CYBER CHALLENGE
CyberChallenge.IT

cini
Cybersecurity National Lab

## SPONSOR PLATINUM

accenturesecurity

aizoon AUSTRALIA EUROPE USA
TECHNOLOGY CONSULTING

B5

EY Building a better working world

eni

exprivia | ITALTEL

IBM

KPMG

LEONARDO

NTT DATA
Trusted Global Innovator

NUMERA
SISTEMI E INFORMATICA S.p.A.

Telsy

## SPONSOR GOLD

bip.

cisco

MONTE DEI PASCHI DI SIENA
BANCA DAL 1472

negg

NOVANEXT
connecting the future

pwc

## SPONSOR SILVER

DiGi ONE
the leading digital company

ICT CYBER CONSULTING

# Hash functions

**Rocco DE NICOLA**
IMT Lucca

CYBER CHALLENGE
CyberChallenge.IT

cini
Cybersecurity
National Lab

https://cybersecnatlab.it

# License & Disclaimer



## License Information



## Disclaimer

- We disclaim any warranties or representations as to the accuracy or completeness of this material.
- Materials are provided “as is” without warranty of any kind, either express or implied, including without limitation, warranties of merchantability, fitness for a particular purpose, and non-infringement.
- Under no circumstances shall we be liable for any loss, damage, liability or expense incurred or suffered which is claimed to have resulted from use of this material.

# Outline



- Hash Functions
- Cryptographic Hash Functions
- Message Authentication
- Digital Signature and Other Applications

# Outline

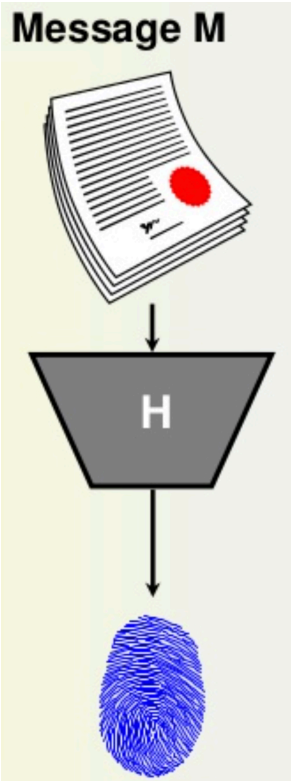


- Hash Functions
- Cryptographic Hash Functions
- Message Authentication
- Digital Signature and Other Applications

# Hash functions



- The hash functions:
  - take as input a block M of variable length
  - generate a fixed length fingerprint message h = H(M).
- No Key is involved



# Outline



- Hash Functions
- Cryptographic Hash Functions
- Message Authentication
- Digital Signature and Other Applications

# Cryptographic Hash Functions



- The hash functions of interest in cybersecurity are cryptographic hash functions, namely those members of a family of functions that are:
  - One-way (non-invertible)
  - Collision resistant
- To be effective cryptographic hash functions have to avoid that two messages are associated to the same hash and guarantee resistance to message forging.
- N.B.: Not necessarily cryptography is involved

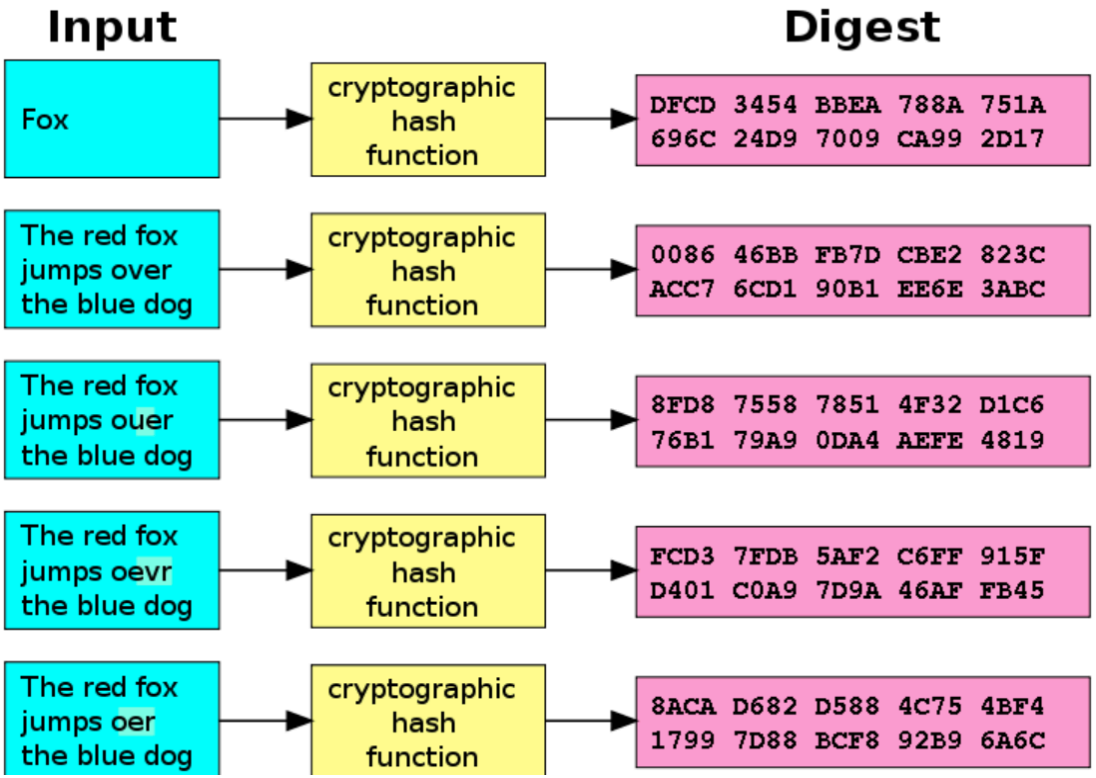
# Resistance of Cryptographic Hash



- Preimage resistance (One-way function): Given h = H(m), it is un-feasible to derive m knowing only h (i.e. to find x such that H(x) = m)
- 2nd preimage resistance (Weak collision resistance): Given h = H(m1) and m1, it is computationally unfeasible to find m2 ≠ m1 such that H(m1) = H(m2).
- Strong collision resistance: it must be computationally impracticable to find a pair of messages m1 and m2 such that H(m1) = H(m2).
- Weak collision resistance is bound to a specific input, whereas strong collision resistance applies to any two arbitrary inputs.

# Cryptographic Hash






- A cryptographic hash function at work: A small change in the input "over") changes drastically the output.
- Avalanche effects if an input is changed slightly (even a single bit), the output changes significantly (half the output bits).

# Secure Hash Algorithm - SHA



- There are “standards” for cryptographic hash functions, as was DES and now AES for symmetric encryption.
- The standard for cryptographic functions is called Secure Hash Algorithm (SHA), a suite of cryptographic functions developed by the NSA since 1993.
- Two main families:
  - SHA-1: returns a  Message Digest (MD) of 160 bits
  - SHA-2: consists of six hash functions differing for the length of the digest that is 224, 256, 384 or 512 bits: SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224, SHA-512/256.

# SHA-1 Usage



SHA-1 has been widely used in:

- Many protocols such as TLS and SSL, PGP, and SSH.
- Some versioning services such as Git and Mercurial that allow people to edit the same files on a remote server and synchronize the changes while avoiding conflicts and unnecessary file transfers.

# Weaknesses of SHA-1



- The SHA-1 returns a  Message Digest (MD) of 160 bits, typically rendered as a hexadecimal number, 40-digit long. It was designed by the United States National Security Agency.
- SHA-1 dates back to 1995 and has been known to be vulnerable to theoretical attacks since 2005, as of 2010 many organizations have recommended its replacement.
- All major web browser ceased to accept SSL certificates based on SHA-1 in 2017.
- As of 2020, it is recommended to remove SHA-1 from products as soon as possible and use SHA-x variants.

# SHA-2



- SHA-2 family consists of six hash functions with digests (hash values) that are 224, 256, 384 or 512 bits:
  - SHA-256 and SHA-512 use different shift amounts and additive constants, but their structures are virtually identical, differing only in the number of rounds.
  - SHA-224 and SHA-384 are truncated versions of SHA-256 and SHA-512, computed with different initial values.
  - SHA-512/224 and SHA-512/256 are also truncated versions of SHA-512, but the initial values are generated differently
- SHA-2 was first published by the US – NIST, the algorithms algorithms are patented but US released the patent under a royalty-free license.

# Outline



- Hash Functions
- Cryptographic Hash Functions
- Message Authentication
- Digital Signature and Other Applications

# Hash functions @ Work

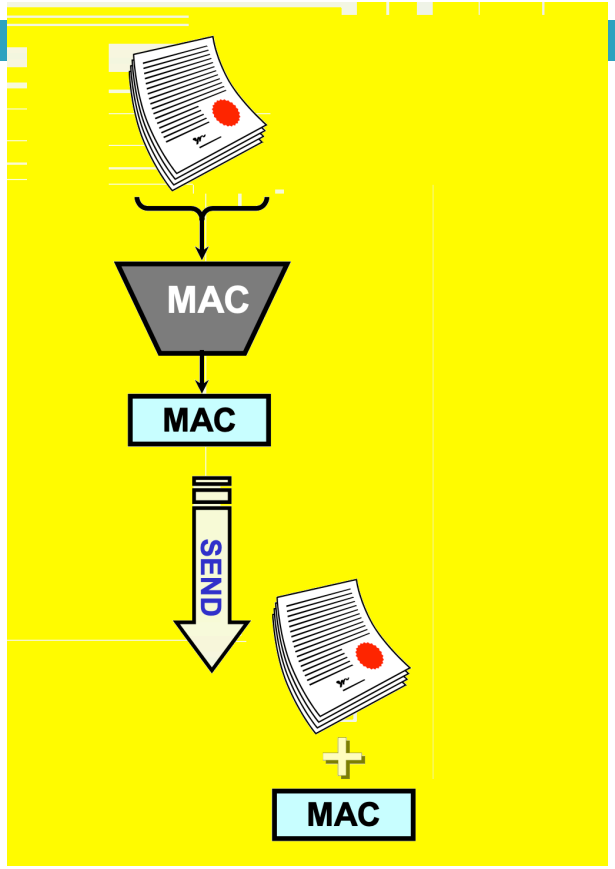


- The main use of hash functions is for guaranteeing *integrity* of a message M.
  - A message M is sent together with the result of the hash function applied to it
  - In case M is changed on the way to become M', the receiver, by calculating the hash function on M', will get, with a very high probability, a different value than the one sent together with M.

# Message Authentication



- So-called Message Authentication Codes (MAC) software is used to produce a message digest (MD - MAC) from a message M.
- Once a MAC has produced an MD, the pair (M, MD) is sent over for subsequent check.



# Message autenthication

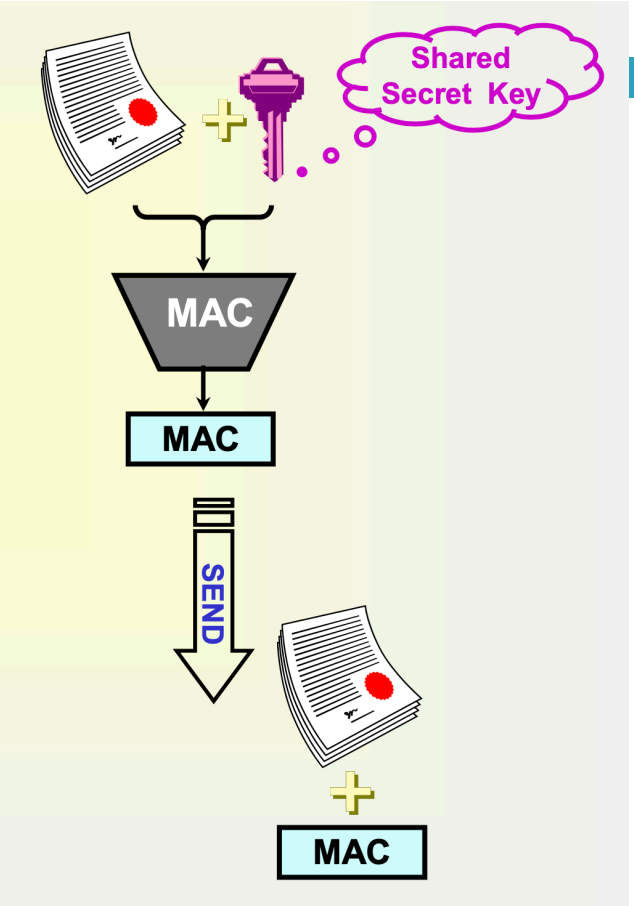


- The sender A calculates the hash of his message, attaches it to the message itself and sends it to B.
- The receiver, upon receipt, separates the message from the hash and recalculates the hash with the secret key.
- At this point, the comparison between the calculated hash and the received hash allows the receiver to check if the message is intact (i.e., the hashes match).
- If the hashes do not match, the message may have come from another source or might have been modified during the trip.

# Crypted Message Authentication



- A shared secret key is used that could be generated via Diffie-Hellman.
- Cryptographic hash functions ensure integrity.
- If the MAC is encrypted authenticity is guaranteed.
- If the message (M) is encrypted, then confidentiality is guaranteed.
- Obviously, both M and MD can be encrypted.





# Message Authentication/2



- If Alice (A) and Bob (B) share a key k (unknown to others) for a symmetrical cipher, then A can encrypt M and MD or both and send them to B:
  - If B can decrypt M then he can consider the message as authentic (i.e., B can assume that it actually cames from A, as she is the only one who could have encrypted that message with the shared key);
  - If B calculates the hash of M with the same function used by A and the output of such function is equal to the received MD then B can assume integrity (the message has not been modified):
  - When a message is not confidential, to save computational resources necessary to encrypt an entire message (which can be arbitrary in length), A can encrypt only MD

# Building a MAC function



- To build a good MAC function, cryptographic hashes can be used for the following reasons:
  - Hash functions such as SHA are computationally faster than symmetric encryption and are sufficient to verify the integrity of a message.
  - Symmetric encryption algorithms would generate MACs of the same size as the original message (which should then be reduced).
  - Many libraries are available for cryptographic hashes.

# Message Authentication Codes (MAC)



- A MAC function can be defined as

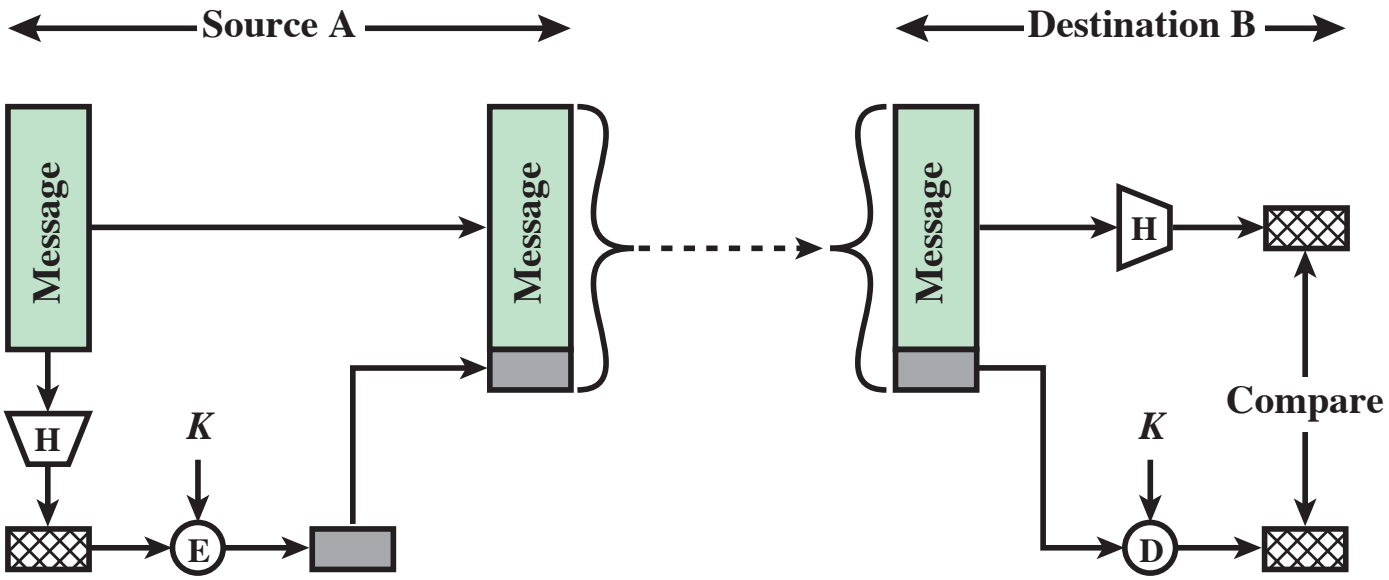
$$MAC = C(K, M)$$

where

- M is the input message
- C is the MAC function
- K is the shared secret key
- MAC is the output control value that is sent with the message.

# MAC with Hashing and Symmetric Encryption






(a) Using symmetric encryption

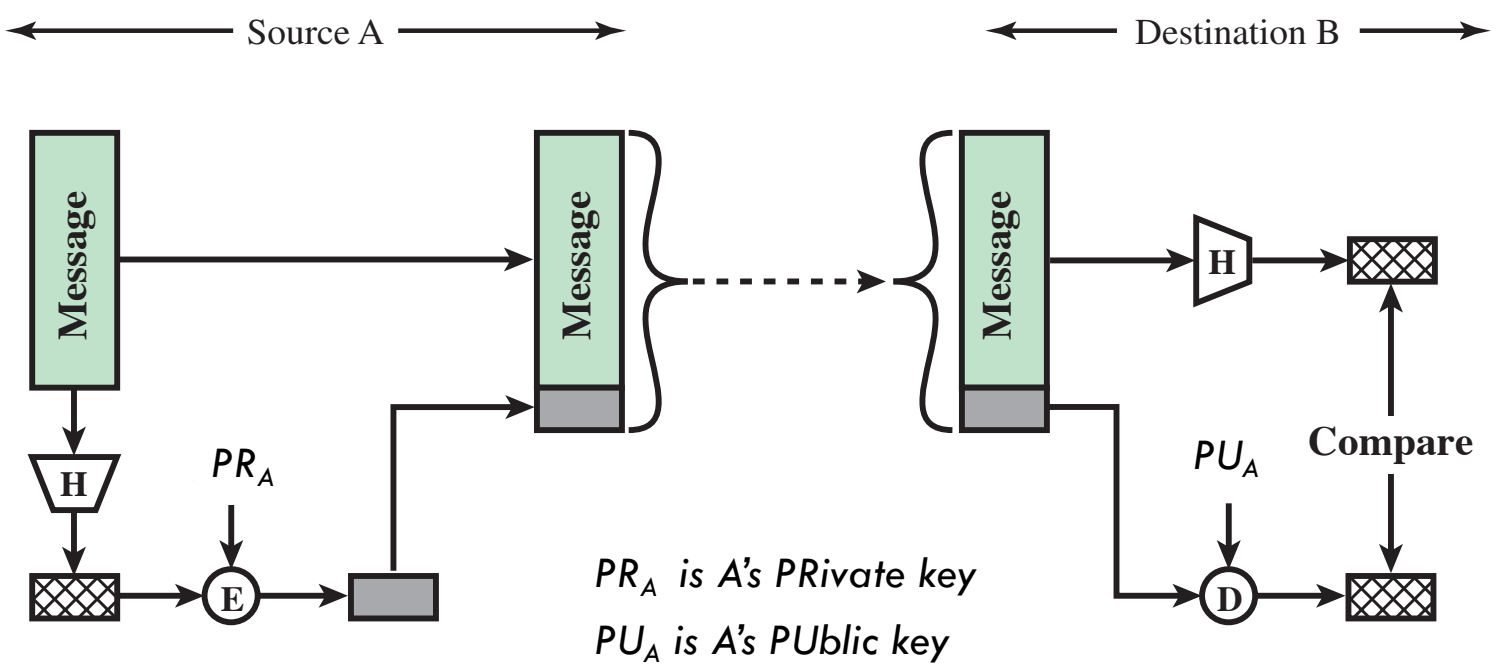
If only sender and receiver share the encryption key, authenticity is assured.

Picture from: *W. Stalling: Cryptography and Network Security, International Edition, Pearson*

# MA with Hashing and Asymmetric Encryption



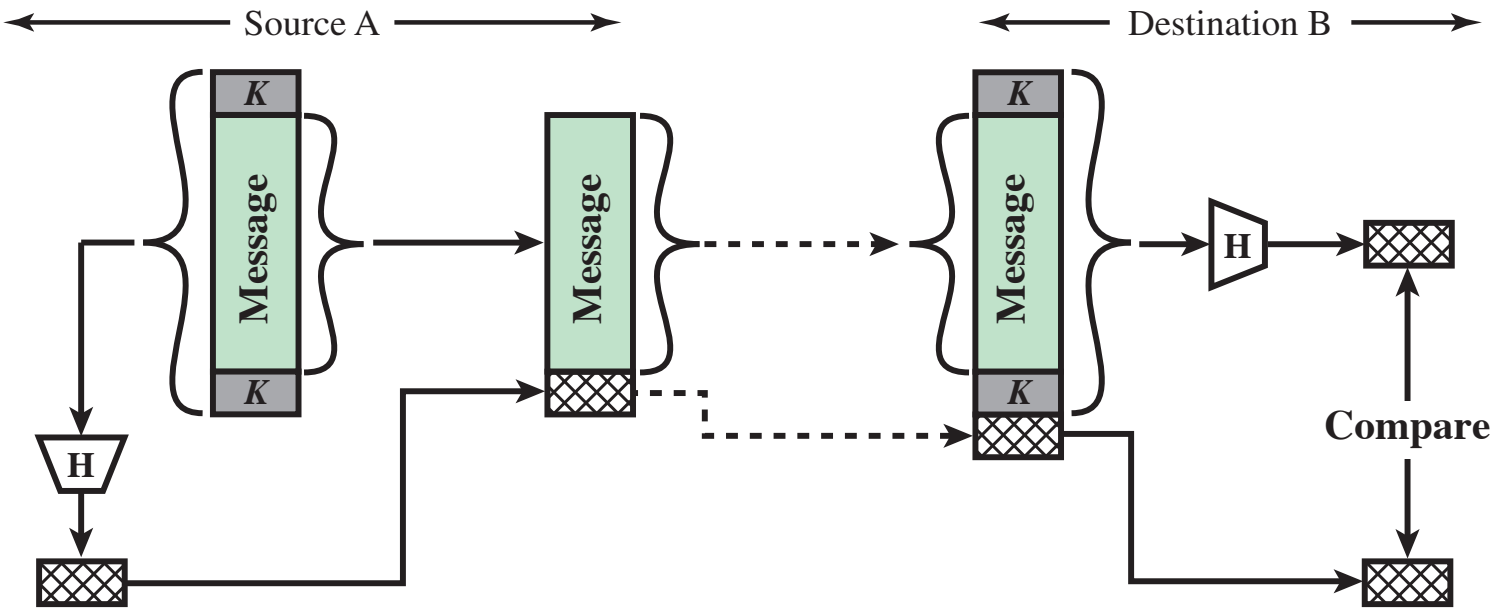



Public-key encryption has the advantages of not requiring keys distribution

Picture from: *W. Stalling: Cryptography and Network Security, International Edition, Pearson*

# MAC with Hashing and no Encryption






Shared secret key K is incorporated into the process of generating a hash code

Picture from: *W. Stalling: Cryptography and Network Security, International Edition, Pearson*

# Attacks to MAC



A MAC may be subject to different types of attacks by external attackers who do not have the key:

- Existential Forgery Attack: The attacker is able to create a valid M message and MAC for M without having the key. The attacker defines both M and the corresponding MAC.
- Selective Forgery Attack: Given an M message (not chosen by the attacker), the attacker is able to produce a valid MAC for M.
- Universal Forgery Attack: The opponent is able to create a valid MAC for any possible M message. Such an attack is the most powerful one and sanctions the total lack of security of the MAC scheme.

# HMAC



- HMAC (Hash-based MAC) is a specific type of MAC involving a cryptographic hash function and a secret cryptographic key.
- It may be used to simultaneously verify data integrity and message authenticity.
- Any cryptographic hash function, such as SHA-1 or SHA-2, may be used in the calculation of an HMAC; the resulting MAC algorithm is termed HMAC-X, where X is the hash function used.

# HMAC: requirements



- The HMAC shall ensure at least the following requirements:
  - Use the available hash functions without having to change them.
  - Use functions with good performance and open source code.
  - Allow a quick replacement of one hash function with a possibly faster or safer one.
  - Use and manage keys in an easy way.
  - Be robust against MAC attacks.

# Outline



- Hash Functions
- Cryptographic Hash Functions
- Message Authentication
- Digital Signature and Other Applications

# Using hashes



- Apart for message authentication, hashing can be used for a number of other reasons:
  - One-way password
  - Intrusion Detection
  - Pseudo Random Number Generator (PRNG)
  - Digital Signature

# One-way passwords



- In some operating systems passwords are not stored in full, only their hash is kept in the password file.
- Those who came in possession of the password file would still not be able to get the password from the hash.
- When a user types in a password, the system calculates the hash and compares it with the corresponding value in the password file.
- When you reset your password and don't receive your plaintext password in return, often it is because the system doesn't store your plaintext password.

# Optimized differences check



- Intrusion Detection: Hash functions are calculated for each file in a computer, if a virus modifies the file, the modification can be detected by recalculating the hash and comparing it with the original one.
- Version control: Many application allow different user to edit concurrently and synchronize the change on shared files:
  - versioning services use hash functions to determine whether files on the server and local files are the same and transfer only those that have been changed.

# Pseudo Random Number Generator



- Hashing is also used to generate random number:

  n = hash(seed).

- The result is an effectively pseudo-random number because a cryptographic hash function takes in input a variable amount of data and outputs a fixed-length block.
- If the seed is carefully chosen and given as input to a cryptographic hash, due to the avalanche effect, the output will contain a number of uniformly distributed bits.

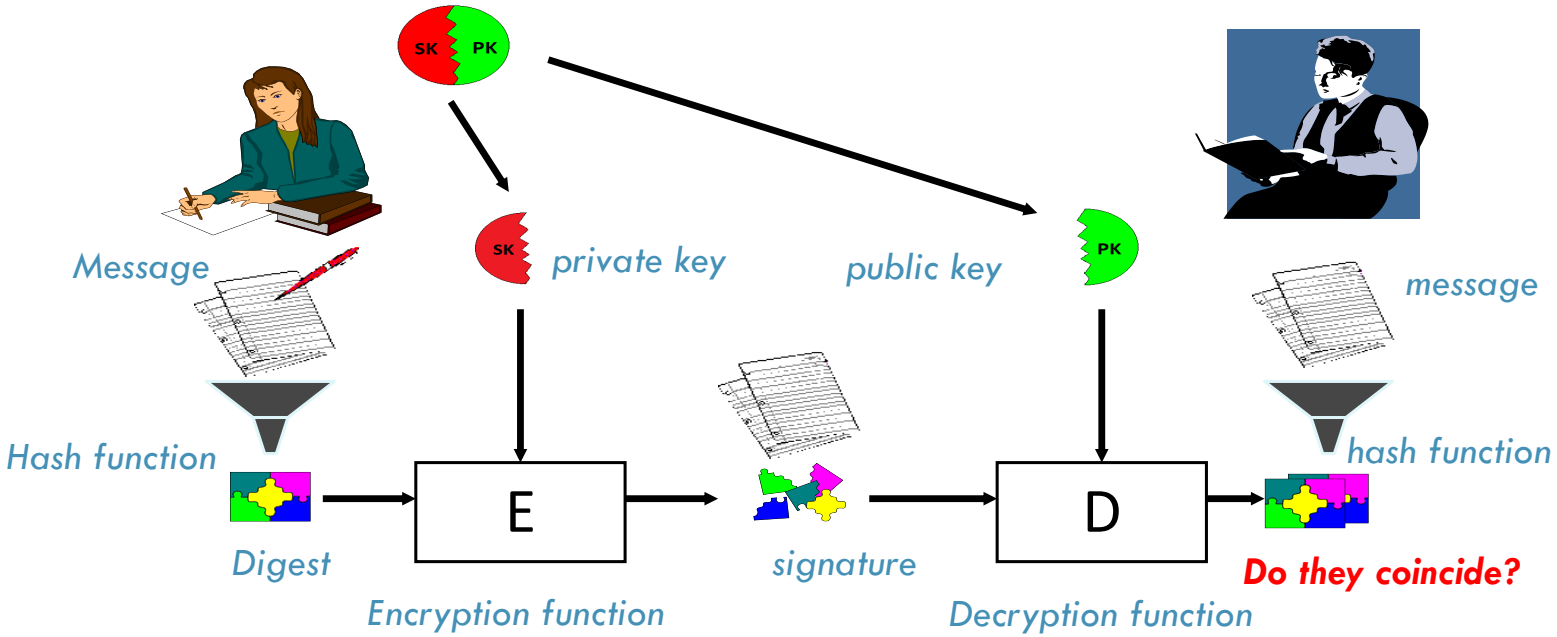
# Digital Signature



- A digital signature is a mathematical scheme for verifying the authenticity of digital messages or documents.
- A valid digital signature, where the prerequisites are satisfied, gives a recipient very strong reason to believe that the message was created by a known sender (authentication), and that the message was not altered in transit (integrity).
- The basic idea is similar to MAC, but the hash value is encrypted with the sender's private key.
- Anyone with the sender's public key can check the integrity of the message!

# Hashing and Digital Signature





# Objectives of Digital Signatures



- The objective of the digital signature is to provide message authentication and guarantee non-repudiation.
- The difference with message authentication codes (MAC) is the public verifiability of the signature
- To ensure non-repudiation, the security level of the whole process must be increased and robust trust mechanisms need be introduced.
- From the regulatory point of view, as of 1 July 2016, reference should be made to the European eIDAS (electronic IDentification Authentication and Signature) regulation no. 910/2014: `https://eurlex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32014R0910&from=EN.`

# Functions of a signature

- Indicative function: the ability to identify a signatory from the signature.
- Declaratory function: the signatory, through his signature, declares that he takes ownership of what he signs, or approves or recognises.
- Probatory function: The signature constitutes the proof of the authenticity and genuineness of the signed document.
  - Authenticity refers to the origin of the document; it is linked to the indicative function.
  - Genuineness means that the content has not been altered; it is linked to the declarative function.

# Digital Guarantees

- Indicative function
  - Public key holder and Public Key Infrastructure (PKI)
- Declaratory function
  - The hashing process applied to the document content
- Probatory function
  - it is guaranteed by the security of the set of used cryptographic algorithms, technologies, processes.

# Public Key Certification



- The indicative function is implemented through a public key certification infrastructure (PKI)
- The PKI used for digital signature certificates follows the X.509 standard
- The signature holder's public key is included in an X.509 digital certificate issued and signed (in turn) by a trust service provider (the Certification Authority - CA) also called Trusted Third Party.
- For signatures with full probatory value (Qualified Electronic Signature), the Certification Authority, in Italy, must be Qualified and Accredited by AGID (Agenzia per L'Italia Digitale).

# Certificate X.509 (v3)



- CERTIFICATE
  - V = Version
  - SN = Serial number
  - AI = Signature Algorithm ID
  - ISSUER = CA ← Certification Authority
  - VALIDITY (not before, not after)
  - SUBJECT NAME
  - SUBJECT PUBLIC KEY INFO (algorithm, key)
  - ISSUER ID
  - SUBJECT ID
  - EXTENSIONS (key usage, …) optional
- CERTIFICATE SIGNATURE ALGORITHM
- CERTIFICATE SIGNATURE

# Certification hierarchies



- X.509 is based on an infrastructure that creates chains of certificates that reach a trusted point (trust anchor), called Root
- Given two certification authorities CA-1 and CA-2, one link in the chain from CA-1 to CA-2 is given by a certificate signed with a CA-1 private key from a CA-2 public key.
- In the eIDAS signature domain, PKIs are implemented at national level, but there is also a European trust anchor mechanism

# Modifiability



## Non-Modifiability:

- a feature that makes the content of the computer document unalterable in form and content throughout the entire management cycle and guarantees its static nature in the conservation of the document itself

## Possible threats:

- Threats are connected to the possibility of changing a previously signed document

# Threats: Fonts



- Changes based on abnormal use of some fonts: "Times New Roman1 exchanges "A", "i", "c" and "e" for "C", "a", "r" and "k", using this font if you write "Alice" you get "Clark" and vice versa.
- Countermeasures: static formats, font hashing, use of sandbox, use a document parser when signing.

# Threats: Macros



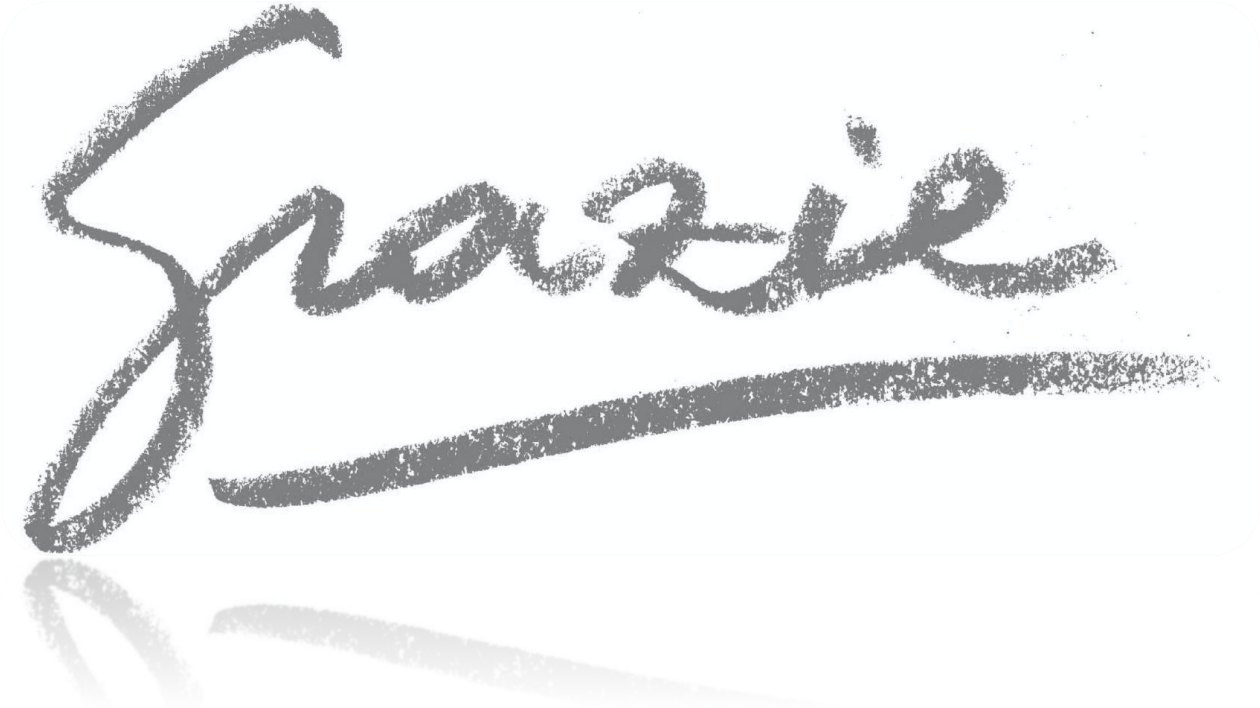
- It is possible  to embed in documents
  - macro instructions (macro of Word documents),
  - executable code (Javascript of PDF documents).
- The part of a documents that is visualized depends on variables related to the environment in which the document is presented.
- A (malicious) contract may include (through macrocode) a value dependent on the date of the system and provides that after a certain date, a specific quantity is increased or decreased.
- Countermeasures: Verify documents before signing or to restrict permitted documents formats to those not allowing the inclusion of (macro) instructions pure text (ASCII) or PDF/A (PDF Archive),

# Threats: Polymorphic files



Dalì attack - Buccafurri, F., Caminiti, G., & Lax, G. (2008), name inspired by the painting of Dali “the image disappears”.

- The attack is based on the insertion of two contents within a single file, each content will be shown by different applications.
- Depending on the file extension, either of the two is shown: copyright.pdf.p7m→auth.tif.p7m
- A signature format based on “cryptographic envelope” does not detect the anomaly.




CYBER CHALLENGE
CyberChallenge.IT

 Rel. 02.04.2020

cini Cybersecurity National Lab

CYBER CHALLENGE
CyberChallenge.IT

cini
Cybersecurity National Lab

## SPONSOR PLATINUM

accenturesecurity

aizoon AUSTRALIA EUROPE USA
TECHNOLOGY CONSULTING

B5

EY Building a better working world

eni

exprivia | ITALTEL

IBM

KPMG

LEONARDO

NTT DATA
Trusted Global Innovator

NUMERA
SISTEMI E INFORMATICA S.p.A.

Telsy

## SPONSOR GOLD

bip.

cisco

MONTE DEI PASCHI DI SIENA
BANCA DAL 1472

negg

NOVANEXT
connecting the future

pwc

## SPONSOR SILVER

DiGi ONE
the leading digital company

ICT CYBER CONSULTING