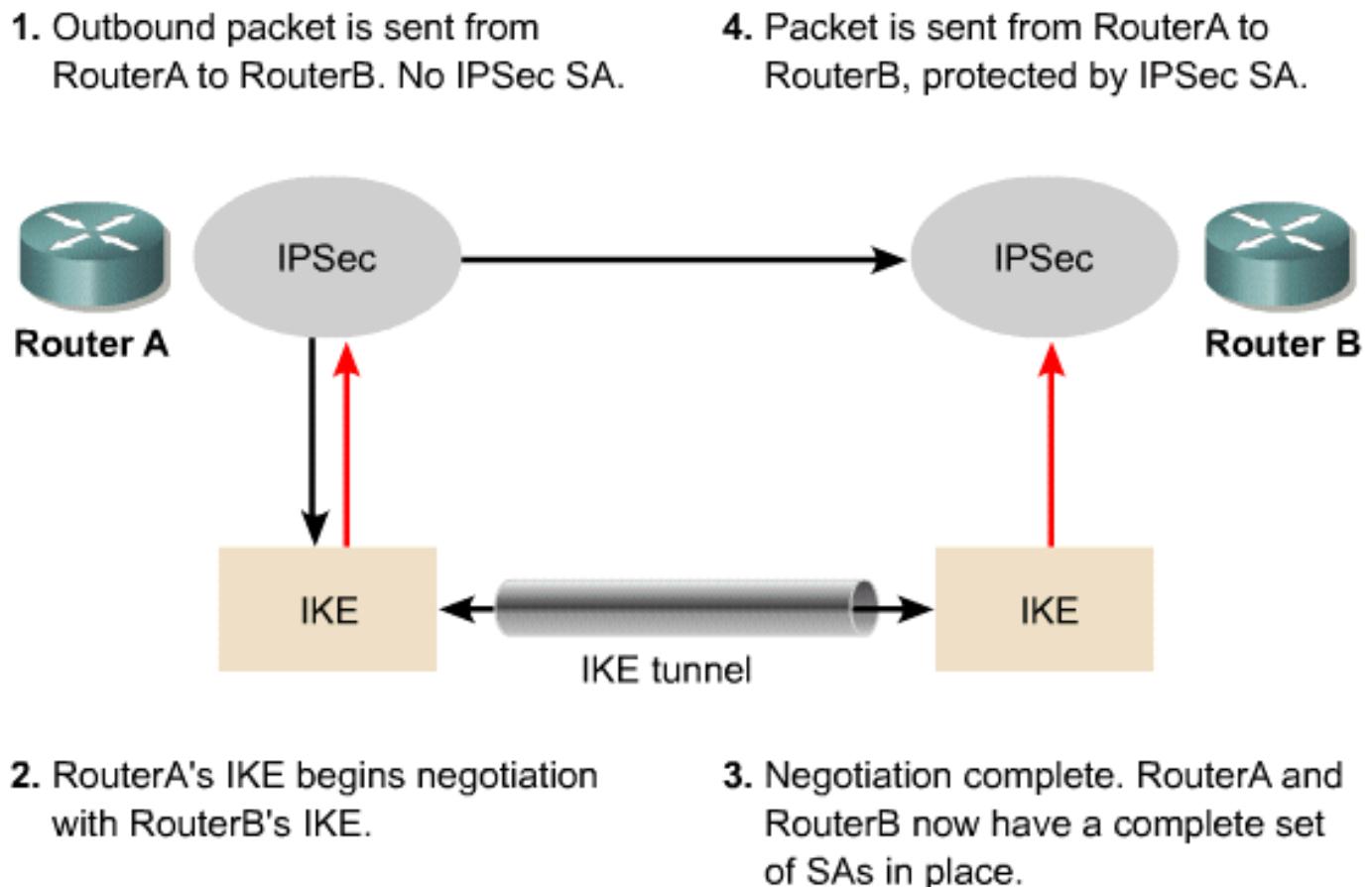
- ◆ ISAKMP/Oakley
  - Oakley Key Determination Protocol
    - Protocolo baseado no algoritmo Diffie-Hellman, mas mais seguro
    - Protocolo genérico que não especifica formatos
  - Internet Security Association and Key Management Protocol
    - Estrutura para a gestão de chaves na Internet
    - Especifica formatos



# Gestão de Chaves (3)

- ◆ Oakley Key Determination Protocol
  - Cada estação envolvida tem uma chave privada e uma pública
  - A chave secreta da sessão é calculada com base na chave privada e na chave pública da estação remota
  - Utilização de um mecanismo de *cookies* para evitar os ataques por entupimento
  - Prevenção de ataques de *replay* através de *nonces*
  - Usa autenticação para prevenir ataques *man-in-the-middle*

# Gestão de Chaves (3)





# Gestão de Chaves (4)

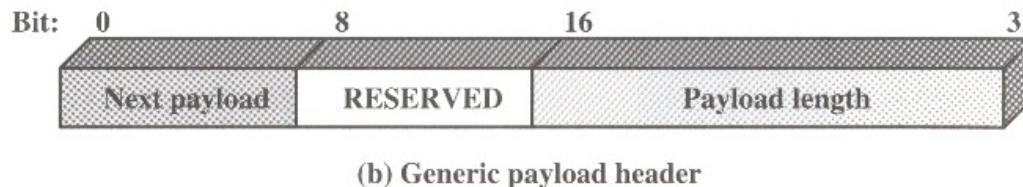
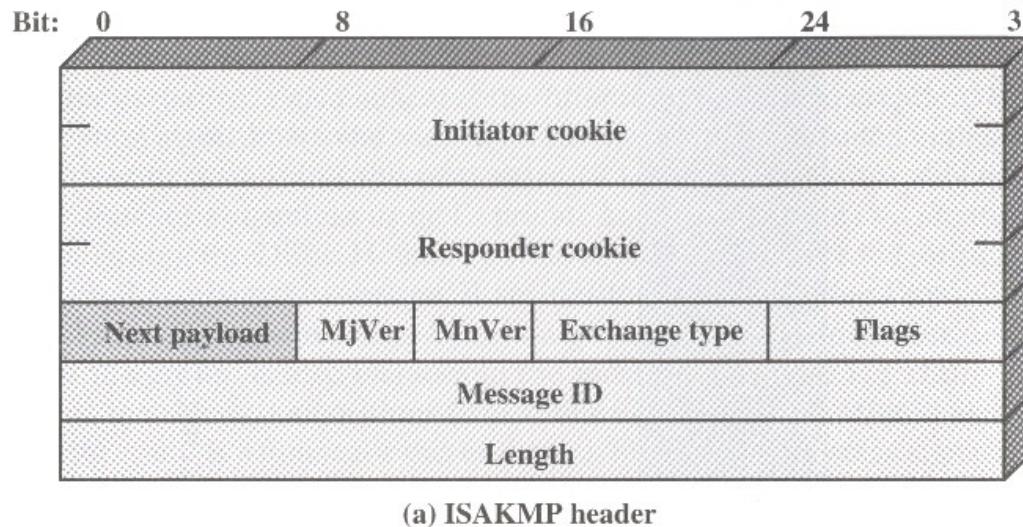
## IKE Policy Parameters

Parameter	Strong	Keyword	Default
Message encryption algorithm	DES 3-DES	des 3des	768-bit Diffie-Hellman
Message integrity has algorithm	SHA-1, HMAC variant MD5, HMAC variant	sha md5	86400 seconds, or one day
Peer authentication method	Pre-shared keys RSA encrypted nonces RSA signatures	pre-share rsa-encr rsa-sig	768-bit Diffie-Hellman
Key exchange parameters, Diffie-Hellman group identifier	768-bit Diffie-Hellman or 1024-bit Diffie-Hellman	1 2	768-bit Diffie-Hellman
ISAKMP-established security associations lifetime	Can specify any number of seconds	-	86400 seconds, or one day

# Gestão de Chaves (5)

## ◆ ISAKMP

- Mensagens trocadas sobre UDP





# Gestão de Chaves (6)

## ◆ Trocas ISAKMP

- *Base Exchange*
  - Troca de chaves e autenticação transmitidas juntas
  - Minimiza as trocas (4 mensagens)
  - Mas não fornece protecção de identidade
- *Identity Protection Exchange*
  - Extensão do *Base Exchange* para dar protecção de identidade (6 mensagens)
- *Authentication Only Exchange*
  - Efectua autenticação mútua sem troca de chaves (3 mensagens)
- *Aggressive Exchange*
  - Minimiza a troca de mensagens (3 mensagens)
- *Informational Exchange*
  - Usado para transportar informação de gestão de SA's

# Políticas de Segurança

# Introdução

1. Potencial de Ataque
2. Nível de Segurança
3. Análise de Risco
4. Política de Segurança

# Motivações para um ataque

- Ataques do interior da organização – representam 70% das ocorrências
  - Intensionais – colaboradores insatisfeitos, ...
  - Acidentais – equipa de limpeza
  - Difíceis de proteger
- Ataques do exterior – representam 30% das ocorrências
  - Concorrência – espionagem industrial, DoS, alteração de conteúdos web (*defacing*), etc
  - Militância – ambientalistas, pacifistas, ...
  - Visibilidade – ataques ao site da NASA ou do FBI
  - Publicidade indesejada – relay de mensagens SPAM

# Potencial de ataque (1)

## □ Classificar de 1 a 5 treze questões:

1. Acesso físico de público ao interior das instalações?
2. Acesso de estranhos à organização aos recursos?
3. Suporte de serviços de comunicação para o público em geral (ex. ISP)?
4. Além da equipa de gestão, mais alguém tem acesso a privilégios de administração?
5. Existe partilhas de contas entre utilizadores ou contas genéricas?
6. A actividade da organização pode ser considerada controversa?
7. A actividade da organização está relacionada com a área financeira?
8. Existem servidores expostos à Internet?
9. São usadas redes públicas (Internet, Frame Relay, RDIS) para dados sensíveis?
10. A actividade da organização é altamente especializada?
11. A organização teve um crescimento muito rápido?
12. A organização tem tido muita visibilidade nos media?
13. Os utilizadores são especialistas em informática?

# Potencial de ataque (2)

- Pontuação
  - <15 pode dormir descansado (mais ou menos)!
  - >15 e <30 potencial de ser alvo de ataque baixo
  - >30 e <40 potencial de ser alvo de ataque médio
  - >40 e <50 potencial de ser alvo de ataque elevado
  - >50 e <60 potencial de ser alvo de ataque muito elevado
  - >60 você está neste momento a ser atacado!

# Nível de Segurança

- Análise de risco
  - Identificar os bens a proteger
  - Identificar as ameaças a esses bens
  - Identificar os custos associados
- Definição de uma política de segurança
  - Objectivos
  - Definição das medidas a implementar
  - Papel dos vários elementos da organização
  - Definição das medidas para impor a política de segurança
  - Relação com legislação em vigor
- Implementação da política de segurança
  - Instalação de mecanismos de segurança
  - Monitorização de segurança
  - Auditorias de segurança

# Análise de Risco (1)

- O que é que necessita de protecção?
  - Recursos físicos (computadores, impressoras, ...)
  - Recursos intelectuais (informação)
  - Tempo (tempo de reparação, tempo de indisponibilidade)
- De quem é necessário garantir protecção?
  - Rede local
  - Redes remotas de outros ofícios
  - Redes de clientes ou fornecedores
  - Internet
  - Acesso comutado através de modems ou RDIS

# Análise de Risco (2)

- Quem é que pode estar interessado em atacar os recursos informáticos?
  - Colaboradores
  - Concorrência
  - ...
- Qual é a probabilidade de uma tentativa ser bem sucedida?
  - Quais os acessos ao exterior existentes?
  - Quais os mecanismos de autenticação existentes?
  - Quais os mecanismos de firewall existentes?
  - Que ganhos pode ter o atacante?

# Análise de Risco (3)

- Quais são os custos imediatos de uma intrusão?
  - Custos de reparação
  - Custos de produtividade
  - Implicações na vida humana (ex. Hospital)
- Quais são os custos de recuperar de um ataque?
  - Custos de recuperação de sistemas
  - Custos de recuperação de informação
  - Custos de negação de serviços (DoS)
  - Custos devidos a acessos não autorizados e não detectados a informação

# Análise de Risco (4)

- Como garantir a protecção a custos controlados?
  - Que nível de protecção é necessário?
  - Instalar firewall?
  - Contratar perito em segurança?
  - Quais os custos de produtividade dos mecanismos a instalar?
  - Não sobredimensionar as soluções (*overkill*)!
  - O custo de garantir protecção deve ser inferior ao custo de recuperação
- Existe alguma legislação que regulamente as medidas de segurança a adoptar?
  - É obrigatória a instalação de mecanismos de segurança (ex bancos)?
  - É proibido a utilização de algum mecanismo de segurança (ex cifragem)?

# Política de Segurança (1)

- Introdução
  - Destinatários
    - Administradores de redes informáticas
    - Gestores (*decision makers*)
    - Utilizadores
  - Compromissos
    - Serviços oferecidos – segurança proporcionada
    - Facilidade de utilização – segurança
    - Custo da segurança – risco de perdas

# Política de Segurança (2)

- Plano de Segurança (1)
  - Identificar o que se quer proteger
  - Determinar do que é que (ou quem) se está a proteger
  - Determinar o risco - quanto provável é a ameaça
  - Implementar medidas de protecção tendo em conta a relação custo-eficácia
  - Rever e aperfeiçoar o processo quando existem pontos fracos

# Política de Segurança (3)

- Plano de Segurança (2)

**“O custo da protecção deve ser menor  
do que o custo do restabelecimento  
se uma ameaça se concretizar”**

# Política de Segurança (4)

- O que é?
  - É uma declaração formal das regras pelas quais as pessoas a quem é dado acesso ao **activo tecnológico e à informação** duma organização se devem reger

# Política de Segurança (5)

- Porque deve existir?
  - Para informar as pessoas acerca dos requisitos obrigatórios para proteger o activo tecnológico e informação da organização
  - Para especificar os mecanismos pelos quais se podem atingir os requisitos de segurança
  - Para estabelecer uma base a partir da qual se pode configurar, auditar e adquirir sistemas informáticos para atingir os requisitos de segurança

# Política de Segurança (6)

- Quem deve ser envolvido na definição da Política de Segurança?
  - Administrador(es) da rede
  - Responsáveis pela gestão da organização
  - Representantes dos utilizadores afectados pela Política de Segurança
  - Conselheiro legal (se apropriado)

# Política de Segurança (7)

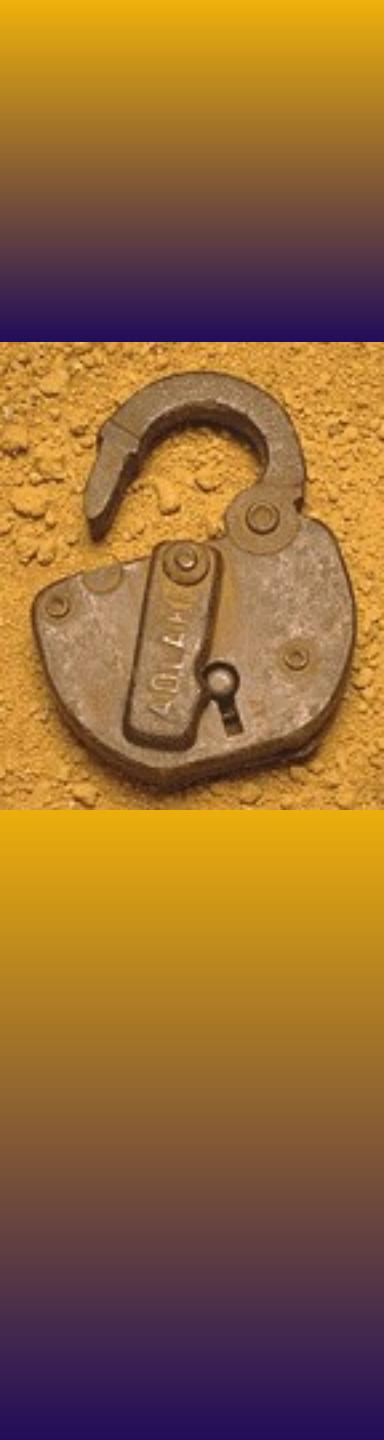
- Características
  - Deve definir claramente as responsabilidades dos utilizadores, administradores e gestores
  - Deve ser executável com ferramentas de segurança e prever sanções onde a prevenção não é tecnicamente viável
  - Deve ser implementada através dos procedimentos de administração dos sistemas

# Política de Segurança (8)

- Componentes
  - Guia de aquisição de material informático
  - Política de privacidade
  - Política de acesso aos recursos
  - Política de responsabilidade
  - Política de autenticação
  - Disponibilidade dos recursos
  - Política de manutenção
  - Procedimentos em caso de transgressão
  - Informação de apoio

# Política de Segurança (9)

- Flexibilidade
  - Independência do hardware
  - Independência do software
  - Mecanismos de actualização da Política de Segurança
  - Contemplar excepções
  - Partilha de informação
- Para saber mais:
  - RFC 2196 “Site Security Handbook”



# Políticas de Segurança

Exemplo prático



# Cenário (1)

- ◆ Empresa fictícia FazSoftware
  - Produz diversas aplicações para computadores pessoais
  - A sede da empresa é em Lisboa e tem uma delegação comercial em Leiria. A sede tem um acesso à Internet via ADSL de 2 Mbps e a delegação de 1 Mbps.
  - Existem mais de 200 colaboradores que trabalham fora das instalações da empresa, 50% são programadores
  - A empresa permite que os seus colaboradores trabalhem um dia por semana a partir de casa



# Cenário (2)

## ♦ Empresa fictícia FazSoftware

- Os vendedores passam a maior parte do tempo em viagens entre clientes e potenciais clientes. Por isso a FazSoftware implementou duas solução de acesso remoto:
  - Uma através de dial-in
  - Outra através da Internet via VPN's
- A informação que entra através dos acessos remotos é considerada sensível
  - Os programadores trabalham sobre a última versão duma aplicação. Por isso a FazSoftware pode perder a sua vantagem competitiva se o código cair nas mãos da concorrência
  - A informação das vendas é altamente sensível, porque pode dar pistas aos concorrentes sobre os novos produtos.



# Política de segurança

## ◆ Sumário

- Abrangência
- Gestão da infra-estrutura
- Requisitos das passwords
- Prevenção de vírus
- Cópias de segurança
- Acessos remotos
- Acesso à Internet
- Privacidade e registos
- Informação adicional



# Abrangência (1)

- ◆ Definir as políticas sobre o uso correcto da infra-estrutura informática. Esta representa um investimento com vista ao aumento da produtividade e da eficiência.
- ◆ São considerados componentes da infra-estrutura informática :
  - Toda a cablagem de suporte para dados e voz
  - Todos os equipamentos usados para o controlo do fluxo de dados e voz
  - Todo o software
  - Todos os dispositivos de entrada e saída como as impressoras, fax's e digitalizadores
  - Todos os componentes dos computadores, tais como: monitores, caixas, dispositivos de armazenamento, modems, placas de rede, memórias, teclados, ratos e fios



## Abrangência (2)

- ◆ As acções não conformes com esta política de segurança serão alvo de acção disciplinar, sendo tratadas caso a caso
- ◆ A empresa reserva-se o direito de proceder a acções legais sempre que a lei vigente no país seja infringida ou quando ocorrerem prejuízos financeiros.



# Gestão da infra-estrutura (1)

- ◆ Toda a manutenção, incluindo alterações de configuração nos computadores, deverão ser efectuadas apenas pelo pessoal da equipa técnica.
- ◆ Os colaboradores que não façam parte da equipa técnica não estão autorizados a efectuar qualquer modificação, mesmo aos computadores que lhe foram atribuídos pela empresa para desempenharem as suas funções



# Gestão da infra-estrutura (2)

- ◆ As seguintes acções são consideradas modificações ao sistema:
  - Mudar a ligação de rede para outra tomada
  - Usar um dispositivo que permita efectuar o arranque de um sistema operativo alternativo (ex. disquetes ou cd-rom's)
  - Remover a tampa da caixa do computador
  - Instalar qualquer software, incluindo o software descarregado da Internet



# Gestão da infra-estrutura (3)

- ◆ A gestão do hardware é restringida para:
  - evitar que as garantias dos fabricantes não fiquem inadvertidamente inutilizadas
  - que as precauções de segurança não são contornadas
- ◆ A instalação de software é restringida para garantir que:
  - a empresa cumpre todos os requisitos legais respeitantes a licenças
  - existe suporte adequado, pela equipa técnica da empresa, a todo o software existente
  - não existem incompatibilidades de software



# Requisitos das passwords (1)

- ◆ A cada utilizador será atribuído um username ao qual estará associada uma password.
- ◆ A password garante que apenas os utilizadores autorizados acedem aos recursos da rede.
- ◆ É da responsabilidade de cada utilizador garantir que a sua password permanece secreta



# Requisitos das passwords (2)

- ◆ Regras:
  - As passwords têm de ter pelo menos 8 caracteres alfanuméricos
  - As passwords não podem consistir em palavras, ou variantes, do nome do utilizador, do username, do nome do servidor ou da empresa
  - Os utilizadores são obrigados a mudar de password de 60 em 60 dias. Caso isso não aconteça a conta do utilizador será bloqueada. Para reactivar a conta o superior do utilizador terá de efectuar um pedido à equipa técnica.



# Requisitos das passwords (3)

- ◆ Regras:
  - Durante a autenticação o utilizador tem 3 tentativas para introduzir a password correctamente. Se todas falharem a conta será automaticamente bloqueada. Para reactivar a conta o superior do utilizador terá de efectuar um pedido à equipa técnica.
  - Todos os computadores da empresa devem ter um screen saver que é activado ao fim de 15 minutos de inactividade. Depois de activado o sistema deve pedir novamente a autenticação do utilizador antes de conceder acesso.



# Requisitos das passwords (4)

- ◆ Regras:
  - Para os acessos remotos (por dial-in ou por VPN's) os utilizadores têm de utilizar o token de segurança que lhes foi atribuído. Este irá gerar automaticamente uma password nova a cada 60 segundos.
  - As passwords são privadas e intransmissíveis. Espera-se que o colaborador não escreva a sua password num papel nem a partilhe com outras pessoas. A única excepção será quando tal for pedido pela equipa técnica na presença do seu superior directo.
  - As passwords de acesso remoto têm de ser diferentes das usadas internamente.



# Requisitos das passwords (5)

- ◆ Regras:
  - A empresa reserva-se o direito de imputar responsabilidades por danos causados pelo facto do colaborador não preservar a confidencialidade da sua password, de acordo com as regras estabelecidas.
- ◆ Uma política de passwords forte visa garantir a segurança de todos os recursos



# Prevenção de vírus

- ◆ Todos os recursos informáticos devem ser protegidos por software antivírus
- ◆ É da responsabilidade dos colaboradores verificarem que o antivírus não está desligado
- ◆ Se um colaborador receber um aviso do software antivírus deverá cessar imediatamente a sua actividade e contactar a equipa técnica.
- ◆ É da responsabilidade da equipa técnica fazer a actualização do software antivírus. Esta será efectuada através de um mecanismo automático.



# Cópias de segurança

- ◆ Uma vez por semana a equipa técnica fará uma cópia de segurança dos documentos guardados no computador de cada colaborador
- ◆ Será atribuído um dia da semana a cada colaborador para ser efectuada a cópia de segurança.
- ◆ Só serão feitas cópias de segurança dos documentos que estiverem dentro da directoria:
  - c:\My Documents
- ◆ É da responsabilidade do colaborador garantir que todos os ficheiros importantes são guardados na referida directória
- ◆ As aplicações devem estar configuradas, por omissão, para guardar os seus documentos na referida directória



# Acessos remotos (1)

- ◆ Existem apenas dois métodos de acesso remoto permitidos pela empresa: dial-in e VPN's
- ◆ A ligação de modems à linha telefónica, nos computadores atribuídos aos colaboradores, é expressamente proibida e poderá servir como despedimento com justa causa
- ◆ O acesso remoto é concedido consoantes as necessidades. Assim o colaborador que queira obter permissões de acesso remoto terá que fazer um pedido ao seu superior directo que terá de dar o seu parecer e reencaminhar o pedido para a equipa técnica



# Acessos remotos (2)

- ◆ Para os acessos remotos serão dados aos colaboradores:
  - Um token de segurança
  - Uma lista com os números de telefone dos modems disponibilizados
  - O software apropriado para as ligações VPN através da internet
  - Um guia sobre o acesso remoto aos recursos da rede da empresa



# Acessos remotos (3)

- ◆ A empresa não é responsável pelo suporte do sistema que o colaborador usará para efectuar o acesso remoto.
- ◆ O colaborador ao aceitar o software será responsável, no seu sistema, pelas actualizações necessárias para obter acesso remoto, nomeadamente:
  - Uma linha telefónica
  - Um modem
  - Um processador mais rápido
  - Memória adicional
  - Etc.



# Acessos remotos (4)

- ◆ A empresa é responsável pelo suporte técnico apenas na rede interna e respectivo perímetro.
- ◆ A resolução de todos os problemas de ligação, fora do âmbito referido, são da responsabilidade do utilizador
- ◆ O colaborador terá de assinar um termo de responsabilidade onde se compromete a manter a confidencialidade dos dados de acesso remoto, incluído o software.
- ◆ A não verificação do ponto anterior poderá dar origem a um processo de despedimento com justa causa.



# Acesso à Internet (1)

- ◆ Os recursos da empresa, incluindo os que são usados para o acesso à Internet, são para o uso decorrente das actividades laborais. Esta política visa a utilização correcta dos recursos da empresa e aplica-se de equitativamente a todos os colaboradores.
- ◆ Os superiores directos dos colaboradores podem autorizar excepções desde que cumpram os seguintes requisitos:
  - O uso pretendido seja ocasional



# Acesso à Internet (2)

## ◆ (continuação)

- Não interfira com as tarefas habituais do colaborador
- Serve os interesses legítimos da empresa
- Tem fins educativos dentro do âmbito das funções do colaborador
- Não infrinja as leis nacionais
- Não sobrecarregue a rede



# Acesso à Internet (3)

## ◆ Navegar na Internet

- É obrigatório o uso de um browser que permita as seguintes configurações/requisitos:
  - Não conter plugins adicionais, além da versão base
  - Desligar a execução de Java e Java Scripting
  - Desligar os a aceitação de cookies
- Estes requisitos visam garantir que os colaboradores não executem, inadvertidamente, código malicioso.
- O não cumprimento destes requisitos resultará na perca de privilégios de navegação na internet
- A instalação de browser deverá ser efectuada apenas pela equipa técnica



# Acesso à Internet (4)

## ◆ E-mails

- A recepção e envio de e-mails está limitado a mensagens com um tamanho igual ou inferior a 10 MB
- Sempre que for necessário efectuar transferências de ficheiros de tamanho superior deverá contactar a equipa técnica para usar o servidor de FTP
- Todas as mensagens transmitidas para listas de e-mail devem conter a seguinte informação como parte integrante da mensagem:
  - As opiniões expressas nesta mensagem não reflectem a posição do meu empregador
- A empresa reserva-se o direito de impedir a transmissão das mensagens que não estejam conforme o ponto anterior
- Os recursos informáticos da empresa não podem ser usados para aceder a contas de e-mail pessoais baseadas em servidores na Internet



# Privacidade e registos

- ◆ Todos os recursos informáticos são propriedade exclusiva da empresa, incluindo: e-mails, ficheiros armazenados, transmissões de dados, etc
- ◆ A empresa reserva-se o direito de monitorizar e registar toda a actividade existente na rede informática
- ◆ O colaborador é responsável pela entrega de todas as suas passwords, ficheiros, ou outros recursos, se tal lhe for pedido pelo seu superior directo.



# Informação adicional

- ◆ Todas as dúvidas ou omissões que existam relativamente a este documento devem ser colocadas ao superior directo do colaborador
- ◆ Os superiores directos dos colaboradores têm a responsabilidade de encaminhar as questões para o departamento mais apropriado da empresa



# Bibliografia

- ◆ Este exemplo é uma adaptação da política de segurança existente no anexo B do livro de Chris Brenton “Mastering network security” da Sybex.