Introduction to Security and to Cryptography

3º MIEIC

Pedro F. Souto (pfs@fe.up.pt)

March 11, 2021

Roadmap

Computer Security/Cybersecurity Incidents

Ransoms

Web page defacing usually for political reasons

Online Banking Credentials Theft

Credit Card Theft and also of personal information

Denial-of-Service often tied to ransoms or political statements

Industrial/military sabotage e.g. Stuxnet

Intellectual Property Theft

Misinformation campaigns using fake news, usual for political gains, both nationally and internationally

Check

- Wikipedia's List of Security Hacking Incidents, for a list of high-profile incidents
- List of Significant Cyber Events Since 2006 by the Center for Strategic & International Studies, a USA Think Tank

Computer Security: A Definition

Security in a Computational System: *"deals with the prevention and detection of unauthorised actions by users of a computer system"*, Dieter Gollmann in Computer Security, John Wiley & Sons, 1999

- Need to specify which actions are authorized to each user (the remaining actions are unauthorized)
  - In other words, we need to specify a **security policy**, i.e. the security requirements
- Authorization requires ***authentication*** and ***access control***.
- To prevent unauthorized actions it is not always possible or may not make economic sense, in this case we will need to content ourselves with the **detection** of these actions

Security: Other definitions

- Often, security is defined in terms of ensuring:

  Confidentiality: that is, prevent unauthorized access to computer-related assets;

  Integrity: that is, prevent unauthorized modification of computer-related assets;

  Availability: that is, prevent that authorized access to computer-related assets be denied

  This is often called the **CIA triad**
- Like in Dieter Gollmann's definition, it is clear that to ensure security it is crucial to define what is authorized.

Security Process

- There are no systems 100% secure.
  - Even if this is technically possible, its economic costs may not be justifiable;
- Implementing security requires a ***risk analysis***, formal or not. This allows to identify:
  - The assets that we need to protect;
  - The threats to these assets.
- The outcome of this analysis is the specification of a security policy, i.e. of the security requirements
- To implement a security policiy, we use security mechanisms
- To verify the conformity of the implementation with the security policy, one needs to audit and to monitor system operation, usually with the help of logs.

Security Threats

Definition (ISO 27005) A potential cause of an incident, that may result in harm of systems and organization

- Internal vs. external;
- Passive vs. active;
- Or also with respect to the consequence:

  Interception e.g. snooping the communication between 2 entities;

  Interruption e.g. deny access to a Web service, via a denial of service attack

  Modification e.g. changing the contents of a message of a DB record;

  Fabrication e.g. add a *password* to an account (that should have none).
- To mitigate the consequences of a threat, so as to compy with the security policy (requirements), we need to use **security mechanisms**

Security Design

- Security cannot be implemented by adding one layer at the end of a design
  - At that time, decisions previously made may seriously restrict the options
- Some design aspects that we need to consider are:

Layer
: in which layer of the computational system (e.g. network, OS, application) are security mechanisms implemented?

Complexity vs. Simplicity
: shall the system have lots of functionality or is it more important to ensure high reliability?

> *The unavoidable cost of reliability is simplicity.*
>
> Anthony Hoare

Centralization vs. Decentralization
: on which components does the system's security depends? I.e. what is the system's ***Trusted Computing Base (TCB)***?

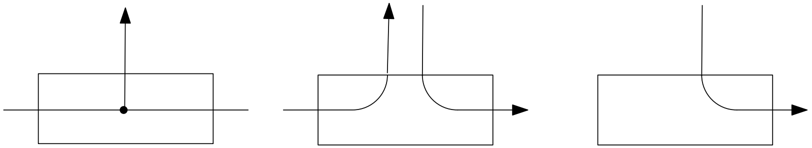
Further Reading

- Watch this great introduction to computer security of an MIT lecture by Prof. Nickolai Zeldovich
  - And, if you have time, explore the the whole course

Roadmap

Criptography

- Is one of the most used security mechanisms in distributed systems
  - Allows to protect the communication among principals against different threats:



Cryptographic Primitives

1. Encryption/Decryption algorithms
2. Cryptographic Hash Functions
3. Digital Signature Algorithms

Fundamental Principle Algorithms should be public. The security is provided by parametrizing the algorithms with ***keys***.

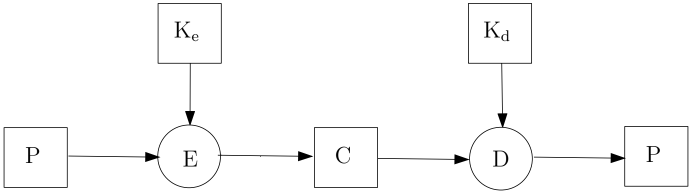
Cryptographic Systems Two:

Symmetrical (or of shared key) use a single key that is ***shared*** (K)

Asymmetrical (or of public key) use two keys one of which is ***public*** (K^+) and the other is ***private*** (K^-).

Roadmap

Encryption/Decryption Algorithms



Symmetrical, or with shared key: in this case, the keys for encrypting and decrypting are the same:

$$K_e = K_d = K$$

- The key is shared among all principals authorized to access information
- The key must be known to those principals only

Asymmetric, or with public key: in this case the encryption and the decryption keys are different:

$$K_e \neq K_d$$

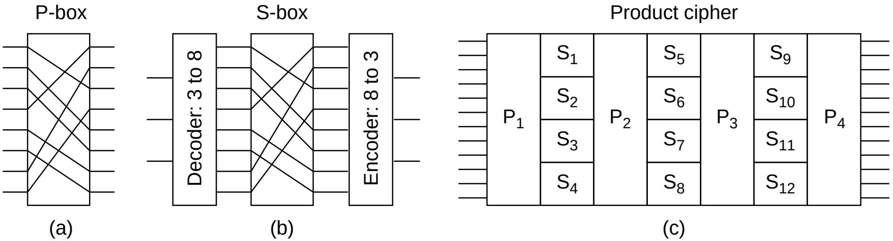
- One of these is public and the other private. Which is which?

Encryption with Shared Key: DES (1/3)

- *Data Encryption Standard (DES)* was a USA encryption standard, considered vulnerable since the mid 90's:
  - It was defeated by Moore's law, as its designers predicted
- The algorithm is relatively simple. It is based on the repeated application of 2 basic operations on bit blocks:

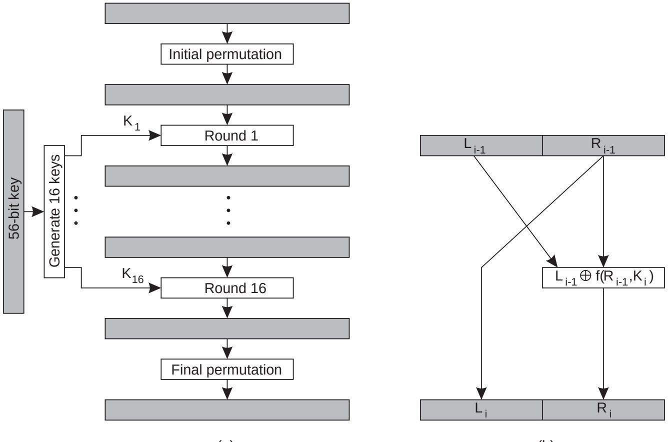
  Permutation of bits in a block;

  Substitution of 6-bit sub-blocks with 4-bit sub-blocks.



Encryption with Shared Key: DES (2/3)

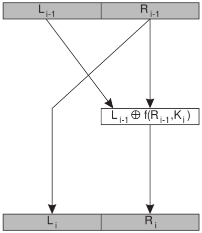
- The basic algorithm operates on 64-bit blocks that are transformed in blocks with the same length;
- The encryption of a block takes 16 rounds.
    - Each round uses a different 48-bit key that is generated from the 56-bit master key



(a) (b)

Encryption with Shared Key: DES (3/3)

- The final permutation is the reverse of the initial permutation
- The real work is performed by a *(mangler function) (f)*



1. Expands R_{i-1} to a 48-bit block;
2. Computes the XOR of the result with the round's key, K_i;
3. Breaks the result in eight 6-bit sub-blocks;
4. Each sub-block is processed by a different substitution function that converts a 6-bit block into a 4-bit block
5. The eight 4-bit sub-blocks are combined into a single 32-bit block which is permutated

- The same algorithm is used for both encryption and decryption
- DES was replaced by AES as a US standard in 2001

Public Key Encryption: RSA (1/3)

- RSA is based on the following property of modular arithmetic:
  - Let p and q be two prime numbers;
  - Let $n = p.q$ and $z = (p-1)(q-1)$
  - Let d and e two numbers that $d.e = 1 \; mod \; z$
  - Then for any x $(0 \leq x < n)$:

    $$x^{d.e} = x \; mod \; n$$

Public Key Encryption: RSA (2/3)

- To encrypt a message:
  1. Split it in blocks of a fixed pre-determined length, such that each block m_i, interpreted as a binary number, be less than n
  2. For each block compute:

     $$c_i = m_i^e \ mod\ n$$
- To decrypt an encrypted message;
  1. Split the received (encrypted) message in fixed-length blocks;
  2. Compute:

     $$m_i = c_i^d \ mod\ n$$
- So, to ensure confidentiality with RSA:
  - The encryption key, $K_e = (e, n)$, must be public;
  - The decryption key, $K_d = (d, n)$, must be secret;

Public Key Encryption: RSA (3/3)

- How to compute the two keys?
  1. Pick p e q, 2 very large prime numbers, e.g. $> 10^{100}$;
  2. Compute $n = pq$ e $z = (p-1)(q-1)$
  3. Select value e (surprisingly, or may be not, it can be small)
  4. Use Euclides algorithm to compute d:

     $$ed = 1 \; mod \; z$$
- The security of RSA relies on the dificulty of factoring a very large number (n)
  - NIST recommends 2048-bit keys
    - 3072-bit keys if security is required beyond 2030

Cipher Block Modes of Operation (1/3)

Observation The majority of the encryption algorithms encrypt fixed-size blocks (e.g. 64-bit blocks in the case of DES)

- For this reason they are known as ***block ciphers***
- ***Stream ciphers*** are another class of encryption algorithms that operate directly on sequences of bytes with an arbitrary length

Problem How can we encrypt data/messages whose length is larger than that of a block?

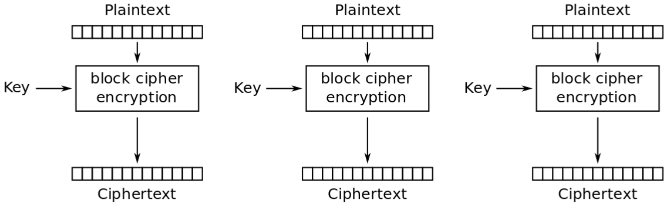
Solution Just:

1. Use **padding** so that the length of the data to encrypt is a multiple of the length of a data block
2. Split the data/message in blocks and encrypt each of the blocks

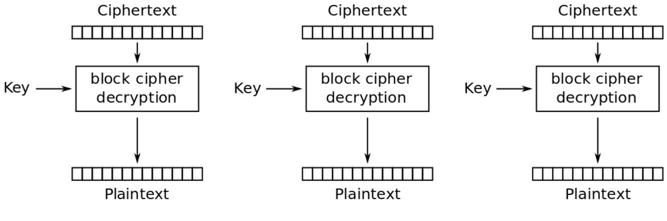
The last step can be carried out in different ways that are known as ***cipher block modes of operation***

Cipher Block Modes of Operation (2/3)

Electronic Code Book (ECB)



Electronic Codebook (ECB) mode encryption



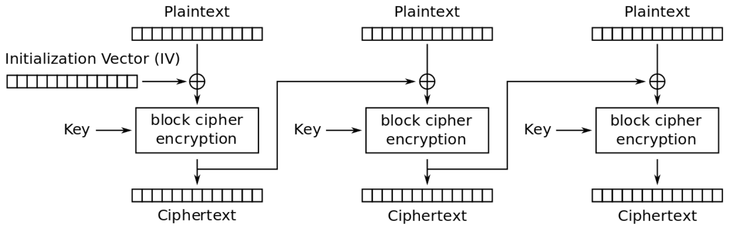
Electronic Codebook (ECB) mode decryption

Problem Facilitates cryptanalysis

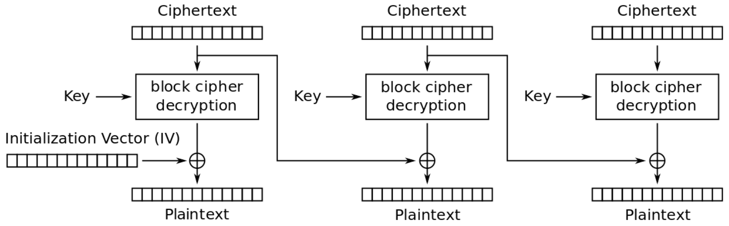
- Identical data blocks are encrypted in identical cyphered-blocks

Cipher Block Modes of Operation (3/3)

Cipher Block Chaining (CBC)



Cipher Block Chaining (CBC) mode encryption



Cipher Block Chaining (CBC) mode decryption

- The ***Initialization Vector*** is a (pseudo-)random number

Cryptographic Hash Functions

- Are used to check data integrity, among many other things
  - Someone has called them the cryptography's work horse
- Poperties a cryptographic hash function, $h()$, should have:

Compression maps an input value of arbitrary length into a fixed-length hash-value;

Ease of computation

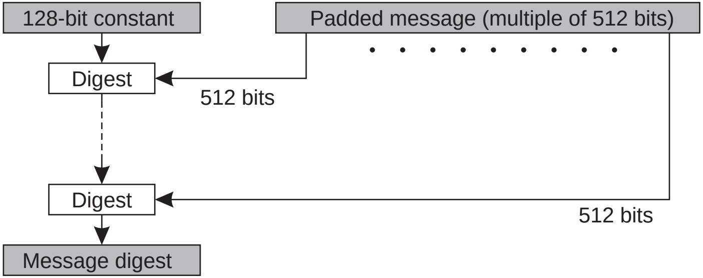
Non reversible, i.e. (*One-way*) given a hash value, y, it is computationally infeasible to find a value x such that $y = h(x)$

Weakly Collision Resistant given a value x is computationally infeasible to find a different value x' such that $h(x) = h(x')$

Strongly Collision Resistant it is computationally infeasible to find two different values x and x' such that $h(x) = h(x')$

Cryptographic Hash Functions: MD5

- The algorithm is executed in *k* stages, where *k* is the number of 512-bit blocks;
  - If necessary, the input is padded so that it length is a multiple of 512 bits.
- Each stage takes as input a 128-bit number and a 512-bit block and its output is a 128-bit number



- Each stage makes 4 passes over message block

MD5: First pass

- Each 512-bit block is split into sixteen 32-bit blocks$(b_0, b_1, \ldots, b_{15})$
- The operations performed on the first pass are:

Iterations 1-8	Iterations 9-16
$p \leftarrow (p + F(q,r,s) + b_0 + C_1) \lll 7$	$p \leftarrow (p + F(q,r,s) + b_8 + C_9) \lll 7$
$s \leftarrow (s + F(p,q,r) + b_1 + C_2) \lll 12$	$s \leftarrow (s + F(p,q,r) + b_9 + C_{10}) \lll 12$
$r \leftarrow (r + F(s,p,q) + b_2 + C_3) \lll 17$	$r \leftarrow (r + F(s,p,q) + b_{10} + C_{11}) \lll 17$
$q \leftarrow (q + F(r,s,p) + b_3 + C_3) \lll 22$	$q \leftarrow (q + F(r,s,p) + b_{11} + C_{12}) \lll 22$
$p \leftarrow (p + F(q,r,s) + b_4 + C_5) \lll 7$	$p \leftarrow (p + F(q,r,s) + b_{12} + C_{13}) \lll 7$
$s \leftarrow (s + F(p,q,r) + b_5 + C_6) \lll 12$	$s \leftarrow (s + F(p,q,r) + b_{13} + C_{14}) \lll 12$
$r \leftarrow (r + F(s,p,q) + b_6 + C_7) \lll 17$	$r \leftarrow (r + F(s,p,q) + b_{14} + C_{15}) \lll 17$
$q \leftarrow (q + F(r,s,p) + b_7 + C_8) \lll 22$	$q \leftarrow (q + F(r,s,p) + b_{15} + C_{16}) \lll 22$

  - p, q, r and s are 32-bit variables, which move from one pass to the next
    - In the first stage, p, q, r, s are initialized to pre-defined values
  - *F* is *F(x,y,z) = (x AND y) XOR ((NOT x) AND z)*;
    - Each of the other 3 passes uses similar functions G, H, I
  - The C_i are 32-bit constants
    - Each pass uses its own set of 16 constants, so there are 64 constants C_1 a C_{64}

Authentication with Hash Functions

- By "adding" a key to the message/data hash functions can be used to authenticate the sender and to check message integrity
  - In this case, the hash value, and also the hash function, are known as **message authentication code (MAC)**.
- In this case the hash function must satifsfy the additional property:

  Computational resistance for any unknown key, k, given the values $(x, h(k, x))$ it is computationally infeasible to compute $h(k, y)$ for a different value y

  Why?
- HMAC (RFC) is a MAC that ensures the same security as the hash function used:
  - MD5 is considered unsafe since 2004
- The key must be shared by all communicating ends
  - A MAC is not the same as a digital signature

Digital Signatures

- A digital signature should:
  1. Identify its author
  2. Be verifiable by others
- On a point-to-point channel, MACs allow the receiver to identify the sender, but does not allow a third party to identify the sender
  - Anyone knowing the message and the key, in principle the 2 communicating parties, nay generate an appropriate MAC

  I.e., MACs do not allow **no-repudiation**.
- Digital signature primitives are usually based on asymmetric encryption systems

Digital Signature with RSA

- Public key encryption algorithms, e.g. RSA, may be used to generate digital signatures
- In its basic form, the encrypted of a message with the senders private key can considered as a signature
  - Decryption of the encrypted message using the public key for deciphering, is the best proof that can be presented
- In practice, signing digitally comprises 2 steps:
  1. Compute the hash value of the data to sign
  2. Encrypt that hash value

  The output of the second step is a digital signature
- Not all digital signature algorithms are based in this algorithm, e.g. DSA

  `Signature sign(Message m, Key` K^-`)`

  `Boolean check(Message m, Signature s, Key` K^+`)`

Roadmap

Strength of the Cryptographic Mechanisms (1/2)

Empirically secure based on the test of time. For example DES

- There are no known vulnerabilities
- Although there is no proof of its security, it is considered secure by the cryptographic community

Provably secure based on complexity theory. If its security depends on solving a problem for which no computationally feasible solution is known. For example, RSA:

- The complexity is measured in asymptotic terms: how big is sufficiently large?
- Actually, there is no proof that factoring cannot be done in polynomial time

This type of algorithm can be cracked by an attacker that has enough computing power

- It is a question of time
- ... and of keys' length

Strength of the Cryptographic Mechanisms (2/2)

Unconditionally secure based on the information theory. An algorithm is secure if an attacker cannot extract information about the plaintext by observing the ciphertext

- History shows that published cryptographic algorithms are broken mostly because of the length of the keys and not so much because of algorithm vulnerabilities
- With unpublished algorithms the history is different
  - DeCSS is may be the most recent and publicized example

The Last Word to the Experts

- *"If you think cryptography will solve your problem then you don't understand cryptography ... and you don't understand your problem."*, Bruce Schneier
- *"Cryptography is rarely ever the solution to a security problem. Cryptography is a translation mechanism, usually converting a communications security problem into a key management problem and ultimately into a computer security problem."*, Dieter Gollmann in
  Computer Security, John Wiley & Sons, 1999

Roadmap

Further Reading

- Chapter 9, Tanenbaum e van Steen, *Distributed Systems, 3rd Ed.*
  - Section 9.1: *Introduction to Security*