Loriano Storchi

loriano@storchi.org

http::/www.storchi.org/

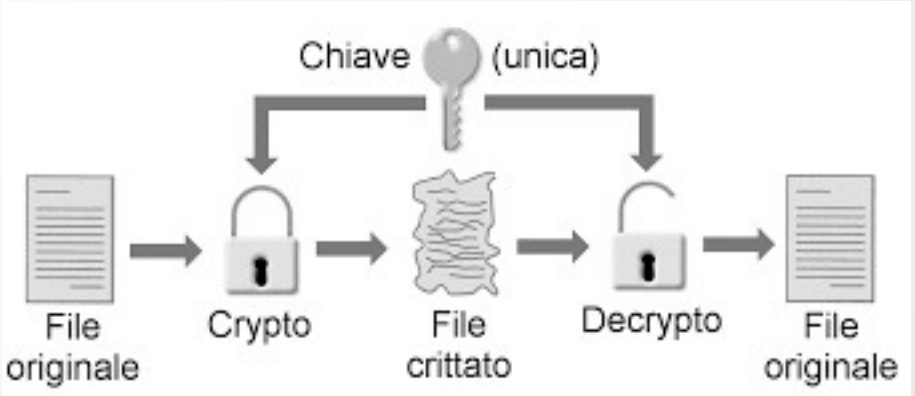
CRYPTOGRAPHY BASICS

Basic Concepts

- In computers, **information is stored as sequences of bits**
- **Cryptographic techniques modify these sequences (strings) to obtain different sequences which can then be transmitted and retransformed** by the recipient in the original sequence. The mathematical functions used in the transformation use one or more **secret keys**
  - **Symmetrical encryption**: used even since the Egyptians and the ancient Romans
  - **Asymmetric encryption:** dates back to the 70s

Symmetric encryption

- **The key used to encrypt and decrypt, and therefore by sender and recipient, is unique**. For example, Caesar's cipher replace each character with another offset of k places (the key is the value of k)
- Single alphabetic cipher



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
m	n	b	v	c	x	z	a	s	d	f	g	h	j	k	l	p	o	i	u	y	t	r	e	w	q

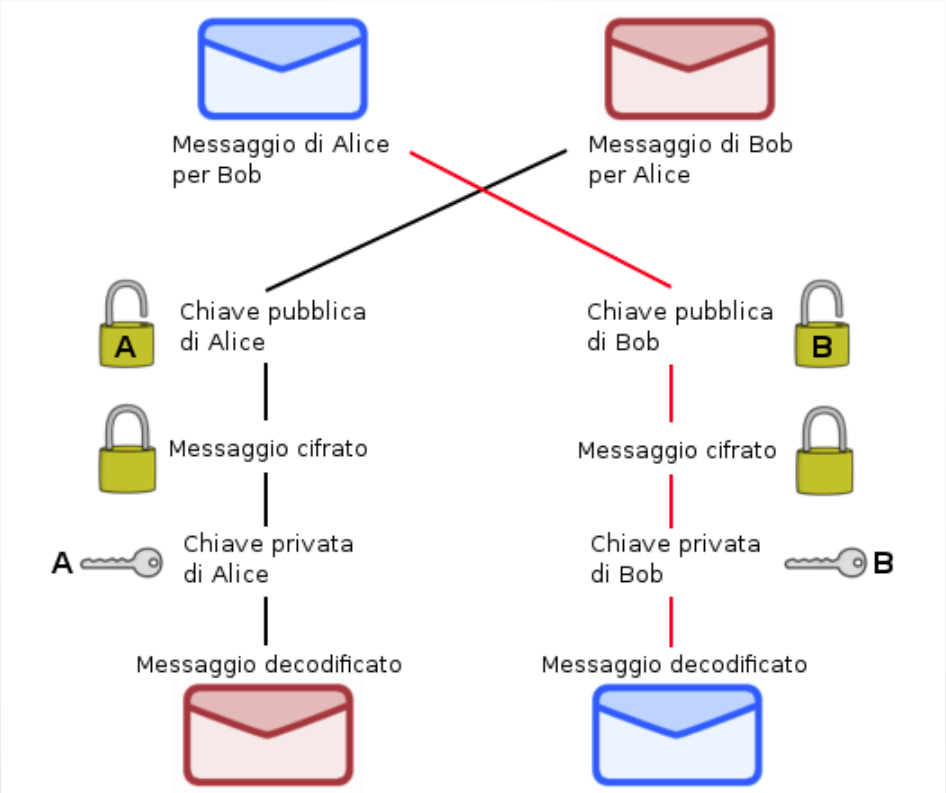
Symmetric encryption

- **I have to find a safe way to exchange the secret key** which is unique for sender and recipient
- **Brute-force attack** in the case of Caesar's cipher, for example, I try all the values of k and see when I get "correct" phrases and words
- Examples of modern algorithms: **Blowfish, Twofish, Standard DES or Triple DES, Standard AES**. They are all based on mathematical problems that are difficult to solve if you don't know the secret key

ASYMMETRIC ENCRYPTION

Asymmetric Encryption

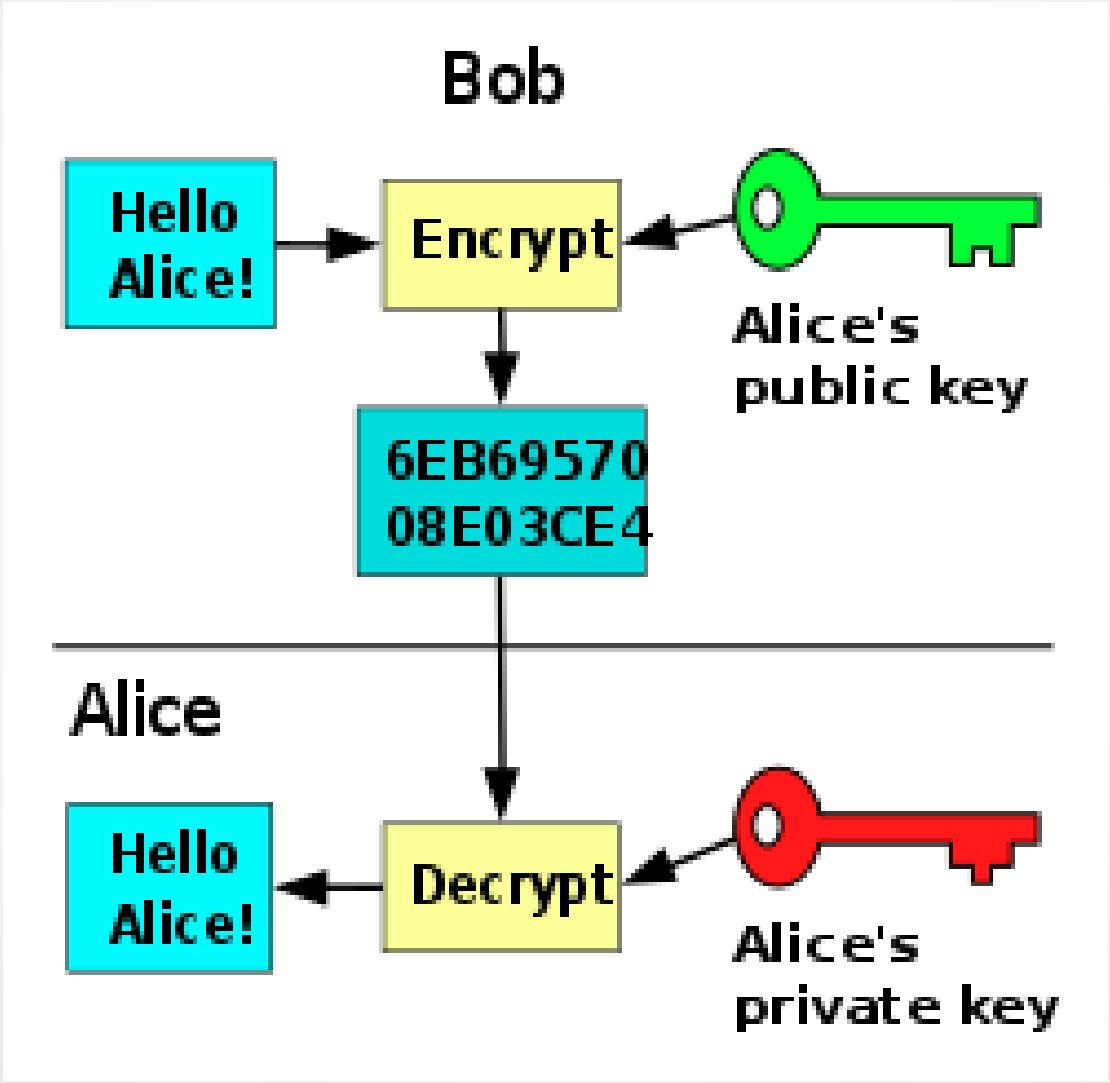
- **The keys used to encrypt and decrypt are different**. Private key that must be kept secret is used to decrypt, therefore to get back the original message, the public key to encrypt



Asymmetric Encryption

- **When the user generates the key pair, he must jealously guard the private (secret) key (KS) and instead distribute only the public one (KP)** . KS for example in a smartcard and in that case it will be the smartcard itself that performs the encryption.
- If user Bob wants to write a private message to user Alice, Bob will use Alice's public key KP and send the obtained cryptogram, Only Alice in possession of the private part of the KS key will be able to get back the original message (in clear)
- **Asymmetric encryption also used in authentication processes**

Asymmetric Encryption



Asymmetric Encryption: RSA

- **The most known and used algorithm is the RSA (names of the inventors Rivest, Shamir, Adleman)**
- Also for authentication or guaranteeing the integrity of a document (**including digital signature**)
- Based on prime numbers, i.e. those natural numbers that are divisible only by 1 or by themselves (2, 3, 5, ..., $19249 \cdot 2^{13018586} + 1$)
- In practice, KS and KP are prime factors of a large number
- Interest in prime numbers and factorization algorithm (Shor quantum computer algorithm)

PGP - OpenPGP

OpenPGP

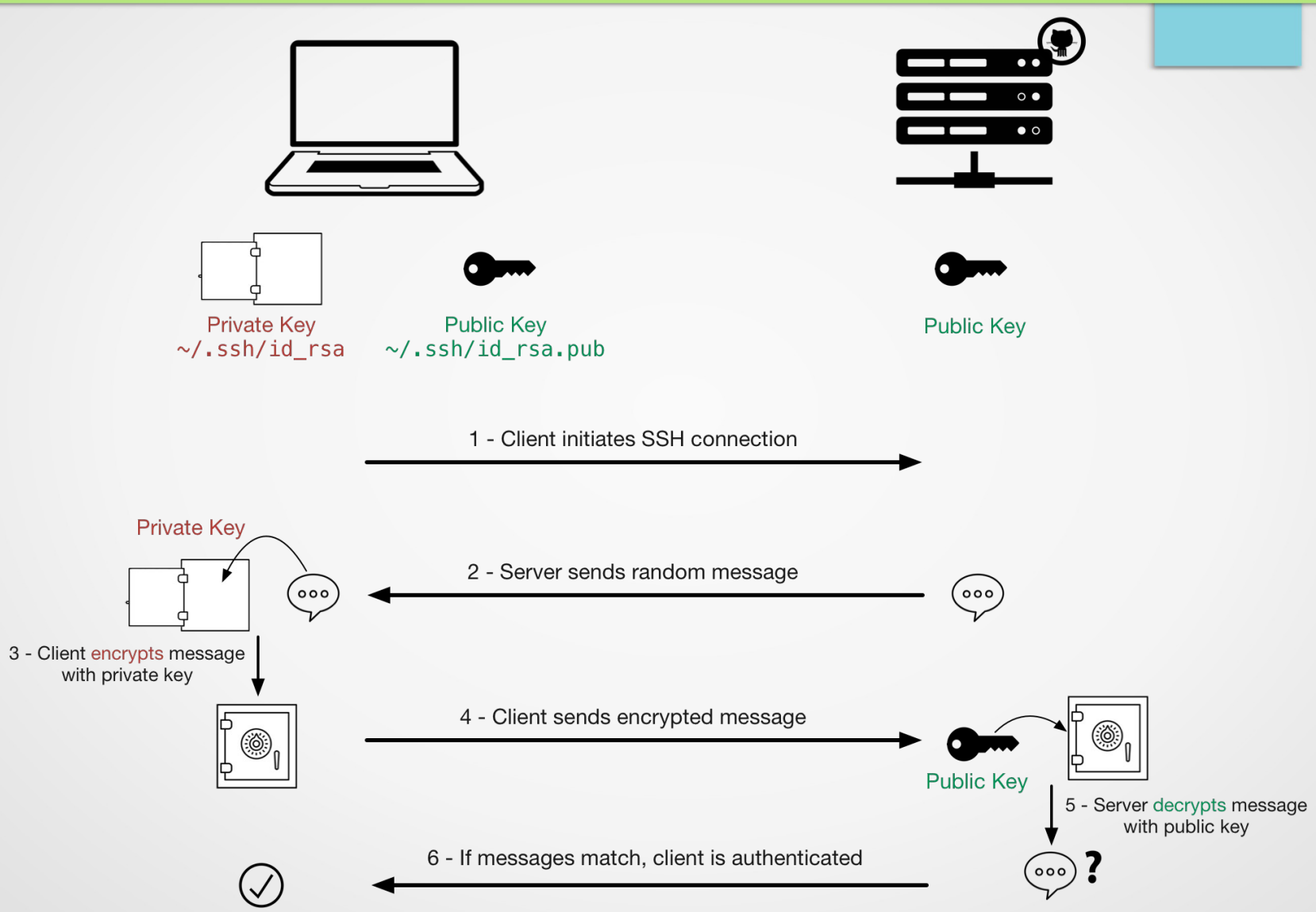
- RSA is an algorithm (actually, two algorithms: one for asymmetric encryption and one for digital signatures - with several variations). **PGP is software, now a standard protocol, generally known as OpenPGP.**
- **OpenPGP defines formats for data elements** that support secure messaging, with encryption and signatures, and various related operations such as key distribution.
- As a protocol, OpenPGP relies on a wide range of cryptographic algorithms. Among the algorithms that OpenPGP can use is RSA.

OpenPGP

- **Philip R. Zimmermann is the creator of Pretty Good Privacy**, an email encryption software package. **Originally conceived as a human rights tool, the PGP was released for free on the Internet in 1991.**
- This mad**e Zimmermann the target of a three-year criminal investigation**, because the government believed U.S. export restrictions for cryptographic software were breached as PGP spread around the world.
- GNU Privacy Guard (GnuPG or GPG) is free software designed to replace the PGP cryptographic suite.

AUTHENTICATION

SSH example


Private Key
~/.ssh/id_rsa
Public Key
~/.ssh/id_rsa.pub
Public Key
1 - Client initiates SSH connection
Private Key
2 - Server sends random message
3 - Client encrypts message
with private key
4 - Client sends encrypted message
Public Key
5 - Server decrypts message
with public key
?
6 - If messages match, client is authenticated

DIGITAL SIGNATURE

Digital Signature

- **It is affixed to digital documents in order to guarantee**
  - **Authenticity: therefore a guarantee of the origin of the message**
  - **Integrity: the message has not been modified in any way**
  - **Non-repudiation: The source of the message cannot deny having signed it**
- **Only the sender can affix that particular signature**
- **Anyone can check who signed the message (digital document, text, sound, image, video)**
- **Basic ingredients asymmetric cipher system and hash function**

Digital Signature: Method of signature

- **CASDES**: extension pdf.p7m the file can be read with signature software (File protector, DiKe ....)
- **PADES**: pdf extension the file is signed with signature software and read with Acrobat Reader
- **XADES**: extension xml.p7m, the xml file is automatically read by the software that receives it

Digital Signature: HASH

- Hash algorithm: **MD5, SHA-1, RIPEMD, SHA-256**:
  - A function that, given a stream of bits of variable size, returns a string of letters or numbers of a fixed size (a sort of single dot)
  - The string is a unique identifier, the modification of only 1 bit of the source stream produces a different HASH
  - It is not invertible so starting from the returned string it is not possible to determine the original flow

```
redo@raspberrypi:~$ date
Fri Nov  3 12:42:50 CET 2017
redo@raspberrypi:~$ date | md5sum
628a589b0ecb099db2cf4d9f4235f97f  -
redo@raspberrypi:~$ date | md5sum
4c0b372ca0fe4cd391d1eb02deaae7b8  -
redo@raspberrypi:~$
```

Digital Signature: How it works

- Alice to sign a given object OR (a document for example):
  - Calculate the HASH of O (also called digest)
  - Encrypt the HASH obtained with her key
  - It appends the encrypted HASH (the signature) together with its public key on object O, let's call it F
- Bob to verify Alice's signature:
  - Calculate the HASH of O
  - Decrypts the encrypted HASH (hence Alice's F signature found together with the document) using Alice's public key
  - Check at this point that the values are the same

CA AND CERTIFICATES

CA and Certificates

- **How can you be sure that the signature used actually belongs to the signatory ?** Paraphrasing how can I be sure of the public key user association ?
- **Digital Certificates serve this purpose. They contain a lot of information, such as the public key and user data**
- Just as paper certificates allow you to have information about the user
- **CA, Certification Authorithy, guarantees the association between the digital signature and the owner's identity**

CA and Certificates

- **Digital certificates: consists of (X.509):**
  - **Public key of the signatory**
  - **From your identity (name surname date of birth etc etc)**
  - **Public key expiration date**
  - **Name of the CA that issued it**
  - **Digital signature of the CA that placed the certificate**
- If we trust the CA (Certification Authority) we can verify its signature and therefore the identity of the signatory

CA and Certificates

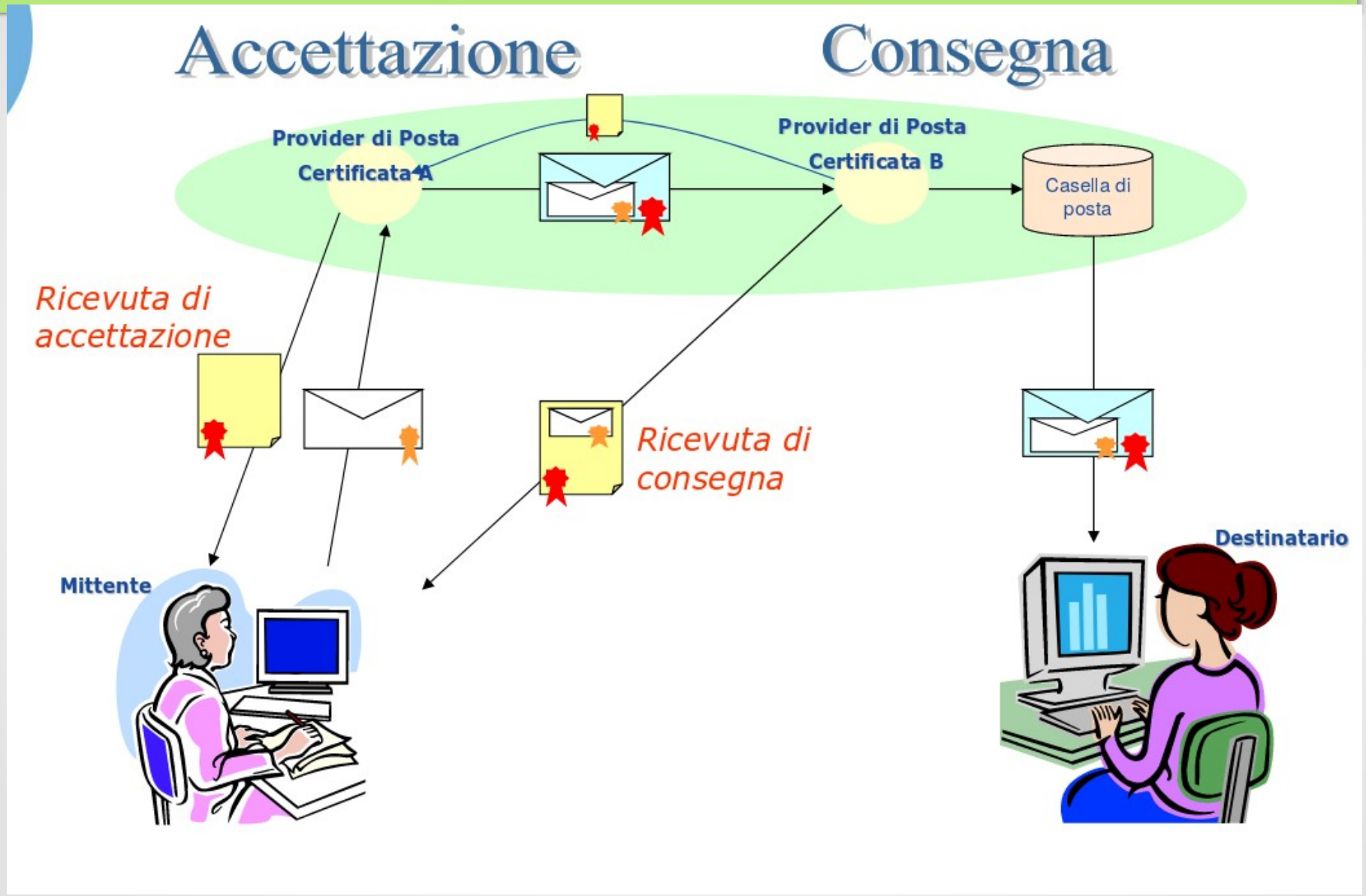
- **CA, Certification Authorithy,** guarantees the association between the digital signature and the owner's identity
- The digital certificate is signed by an entity called CA which certifies its validity. The signature operation takes place by the CA by attaching its references to the certificate and encrypting everything with its private key
- A given CA that signs the certificate may not be considered reliable at which point we will repeat the certificate from the suspect CA we can contact the higher level CA and so on until we trace a CA deemed reliable, and which therefore validates all the chain

HTTPS AND PEC

HTTPS

- Fundamental protocol for example in e-commerce and home banking
- When I connect to a site via https:
  - **The server declares its identity by sending its public key certificate guaranteed by a CA**
  - **My browser (client) verifies that the URL matches the identity contained in the certificate**
  - **The client (browser) using the server's public key sends a temporary symmetric key to encrypt all data traffic**

PEC

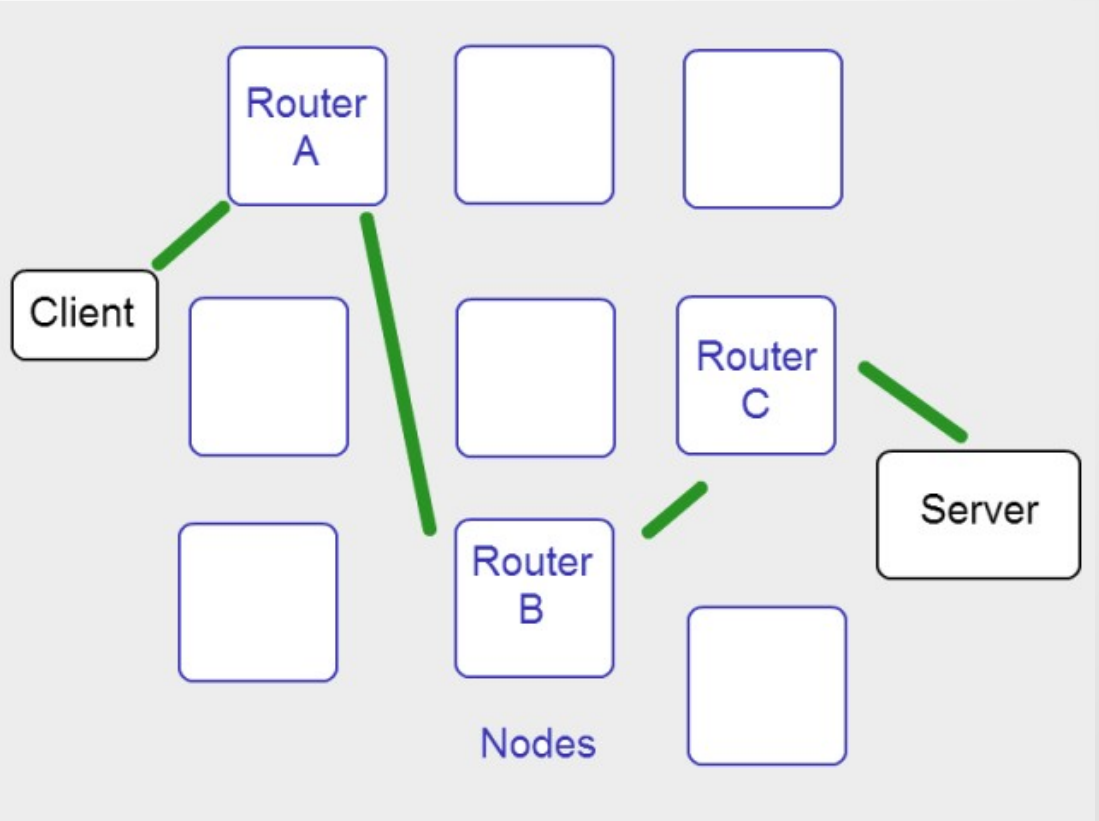

Accettazione
Consegna
Provider di Posta Certificata A
Provider di Posta Certificata B
Casella di posta
Ricevuta di accettazione
Ricevuta di consegna
Mittente
Destinatario

INTERNET

Deep web, Tor, bitcoin

- Onion routing a mention. The tor network is made up of volunteers who use their computers as nodes:

Tor creates a random path between the various nodes so as to reach the server starting from the client

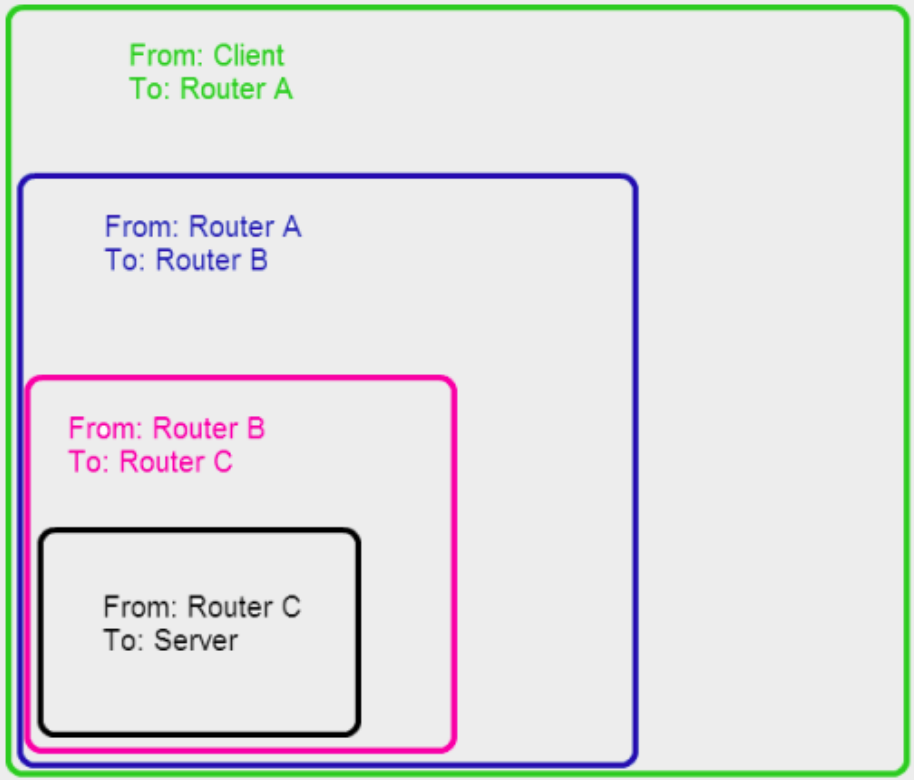


Deep web, Tor, bitcoin

- I can reach “hidden” hidden services, but also normal servers via exit nodes

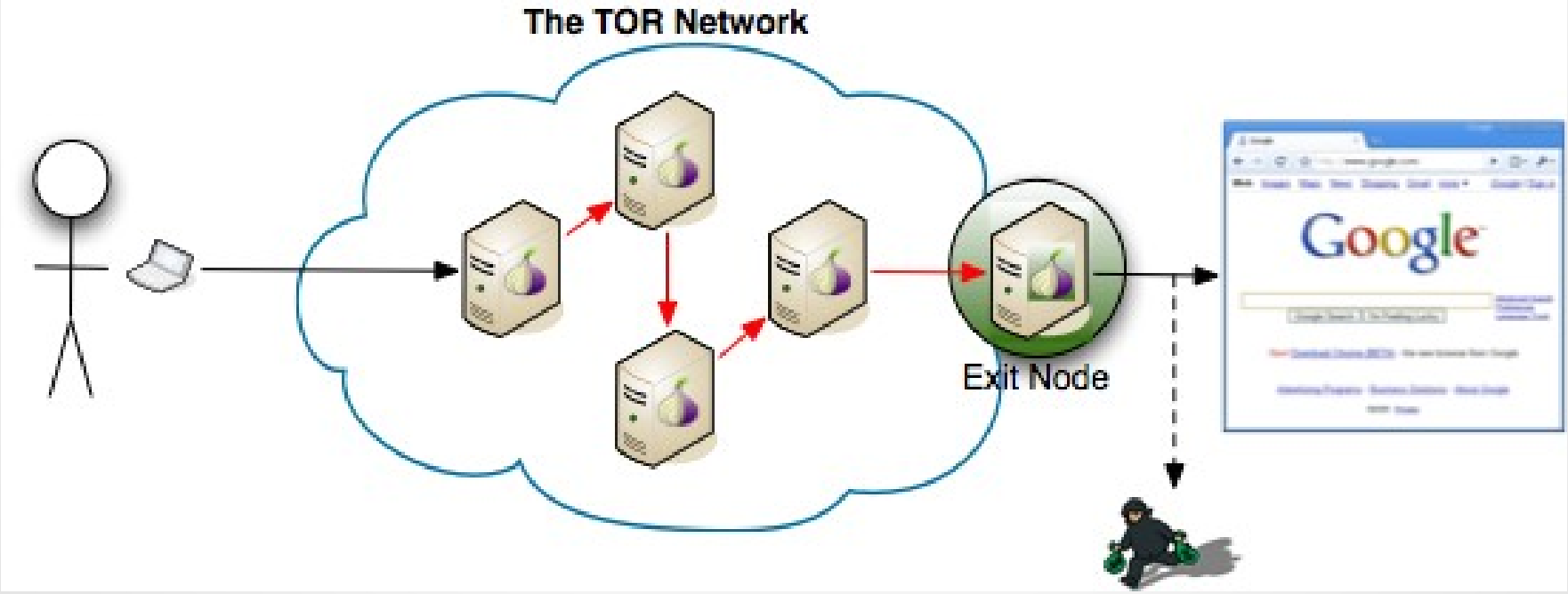
**The packets passing between t
various nodes are encapsulate
in successive encrypted layers
just like the layers of an "onion**

At each step the node only knows
the point of origin and the point o
arrival



Deep web, Tor, bitcoin

- I can reach “hidden” hidden services, but also normal servers via exit nodes



Deep web, Tor, bitcoin

- **Bitcoin** (I quote wikipedia) Unlike most traditional currencies, Bitcoin does not use a central body: it uses a database distributed among the network nodes that keep track of transactions, but uses cryptography to manage functional aspects, such as the generation of new money and the attribution of ownership of bitcoins.
- Based on a P2P network (**blockchain distributed block structured database**)
- Based on cryptography and hashing algorithms (SHA-256), **BitCoin uses the SHA-256 hash algorithm to generate "random" verifiable numbers in a way that requires a predictable amount of computation time.** Generating a SHA-256 hash with a value less than the current target resolves a block.