Introduzione all'Informatica

Loriano Storchi

loriano@storchi.org

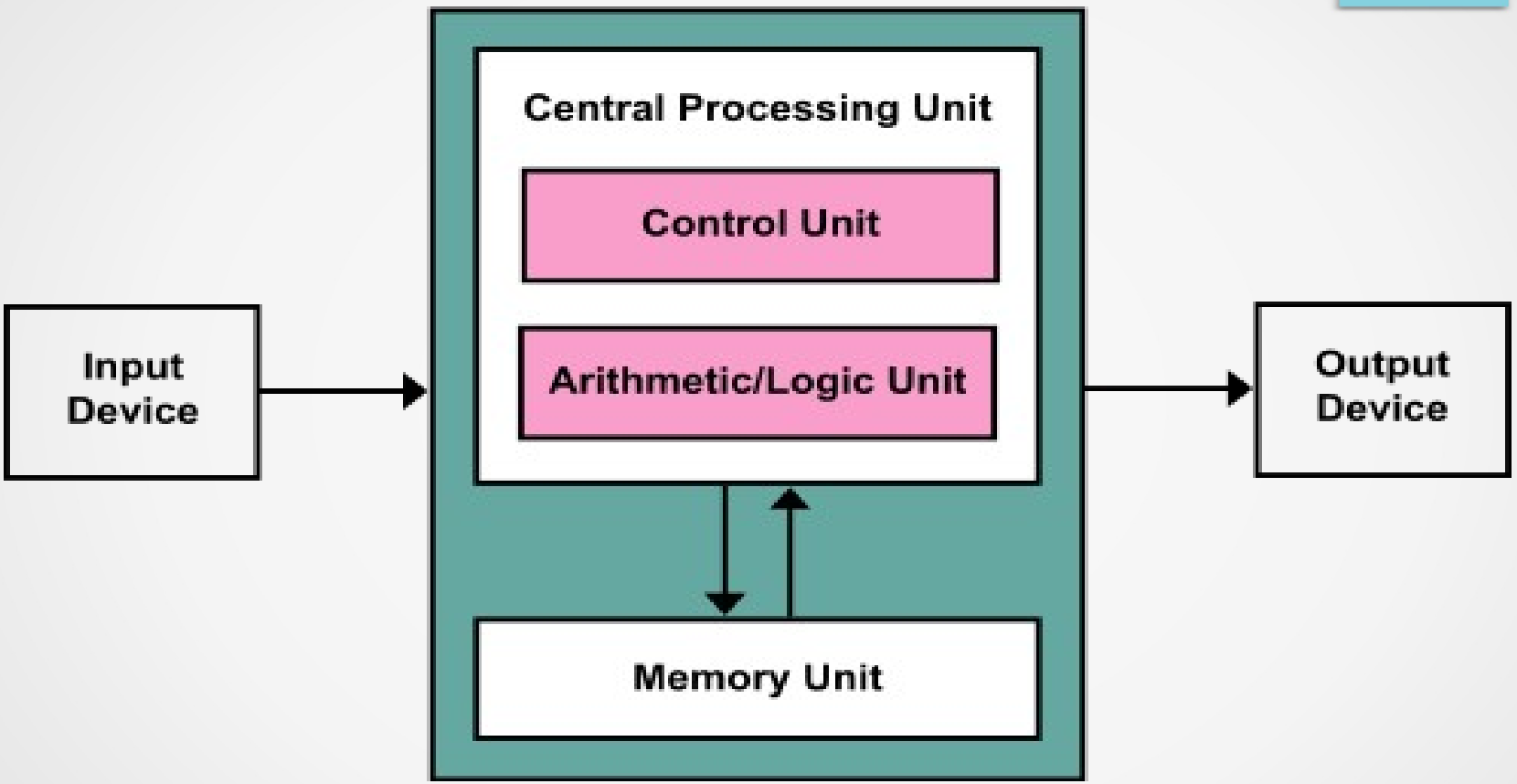
http://www.storchi.org/

Informatica

- Possiamo usare diverse definizioni come ad esempio: l'informatica e' la scienza della rappresentazione e dell'elaborazione dell'informazione, parafrasando lo studio degli algoritmi che descrivono e trasformano l'informazione

HARDWARE

Macchina di Von Neumann (Zuse)



BUS di SISTEMA , collegamento (architettura Harvard separazione tra memoria dati e memoria contente il programma)

Macchina di Von Neumann (Zuse)

L’architettura descritta da Von Neumann (Zuse) si compone dunque di :

- La CPU e quindi una unità di elaborazione centrale
- Un dispositivo di memoria che serve appunto a memorizzare i dati che sono poi identificabile mediante il lro indirizzo
- I vari dispositivi di Input Output (I/O) che servono appunto ad interagire con l’utente esterno od altri sistemi
- Una linea di interconnessione che collega i vari sottosistemi , il bus

Macchina di Von Neumann (Zuse)

Un calcolatore così costituito è una macchina estremamente flessibile. L'Hardware mette a disposizione funzionalità di base sarà poi il software a specializzare la macchina cosiì da svolgere compiti specifici.

Di seguito presenteremo le varie componenti di base del calcolatore con una prospettiva "moderna"

CPU

C.P.U.

C.P.U. Central Processing Unit o Unità di Elaborazione Centrale, coordina e gestisce tutti i vari dispositivi hardware per acquisire, interpretare ed eseguire le istruzioni dei programmi. **Oggi è costituita da un solo chip** e come ogni chip **comunica con l'esterno mediante i piedini**. Usando i piedini riceve segnali ed invia segnali elettrici (informazione costituita da sequenze di bit)

- **Unita' di controllo** (nota anche come **CU**) legge i dati dalla memoria istruzione e dati esegue le istruzioni e copia i risultati nella memoria o nei registri

C.P.U. - A.L.U.

- **A.L.U.** (unita' logico-Aritmetica) svolge le operazioni logiche ed aritmetiche.
- Strettamente collegato all’ALU c’è lo **shifter** che esegue appunto lo shift verso destra o sinistra dei risultato dell’ALU , corrispondente a divisioni o moltiplicazioni per potenze di due

C.P.U. - Registri

- **Registri**, in pratica memoria interna della CPU che permette di accedere ai dati in moto molto piu' rapido
- Registri: In tutte le CPU sono sempre presenti almeno due registri:
- **IP (Instruction Pointer o Program Counter PC)** che contiene il puntatore alla prossima istruzione da eseguire.
- **Registro dei Flags**. Questo registro è in sostanza una serie di bit che rappresentano un particolare stato della CPU , ad **esempio il flag dell'Overflow** è messo ad 1 nel caso in cui il risultato dell'operazione appena effettuata è troppo grande per il campo dei risultati

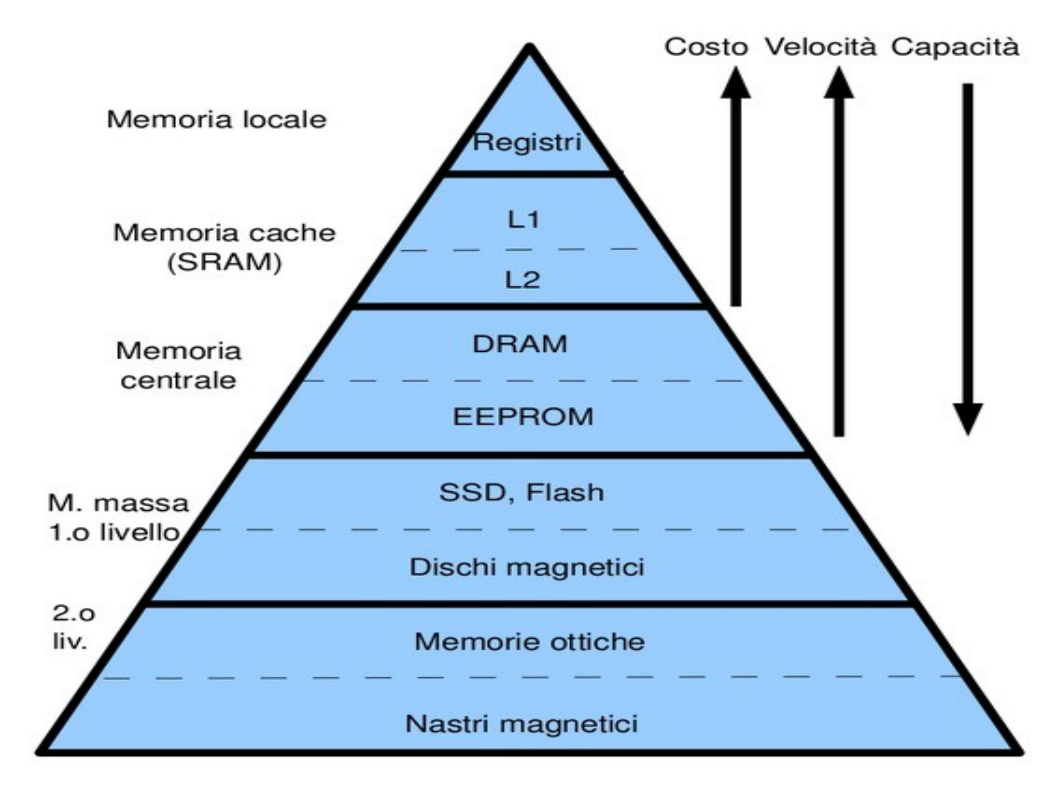
C.P.U. - CLOCK

- **CLOCK**: scandisce gli intervalli di tempo in cui agiscono i dispositivi interni della CPU . Ne determina la **velocita' espressa come numero di intervalli nell'unita' di tempo**.
- **Lo stato della CPU cambia ogni volta che viene inviato un impulso** Quindi il tempo di esecuzione di una data operazione viene misurato in numero di cicli di clock.
- Una parte importate delle CPU e' proprio la serie di “circuiti” che servono a propagare questo impulso fra tutte le componenti della CPU.
- Nessuna CPU può operare più velocemente del tempo impegnato dal segnale di clock per percorrere il percorso piu' lungo di questo “circuito” di distribuzione del segnale, **critical path**

MEMORIA

Gerarchia di memoria

La gerarchia di memoria nei processori attuali e' compostada diversi livelli, caratterizzati ciascuno da velocita' di accesso ai dati inversamente proporzionali alla dimensione: piu' sono grandi queste aree e maggiore e' il temporichiesto per recuperare i dati in esso contenuti.



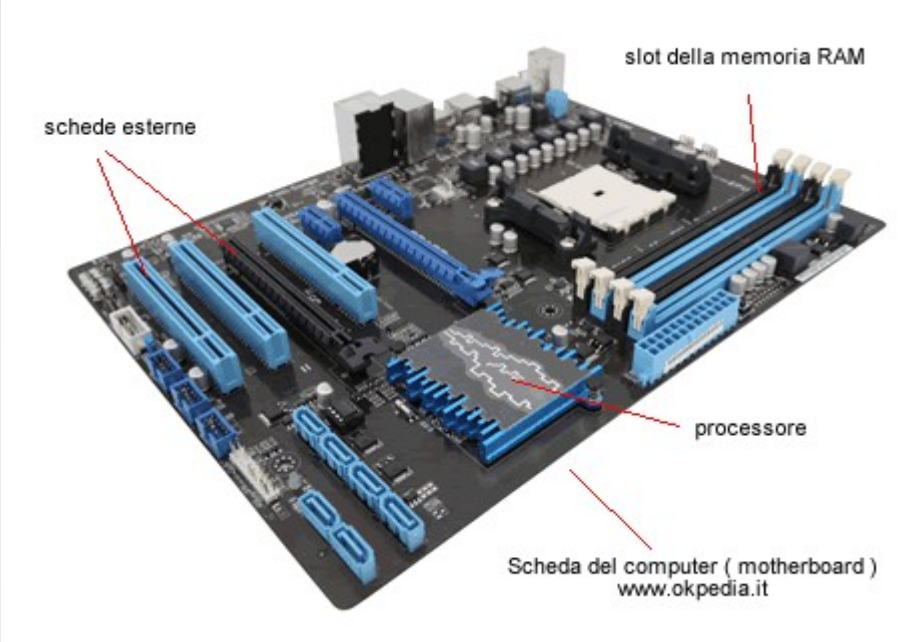
Memoria Centrale

- **E’ la memoria che e’ direttamente interfacciata alla CPU** , ad esempio è direttamente connessa alla scheda madre del computer. Questo consente un flusso continuo di dati da e verso la CPU.
- Questo tipo di memoria è caratterizzata da una velocità di accesso ai dati estremamente rapido
- La memoria centrale (anche ***Primary Storage***) può essere di sola lettura come la **ROM** oppure di lettura e scrittura come la **RAM**
- N.B. in questa categoria sono classificati anche i registri della CPU

Memoria Centrale - RAM

- **RAM (Random Access Memory)**. E’ divisa in celle di memoria ed **ogni cella e’ identificata ad un indirizzo** che viene usato per accedere il lettura e scrittura ai dati in essa contenuti
- Nella **RAM sono contenute anche le istruzioni (opcode)** che verranno eseguite ed i dati su cui tali istruzioni opereranno
- Caratteristiche:
  - **Volatile**: il contenuto e' perso quanto l'elaboratore e' spento
  - **Veloce**: 100 cicli di clock circa . Veloce quindi costosa
  - **Dimensioni piccole** rispetto alla memoria di massa ordine di qualche GiB

Memoria Centrale - RAM



Esistono diverse tecnologie di RAM, quella modernamente più utilizzata è la **DRAM (Dynamic RAM)**
In pratica ogni bit è memorizzato in un condensatore (**bit 1 o 0 dipende dalla carica del condensatore**) , ogni condensatore deve essere ricaricato periodicamente altrimenti si avrebbe una perdita di carica e quindi di informazione. Ci sono poi diverse varianti tecnologicamente diverse di DRAM.

SRAM Static RAM sono invece memoria con una tecnologia che non richiede il refresh continuo, ma sono in grado di mantenere l'informazione per tempi molto lunghi. **Bassi consumi e tempi di accesso brevi, ma costi di costruzione elevati. Sono generalmente usati usati per le memoria cache**

Memoria Centrale - ROM

- **R.O.M. (Read Only Memory)** memoria di sola lettura ad alta velocità di accesso rispetto alla memoria di massa
- **Consenti di memorizzare dati in modo permanente**. Questo tipo di memoria serve ad immagazzinare i dati ed il codice necessario alla procedura di avvio dei computer ed altri programmi/procedure (**Firmware**)
- **BIOS Basic Input/Output System** , nei sistemi moderni rimpiazzato dalle **UEFI Unified Extensible Firmware Interface**
- **EPROM Erasable Programmable Read Only Memory**, ovvero memoria di sola lettura programmabile e cancellabile, Possono essere cancellate e riscritte per un numero generalmente limitato di volte

Memoria Centrale - ROM

- **R.O.M. (Read Only Memory)** memoria di sola lettura ad alta velocità di accesso rispetto alla memoria di massa
- **Consenti di memorizzare dati in modo permanente**. Questo tipo di memoria serve ad immagazzinare i dati ed il codice necessario alla procedura di avvio dei computer ed altri programmi/procedure (**Firmware**)
- **BIOS Basic Input/Output System** , nei sistemi moderni rimpiazzato dalle **UEFI Unified Extensible Firmware Interface**
- **EPROM Erasable Programmable Read Only Memory**, ovvero memoria di sola lettura programmabile e cancellabile, Possono essere cancellate e riscritte per un numero generalmente limitato di volte

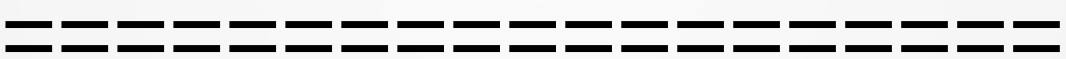
BREVE INTERMEZZO IL BOOT

BOOTUP process

- **Power-on**: Ovvio che la prima fase del processo è l’accensione, che viene in genere avviata dall'utente.
  - Più recentemente è stata aggiunta la possibilità consentono un'impostazione "**wake on LAN**", che facilita l'accensione attraverso la rete, quindi non è richiesta alcuna presenza o azione fisica dell'utente.
  - Appena alimentata la CPU esegue il codice che si trova, appunto, nella **ROM** memoria di sola lettura presente nella scheda madre

BOOTUP process

- **POST**: Il sistema esegue poi una procedura detta POST (**Power-On Self Test**), che garantisce che tutto l'hardware sia operativo e pronto all'uso. Ciò include il c**ontrollo della memoria e dei dischi rigidi e ...**. Una volta terminato il POST, il sistema cerca il primo dispositivo nell'elenco degli ordini di avvio.
- A questo punto dopo il **Load del BIOS o UEFI** Il sistema cerca un dispositivo attivo nell'elenco dei dispositivi di avvio. Quando trova un dispositivo disponibile il mediante l'uso delle funzionalità di base del **BIOS/UEFI il sistema e' in grado di avviare il sistema operativo.**



Memoria Centrale - Cache

Cache In generale da 1 a 3 livelli di memoria cache sono installati. La memoria cache e' caratterizzata da diversi tempi di accesso e dimensioni, a seconda che essa sia all'interno del chip (on chip) o fuori dal chip (offchip), e dalla tecnologia con cui sono realizzate le celle di memoria. (cat /proc/cpuinfo per vedere le dimensioni della cache)

L'uso e il vantaggio di una cache si basa sul principio di localita' ovvero che un programma tende a riutilizzare dati ed istruzioni usate recentemente Una conseguenza e una regola del pollice che dice che in genere il 90 % del tempo totale viene impiegato ad eseguire il 10% delle istruzioni. Per essere piu precisi l'uso della cache e' vantaggioso quando di un codice si sfrutta sia la localita' spaziale che la localita temporale dei dati.

Memoria Centrale - Cache

- Localita' temporale: una volta che viene usato un generico dato "a" questo verra riusato piu volte in un breve arco di tempo
- Localita' spaziale: se si usa il generico dato "a" allora anche il dato "b", che e conservato nella locazione di memoria contigua a quella di "a" verra' usato a breve.

Il fatto che nella maggior parte dei casi si lavori essenzialmente su un piccolo sottoinsieme dei dati totali permette di copiare nella cache L1, che ha
latenza di pochi cicli, solo quei dati necessari alla CPU. Inoltre, grazie alla localita' spaziale, ogni volta che carico un dato dalla memoria alla cache non
prendo solo quel dato ma anche altri dati contigui, ovvero la cosiddetta linea di cache, che e composta da 64-128 byte. Quando si richiede un dato non viene spostato solo quello ma anche i dati contigui.

Memoria di massa – Storage Secondario

- **A differenza della memoria centrale non e' direttamente accessibile dalla CPU, ma la CPU comunica con un controller (bus I/O)**
- I dati sono trasferiti dalla memoria secondaria ad un'area nella memoria centrale e da li letti direttamente dalla CPU
- Fisicamente questi dispositivi sono collegati alla scheda madre con un cavo ad alta velocità oppure mediante cavi collegati ad interfacce esterne

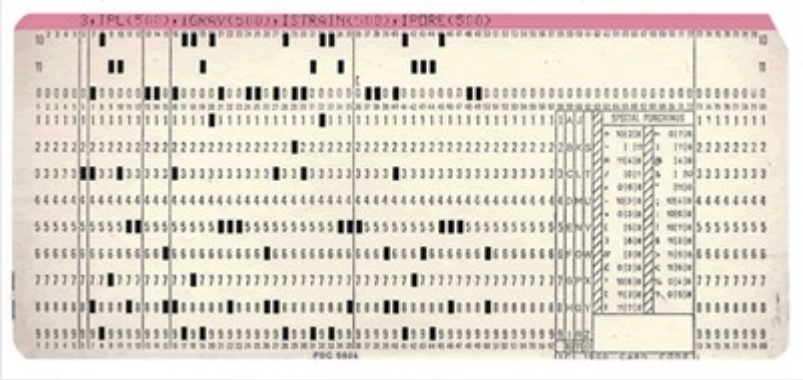
Possono essere realizzate con tecnologia magnetica, ottica oppure a stato solido

Memoria di massa

- Costituita da Dischi Rigidi, CD, DVD, dischi stato solido , nastri
  - **Non volatile**, quindi i dati rimangono memorizzato al riavvio del computer
  - **Lenta rispetto alla RAM** (> 10000 cicli), ovviamente molto variabile dipende dal tipo di supporto
  - **Mediamente economica** o comunque piu' economica della DRAM o SRAM
  - **Di grandi dimensioni** (oramai centinaia di GiB o qualche TiB)
  - **Possono essere sia di sola lettura che di lettura e scrittura** (pensiamo ad HARD-Disk e DVD-ROM)

Memoria di massa – Gli albori

- Schede perforate , e’ la prima memoria secondaria della storia dei computer. I dati sono memorizzati su sched di cartone e registrati mediante perforazione, e successivamente letti dal calcolatore
- Nastri perforati simile alle schede perforate come tipologia



Memoria di massa – Nastro magnetico

- In questo caso i dati sono memorizzati su un nastro in forma magnetica , Questi venivano usati storicamente nei mainframe (computer di grandi dimensioni in grado di eseguire elaborazioni molto velocemente ed immagazzinare grandi quantità di dati) degli anni 70-80, ma **usati anche oggi per il backup di dati**

Memoria di massa – Hard-Disk

- **Composto da più piatti magnetici** sovrapposti uno sull’altro e ruotanti
- I dati sono memorizzati su tutti e due i lati dei singoli piatti e sono organizzati in **tracce e settori**
- Le **testine** servono e leggere e scrivere i dati
- La velocità di rotazione dei piatti (**rpm giri al minuto**) è uno dei fattori determinanti nella velocità di lettura e scrittura
- **SSD dischi a stato solido** latenze inferiori meno sensibili a rotture dovute e shock meccanici

Memoria di massa – Rimovibile

- Può essere rimossa e separata dal computer
- **CD e DVD**, sia di sola lettura che di lettura e scrittura, i dati sono memorizzati e letti usando una luce laser (l'idea di base e' semplice riflessione o non riflessione per identificare bit a 1 o a 0)
- **USB flash drive**
- **Foppy-disk** non più usati

USB

- **3.0 azzurro** o blu scuro
- **2.0 nera , bianca, grigia**
- 3.0 retro-compatibile
- 3.0 150 mA (4.5 w)
- 2.0 100 mA (2.5 W)

- C’è poi una differenza notevole nella velocità di trasferimento dati che passa da **480 Mbit/s a 5 Gbit/s nel caso della USB 3**
- **USB Type-C (3.1) alimenta dispositivi fino a 100 W e permette una velocità di trasferimento dati di 10 Gbit/s (connettore diverso)**

Introduzione all'Informatica

Loriano Storchi

loriano@storchi.org

http://www.storchi.org/

UNITA’ DI MISURA

Unita' di misura dell'informazione

- **BIT** = è l'unità di misura dell'informazione (dall'inglese "binary digit"), definita come **la quantità minima di informazione che serve a discernere tra due possibili eventi equiprobabili**. (Wikipedia)
- **BYTE** = 8 BIT (storicamente i caratteri erano rappresentati da 8 BIT , motivo per cui **1 Byte rimane tutt'oggi l'unita' di memoria minima indirizzabile**)
- “KiloByte” meglio “kibibyte” KiB = 2^{10} Byte = 1024 Byte
- “MegaByte” meglio “mebibyte” MiB = 1024 * 1024 Byte
- “GigaByte” meglio “gibibyte” GiB = 1024 * 1024 * 1024 Byte
- “TeraByte” meglio “tebibyte” TiB = 1024 * 1024 * 1024 * 1024 Byte

Prestazioni dei calcolatori

- Chiaramente aumentare le prestazioni di un calcolatore significa diminuire il tempo che esso impiega nell'eseguire un'operazione.
- **T_{clock}** e' il periodo di clock della macchina (**aumento della frequenza**)
- **CPI_i** e' invece il numero di "colpi" di clock necessari ad eseguire la data istruzione i (**riduzione della complessità per la singola istruzione**)
- **N_i** e' infine il numero di istruzioni di tipo i (ad esempio somme, salti ...)

$$T_{esecuzione} = T_{clock} \sum_{i=0}^{n} N_i CPI_i$$

MIPS

MIPS e' un'abbreviazione per **Mega Instuctions Per Second**, ed indica appunto il numero di istruzioni generiche che una CPU esegue in un secondo. E’ un’unita’ di misura usato per misurare le prestazioni di un calcolatore di uso più generale rispetto al FLOPS che vedremo a breve:

$$\textbf{MIPS} = \textbf{(Frequenza del clock)} / (10^6 \textbf{ CPI})$$

Questo tipo di misura non tiene conto ad esempio delle ottimizzazioni dovute alla presenza della cache e delle percentuali delle diverse istruzioni all’interno di programmi reali, e non solo.

FLOPS

FLOPS e' un'abbreviazione per **Floating Point Operations Per Second,** ed indica appunto il numero di operazioni in virgola mobile che una CPU esegue in un secondo. E' un'unita' di misura usato per misurare le prestazioni di un calcolatore diffusa in modo particolare nell'ambito del calcolo scientifico

Ad esempio nel caso di un prodotto classico tra matrici, vengono eseguite $2*N^3$ operazioni, quindi posso valutare i FLOPS esattamente misurando il tempo necessario ad eseguire tale moltiplicazione ed ottenere:

$$[\text{flops}] = 2*N^3 / \text{tempo}$$

SPEC

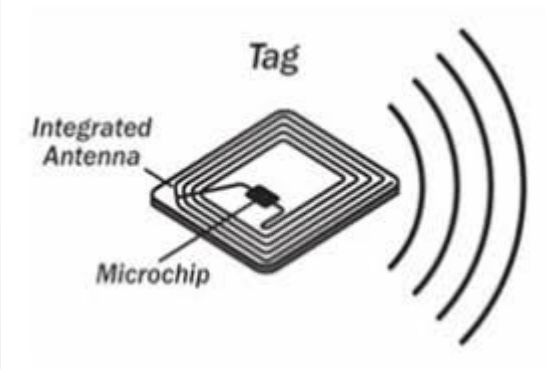
Standard Performance Evaluation Corporation e' un'organizzazione non-profit che produce e mantiene un insieme standardizzato di benchmark per computer (quindi un insieme di programmi di test che sono rappresentativi delle applicazioni reali di un calcolatore).

Ci sono diversi set di SPEC che sono specifici ad esempio per usi diversi previsti del computer

CALCOLATORI MODERNI

RFID

- **RFID Radio-frequency identification** costano pochi centesimi e possono raggiungere **qualche MIPS**
- Usano campi elettromagnetici per identificare e tracciare automaticamente oggetti mediante un ID o comunque informazioni
- **Passivi** se usano energia emessa dal lettore di RFID. Anche con batteria
- **Attivi** necessariamente con batteria inviano periodicamente un segnale



Sistemi embedded e SmartPhone Tablet

- **Sistemi embedded** oggi molto diffusi sono calcolatori pensati per Orologi, automobili, Elettrodomestici , apparati Medicali, lettori audio/video (Android Box). Sono calcolatori del costo diu qualche decina di euro con prestazioni dell'ordine di **qualche centinaio di MIPS** (**spesso equipaggiati con OS Linux**)
- **SmartPhone e Tablet** sono invece sistemi con potenze di calcolo decisamente superiori dell'ordine di **centinaia di GFLOPS per le solo CPU** e costi dell'ordine di centinaia di euro

Console di Gioco, PC e Workstation

- **Console di Gioco** sono sistemi con prestazioni complessive che possono arrivare ai **2000 GFLOPS circa considerando anche le GPU**
- **PC considerando Desktop e Portatili** coprono una fascia ampia di possibilità partendo dalle poche centinaia di euro fino a qualche migliaia con prestazioni che quindi **vanno dalle decine di GFLOPS ai 2000 GFLOPS considerando le GPU**
- Ci sono **Server e Workstation** che sono pensati per **High Performance Computing** e centralizzazione di servizi che possono arrivare a prestazioni di **qualche decina di TFLOPS con costi fino a 10/20,000 euro**

HPC

- Per poter aumentare le prestazioni della risorsa di calcolo in generale e' necessario accoppiare assieme numerosi calcolatori **interconnessi con reti ad alte prestazioni (calcolo parallelo)**
- Ad esempio **Cluster di workstation che possono arrivare a qualche centinaio di TFLOPS**
- **Supercomputer** con potenze di calcolo che oggi arrivano fino a **qualche decina di PFLOPS**
- Ovviamente aumentano allo stesso modo costi di produzione e costi di mantenimento (anche solo in termini di potenza assorbita)

CPU MODERNE

CISC vs RISC

La velocità di esecuzione di una singola istruzione e' uno dei fattori determinanti delle prestazioni della CPU stessa. Due visioni diverse:

- **CISC (Complex Instruction Set)** in questo caso l'idea di base e' che il set di istruzioni di base di una CPU debba essere la più ricca possibile, anche se **ogni singola istruzione in realtà richiede più cicli di clock per essere eseguita**
- **RISC (Reduced Instruction Set)** in questo caso **ogni istruzione e' eseguita in un solo ciclo di clock**. Ovviamente saranno necessarie più istruzioni RISC per eseguire la stessa singola istruzione di un CISC
- CISC architettura predominante nel mercato negli anni 17 ed 80 oggi c'è una tendenza ad favore della CPU di tipo RISC

Migliorare le Prestazioni

Aumentare le prestazioni di una CPU e' sempre un compromesso fra costi e consumi, ad esempio si possono adottare alcune strategie di base:

- **Ridurre il numero di cicli** necessarie all'esecuzione di una singola istruzione (esempio banale della moltiplicazione)
- **Aumentare la frequenza** (clock) con ovvi limiti fisici (**ad esempio al velocità della luce**)
- **Parallelismo, quindi eseguire ad esempio piu' operazioni in parallelo** (in contemporanea)
  - **A livello di istruzione piu' istruzioni** sono eseguite in parallelo dalla stessa CPU con ad esempio l'uso delle **pipeline o processori superscalari**
  - **Parallelismo a livello di core, quindi più core per CPU**

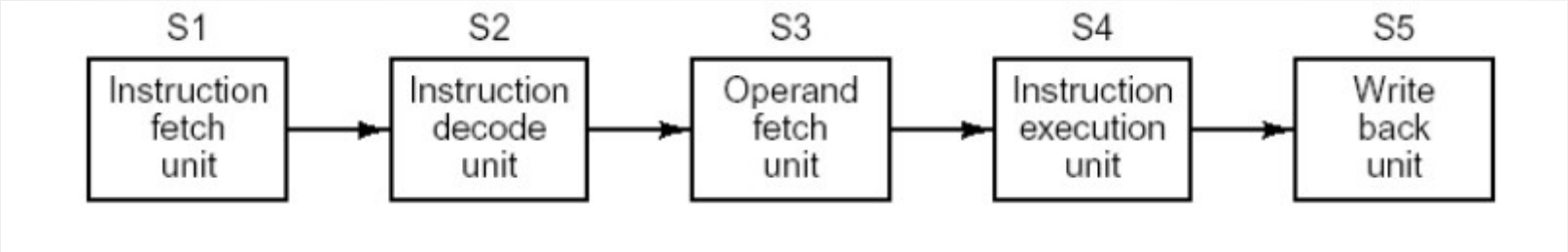
Migliorare le Prestazioni – Aumento frequenza di clock

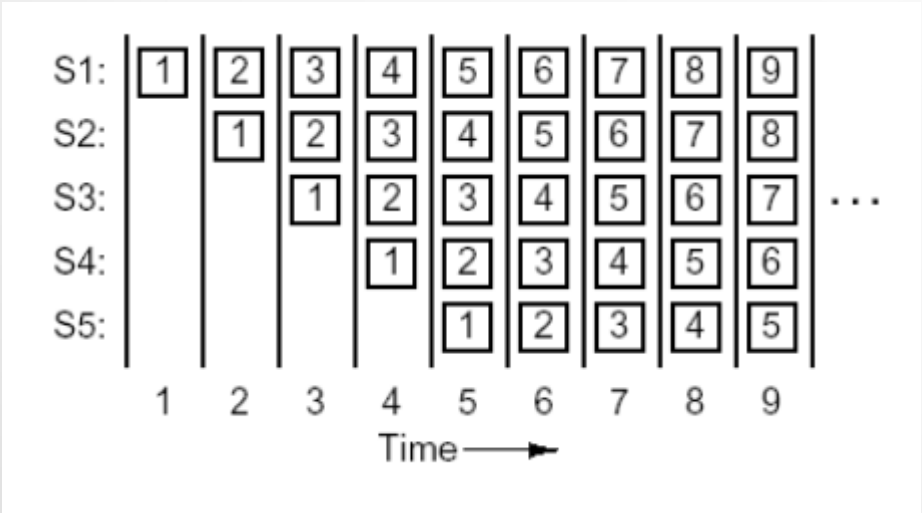
Fino al 2000 l'aumento delle prestazioni di una CPU coincideva per buona parte soprattutto con l'aumento della frequenza di clock. **Siamo arrivati a circa 4 Ghz** Abbiamo raggiunto i limiti fisici (**1 GHz e quindi in un ns la distanza che puo' percorrere l'impulso elettrico, immaginando che viaggia lla velocità delle luce nel vuoto, è di 33 cm circa**):

- Le alte frequenze creano **disturbi** ed **aumentano il calore da dissipare**
- Ritardi nella propagazione del segnale,
- **Bus skew** i segnali che viaggiano su linee diverse viaggiano con velocità diverse

Migliorare le Prestazioni – Pipeline

Ogni singola istruzione viene divisa in più stadi (o fasi) ed ogni fase è gestita da un pezzo di CPU (hardware) dedicato:

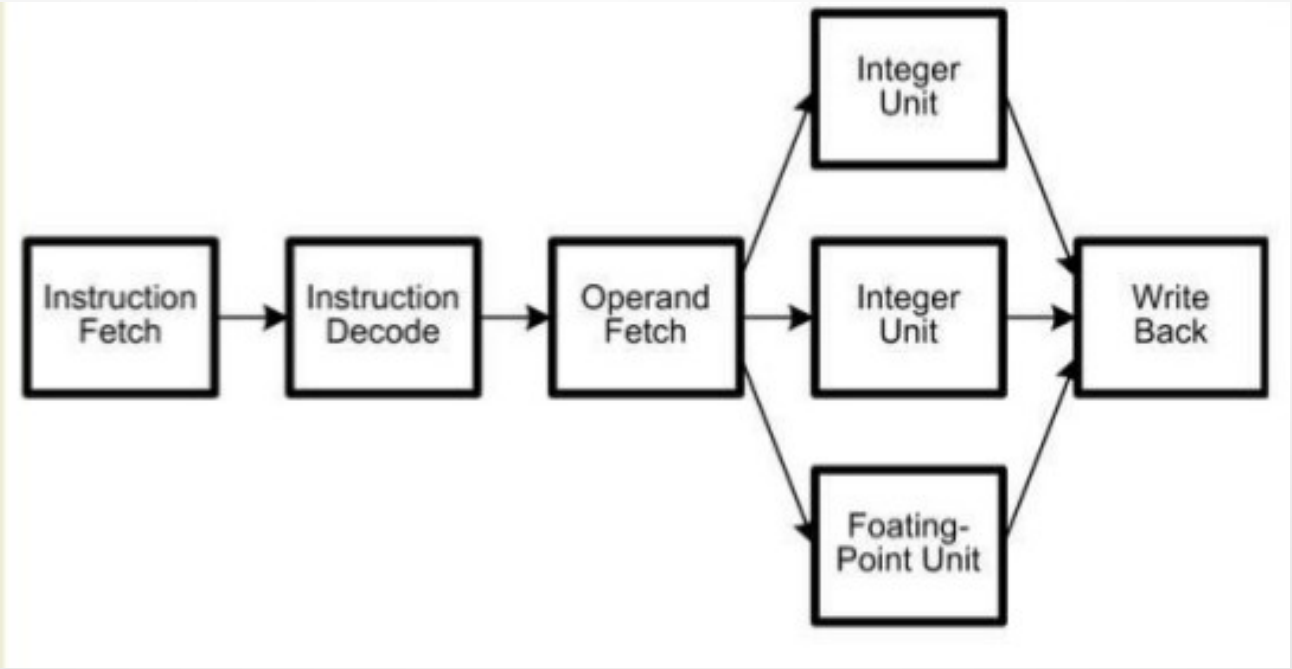




Chiaramente **le operazioni devono essere indipendenti**. In questo caso dopo un tempo (**latenza**) iniziale a caricare la pipeline ci saranno n operazioni eseguite in parallelo. Si possono avere anche più pipeline e quindi vengono lette più istruzioni alla volta ed eseguite

Migliorare le Prestazioni – Superscalari

Istruzioni differenti trattano i propri operandi contemporaneamente su unita' hardware differenti , **in pratica sono presenti diverse unita' funzionali dello stesso tipo, ad esempio sono presenti piu' ALU**



Migliorare le Prestazioni – Predizione di salto

L’uso delle pipeline funziona particolarmente bene in caso di istruzioni sequenziali ma uno dei costrutti basilari della programmazione sono le istruzioni di salto, ad esempio decisioni **IF...THEN...ELSE**:

```
If (a == b)
    printf (“i due numeri sono uguali”);
else
    printf (“Sono diversi \n”);
```

Migliorare le Prestazioni – Predizione di salto

Le CPU moderne possono (pre-fetching) provare ad indovinare se il programma saltera':

- **Predizione statica**: si usano dei criteri che fanno delle assunzioni di "buon senso", ad esempio assumiamo che tutti i salti vengano eseguiti
- **Dinamica**: in pratica la CPU mantiene una tabella che e' basata su una **statistica** di esecuzione

Migliorare le Prestazioni – Esecuzioni fuori ordine e speculativa

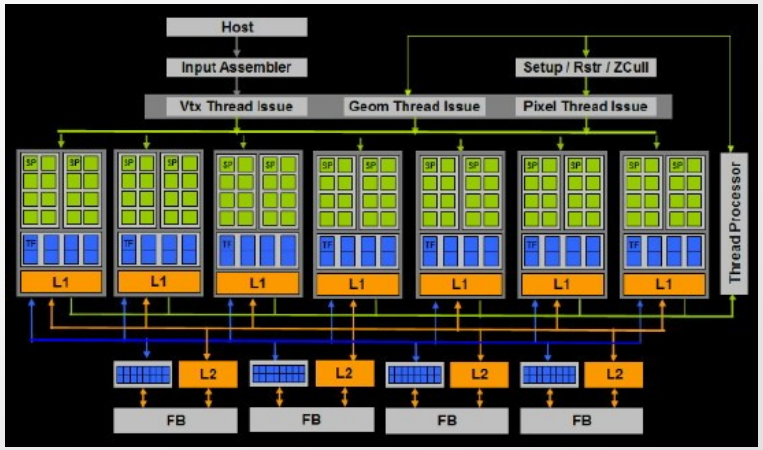
La progettazione delle CPU e' decisamente piu' semplice se tutte le istruzioni sono eseguire in ordina una dopo l'altra, ma come abbiamo visto non possiamo fare questa assunzione a monte. Ci sono delle dipendenze. Ad esempio eseguire una data istruzione richiedere che sia noto il risultato dell'istruzione precedente.

- **Esecuzione fuori ordine**: Le CPU moderne possono per aumentare le prestazione **saltare temporaneamente alcune istruzioni mettendole in attesa** per seguirne altre che non introducono dipendenze
- **Esecuzione speculativa**: eseguire parti del codice particolarmente gravose (pesanti) prima di essere certi che servano davvero

C.P.U. - Considerazioni finali

CPU moderne:

- **Multi-core** : in pratica nello stesso chip ci sono piu' processori indipendenti, **ognuno con le rispettive memoria cache ad esempio , interconnessi fra di loro**. Questo permette di aumentare le prestazioni "teoriche" senza aumentare la frequenza (anche **Hyper-threading**)
- **GPU** (processori grafici) oggi sempre piu' utilizzati nell'accelerazione del calcolo.



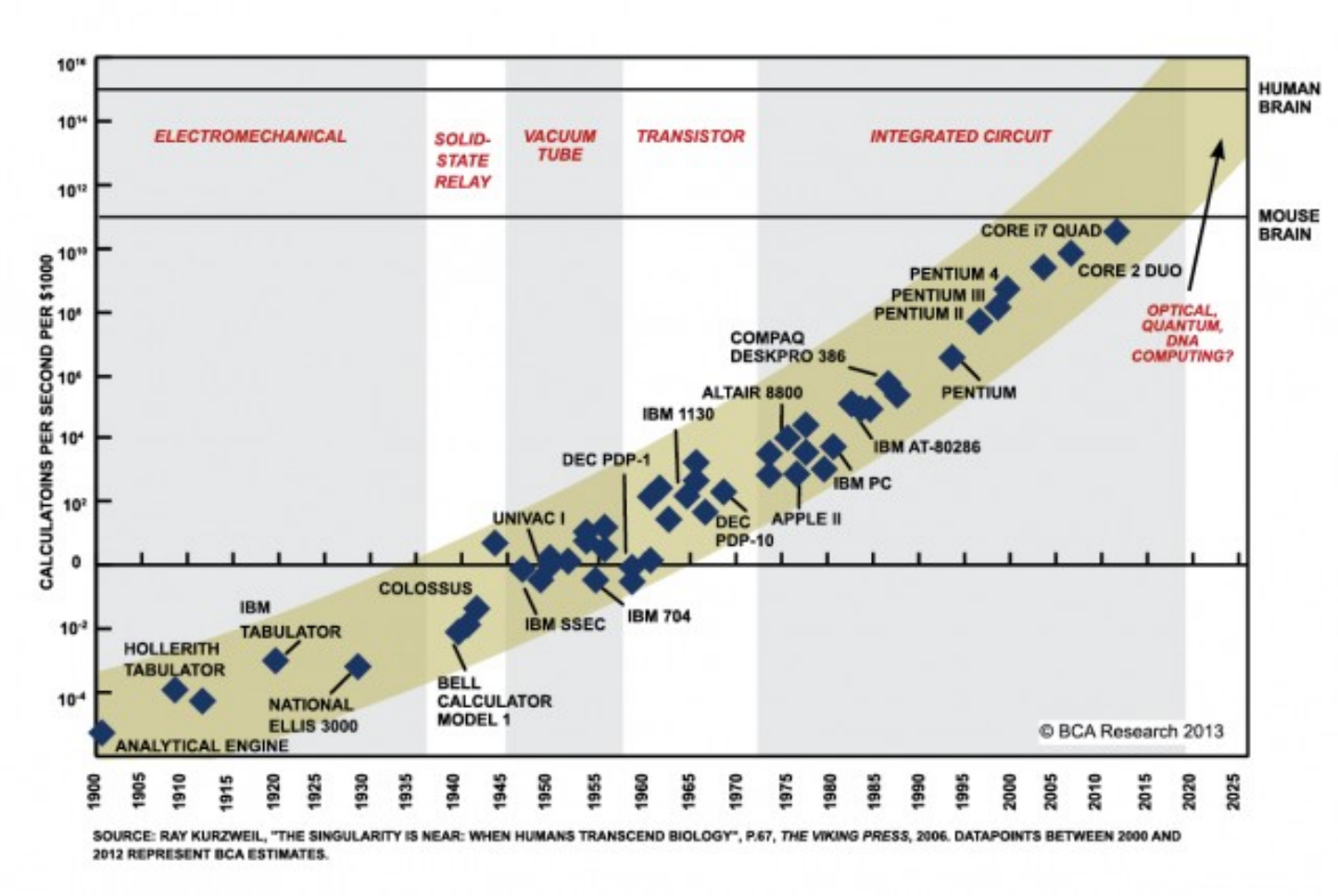
C.P.U. - Considerazioni finali

CPU sistemi embedded (anche Raspberry Pi):

- **ARM** sta ad indicare una classe di processori di tipo RISC (32 bit) (**Advanced RISC Machine**) sviluppati da un'azienda inglese che non li produce direttamente ma detiene le licenze. Questi vengono poi prodotti ad esempio da STMicroelectronics, Samsung, Broadcom, Qualcomm etc etc. (System-On-Chip SOC)
- Sono appunto **CPU RISC** che in funzione del disegno architetturale garantiscono un buon compromesso fra prestazioni e consumo (**ARM e Cortex**)
  - **Il Cortex A9 (1 GHz) consuma 250 mW per core. Una CPU Intel Corei-i7 puo' arrivare a cosumare oltre 100 W**

TOP500 E LEGGE DI MOORE

Legge di Moore



In elettronica e informatica è indicato come prima legge di Moore il seguente enunciato: « **La complessità di un microcircuito, misurata ad esempio tramite il numero di transistori per chip, raddoppia ogni 18 mesi** (e quadruplica quindi ogni 3 anni). »

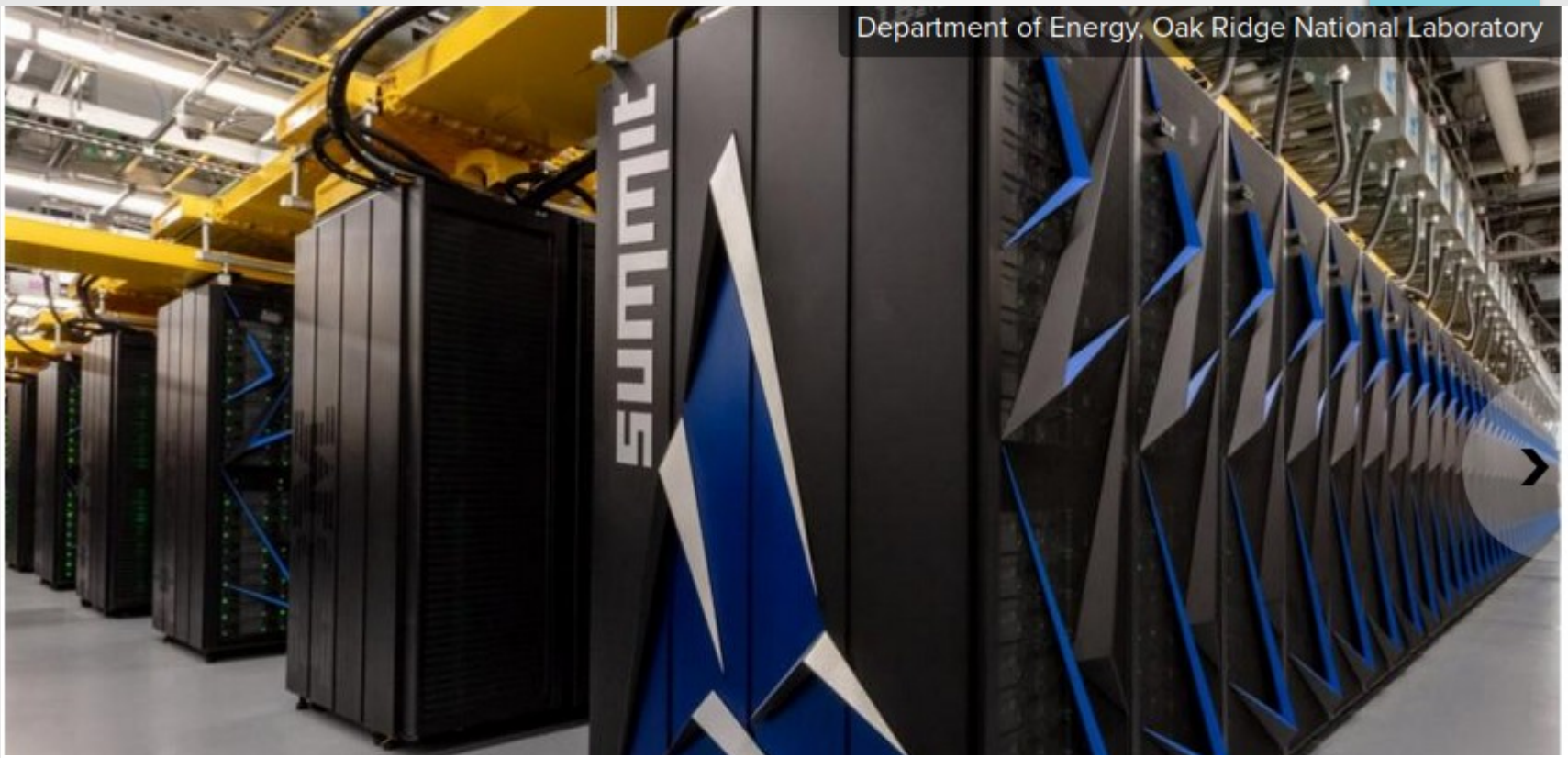
TOP500

- Differenza tra **susteined perfomace e prestazioni di picco**
- Per valutare in modo oggettivo le prestazioni di un computer c'e' bisogno di un test di riferimento, un **benachmark standard, ad esempio Linpack**
- TOP500 http://www.top500.org/, classifica dei 500 computer piu' potenti al mondo

Top500 list Giugno 2018

Rank	Site	System	Cores	Rmax (TFlop/s)	Rpeak (TFlop/s)	Power (kW)
1	DOE/SC/Oak Ridge National Laboratory United States	**Summit** - IBM Power System AC922, IBM POWER9 22C 3.07GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR Infiniband IBM	2,282,544	122,300.0	187,659.3	8,806
2	National Supercomputing Center in Wuxi China	**Sunway TaihuLight** - Sunway MPP, Sunway SW26010 260C 1.45GHz, Sunway NRCPC	10,649,600	93,014.6	125,435.9	15,371
3	DOE/NNSA/LLNL United States	**Sierra** - IBM Power System S922LC, IBM POWER9 22C 3.1GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR Infiniband IBM / NVIDIA / Mellanox	1,572,480	71,610.0	119,193.6	
4	National Super Computer Center in Guangzhou China	**Tianhe-2A** - TH-IVB-FEP Cluster, Intel Xeon E5-2692v2 12C 2.2GHz, TH Express-2, Matrix-2000 NUDT	4,981,760	61,444.5	100,678.7	18,482
5	National Institute of Advanced Industrial Science and Technology (AIST) Japan	**AI Bridging Cloud Infrastructure (ABCI)** - PRIMERGY CX2550 M4, Xeon Gold 6148 20C 2.4GHz, NVIDIA Tesla V100 SXM2, Infiniband EDR Fujitsu	391,680	19,880.0	32,576.6	1,649

Top500 list Giugno 2018



IBM Power System AC922, IBM POWER9 22C 3.07GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR InfiniBand

2.28 million cores

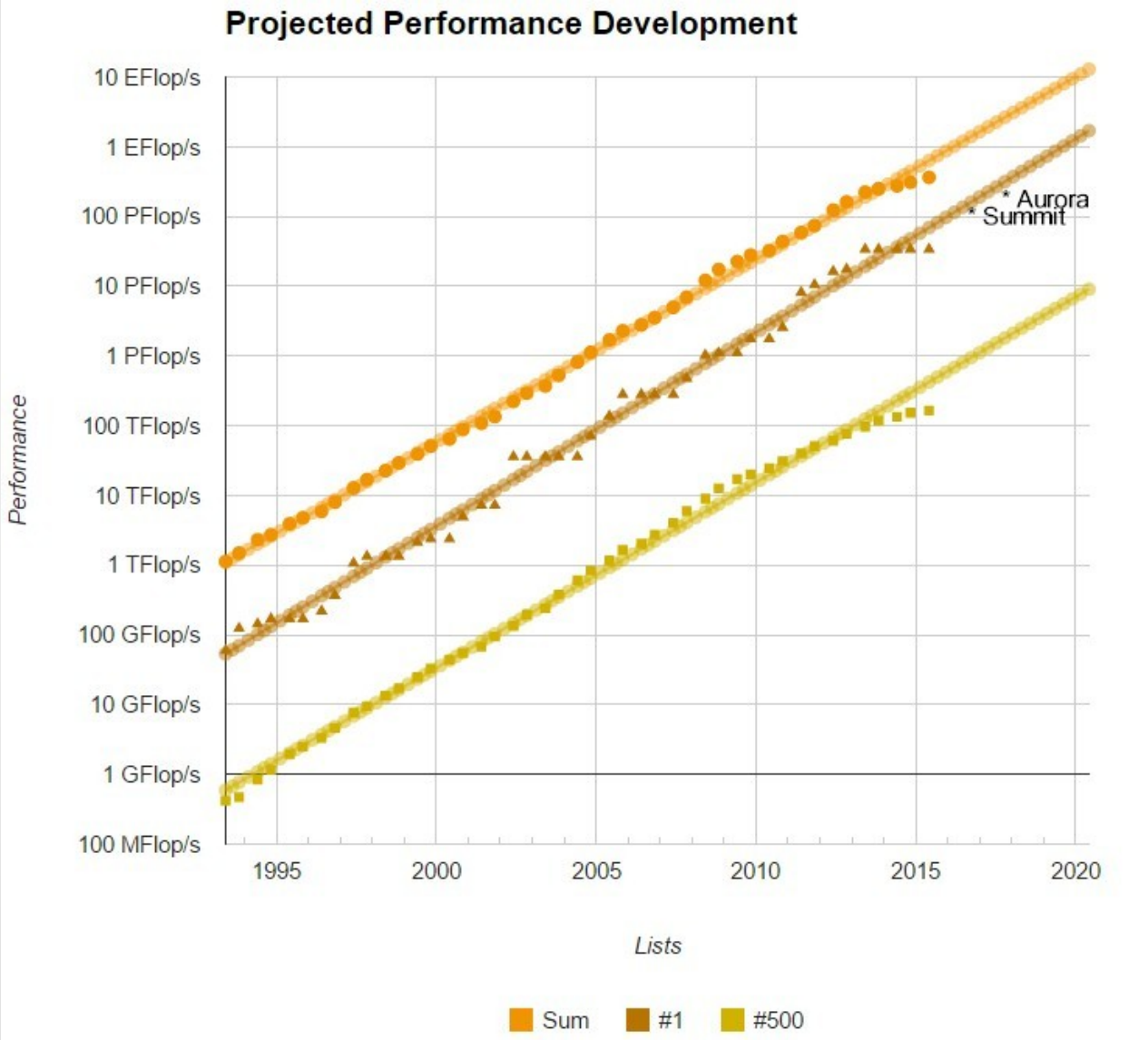
122.3 petaflops

Department of Energy, Oak Ridge National Laboratory

Top500 list Giugno 2017

The Sunway TaihuLight uses a total of 40,960 Chinese-designed SW26010 manycore 64-bit RISC processors based on the Sunway architecture.[5] Each processor chip contains 256 processing cores, and an additional four auxiliary cores for system management (also RISC cores, just more fully featured) for a total of 10,649,600 CPU cores across the entire system.[5]

Top500 list grafico storico



Una stima della **potenza di calcolo della mente umana** qualche decina di **petaflops o oltre**

Introduzione all'Informatica

Loriano Storchi

loriano@storchi.org

http::://www.storchi.org/

SEMPRE A PROPOSITO DELLE ARCHIETTURE

FPGA

- **Field Programmable Gate Array,** e’ un circuito integrato le cui funzionalita’ sono programmabili via software . Gli FPGA (Field Programmable Gate Arrays) sono dispositivi a semiconduttore basati su una matrice di blocchi logici configurabili (CLB) collegati tramite interconnessioni programmabili.
- FPGA possono essere riprogrammati in base all'applicazione desiderata o ai requisiti di funzionalità dopo la produzione. Questa funzione distingue gli FPGA dai circuiti integrati specifici dell'applicazione (ASIC), che sono realizzati su misura per specifiche attività di progettazione. Sebbene siano disponibili FPGA programmabili (OTP) una tantum, i tipi dominanti sono basati su SRAM che possono essere riprogrammati man mano che il design si evolve
- **ASIC (Application Specific Integrated Circuit)**

UNITA’ DI MISURA E PRESTAZIONI

Prestazioni

- **Bandwidth** (Larghezza di banda) la quantita’ di dati (numero di bit) massima di dati che possono essere trasmessi in un canale, La si usa **spesso come approssimazione del rendimento effettivo** (unita’ di misura ad esempio **Mbps**)

- **Throughput** (**Rendimento**) quantita’ di dati (numero di bit) trasmessi sul canale in un certo periodo di tempo (unita’ di misura ad esempio **Mbps**)

Prestazioni

- **Latenza (latency)** o anche ritardo (**delay**) e' il tempo impiegato da un messaggio per andare da un punto all'altro (**unita' di misura il tempo ad esempio ms = millisecondo = (1/1000) s**)
  - Tempi di trasmissione dovuti alla **banale velocita' di propagazione del segnale nel mezzo trasmissivo**, e dunque alla **distanza**
  - **Tempi richiesti per elaborare ad esempio l'intestazione dei pacchetti trasmessi**
- **Round Trip Time (RTT)** tempo impiegato dal messaggio per andare da un punto A ad un punto B e tornare nuovamente da B ad A (**ping e traceroute**)

Prestazioni

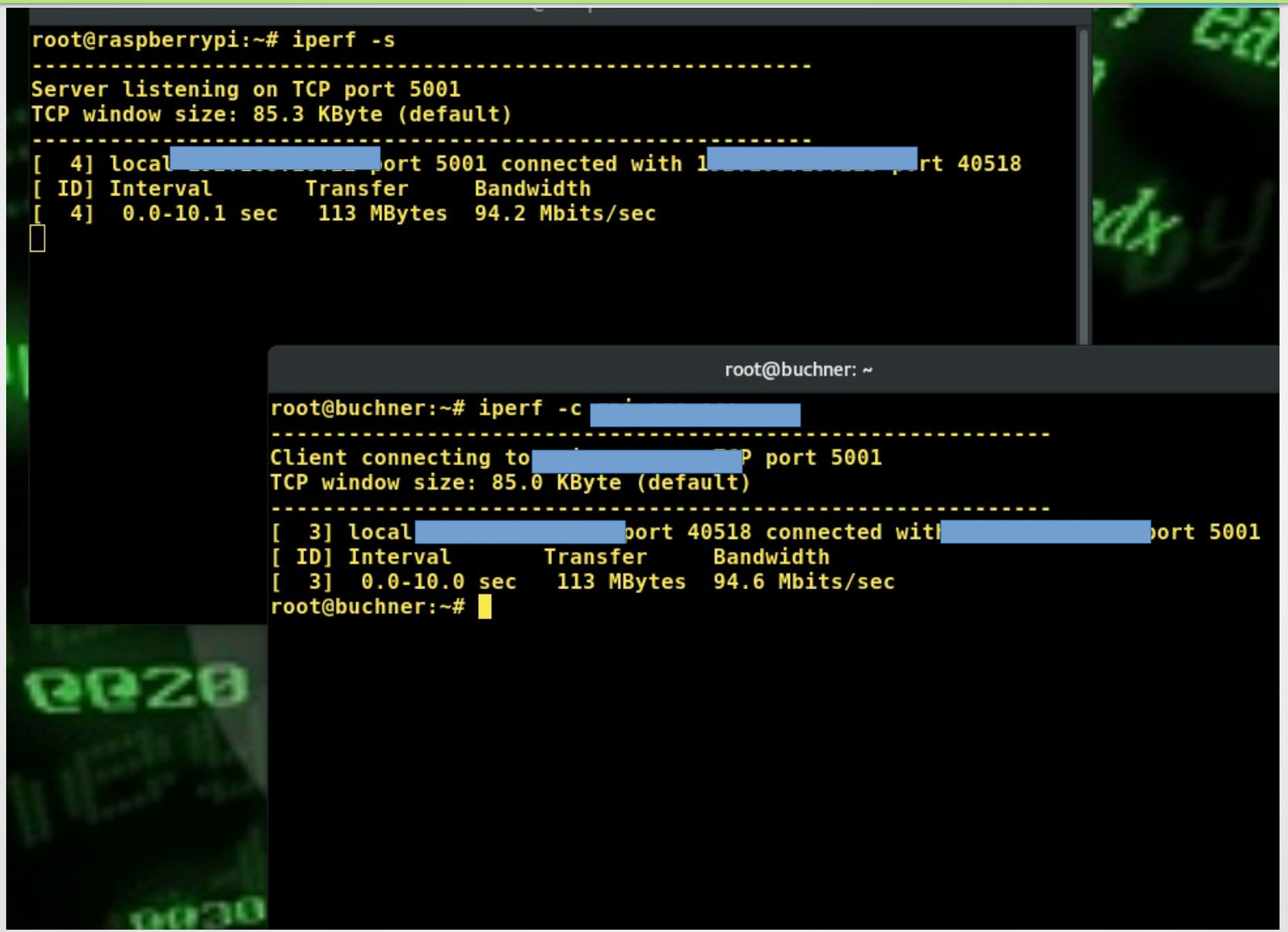
```
redo@eeegw:~$ traceroute www.google.com
traceroute to www.google.com (216.58.205.36), 30 hops max, 60 byte packets
 1  gw.ego.eco (192.168.10.1)  0.497 ms  0.410 ms  0.369 ms
 2  maingw.ego.eco.168.192.in-addr.arpa (192.168.1.1)  1.395 ms  1.278 ms  1.621 ms
 3  * * *
 4  172.18.25.226 (172.18.25.226)  10.313 ms 172.18.25.208 (172.18.25.208)  10.438 ms 172.18.25.230 (172.18.25.230)  10.527
 5  172.18.24.73 (172.18.24.73)  13.664 ms 172.18.24.85 (172.18.24.85)  13.472 ms 172.18.24.81 (172.18.24.81)  12.449 ms
 6  172.19.240.189 (172.19.240.189)  24.534 ms  20.669 ms  20.618 ms
 7  * * *
 8  74.125.51.148 (74.125.51.148)  29.704 ms 72.14.219.236 (72.14.219.236)  35.951 ms 74.125.48.192 (74.125.48.192)  15.297
 9  * * *
10  216.239.42.31 (216.239.42.31)  14.514 ms  14.070 ms  17.031 ms
11  mil04s24-in-f36.1e100.net (216.58.205.36)  15.526 ms  26.030 ms  16.210 ms
```

```
[redo@banquo ~]$ ping 10.0.63.254
PING 10.0.63.254 (10.0.63.254) 56(84) bytes of data.
64 bytes from 10.0.63.254: icmp_seq=1 ttl=64 time=0.209 ms
64 bytes from 10.0.63.254: icmp_seq=2 ttl=64 time=0.196 ms
64 bytes from 10.0.63.254: icmp_seq=3 ttl=64 time=0.181 ms
^C
--- 10.0.63.254 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2074ms
rtt min/avg/max/mdev = 0.181/0.195/0.209/0.016 ms
[redo@banquo ~]$ ping www.google.com
PING www.google.com (216.58.205.196) 56(84) bytes of data.
64 bytes from mil04s29-in-f196.1e100.net (216.58.205.196): icmp_seq=1 ttl=55 time=24.5 ms
64 bytes from mil04s29-in-f196.1e100.net (216.58.205.196): icmp_seq=2 ttl=55 time=24.9 ms
64 bytes from mil04s29-in-f196.1e100.net (216.58.205.196): icmp_seq=3 ttl=55 time=25.0 ms
^C
--- www.google.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 24.582/24.875/25.095/0.215 ms
```

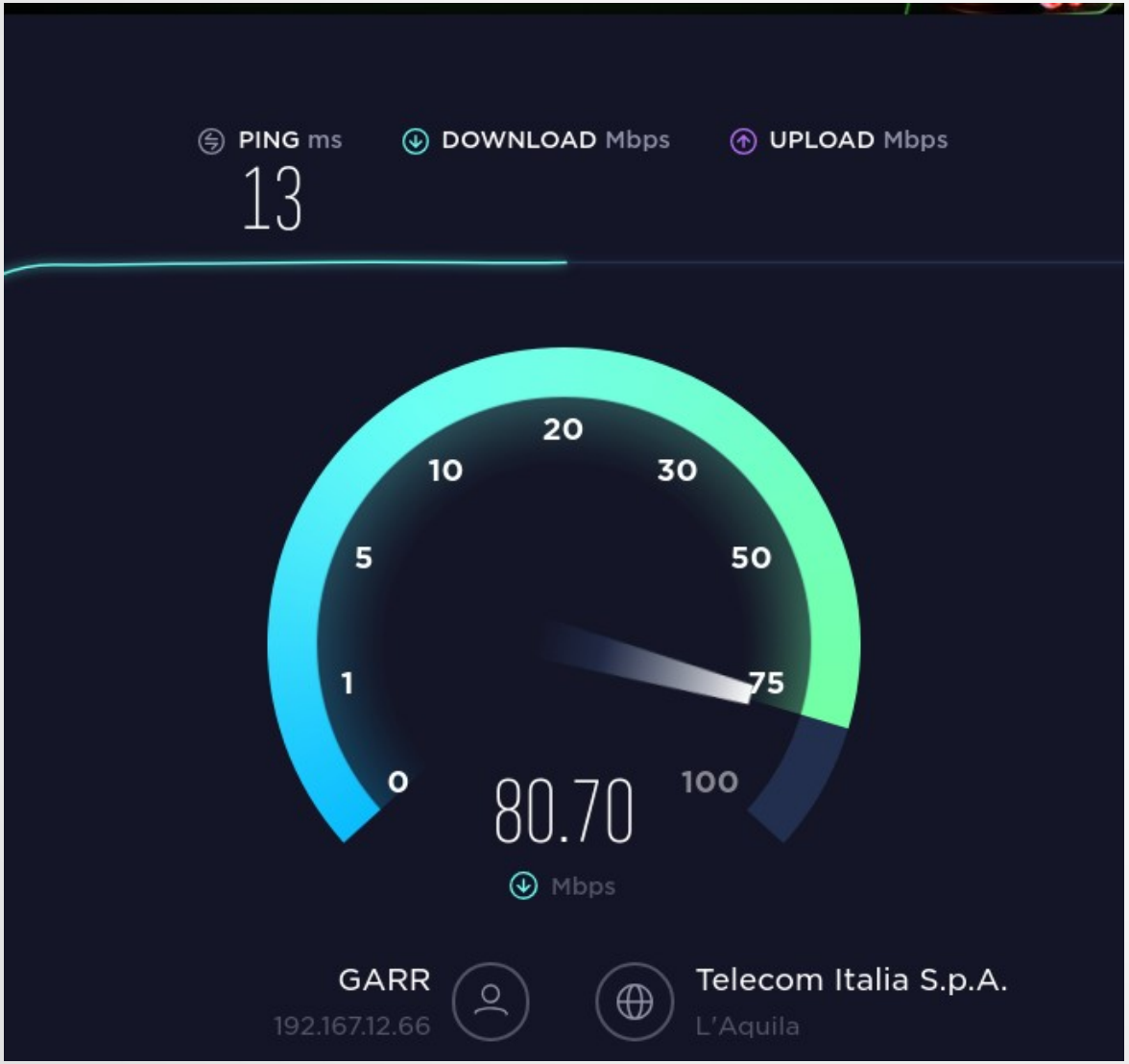
Prestazioni

```
redo@raspberrypi:~$ curl -s https://raw.githubusercontent.com/sivel/speedtest-cl
i/master/speedtest.py | python -
Retrieving speedtest.net configuration...
Testing from Telecom Italia (          )...
Retrieving speedtest.net server list...
Selecting best server based on ping...
Hosted by Inweb Adriatico S.r.l. (Silvi) [18.03 km]: 17.92 ms
Testing download speed.....................................................
.....................
Download: 35.64 Mbit/s
Testing upload speed.......................................................
.....................................
Upload: 15.45 Mbit/s
redo@raspberrypi:~$
```

Prestazioni



Speedtest: facciamo un test



Latenza, e banda

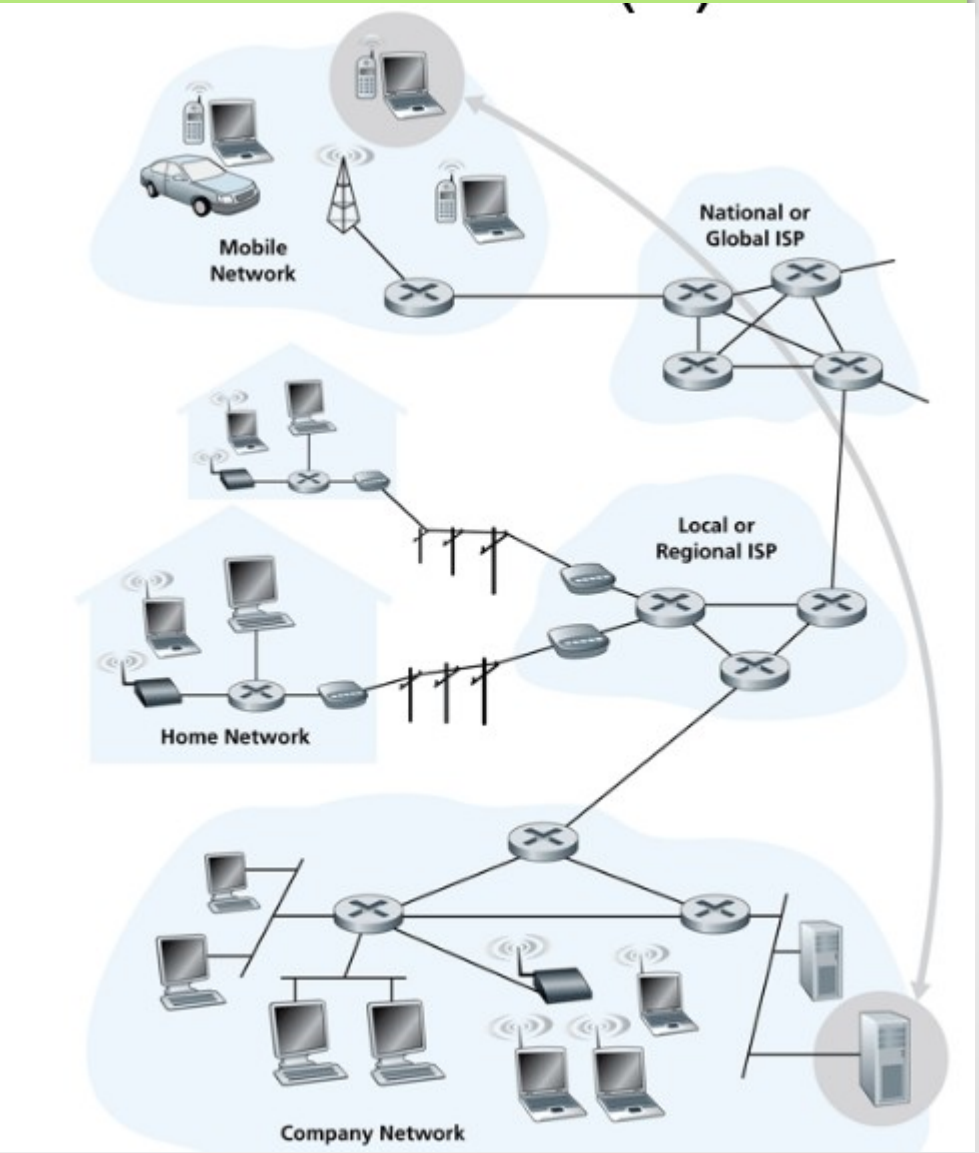
Unita' di misura

- **Velocita' trasmissione dati = quantita' di informazione / tempo di trasferimento**
- In generale questa velocita' viene espressa in bit per secondo cioe' **bit/s** (oppure **bps** detta anche **bit rate**). Si usa anche il byte per secondo **byte/s** (oppure **Bps**).
- Si usano poi i prefissi standard **k (=kilo 10^3) , M (=mega 10^6), G (=giga 10^9)**, quindi non le approssimazioni basaste sulle potenze di due che si utilizzano in ambito informatico.
- Convertire da **bps a Bps e' semplice basta dividere per 8**. Ad esempio ADSL 1**0 Mbps = 10 Mbps/8 = 1250 KBps**
- Dovendo trasferire un file da **10 MiB** con una linea da **5 Mbps** impieghero' circa **(10 * 1024 * 1024 * 8)/ 5 10^6 = 16.8 s** (**trascurando** la **latenza**)

RETI DI COMPUTER

Reti di Calcolatori

Un insieme di calcolatori autonomi collegati, la rete e' vista come una **fornitrice di canali logici** attraverso i quali le varie applicazioni possono comunicare fra di loro.

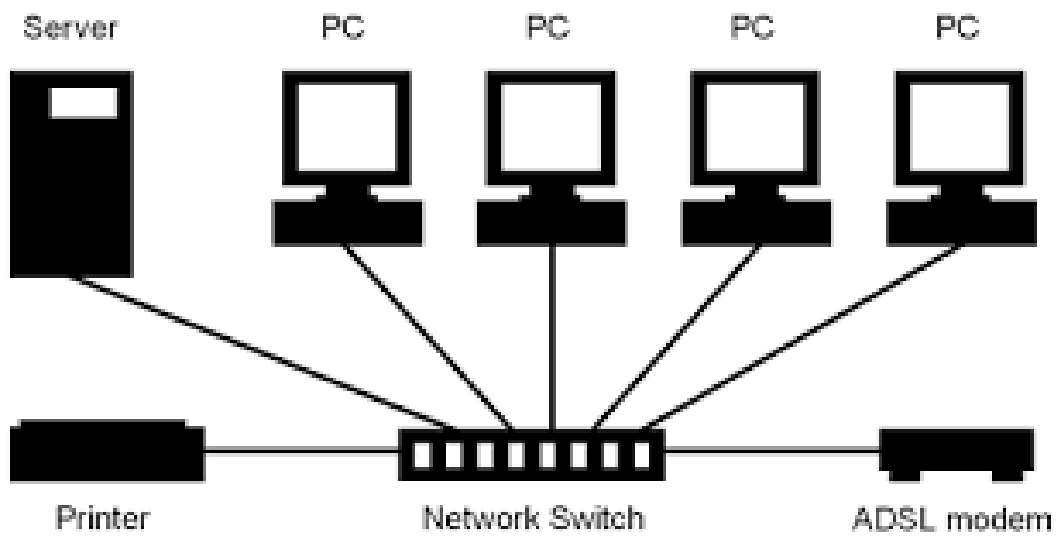


Reti di Computer

- Un insieme di calcolatori autonomi collegati
  - **Nodi (host)** possono essere sia server che desktop PC, oppure dispositivi mobili o altro
  - Collegamenti, sono i canali che permettono ai nodi di comunicare. Ovviamente posso avere **mezzi di trasmissioni molto diversi, ad esempio il doppino telefonico, oppure fibre ottiche, oppure canali radio (wireless)**
    - Posso avere collegamenti diretti od indiretti (**switch**)

Switch

- switch di rete (chiamato anche hub di commutazione, hub di bridging) è un dispositivo di rete che collega i dispositivi insieme computer utilizzando la **commutazione** di **pacchetto** per ricevere, elaborare e inoltrare i dati al dispositivo di destinazione.



Reti di Computer

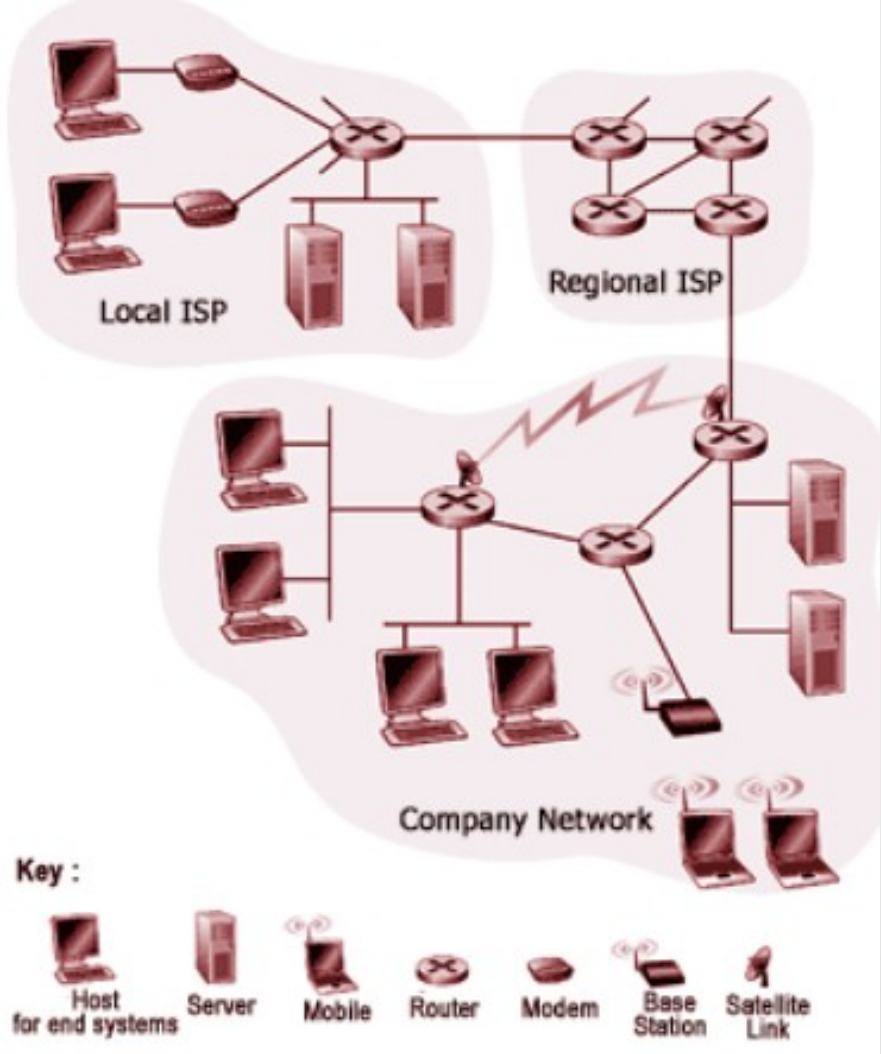
- Posso caratterizzare le reti in base alla loro estensione
  - **LAN (Local Area Network)**: rete su scala locale, si tratta di reti estese a livello di una singola stanza o al massimo di un edificio
  - **MAN (Metropolitan Area Network)**: puo' ad esempio collegare piu' LAN
  - **WAN (Wide Area Network)**: reti estese su aree geografiche. Connettono assieme LAN e MAN (Internet e' la WAN per eccellenza)

INTERNET

Internet

Possiamo descrivere internet innanzi tutto **dal punto di vista della componentistica di base**.

Internet interconnette milioni di dispositivi in tutto il mondo, i tradizionali PCs desktop, i dispositivi mobili, ed i cosi' detti servers (che immagazzinano e trasmettono dati, come pagine html, e-mail etc etc). Questi dispositivi sono detti **hosts o end systems**

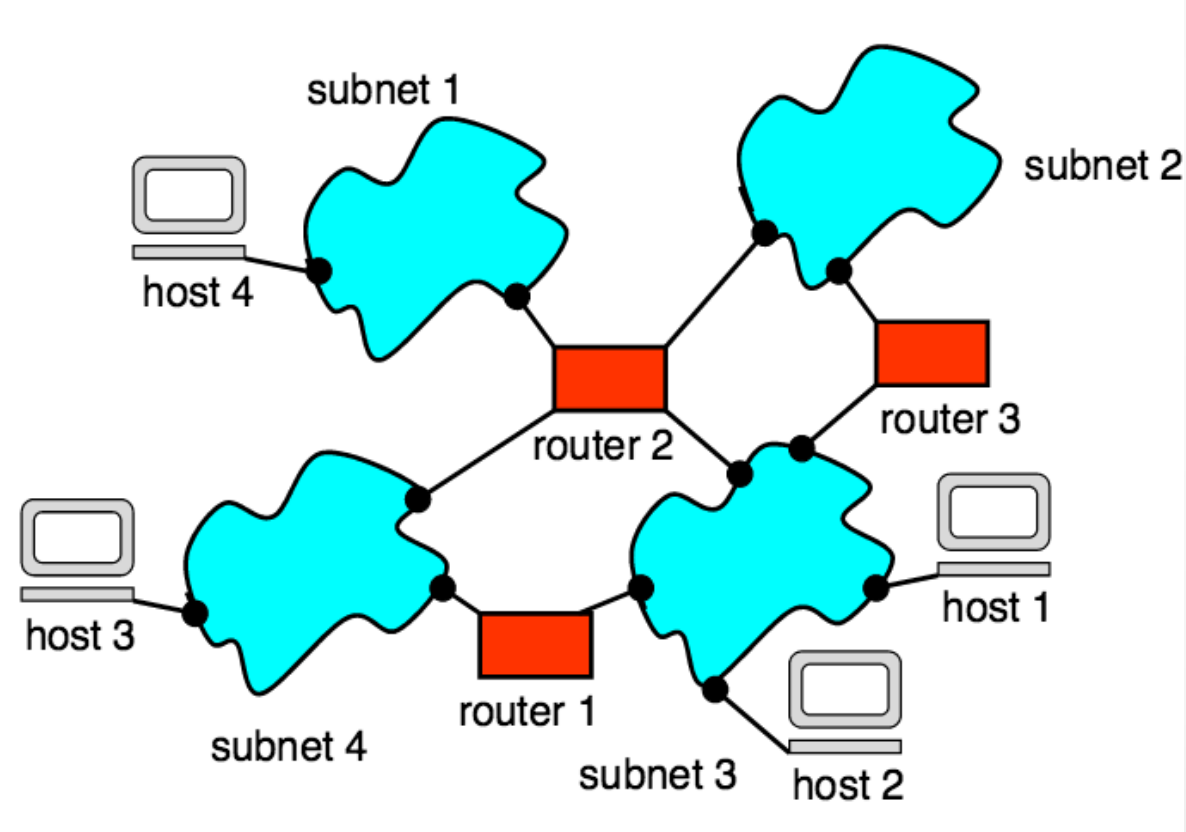


Internet

- I vari hosts sono interconnessi mediante canali di comunicazione. Questi canali possono essere di natura molto diversa , ad esempio cavi coassiali, fili di rame, fibre ottiche o ponti radio
- **Generalmente gli hosts non sono interconnessi direttamente ma ci sono dei dispositivi di “switching” detti router**. **I router servono ad instradare il traffico dati** , e quindi prendono informazioni che arrivano da un canale e le inviano ad un altro canale
- Internet e’ un insieme di reti interconnesse fra di loro . **I vari hosts, cosi’ come gli altri devices infrastrutturali comunicano fra di loro usando protocolli comuni** (IP Internet Protocol e TCP Transmission Control Protocol sono i due protocolli principali)

Internet

Internet oggi interconnette migliaia di sottoreti



Host: computer collegato ad internet, puo' essere si un client che un server a livello applicativo

Router: nodo ce serve ad instradare il traffico (**a Layer 3 network gateway device**,)

Sottorete: insieme di host fra cui c'e' un collegamento di livello 2 (ad esempio una LAN)

Ad oggi ordine di miliardi di Host

Internet: Breve storia

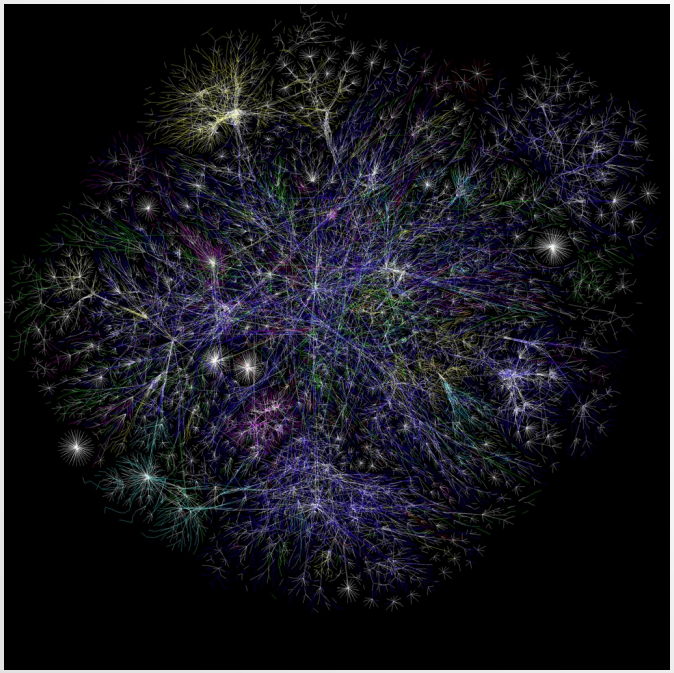
- **1969 parte la progettazione della rete militare ARPANET** . Questa rete viene progettata fra le altre cose anche con lo scopo di resistere ad attacchi nucleari. Infatti era in grado in caso di connettere i dispositivi interconnessi seguendo in caso di rotture strade diverse
- **1972 nasce la posta elettronica (e-mail) trasferimento di files via FTP e collegamento remoto (Remote Login)**
- **1974** vengono **ufficialmente** presentati i protocollo **IP e TCP**.
- **1979** data di nascita della rete **CSNet** che interconnette le **Universita' ed i centri di ricerca degli Stati Uniti**

Internet: Breve storia

- **1982 ARPANET e CSNet** vengono collegate, questa viene considerata la **data di nascita ufficiale** in qualche modo di Internet
- **1990 NSFNet, rete che collega supercomputer soppianta Arpanet.** Questo apre la strada ad usi civili e commerciali di Internet.
- **1990** Stesso anno **Tim Bernes-Lee che allora lavoravga al CERN di Ginevra inventa l'HTML** (Hyper Text Markup Language) che consente la gestione informazioni di natura diversa, testo, immagini etc etc. Questo e\ il primo passo verso il WWW (World Wide Web)

Internet: Breve storia

- **1993** viene realizzato **Mosaic** il primo browser
- **1994 nasco Yahoo!** Il primo motore di ricerca . Nella seconda meta' degli anni 90 ne vengono realizzati molti altri fino ad arrivare al **1998 data della nascita di Google**.



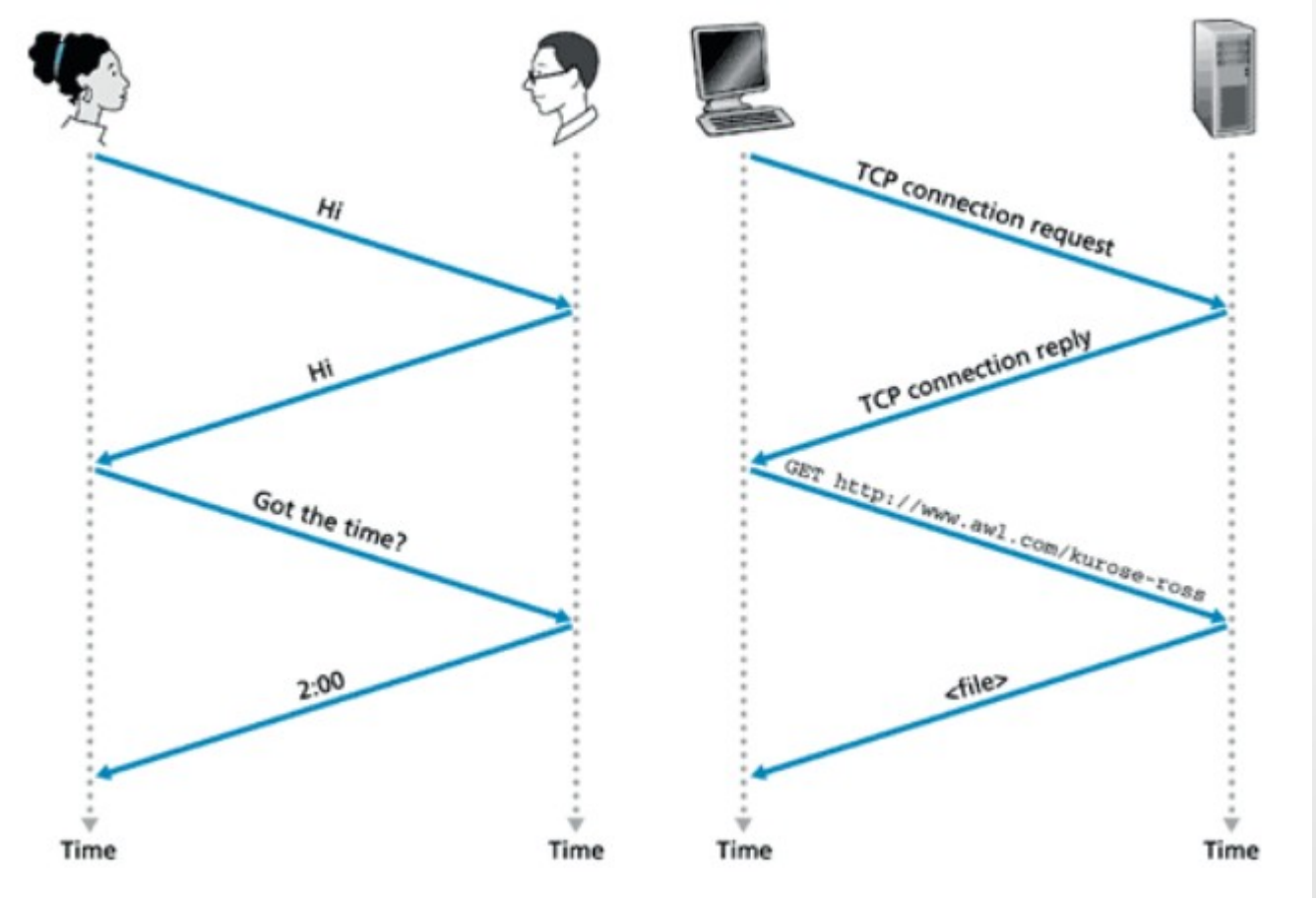
Internet oggi contiene circa **10 miliardi di computer, ogni computer (dispositivo) contiene in media un paio di miliardi di transistor. Quindi internet interconnette 10^{19} transistor , in conclusione ci sono 10000 volte piu' transistor che sinapsi nel cervello umano**

Internet oggi usato nella vendita di beni e servizi. Trasmissione audio video, collaborazione in rete, lavoro, giochi in rete, interazioni sociali.

INTERNET

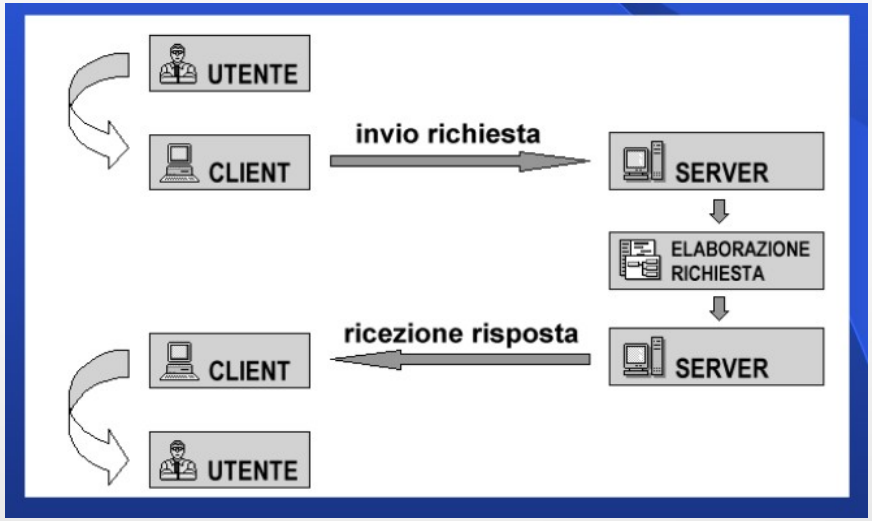
Protocollo di comunicazione

- Insieme di regole (formalmente descritte) che definiscono le modalita’ di comunicazione tra due o piu’ entita’.



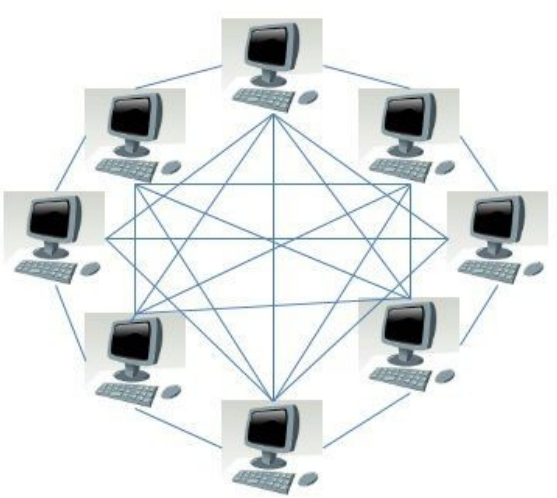
Modello di rete: Client-Server

- La maggior parte dei servizi telematici offerti da internet si basano sulla **modalita' di interazione client/server (diverso ispetto al P2P)**.
- Il client e' dotato di un particolare software client in grado di inviare richieste di servizio ad un articolare server. Il clint formatta le richieste in modo adeguato e comprensibile al server, usando quindi uno specifico protocollo (es: browser server web)



Modello di rete: P2P Peer-to-Peer

- In questo caso **non c'e' una distinzione fra client e server , ogni nodo puo' invece agire sia da client che da server** dipendentemente se il singolo nodo stia richiedendo oppure fornendo dati
- Per diventare parte della rete **P2P dopo il join il nodo deve iniziare a fornire servizi e potra' chiedere a sua volta servizi** ad altri nodi (es: BitTorrent)



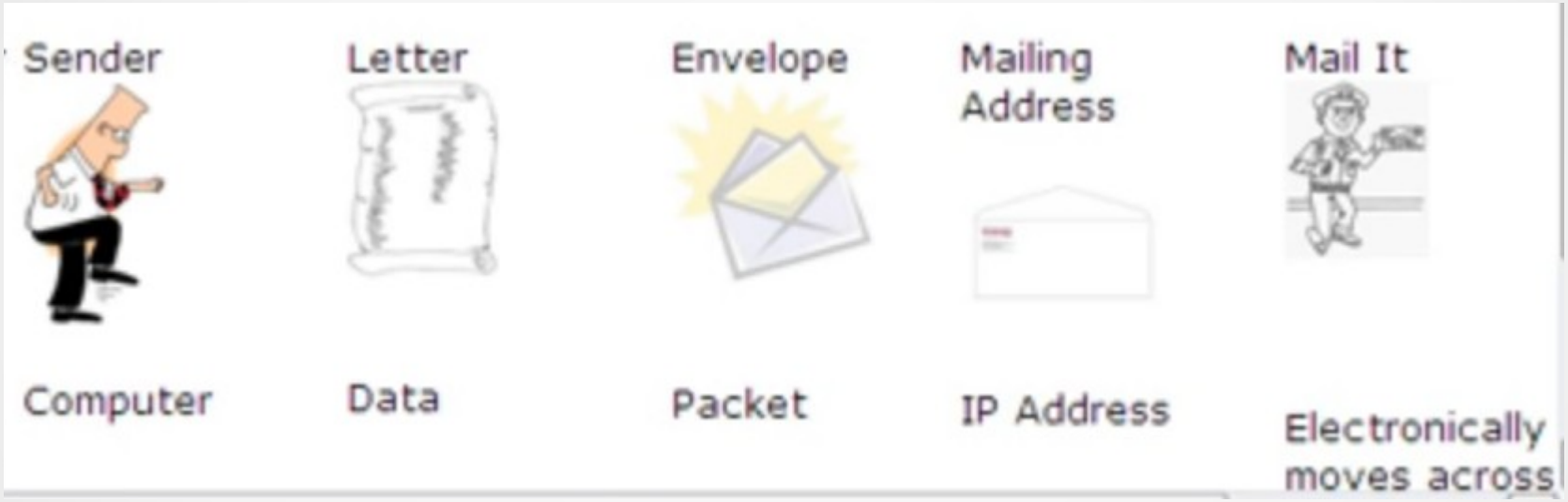
Peer-to-Peer Network Model

TCP/IP

TCP/IP: funzionamento base

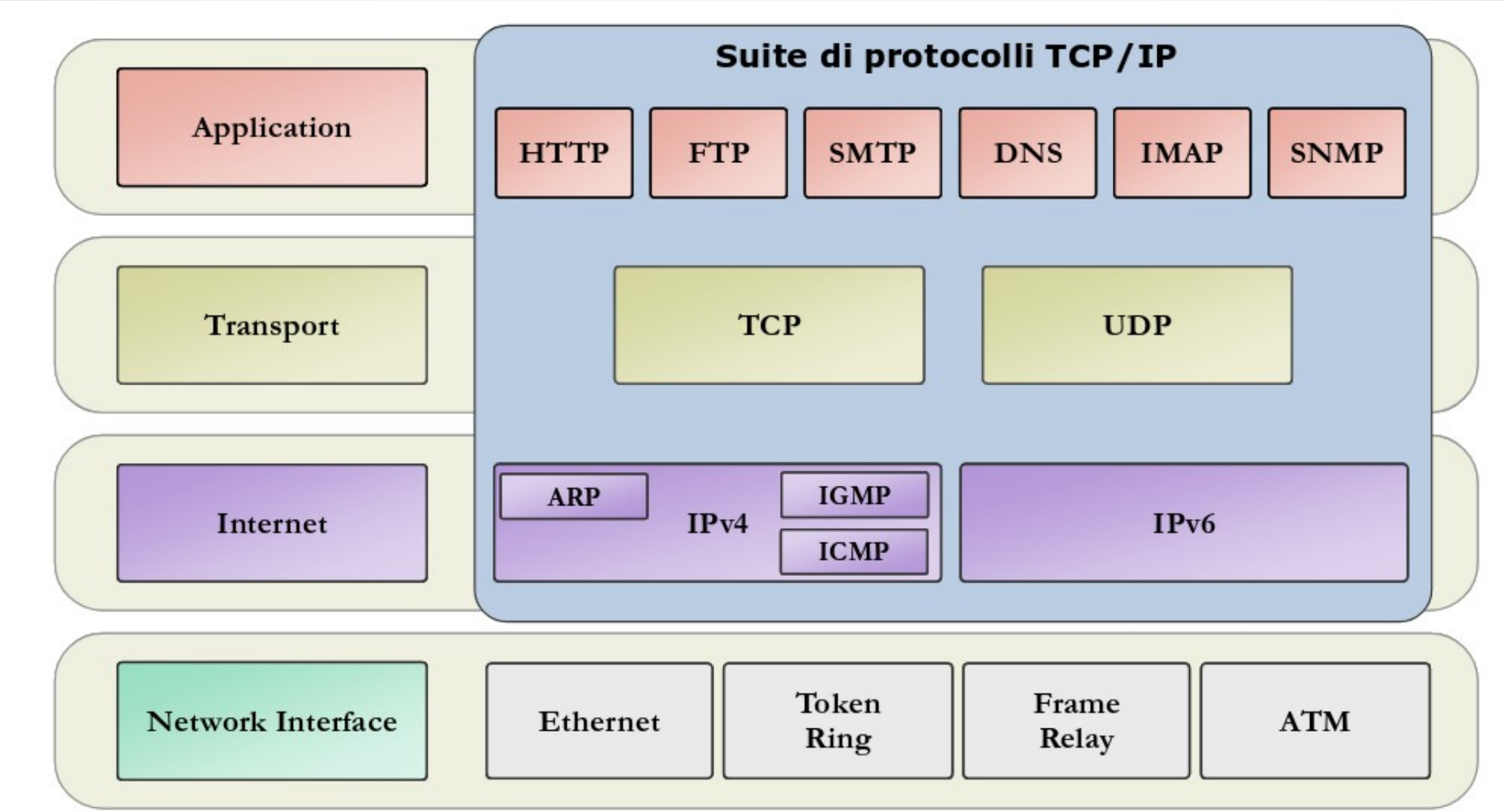
- Un protocollo e' un insieme di regole che chi spedisce dati (mittente) e chi li riceve (destinatario) devo seguire
- Esempio banale di un “protocollo” (non formale) che due persone seguono quando si incontrano ed una chiede l’ora:
  - Bob saluta Alice
  - Alice saluta Bob
  - Bob chiede l’ora
  - Alice dice l’ora
  - Bob ringrazia e saluta
  - Alice saluta

TCP/IP: funzionamento base



Ed alla fine una ricevuta di ricezione (ACK)

Architettura TCP/IP



TCP/IP: funzionamento base

- Idea del **layering**. Ad esempio il protocollo **HTTP e' costruito sopra il TCP**, Un browser eb non si deve preoccupare di come e' implementato il TCP, ma deve solo sapere che funziona.
- **I dati che il livello di trasporto riceve da quello applicativo sono frammentati in pacchetti.** I pacchetti di dati sono ricomposti a destinazione (**ogni pacchetto puo' seguire strade diverse**)
- **Il TCP aggiunge ad ogni pacchetto delle informazioni aggiuntive** come in particolare il numero d'ordine della sequenza di cui il pacchetto fa parte.
- Il pacchetto viene poi passato al livello di rete dove **IP instrada i pacchetti verso l'host di destinazione** nel modo piu' opportuno

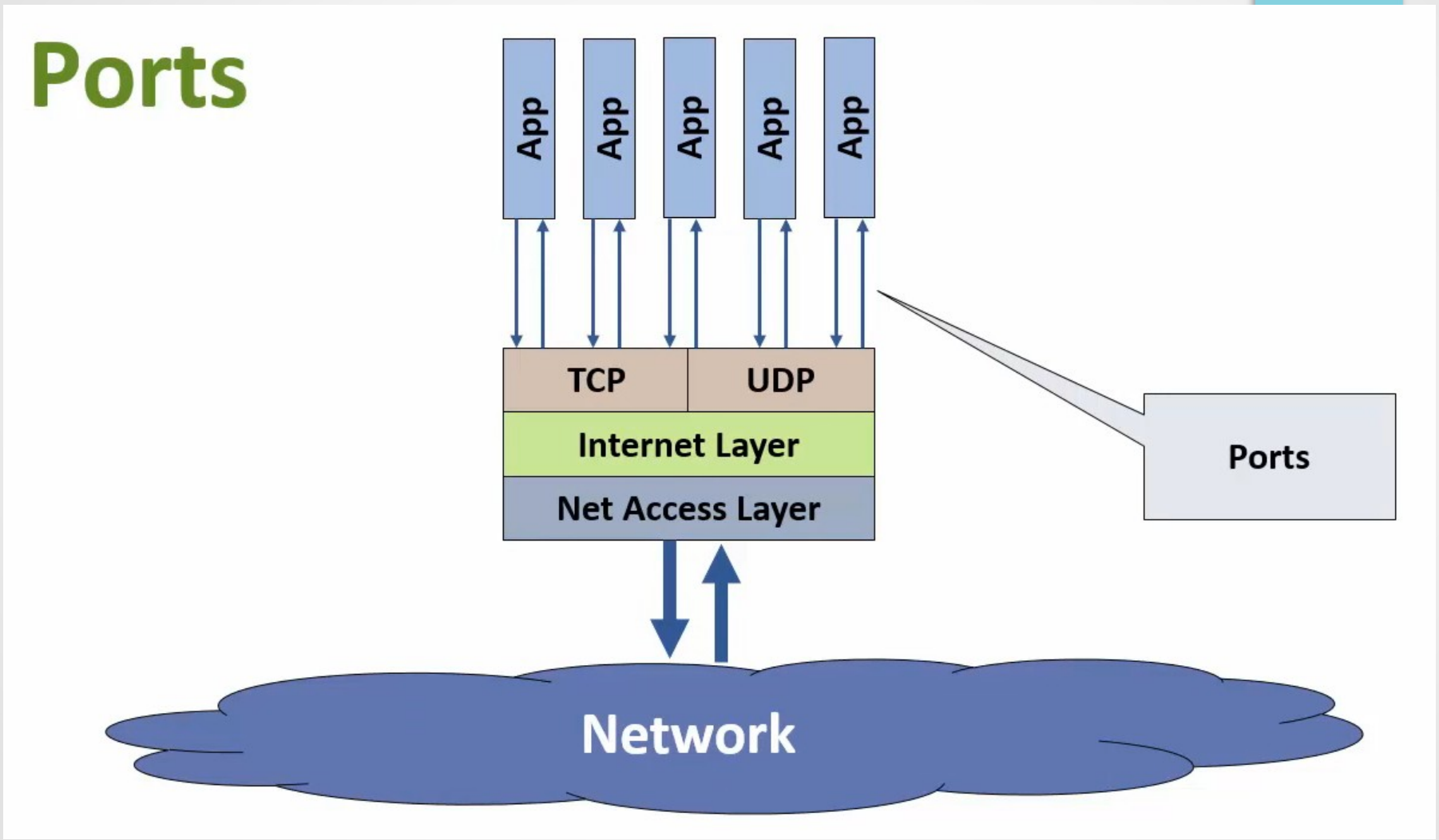
Lo STACK TCP/IP: Livello applicativo

- **Livello applicativo**: troviamo numerosi protocolli come HTTP, FTP, DNS, SMTP, etc etc. A questo livello due applicazioni si scambiano **messaggi** senza preoccuparsi di come questi verranno consegnati.
- Questa e' l'interfaccia con l'utente , ad esempio se consultiamo una pagina web il protocollo gestisce la sessione di interazione fra il nostro browser (client) ed il server web

Lo STACK TCP/IP: Livello di trasporto

- **Livello di Trasporto**: a questo livello troviamo i due protocolli base **TCP** ed **UDP ,** a questo livello due hosts si scambiano i segment .
- I protocolli a questo livello **offrono il servizio di trasporto** al livello applicativo, Per gestire ad esempio **piu' sessioni attive contemporaneamente il TCP e l'UDP usano diversi numeri di porta** (porte logiche)
- **TCP**
  - Ad ogni finestra di pacchetti spediti il TCP fa partire un contatore di tempo
  - Chi riceve invia un **ACK** se ha ricevuto il pacchetto
  - Se chi trasmette non riceve un **ACK** prima che scada il tempo (oppure…) . Il trasmettitore si occupa ad esempio di rinviare i dati

Lo STACK TCP/IP: Livello di trasporto



Lo STACK TCP/IP: Livello di rete

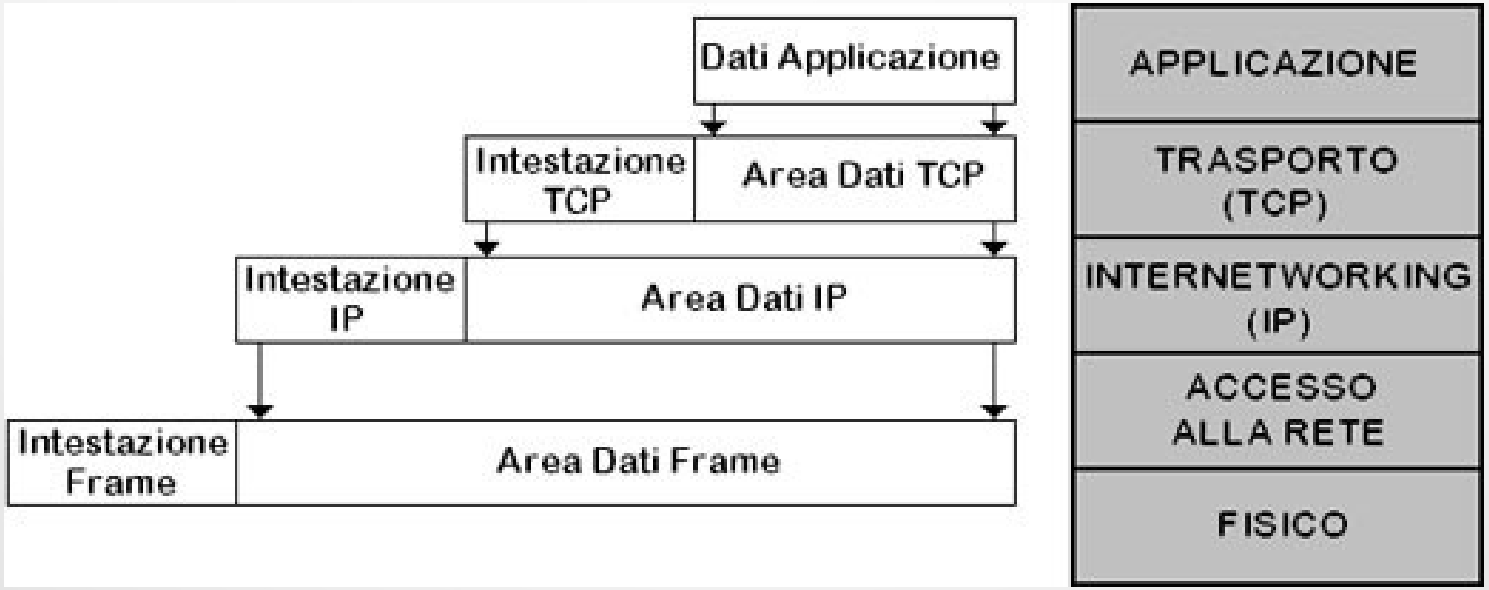
- **Livello di Rete**: a questo livello troviamo il protocollo **IP. Questo protocollo si occupa dell'indirizzamento e dell'instradamento dei pacchetti fra mittente e destinatario.**
- **Indirizzamento:** Ogni nodo e' identificato in modo non ambiguo da un **indirizzo IP**
- **Instradamento:** questa funzionalita' consente di selezionare il percorso migliore da seguire per far transitare i dati dal mittente verso il destinatario

Lo STACK TCP/IP: Livello di accesso alla rete

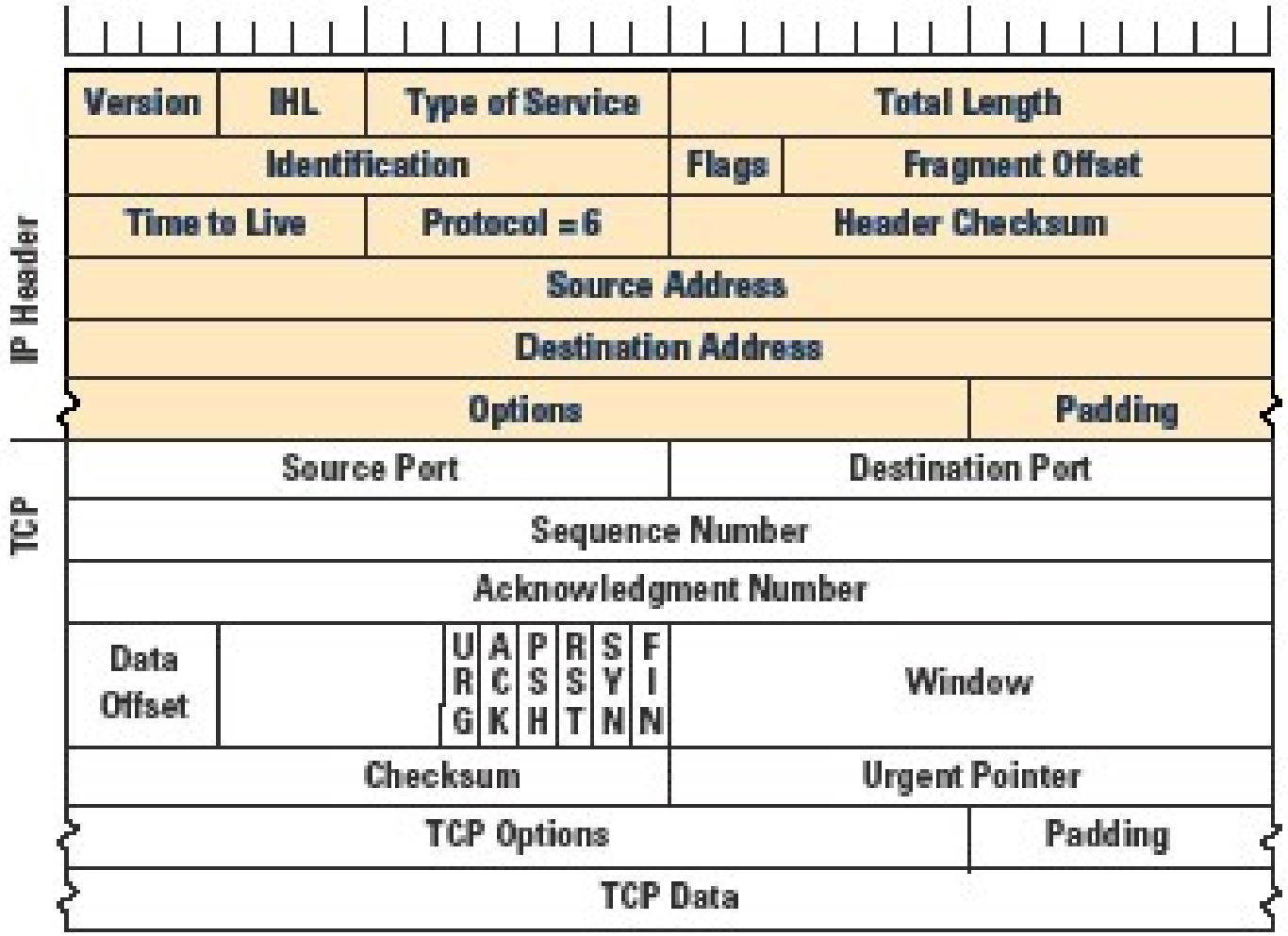
- Il modello TCP/IP, specifica solo che sotto al livello IP ci deve essere un livello di accesso alla rete che si occupi fattivamente di spedire i pacchetti.
- Al livello di collegamento i protocolli decidno come il messaggio debba essere trasferito per ogni tratto del percorso. Quindi ad esempio come andare dal primo host al primo router e cosi' via (**indirizzi MAC, ed ethernet**)
- A livello fisico poi i dati sono convertiti in **segnali elettrici od elettromagnetici o ...**

Lo STACK TCP/IP

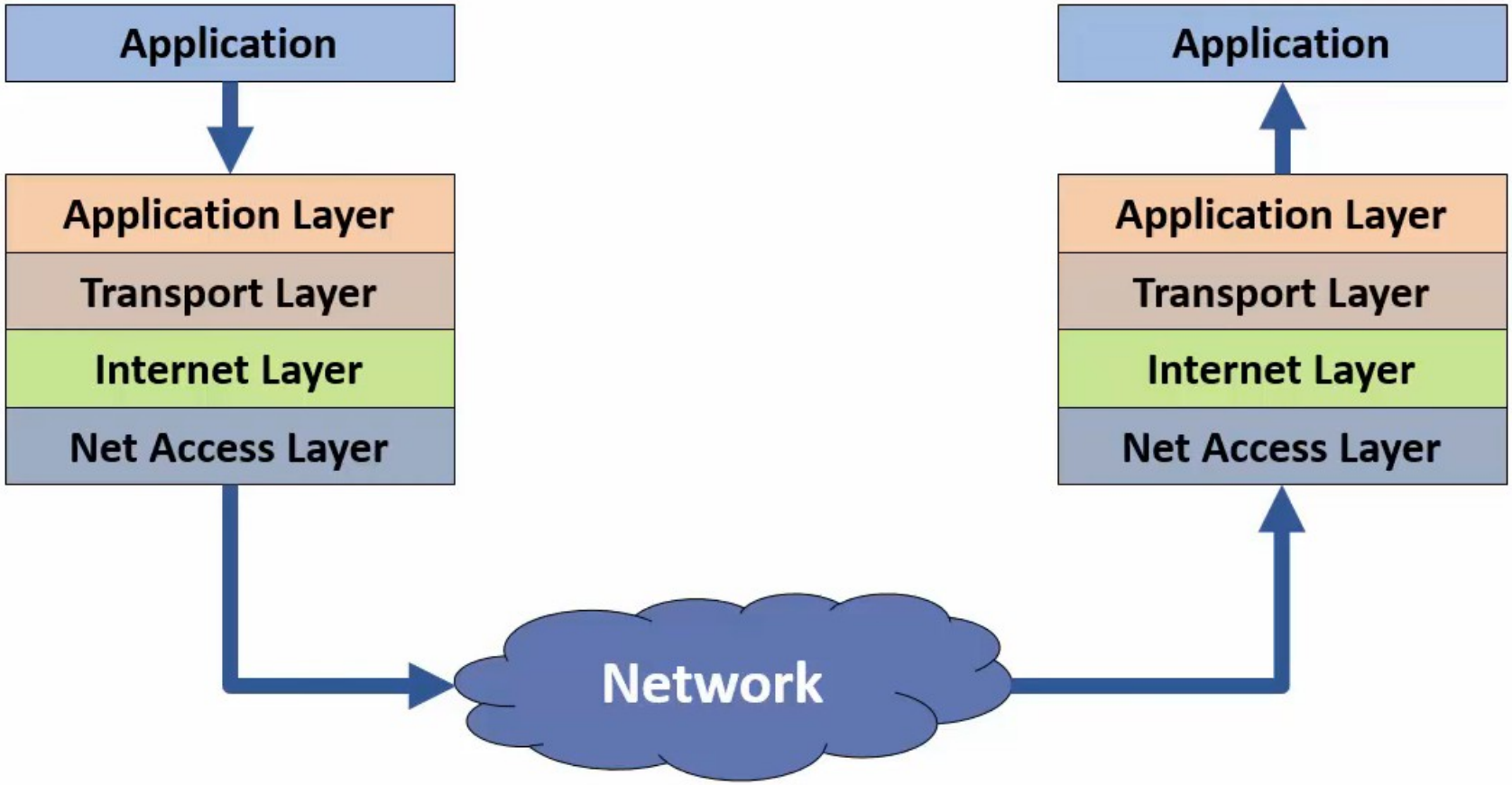
- Quando un’applicazione deve inviare dati, questi vengono passati al livello inferiore, di volta in volta fino a raggiungere la rete fisica sottostante. Durante questo percorso, ogni strato aggiunge informazioni ai dati, fino a creare un “frame di rete” (incapsulamento):



TCP/IP: headers



TCP/IP: headers



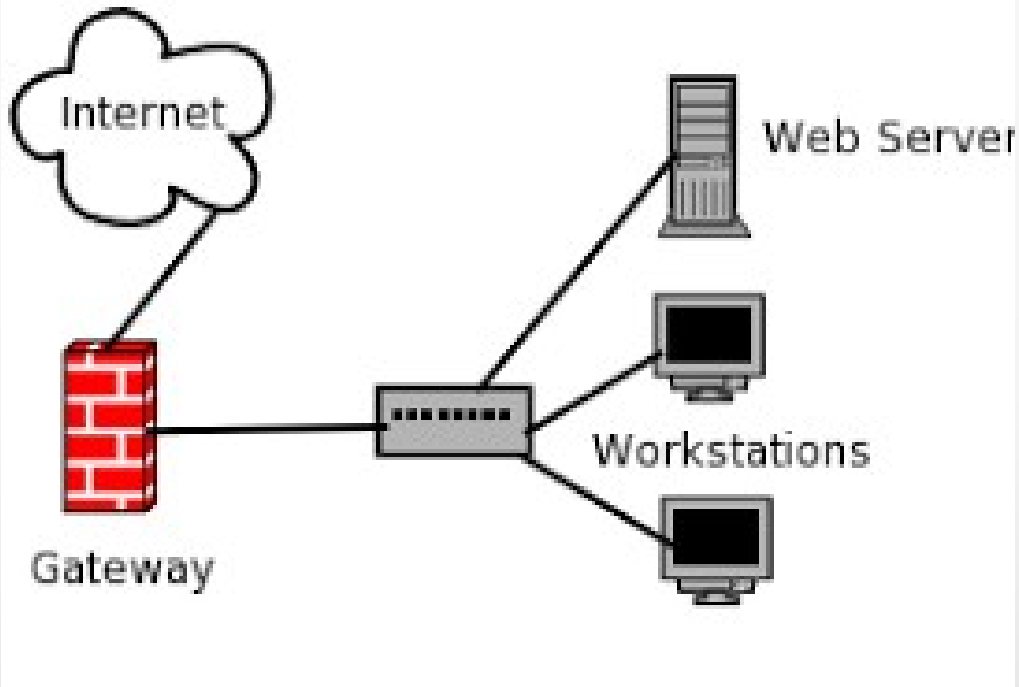
INDIRIZZI IP, DNS, DHCP AND ...

Indirizzi IP

- **Ogni computer collegato ad internet e' identificato dal suo indirizzo IP**, composto da **4 gruppi** di un byte ciascuno (**complessivamente 32-bit**). Ogni numero puo' assumere valori da **0 a 255**
- Ad esempio: **192.167.12.66** (IP statici o Dinamici, IP privati …)
- L'ultimo numero identifica solitamente un Host, i numeri precedenti la sottorete a cui questo Host appartiene.
- **Il massimo numero di indirizzi IPv4 e' dunque 255*255*255*255**
- **IPv6**: **128-bit** e quindi 2^{128} circa 3.4×10^{38} indirizzi (**IoT: Internet of things**)

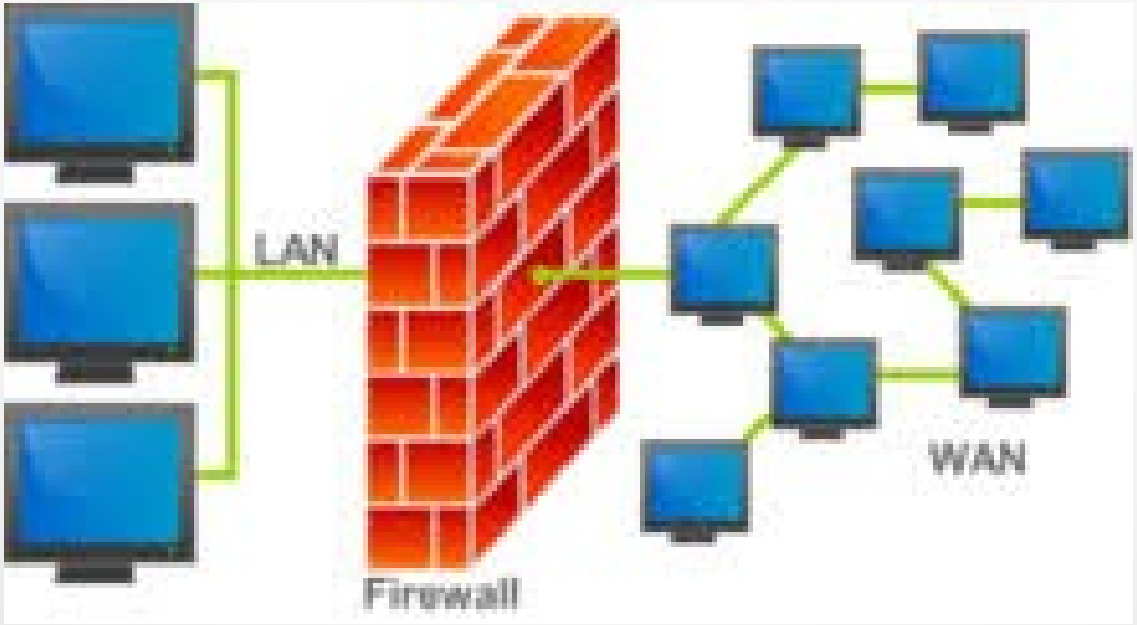
Indirizzi IP: gateway

- Il **gateway** di default viene utilizzato per instradare pacchetti verso altre destinazioni
- Quasi sempre il protocollo **DHCP** viene usato per fornire automaticamente al client l'indirizzo IP del gateway predefinito



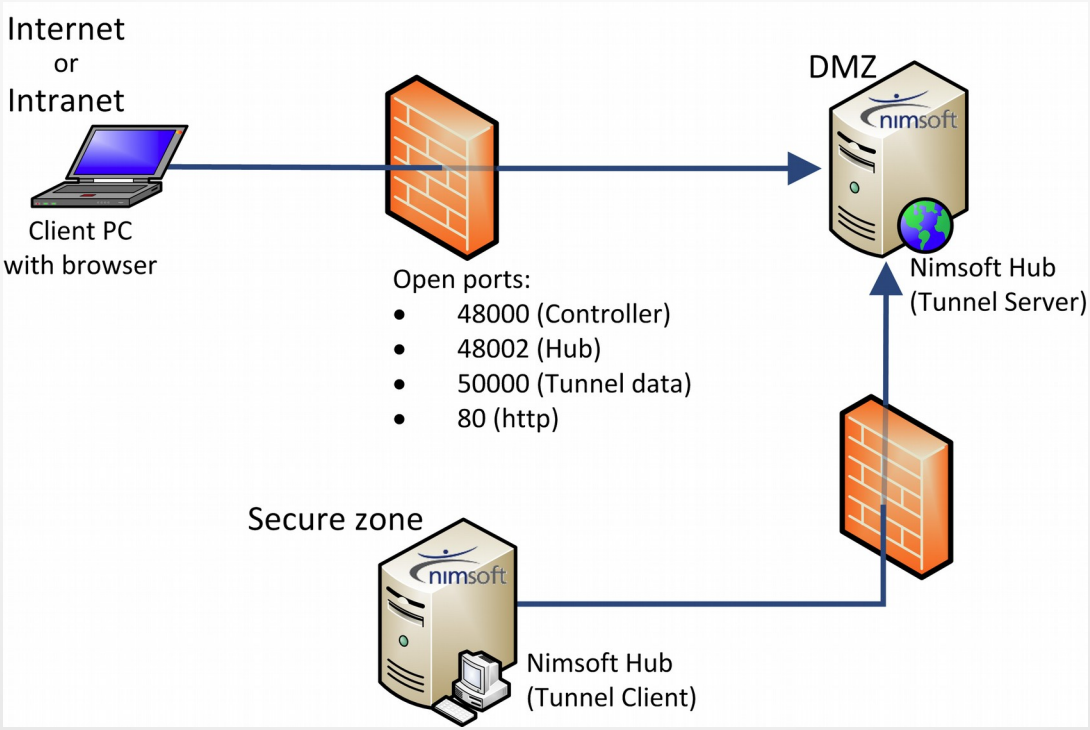
Indirizzi IP: firewall

- Il **firewall** e’ un sistema di sicurezza della rete che controlla tutteo il traffico in ingresso ed in suscita secondo regole ben definite



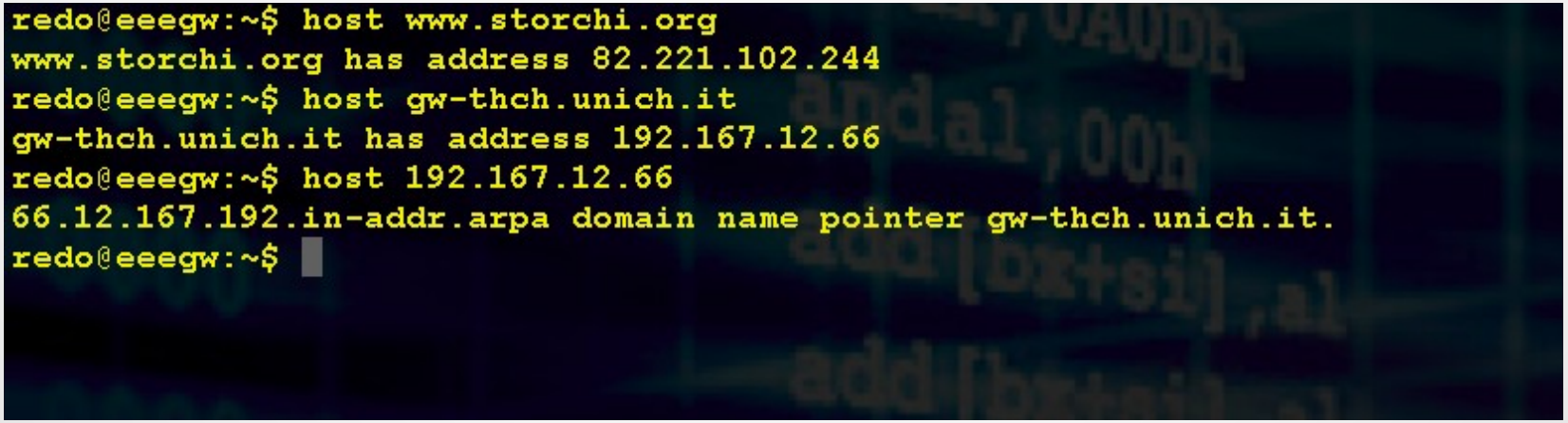
Indirizzi IP: DMZ

- Una **DMZ** (demilitarized zone) è una sottorete fisica o logica che contiene ed espone i servizi di un'organizzazione esterna verso una rete non protetta, solitamente una rete più grande come Internet



DNS

- E' difficile per un essere umano memorizzare numeri, molto piu' facile nomi. Ci sono quindi servizi di DNS (**Domain Name System**). Quindi sistemi utili a tradurre in un verso e nell'altro nomi e indirizzi.

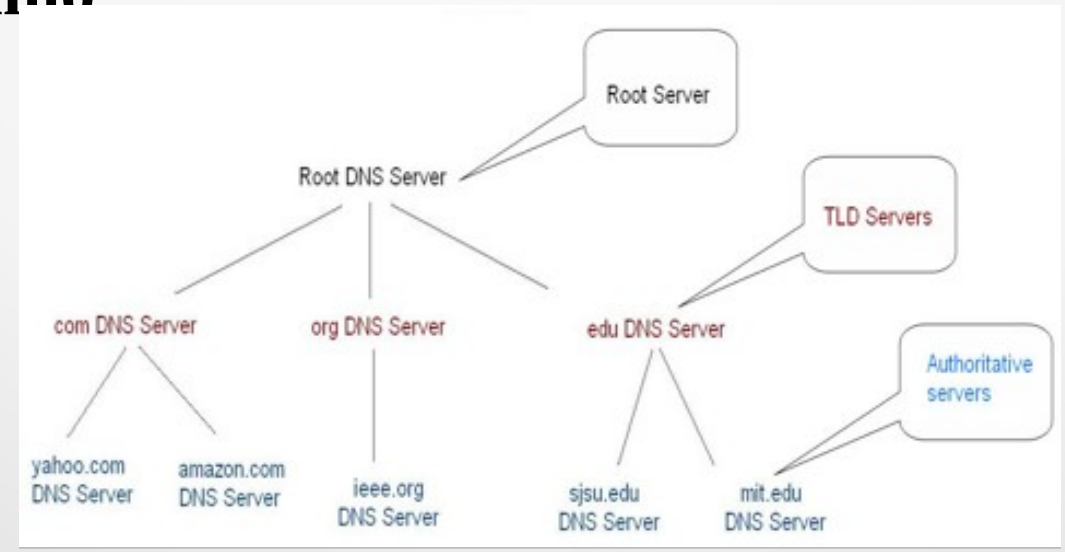


DNS

- Ogni host e' dunque identificato dall'utente da un nome simbolico:
  - gw-thch.unich.it
- I nomi sono assegnati univocmante e gestiti amminsitratativamente in modo gerarchico
- I nomi identificano in modo univocono un host all'interno di un dominio:
  - it e' il dominio
  - unich e' il sotto-dominio all'interno ad it
- I domini principali sono:
  - .gov .edu .com essenzialmente in USA associati al tipo di organizzazione
  - Le varie nazioni invece hanno domini del tipo: .it, .uk, .fr , .de ….

DNS

- Prima dell'introduzione del sistema DNS la corrispondenza fra indirizzi IP e nomi era gestita dallo SRI-NIC che sostanzialmente manteneva una lista in un file hosts.txt
- In pratica il **DNS e' un database distribuito**. Le informazioni sono infatti distribuite su molti computer, server DNS, **ognuno dei quali e' responsabile di una certa porzione del nome, detta dominio**

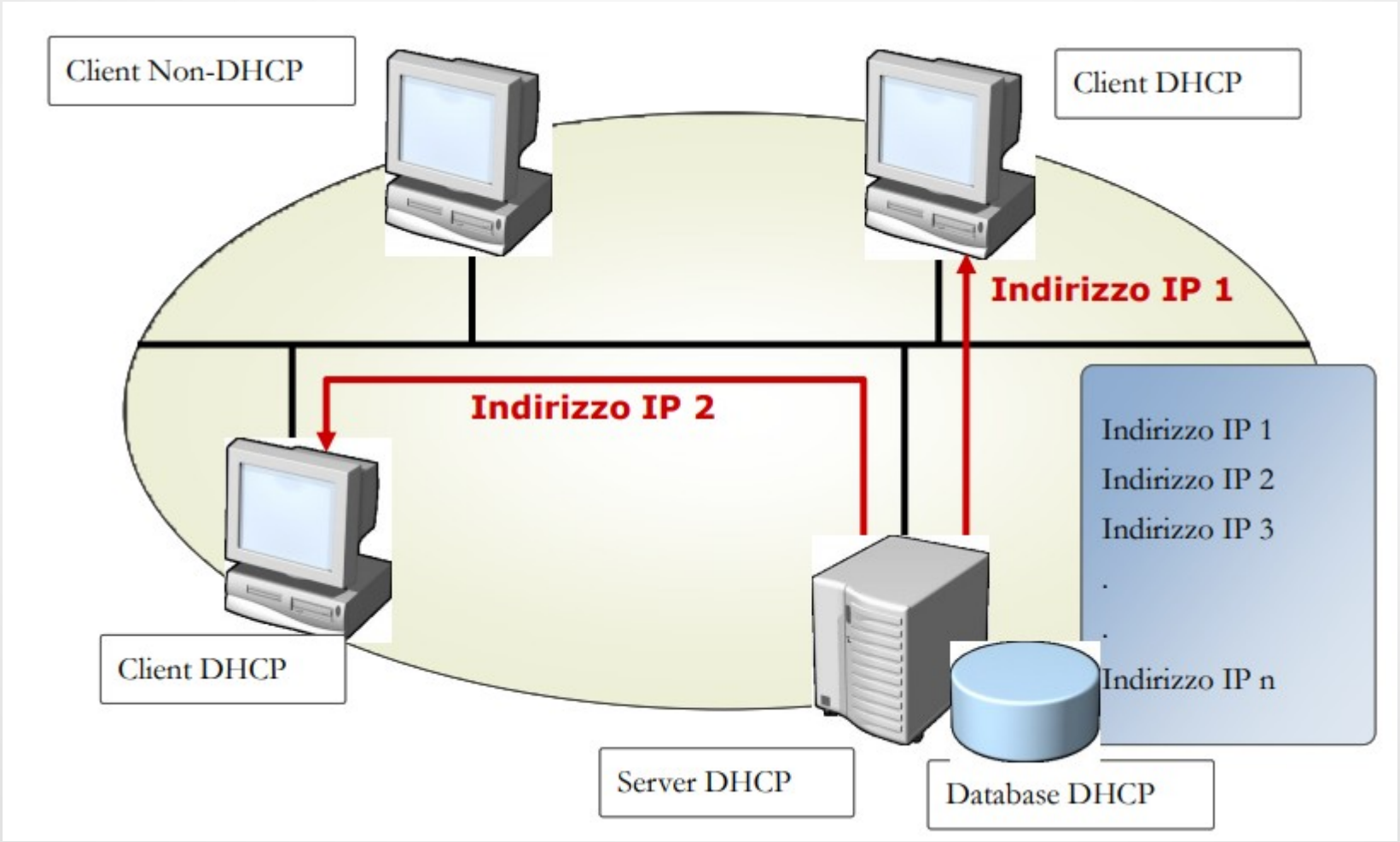


DNS

- I server sono organizzati con una **struttura gerarchica ad albero**
- Al momento della richiesta di un dato indirizzo, ad esempio www.storchi.org, il server DNS della “propria rete” controlla se l’indirizzo corrispondente al **nome e’ presente nella cache**.
- Se non e’ presente **contatta i Root servers (quelli che gestiscono le estensioni .org, .it, .edu …**), in questo caso .org che restituirea’ una lista di **server che gestiscono il dominio storchi.org**
- **Quest’ultimo server restituira’ l’indirizzo IP corrispondente a “WWW” (Domain Name System (DNS) names are "case insensitive")**

DHCP

- Ad esempio il router ADSL che avete a casa



Protocolli ad alto livello

- Vengono usati diversi tipi di protocollo ognuno per ogni specifico servizio:
  - **HTTP (HyperText Transfer Protocol)** Accesso alle pagine ipertestuali (WEB) nell'ambito del WWW (https crittato)
  - **FTP (File Transfer Protocol)** trasferire e copiare file
  - **SMTP (Simple Mail Transfer Protocol)** Spedizione di messaggi di posta elettronica (e-mail) **POP3** per scaricare i messaggi e-mail nel proprio computer . **IMAP utile quando si consultano i messaggi da piu' dispositivi**

Una risorsa in rete e' quindi “identificata” dall'URL:

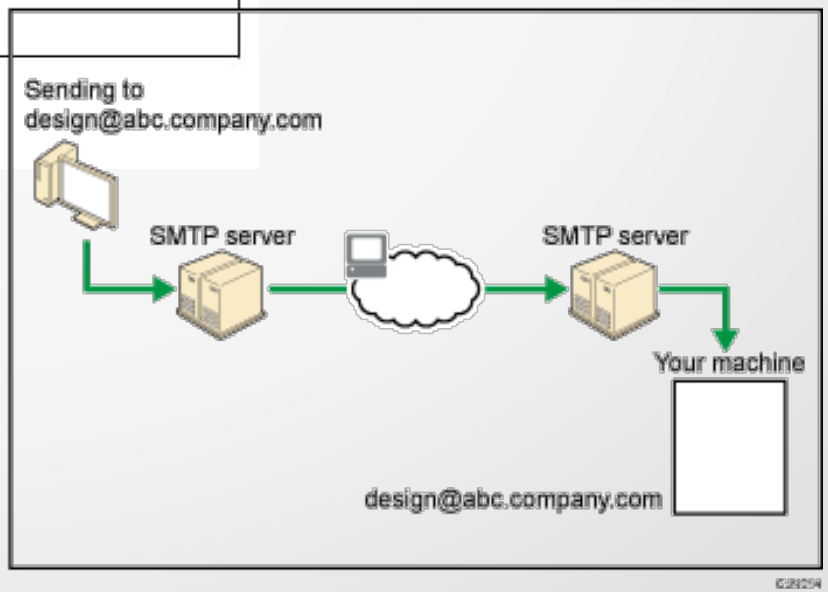
http://nomehost.it/index.html

SMTP esempio

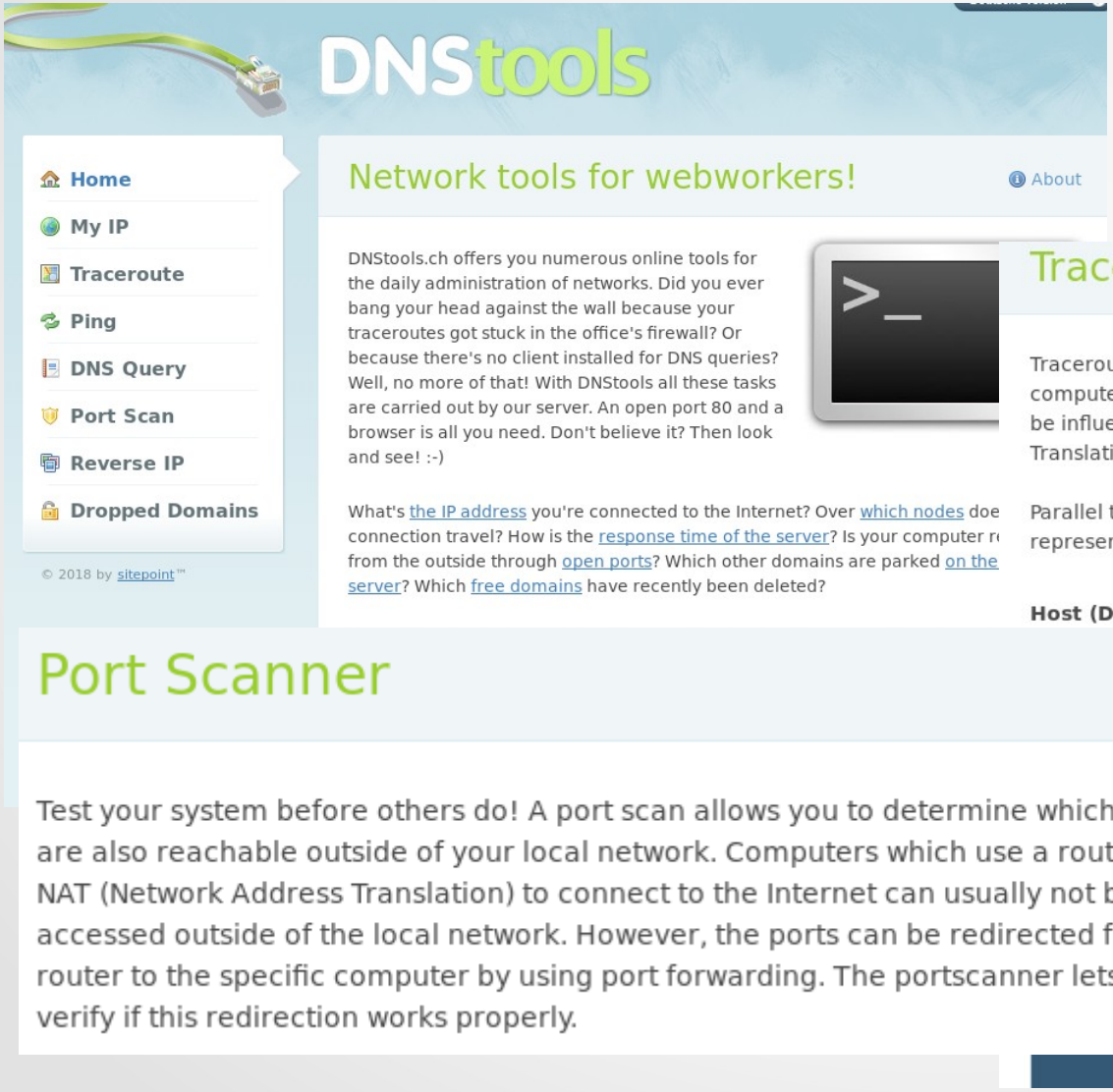
SMTP Protocol Exchange

```
S:   250 SMTPUTF8

C:   EHLO example.com

S:   250-mx.google.com at your service, [999.999.999.999]
S:   250-SIZE 35882577
S:   250-8BITMIME
S:   250-AUTH LOGIN PLAIN XOAUTH XOAUTH2 PLAIN-CLIENTTOKEN
S:   250-ENHANCEDSTATUSCODES
S:   250-PIPELINING
S:   250-CHUNKING
S:   250 SMTPUTF8

C:   AUTH XOAUTH2 dXNlcj1hbWFnYWtpLnRv...

S:   235 2.7.0 Accepted
```

Specify Initial Client Response which is created from username and access token



DNS Tools


DNStools
Home
My IP
Traceroute
Ping
DNS Query
Port Scan
Reverse IP
Dropped Domains
© 2018 by sitepoint™
Network tools for webworkers!
About
DNStools.ch offers you numerous online tools for the daily administration of networks. Did you ever bang your head against the wall because your traceroutes got stuck in the office's firewall? Or because there's no client installed for DNS queries? Well, no more of that! With DNStools all these tasks are carried out by our server. An open port 80 and a browser is all you need. Don't believe it? Then look and see! :-)
What's the IP address you're connected to the Internet? Over which nodes doe connection travel? How is the response time of the server? Is your computer r from the outside through open ports? Which other domains are parked on the server? Which free domains have recently been deleted?
Traceroute on a map
About
Traceroute determines which IP-Router the data packets take to get to the target computer. However, traceroute does not always show the actual route. The result may be influenced by firewalls, flawed implementation of IP-stacks, Network Address Translation and IP tunnels.
Parallel to the traceroute query, locations of the nodes are also determined and represented on the map.
Host (Domain/IP)
192.167.12.66
Trace
microsoft.com or bluewin.ch
Port Scanner
About
Test your system before others do! A port scan allows you to determine which services are also reachable outside of your local network. Computers which use a router with NAT (Network Address Translation) to connect to the Internet can usually not be accessed outside of the local network. However, the ports can be redirected from the router to the specific computer by using port forwarding. The portscanner lets you verify if this redirection works properly.
Netherlands
Berlin
Belgium
Prague
Paris
Czechia
Austria
Hungary
Moldova
France
Romania
Croatia
Serbia
Barcelona
Rome
Bulgaria
Istanbul
Greece
Tyrrhenian Sea
Poland
Belarus
For development purposes only

Introduzione all'Informatica

Loriano Storchi

loriano@storchi.org

http://www.storchi.org/

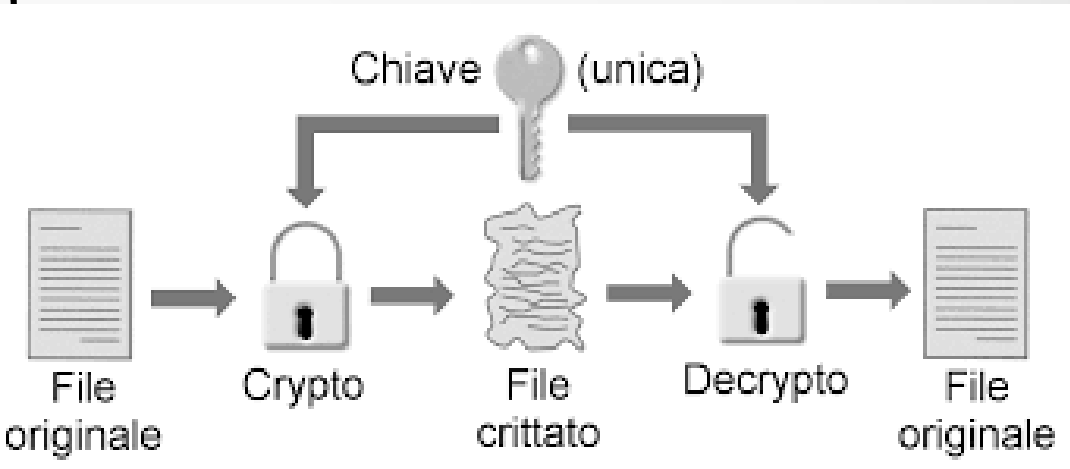
CRITTOGRAFIA CENNI DI BASE

Crittografia cenni di base

- Nei calcolatori le informazioni sono memorizzate come sequenze di bit
- **Le tecniche crittografiche modificano queste sequenze (stringhe) per ottenere sequenze diverse che poi possono essere trasmesse e ritrasformate dal riceventi nella sequenza originale**. Le funzioni matematiche usate nella trasformazione usano una o piou' chiavi segrete
  - **Cifratura Simmetrica:** usata addirittura a partire dagli Egizi e dagli antichi Romani
  - **Cifratura Asimmetrica**: risale agli anni 70

Cifratura simmetrica

- **La chiave usata per cifrare e decifrare, e quindi da mittente e destinatario e’ unica**. Ad esempio cifrarion di Cesare sostituire ogni carattere con altro sfasato di k posti (la chiave e’ il valore di k)
- Cifrario monoalfabetico



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

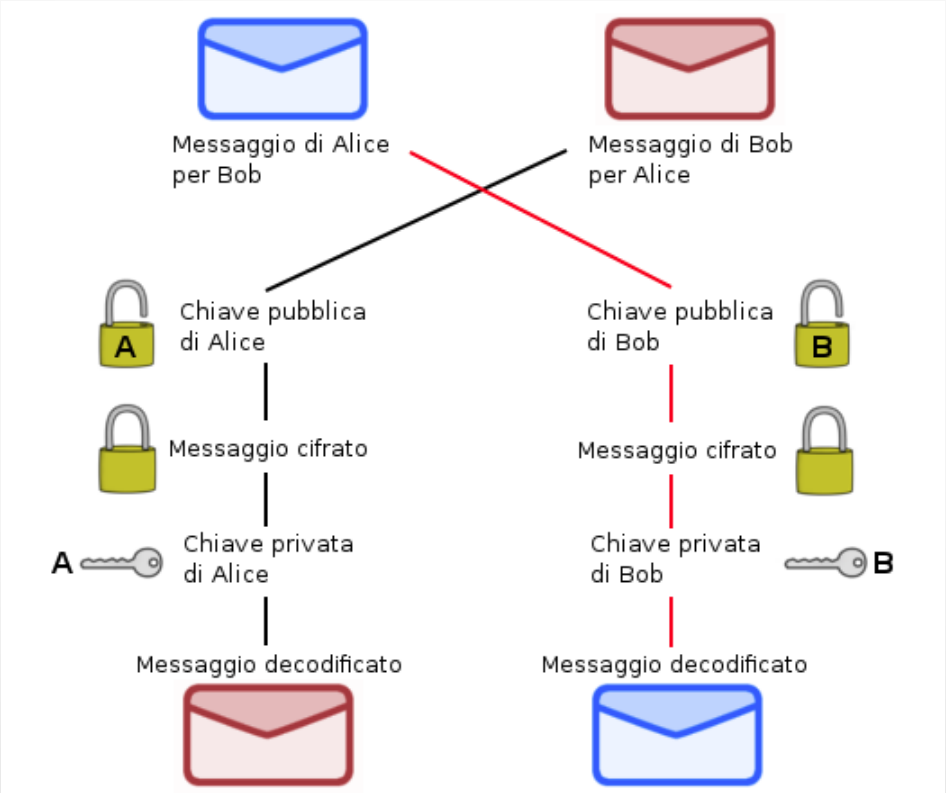
Cifratura simmetrica

- **Devo trovare un modo sicuro per scambiare la chiave** segreta che e' unica per mittente e destinatario
- Attacco a forza bruta (**brute-force**) nel caso ad esempio del cifrario di cesare provo tutti i valori di k e vedo quando ottengo frasi e parole "corrette"
- Esempi di algoritmi moderni : **Blowfish, Twofish, Standard DES o Triple DES, Standard AES**. Sono tutti basati su problemi matematici difficili da risolvere se non si conosce la chiave segreta

CIFRATURA ASIMMETRICA

Cifratura asimmetrica

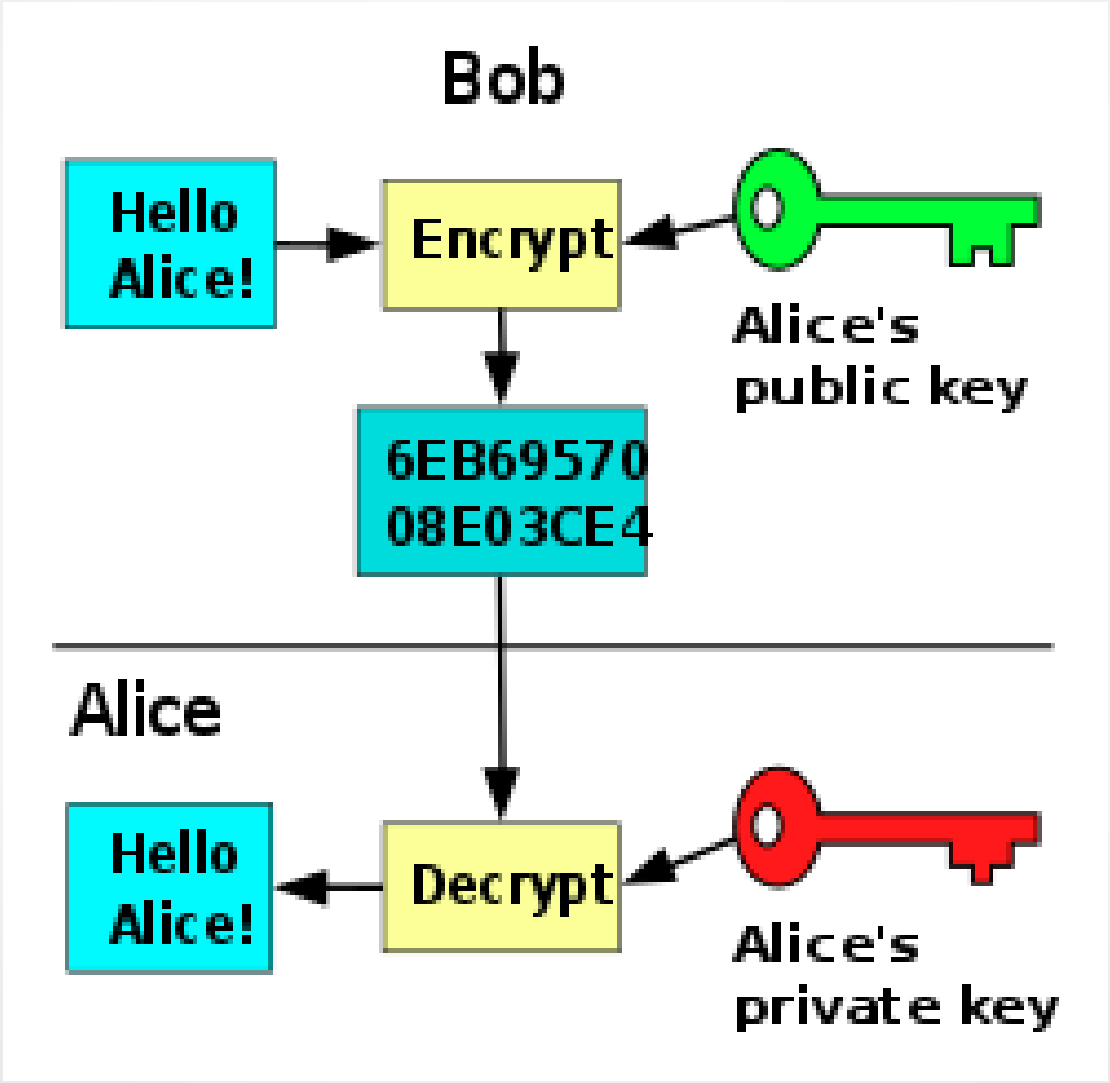
- **Le chiavi usate per cifrare e decifrare sono diverse**. Chiave privata che deve essere tenuta segreta serve a decifrare, quindi a riottenere il messaggio originale, quella pubblica a cifrare



Cifratura asimmetrica

- **Al momento in cui l’utente genera la coppia di chiavi** deve custodire gelosamente la chiave **privata (segreta) KS** ed invece distribuire solo quella **pubblica KP**. **KS ad esempio in una smartcard** e’ sara’ in quel caso la smartcard stessa a operare la cifratura.
- Se l’utente Bob vuole scrivere un messaggio privato all;utente Alice, Bob usera’ la chiave pubblica KP di Alice e spedira’ il **crittogramma** ottenuto, Solo Alice in possesso della parte privata della chiave KS potra’ riottenere il messaggio originale (in chiaro)
- Cifratura asimmetrica usata anche nei processi di autenticazione

Cifratura asimmetrica



Cifratura asimmetrica: RSA

- L'algoritmo piu' noto ed usato e' l'**RSA** (nomi degli inventori Rivest, Shamir, Adleman)
- Anche per autenticazione o garantire integrita' di un documento (anche firma digitale)
- **Basato su numeri primi, cioe' quei numeri naturali che sono divisibili solo per 1 o per se stessi (2, 3, 5, ..., $19249 \cdot 2^{13018586} + 1$)**
- **In pratica KS e KP sono i fattori primi di un numero grande**
- Interesse nei numeri primi e negli algoritmo di fattorizzazione (**algoritmo di Shor quantum computer**)

PGP - OpenPGP

OpenPGP

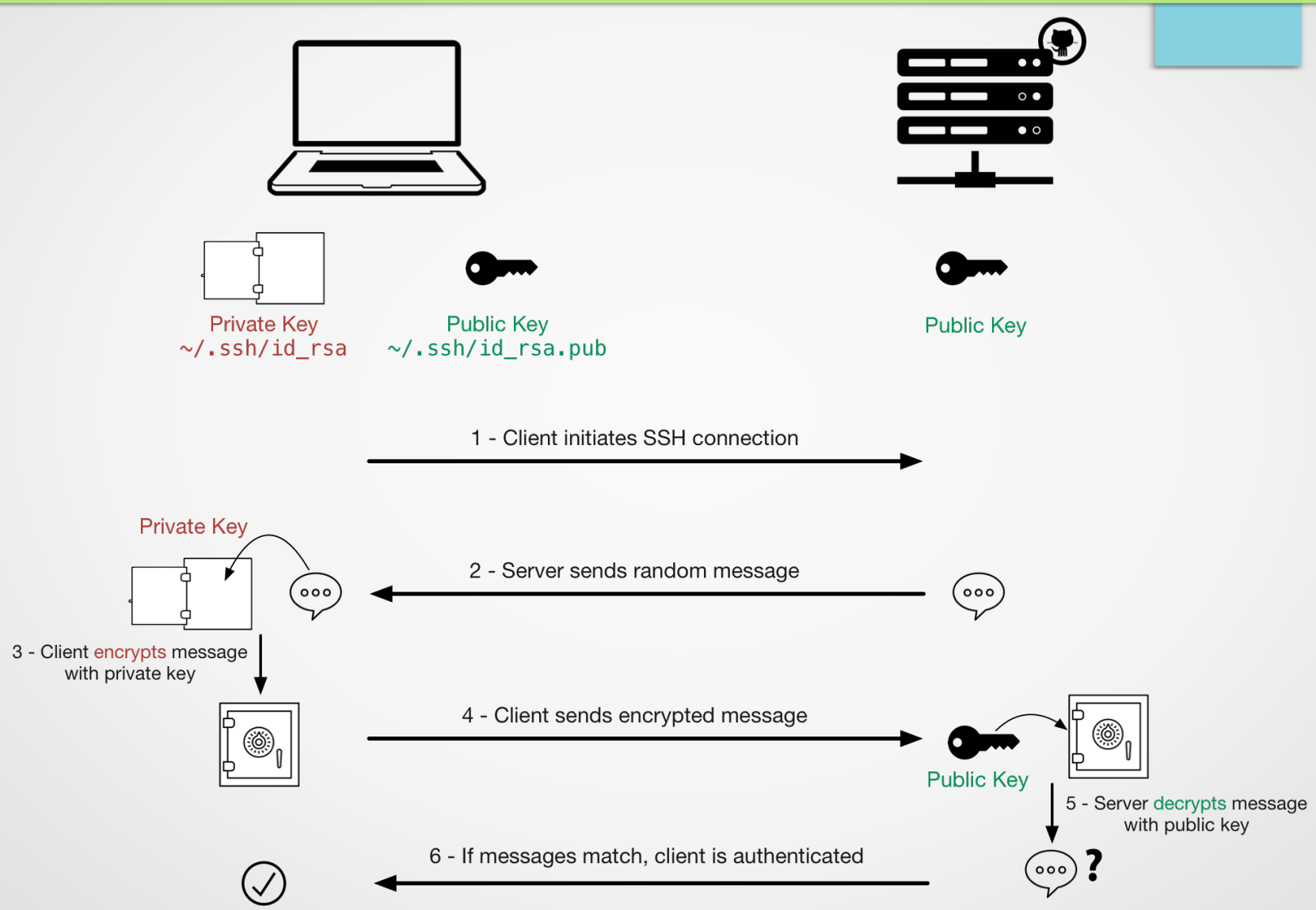
- RSA è un algoritmo (in realtà, due algoritmi: uno per la crittografia asimmetrica e uno per le firme digitali - con diverse varianti). **PGP** è un software, **ora un protocollo standard, generalmente noto come OpenPGP**.
- **OpenPGP definisce i formati per gli elementi di dati che supportano la messaggistica sicura**, con crittografia e firme e varie operazioni correlate come la distribuzione delle chiavi.
- Come protocollo, **OpenPGP si basa su una vasta gamma di algoritmi crittografici.** Tra gli algoritmi che OpenPGP può utilizzare è **RSA**.

OpenPGP

- **Philip R. Zimmermann** è il creatore di **Pretty Good Privacy**, un pacchetto software di crittografia email. Originariamente concepito come uno strumento per i diritti umani, il **PGP** è stato **pubblicato gratuitamente su Internet nel 1991**.
- Ciò ha reso Zimmermann l'obiettivo di un'indagine criminale di tre anni, perché il governo riteneva che le restrizioni all'esportazione degli Stati Uniti per software crittografico fossero violate quando PGP si diffuse in tutto il mondo.
- **GNU Privacy Guard** (GnuPG o GPG) è un software libero progettato per sostituire la suite crittografica PGP.

AUTENTICAZIONE

Esempio autenticazione SSH



FIRMA DIGITALE

Firma Digitale

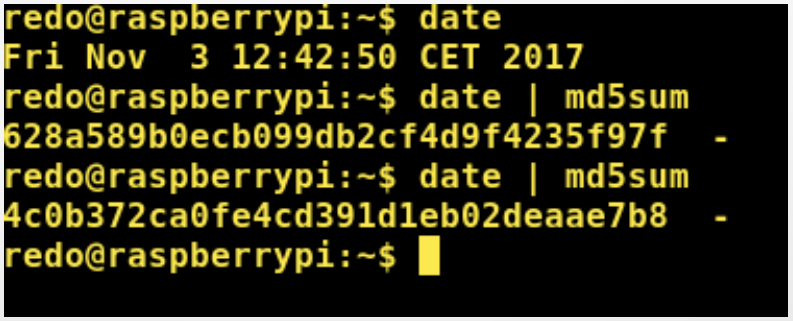
- Viene apposta a documenti digitali in modo da garantire
  - **Autenticità** : quindi garanzia della provenienza del messaggio
  - **Integrità** : il messaggio non e' stato modificato in alcun modo
  - **Non ripudiabilita** : La fonte del messaggio non puo' disconoscere di averlo firmato
- Solo il mittente puo' apporre quelle particolare firma
- Chiunque puo' verificare chi ha firmato il messaggio (documento digitale, testo, suono, immagine, filmato)
- **Ingredienti di base sistema di cifratura asiemmetrico e funzione di hash**

Firma Digitale: Modalita’ di firma

- **CASDES**: estensione **pdf.p7m** il file puo’ essere letto con i software di firma (File protector, DiKe ....)
- **PADES**: estensione **pdf** il file viene firmato con software di firma e letto con Acrobat Reader
- **XADES**: estensione **xml.p7m** , il file xml e’ letto in automatico da software che lo riceve

Firma Digitale: HASH

- Algoritmo di **hash**: **MD5, SHA-1, RIPEMD, SHA-256:**
  - Una funzione che dato un flusso di bit di dimensione variabile restituisce una stringa di lettere o numeri di dimensione fissa (una sorta di bollino unico)
  - La stringa e' un identificativo univoco, la modifica di 1 solo bit del flusso sorgente produce un HASH diverso
  - Non e' invertibile quindi a partire dalla stringa restituita non e' possibile determinare il flusso originale



Firma Digitale: Come funziona

- Alice per firmare un dato oggetto O (un documento ad esempio):
  - **Calcola l'HASH di O** (detto anche **digest**)
  - **Cifra l'HASH ottenuto cona la propria chiave privata**
  - **Appende l'HASH cifrato (la firma) assieme alla sua chiave pubblica** all'oggetto O, chiamiamola F
- Bob per verificare la firma di Alice:
  - **Calcola l'HASH di O**
  - **Decifra l'HASH** cifrato (quindi la firma F di Alice che trova assieme al documento) **usando la chiave pubblica di Alice**
  - **Verifica** a questo punto che i valori siano uguali

CA E CERTIFICATI

CA e Certificati

- Come si puo' essere sicuri che la firma usata sia effettivamente del firmatario ? Parafrasando come posso essere certo dell'associazione utente chiave pubblica ?
- **I Certificati Digitali servono a questo scopo. Essi contengono una seri di informazioni, come ad esempio la chiave pubblica ed i dati dell'utente**
- Cosi' come i certificati cartacei permettono di avere informazioni a proposito dell'utente
- **CA, Certification Authorithy** , garantisce l'associazione fra firma digitale ed identita' del titolare

CA e Certificati

- Certificati digitali: e’ costituito da (X.509):
  - **Chiave pubblica** del firmatario
  - Dalla **sua identita’** (nome cognome data di nascita etc etc)
  - **Data di scadenza** della chiave pubblica
  - **Nome della CA che lo ha rilasciato**
  - **Firma digitale della CA che ha emesso il certificato**
- **Se ci fidiamo della CA (Autorita’ di Certificazione) possiamo verificare la sua firma e dunque l’idenrtita’ del firmatario**

CA e Certificati

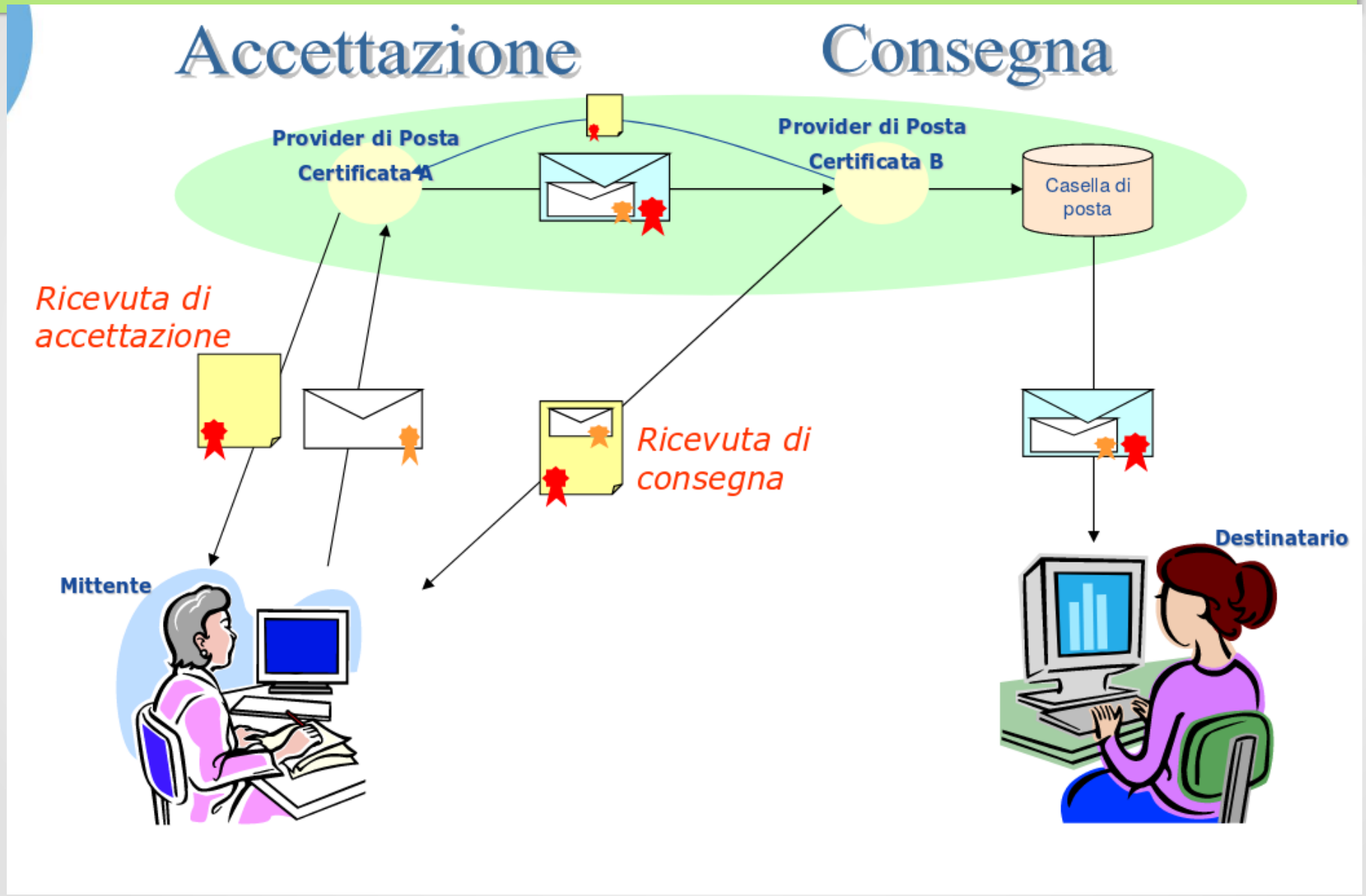
- **CA, Certification Authorithy** , garantisce l’associazione fra firma digitale ed identita’ del titolare
- Il certificato digitale viene firmato da un ente detto CA che ne attesta la bontà. L'operazione di firma avviene,da parte della CA,allegando al certificato i propri riferimenti e cifrando il tutto con la propria chiave privata
- Una data CA che firma il certifica potrebbe non essere ritenuta affidabile a quel punto ripeteremo il certificato dalla CA sospetta possiamo rivolgerci alla CA di livello superiore e cosi via fino a risalire a una CA ritenuta affidabile, e che quindi validi tutti la catena

HTTPS e POSTA CERTIFICATA

HTTPS

- Protocollo fondamentale ad esempio nell'e-commerce e home banking
- Quando mi collego ad un sito via https:
  - **Il server dichiara la sua identita' inviando il proprio certificato** di chiave pubblica **garantito da una CA**
  - **Il mio browser (client) verifica che l'URL corrisponda all'identita' contenuta nel certificato**
  - **Il client (browser) usando la chiave pubblica del server invia una chiave simmetrica temporanea per cifrare tutti il traffico dati**

Posta certificata

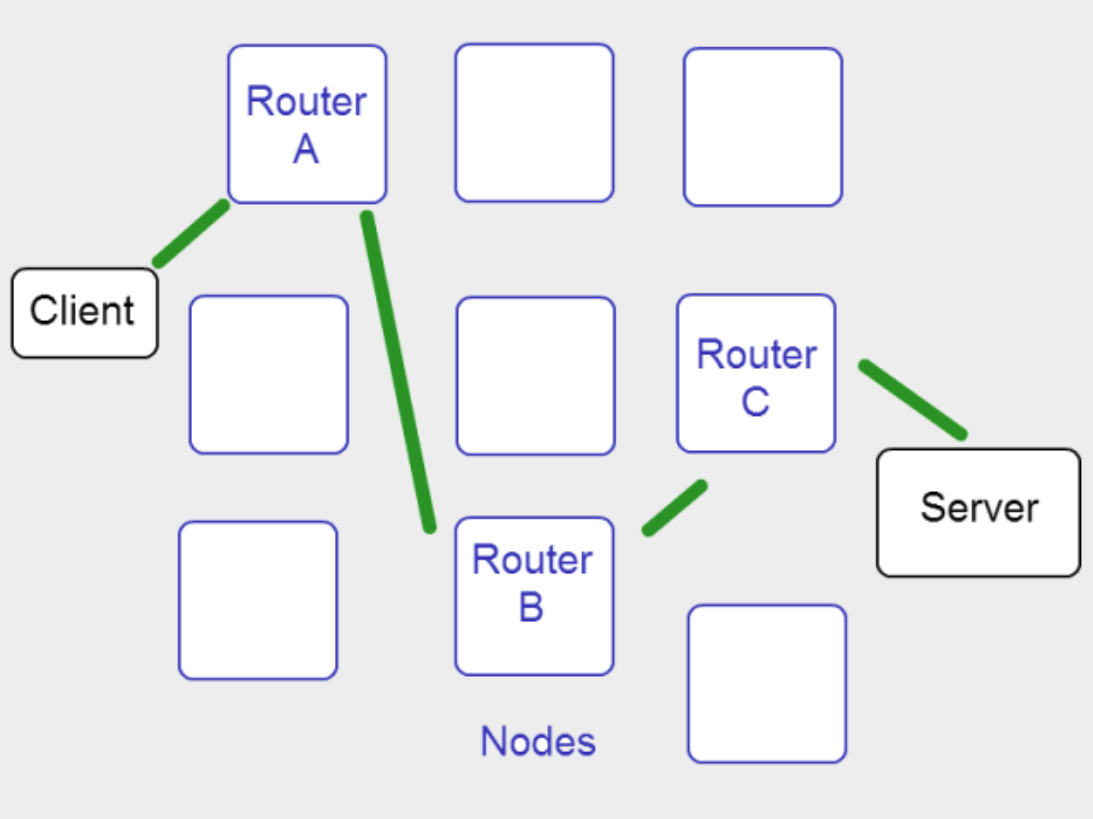


INTERNET, ALTRI ASPETTI

Deep web, Tor, bitcoin

- Onion routing un cenno. La rete tor e composta da volontari che usano i propri computer come nodi:

Tor crea un percorso random fra i vari nodi cosi da raggiungere il server partendo dal client

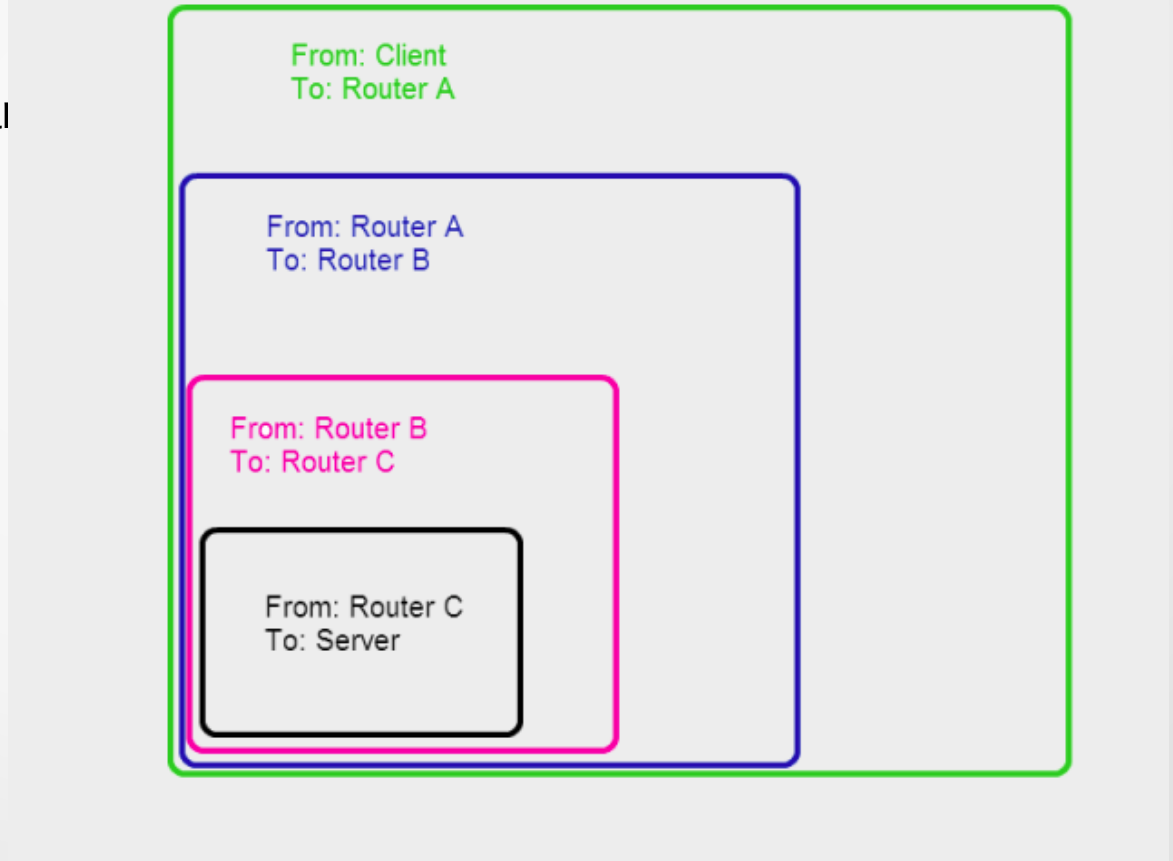


Deep web, Tor, bitcoin

- Posso raggiungere servizi “nascosti” **hidden service**, ma anche normali server via **exit nodes**

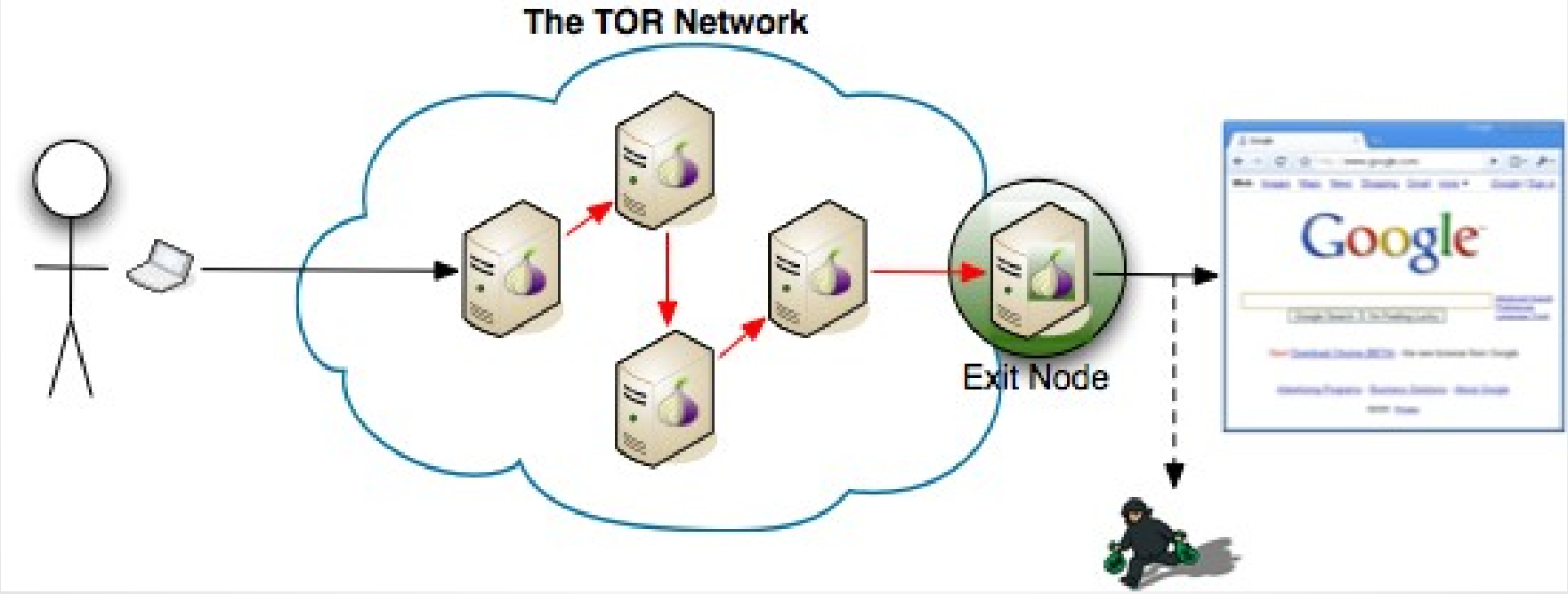
I pacchetti che transitano fra i var nodi sono incapsulati in strati successivi crittati, cosi’ come gli strati di una “cipolla”

Ad ogni passaggio il nodo conoscenze solo il punto di provenienza e quello di arrivo



Deep web, Tor, bitcoin

- Posso raggiungere servizi “nascosti” **hidden service**, ma anche normali server via **exit nodes**



Deep web, Tor, bitcoin

- **Bitcoin** (cito wikipedia) A differenza della maggior parte delle valute tradizionali, Bitcoin non fa uso di un ente centrale: esso utilizza un database distribuito tra i nodi della rete che tengono traccia delle transazioni, ma sfrutta la crittografia per gestire gli aspetti funzionali, come la generazione di nuova moneta e l'attribuzione della proprietà dei bitcoin.
- Basata su una **rete P2P (blockchain** database distribuito strutturato a blocchi**)**
- Basata su crittografia ed algoritmi di hashing (SHA-256), BitCoin utilizza l'algoritmo hash SHA-256 per generare numeri "random" verificabili in un modo tale da richiede una quantità prevedibile di tempo di calcolo. Generare un hash SHA-256 con un valore inferiore al target attuale risolve un blocco.

Rappresentazione binaria e dati

Loriano Storchi

loriano@storchi.org

http::://www.storchi.org/

BREVE DIGRESSIONE

Rappresentazione binaria dei numeri (breve digressione)

- In un sistema di **numerazione posizionale** data la **base** questa definisce direttamente il numero di **simboli (cifre)** che si usano per scrivere il numero.
  - Ad esempio nel sistema decimale usiamo 10 simboli (0,1,2,3,4,5,6,7,8,9)
- I moderni sistemi di numerazione sono posizionali, **quindi il numero e' scritto specificando l'ordine delle cifre, ed ogni cifra assume un valore a seconda della sua posizione**
  - Ad esempio $423 = 4 * 10^2 + 2 * 10^1 + 3 * 10^0$

Se volete 4 centinaia 2 decine e 3 unita'

Rappresentazione binaria dei numeri (breve digressione)

- In generale data una **base b avro' b simboli** (cifre) e quindi **un numero intero N** sara' scritto come:
  - Valore N = $c_n * b^n + c_{n-1} * b^{n-1} + \ldots + c_0 * b^0$
- Similmente se ho un numero N = $0.c_1c_2...c_n$
  - Valore N = $c_1 * b^{-1} + c_2 * b^{-2} + \ldots + c_n * b^{-n}$
- Consideriamo adesso il caso piu' semplice quello delle rappresentazione di **numeri interi senza segno (i Naturali)**
- Se uso un sistema di numerazione con **base b con n cifre** potro' rappresentare un **massimo di b^n numeri diversi**, quindi tutti i numeri da 0 fino a b^n - 1
- Ad esempio in **base 10** e' chiaro che usando **due cifre posso rappresentare tutti i numeri da 0 a 99** quindi **10^2 = 100** numeri distinti

La base binaria

- Usando una base 2, quindi solamente due simboli 0 ed 1, sempre rimanendo nell'ambito della rappresentazione di numeri interi positivi usando n cifre potro' rappresentare al massimo tutti i numeri compresi fra **0 e 2^n -1**
  - Ad esempio usando due cifre potro' rappresentare 4 numeri distinti:
  - $00 = 0 * 2^1 + 0 * 2^0 = 0$
  - $01 = 0 * 2^1 + 1 * 2^0 = 1$
  - $10 = 1 * 2^1 + 0 * 2^0 = 2$
  - $11 = 1 * 2^1 + 1 * 2^0 = 3$

DIGITALIZZAZIONE

La base binaria

- Un byte (un boccone) rappresenta modernamente la sequenza di 8 bit ed e' divenuto storicamente l'elemento base dell'indirizzabilita' e quindi l'unita' di misura base dell'informazione.
- **8 bit significa che con 1 byte posso rappresentare 2^8 = 256 valori differenti**. Quindi nel caso di numeri interi unsigned i numeri da 0 a 255. **Se uso un bit per indicare il segno ad esempio 0 positivo ed 1 negativo,** posso rappresentare i numeri interi da -128 a 127 (da 10000000 a 01111111)
- Oppure con 8 bit posso rappresentare 256 differenti caratteri.

ASCII

- Extended ASCII usa 8-bit
- ASCII originale US-ASCII 7-bit

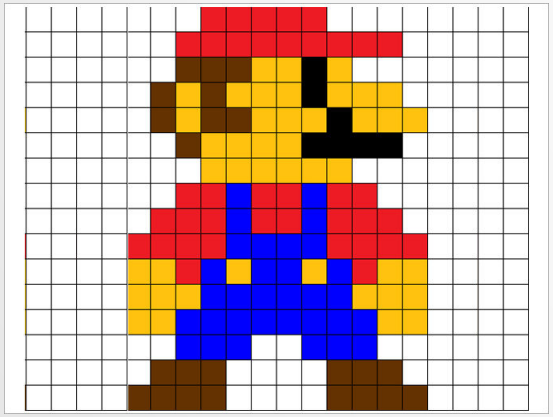
000	NUL	033	!	066	B	099	c	132	ä	165	Ñ	198	ã	231	þ
001	Start Of Header (SOH)	034	"	067	C	100	d	133	à	166	ª	199	Ã	232	Þ
002	Start Of Text (STX)	035	#	068	D	101	e	134	å	167	º	200	╚	233	Ú
003	End Of Text (ETX)	036	$	069	E	102	f	135	ç	168	¿	201	╔	234	Û
004	End Of Transmission (EOT)	037	%	070	F	103	g	136	ê	169	®	202	╩	235	Ù
005	Enquiry	038	&	071	G	104	h	137	ë	170	¬	203	╦	236	ý
006	Acknowledge (ACK)	039	'	072	H	105	i	138	è	171	½	204	╠	237	Ý
007	Bell	040	(	073	I	106	j	139	ï	172	¼	205	═	238	¯
008	Backspace (BS)	041	)	074	J	107	k	140	î	173	¡	206	╬	239	´
009	Horizontal Tab	042	*	075	K	108	l	141	ì	174	«	207	¤	240	-
010	Line Feed **(LF)**	043	+	076	L	109	m	142	Ä	175	»	208	ð	241	±
011	Vertical Tab	044	,	077	M	110	n	143	Å	176	░	209	Ð	242	‗
012	Form Feed **(FF)**	045	-	078	N	111	o	144	É	177	▒	210	Ê	243	¾
013	Carriage Return **(CR)**	046	.	079	O	112	p	145	æ	178	▓	211	Ë	244	¶
014	Shift Out	047	/	080	P	113	q	146	Æ	179	│	212	È	245	§
015	Shift In	048	0	081	Q	114	r	147	ô	180	┤	213	ı	246	÷
016	Dataline Escape (DLE)	049	1	082	R	115	s	148	ö	181	Á	214	Í	247	¸
017	DC 1 (XON)	050	2	083	S	116	t	149	ò	182	Â	215	Î	248	°
018	DC 2	051	3	084	T	117	u	150	û	183	À	216	Ï	249	¨
019	DC 3 (XOFF)	052	4	085	U	118	v	151	ù	184	©	217	┘	250	·
020	DC 4	053	5	086	V	119	w	152	ÿ	185	╣	218	┌	251	¹
021	Negative Acknowledge (NAK)	054	6	087	W	120	x	153	Ö	186	║	219	█	252	³
022	Synchronous Idle	055	7	088	X	121	y	154	Ü	187	╗	220	▄	253	²
023	End Of Transmission Block	056	8	089	Y	122	z	155	ø	188	╝	221	¦	254	■
024	Cancel	057	9	090	Z	123	{	156	£	189	¢	222	Ì	255	
025	End Of Medium	058	:	091	[	124	|	157	Ø	190	¥	223	▀		
026	Substitude	059	;	092	\	125	}	158	×	191	┐	224	Ó		
027	Escape **(ESC)**	060	<	093	]	126	~	159	ƒ	192	└	225	ß		
028	File Seperator	061	=	094	^	127	(DEL) ⌂	160	á	193	┴	226	Ô		
029	Group Seperator	062	>	095	_	128	Ç	161	í	194	┬	227	Ò		
030	Record Seperator	063	?	096	`	129	ü	162	ó	195	├	228	õ		
031	Unit Seperator	064	@	097	a	130	é	163	ú	196	─	229	Õ		
032	*SPACE* (SP)	065	A	098	b	131	â	164	ñ	197	┼	230	µ		

Codifica dell’informazione

- Una codifica e’ una convenzione, come visto prima quindi la sequenza di bit **01001100 puo’ rappresentare il carattere L (L maiuscolo) oppure se invece intendiamo un valore intero unsigned rappresenta il numero decimale 76**.
- Ad esempio **ogni immagine e’ composta da pixel, se usassi 1 solo bit per ogni pixel potrei avere solo immagini in bianco e nero (1 pixel nero , 0 pixel bianco).**

Codifica dell’informazione

- se uso piu’ bit per rappresentare ogni pixel posso invece avere range di grigi o colori. E da qui suoni, video ...



Il pixel rappresenta il piu’ piccolo elemento autonomo dell’immagine. Ogni pixel e’ dunque caratterizzato, dalla propria posizione

Il numero totale di pixel di un’immagine digitale e’ detto **risoluzione** . Ad esempio se ho una griglia 10 x 10 l’immagine sarà composta dal 100 pixel

dpi = numero di punti per pollice , ad esempio in un monitor tipicamente avro’ 72 pixel per pollice

Profondita’ : nel caso di un’immagine in scala di grigi s possono usare 8 bit per ogni pixel avendo cosi’ a disposizione **2^8 = 256 sfumatiure di grigio**

Codifica dell’informazione

- Immagini a colori

Nel caso di immagini a colori **ogni pixel** e’ caratterizzato da **tre scale di colori** per i tre colori fondamentali, **RGB (Red, Green, Blue)**

Ad esempio un’immagine ad **8-bit** avra’ per ogni colore 256 possibili sfumature per il rosso, 256 per il verde e 256 per il blu. Dunque in totale **16777216** possibili sfumature di colore. Nel caso di immagini a 12-bit avremo invece $\mathbf{2^{16} = 4096}$ **sfumature per ogni colore**, in totale **68718476736** possibili colori per ognii pixel.

CENNI; OVERFLOW, UNDERFLOW, TIPI DATO

Le memoria dei calcolatori e’ finita

- **OVERFLOW**: i bit a disposizione non bastano per rappresentare il risultato. **UNDERFLOW** numero troppo piccolo 3/2 non posso rappresentare 1.5 , posso rappresentare solo 1 o 2
- Ad esempio se uso 8 bit per rappresentare i numeri interi positivi posso rappresentare solo i numeri da 0 a 255, quindi cosa succede se provo a sommare 1 a 255 ?

1 1 1 1 1 1 1 1 +

0 0 0 0 0 0 0 1 =

---

1 0 0 0 0 0 0 0 0 per rappresentare il risultato non sono sufficienti 8 bit

Tipi di dato

- Vedi differenza fra linguaggi fortemente tipizzati e non, tipizzazione dinamica e statica
- I linguaggi di programmazione hanno dei tipi di dato nativi, come ad esempio gli integer, i floating-point ,i boolean e i character. Diversi tipi diverse dimensioni e dunque diversi range (Algebra esatta e non …)

label	size (bytes)	smallest value	largest value
byte	1	-128	127
short	2	-32768	32767
int	4	-2147483648	2147483647
long	8	-9223372036854775808	9223372036854775807
char	2	0	65535

CENNI CONVERSIONE ANALOGICO DIGITALE

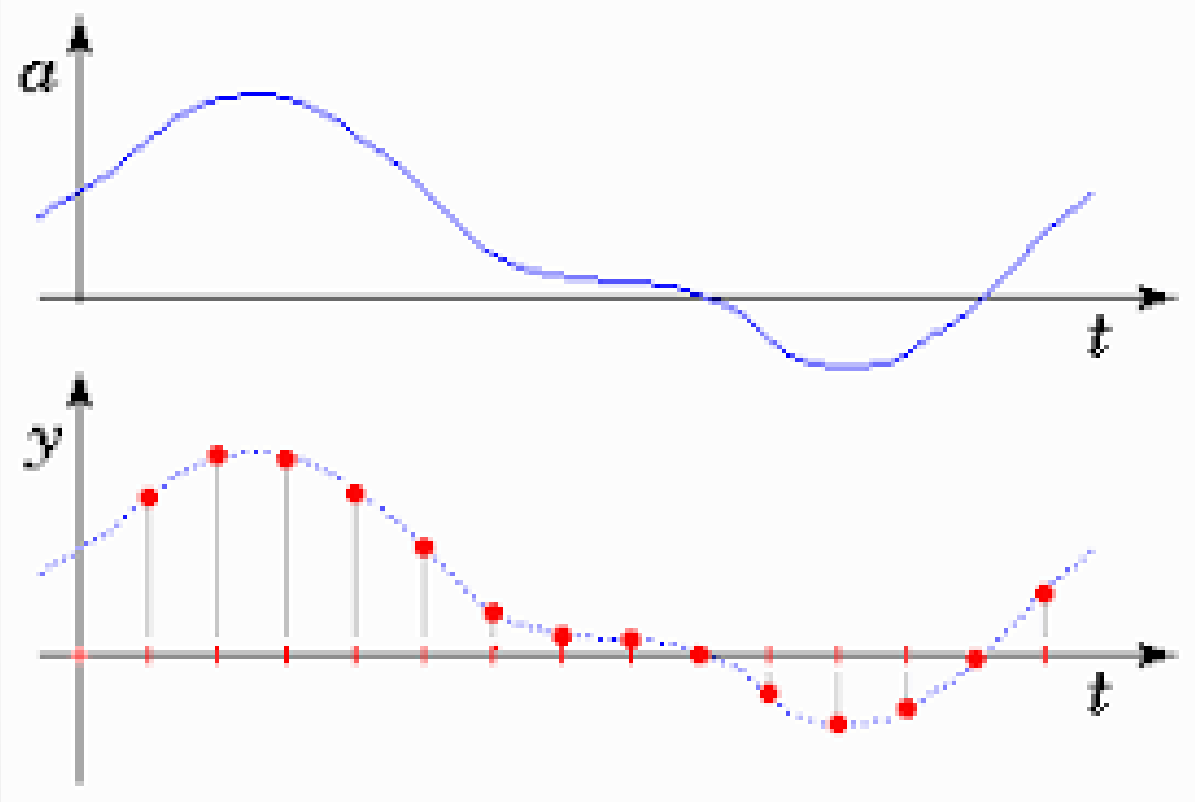
Segnale analogico

- Conversione Analogico Digitale

campionamento - discretizzazione del tempo (teorema del campionamento di Nyquist-Shannon)

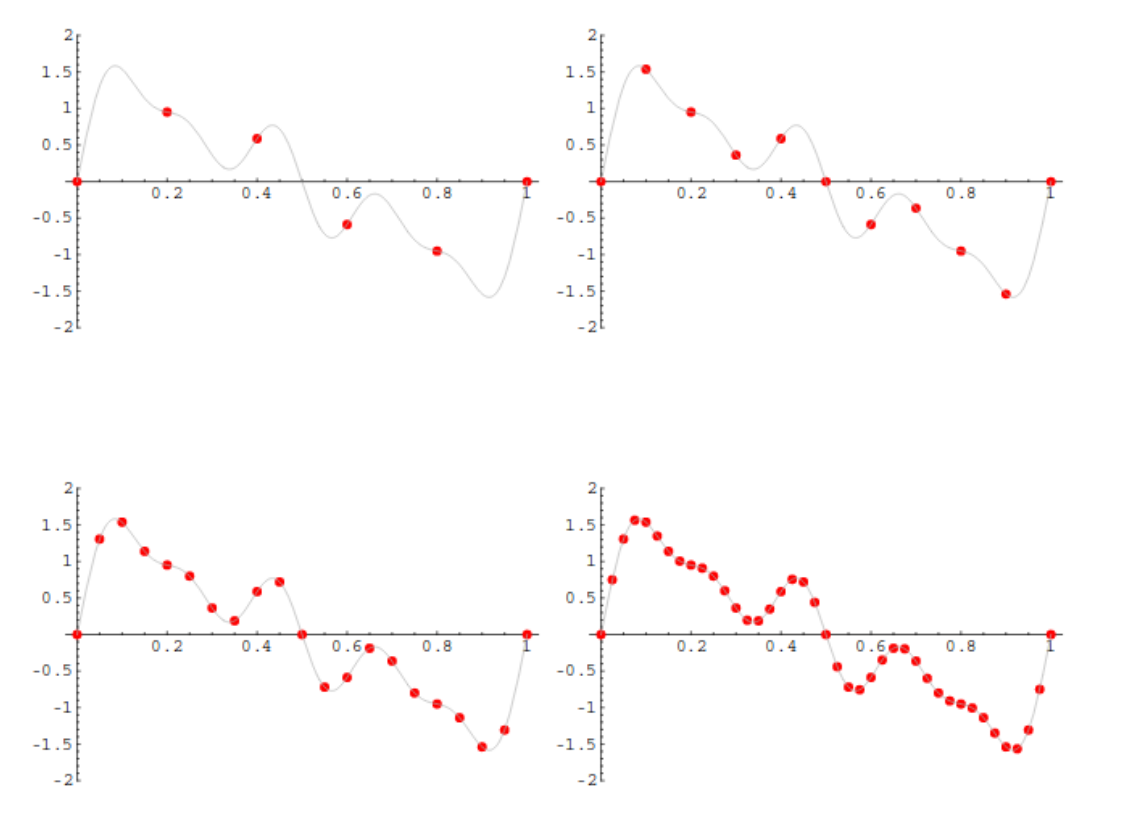
quantizzazione - discretizzazione della ampiezza

codifica - uso di “parole” binarie per esprimere il valore del segnale



Segnale analogico

- Ovviamente la fedelta’ migliora man mano che cresce il numero di campioni per unita’ di tempo (frequenza di campionamento) ed il numero di livelli di quantizzazione



Audio digitale

- Visto che l'orecchio umano riesce a udire frequenze in un certo range (20, 20000 Hz circa) il teorema del campionamento ci dice che dobbiamo campionare a 40 kHz
- Solitamente ssi usa un numero di livelli di quantizzazione molto piu' grande di 256, spesso 16 bit
- Ad esempio nel caso di un **CD** abbiamo due canali (stereo) a **44.1 kHz e 16 bit (2^{16} = 65536)**
  - Quindi il **bit rate = 44100 campioni/s * 16 bit * 2 canali = 1411200 bit/s = 176400 Byte/s**
  - Se vogliamo **1 minuto** di musica abbiamo bisogno di **60*176400 Byte/s = 10584000 Bytes** che sono circa 10 MiB

Considerazioni finali

- Compressione di immagini e suono (quindi anche video) si basa essenzialmente sull’eliminazione di informazioni a cui l’orecchio umano e l’occhio umano sono poco sensibili
- Ma non solo ….

Introduzione all'Informatica

Loriano Storchi

loriano@storchi.org

http://www.storchi.org/

SOFTWARE

Software

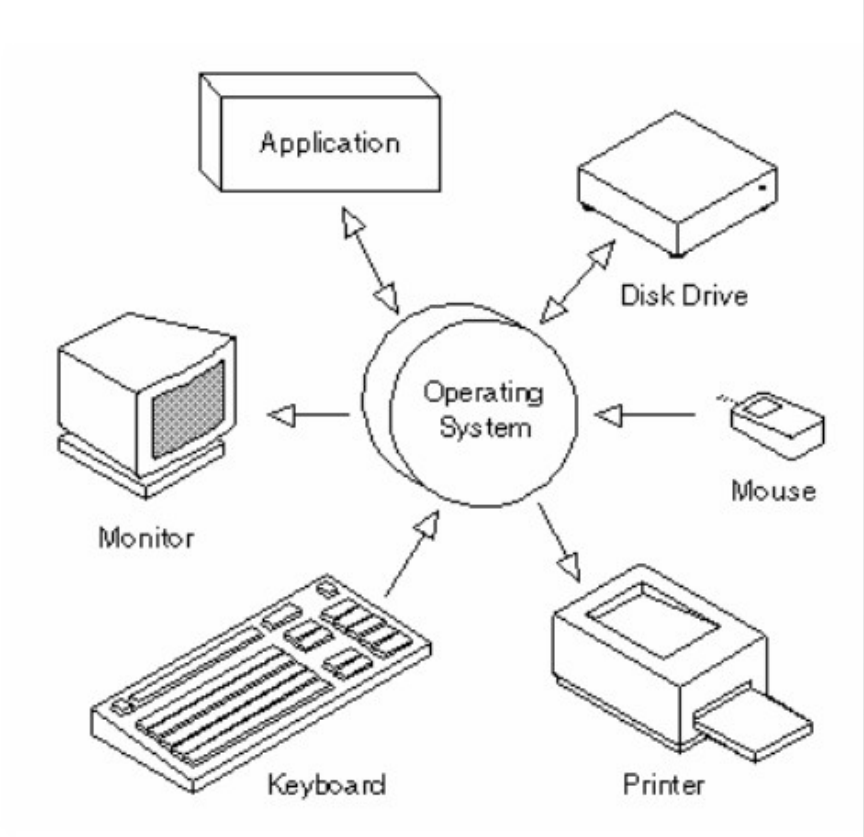
- **Software di base**: dedicato alla gestione dell'elaboratore stesso, ad esempio il Sistema Operativo

- **Software applicativo**: dedicato alla realizzazione di specifiche applicazioni, ad esempio navigazione su internet, o videoscrittura o altro.

SISTEMI OPERATIVI

Sistema Operativo

- **E’ un software low-level che assiste l’utente e le applicazioni ad alto livello** ad interagire con l’hardware ed i dati che i programmi immagazzinano nel computer
- Un **OS esegue le operazioni base**, come ad esempio riconoscere l’input dalla tastiera, mandare l’output al display, tenere traccia delle directory e dei files nel disco, e controllare le periferiche come le stampanti

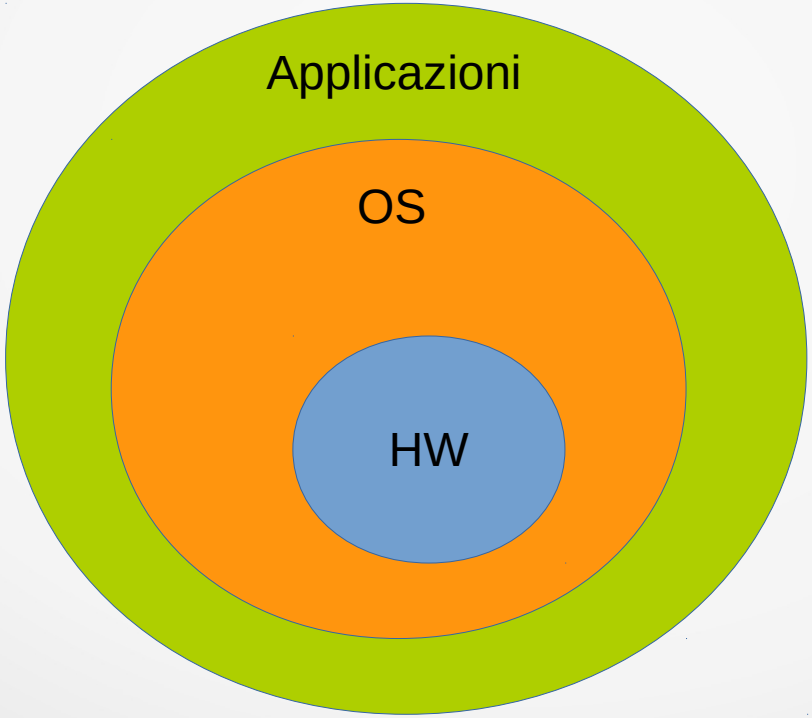


Sistema Operativo

- **Esecuzione dei programmi**: gli OS mettono a disposizione un ambiente dove l'utente puo' eseguire programmi applicativi senza doversi preoccupare dell'**allocazione della memoria o dello scheduling della CPU**
- **Operazioni di I/O**: Ogni applicazione richiede un qualche input e produce un qualche output. L'**OS nasconde i dettagli necessari a gestire questo tipo di operazioni** rispetto all'hardware sottostante
- **Comunicazione**: Ci sono situazioni in cui due **applicazioni devono comunicare l'una con l'altra, sia che si trovino sullo stesso computer (processi differenti), che su computer differenti**. L'OS si occupa di gestire questo tipo di **inter-process communication**

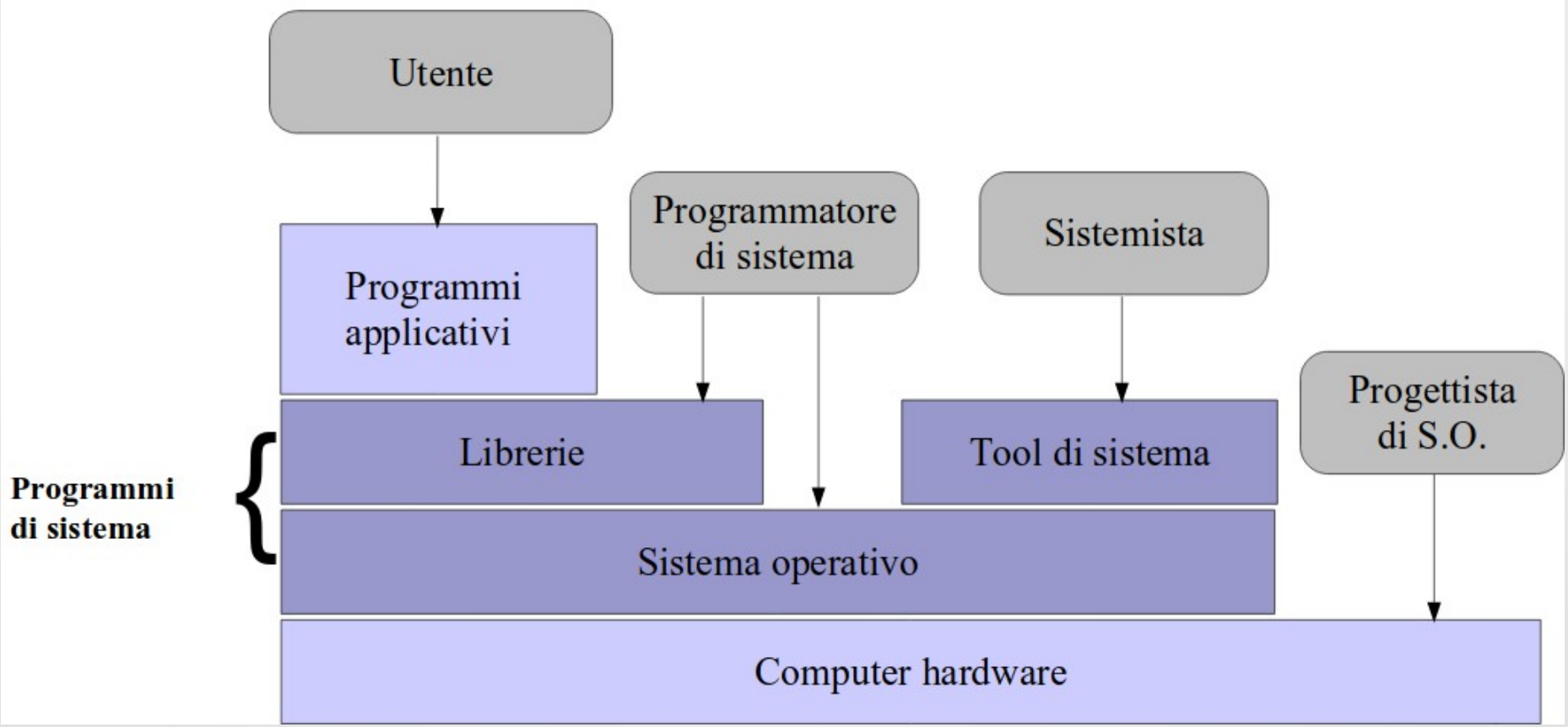
Sistemi Operativi

Rendono piu' semplice la scrittura di programmi applicativi che non devono preoccuparsi delle specifiche caratteristiche dell'HW.



Sistemi Operativi

Visione a strati delle componenti hardware e software, Fornisce ad esempio al programmatore **un API facile da usare**, nasconde i dettagli dell'hardware



SISTEMI OPERATIVI STORIA E CARATTERISTICHE SALIENTI

Sistema Operativo: storia

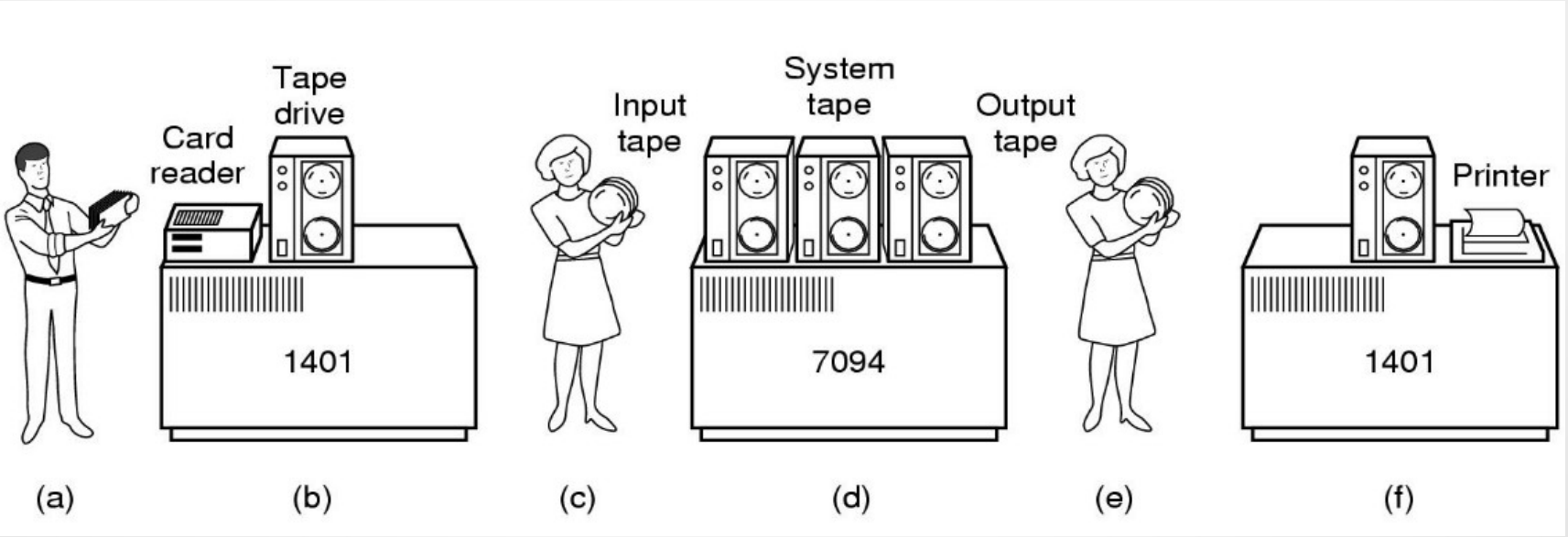
- **Babbage** (1792-1871) Cerca di costruire una **macchina meccanica** analitica e programmabile senza sistema operativo
  - **Prima programmatrice della storia Lady Ada Lovelace** (figlia di Byron)
- **Macchine a valvole** (1944-1955), sono **progettate , costruire e programmate da un singolo gruppo di persone**
  - **Programmate in linguaggio macchina**, usate per il solo calcolo numerico
  - **Non esiste OS**, non esiste distinzione fra progettista, programmatore ed utente
  - **Il singolo utente scrive i programma**, lo carica, carica i dati aspetta l'output e poi passa all'esecuzione del programma successivo
  - **Nessuno immagina che i calcolatori andranno mail oltre i laboratori di ricerca**

Sistema Operativo: storia

- **I transistor** (1955-1965) si possono costruire macchine piu' affidabili ed economiche
  - Si iniziano ad usare per **compiti diversi dal calcolo numerico** di base
  - **Chi costruisce la macchine e' diverso da chi la programma e quindi usa (utente == programmatore)**
  - Si introducono i **primi linguaggi ad "alto livello"** come Assembly e FORTRAN e si usano **schede perforate**
  - **Primi esempi di OS,** detti **Monitor residenti** :
    - **Controllo** sulla macchina e' dato al **monitor**
    - Il **controllo** e' **passato al job** che viene seguito
    - Una volta terminato il job, il **controllo torna al monitor**

Sistema Operativo: storia

- Per evitare i tempi morti fra l'esecuzione di un job e l'altro vengono introdotti i **nastri per la memorizzazione dei job**



Sistema Operativo: storia

- **I circuiti integrati** (1965 – 1980) **sparisce la figura dell'operatore come interfaccia verso il computer**:
  - La **Programmazione** avviene principalmente usando **linguaggi ad alto livello** come il C
  - Arrivano gli **editor di testi e** terminali che permettono di operare
  - **Compaiono i sistemi operativi** con caratteristiche moderne:
    - **Interattivi**
    - **Multi-programmazione**
    - **Time sharing**

Sistema Operativo: Multiprogrammazione

- **Utilizzare il processore mentre altri job stanno facendo i/O**
- Ci sono quindi **piu' job contemporaneamente** in memoria
- Lo **scheduler** (componente dell'OS) **gestisce l'uso della CPU**, mentre un job fa i/O l/altro usa la CPU , si **eliminano cosi' i tempi morti** (idle della CPU)
- Quindi **le routine di i/O devono essere fornite dall'OS**
- Il sistema operativo deve occuparsi di **allocare la memoria per i job multipli**
- Allo stesso modo l'OS deve essere in grado di **allocare le risorse di i/O fra processi diversi**

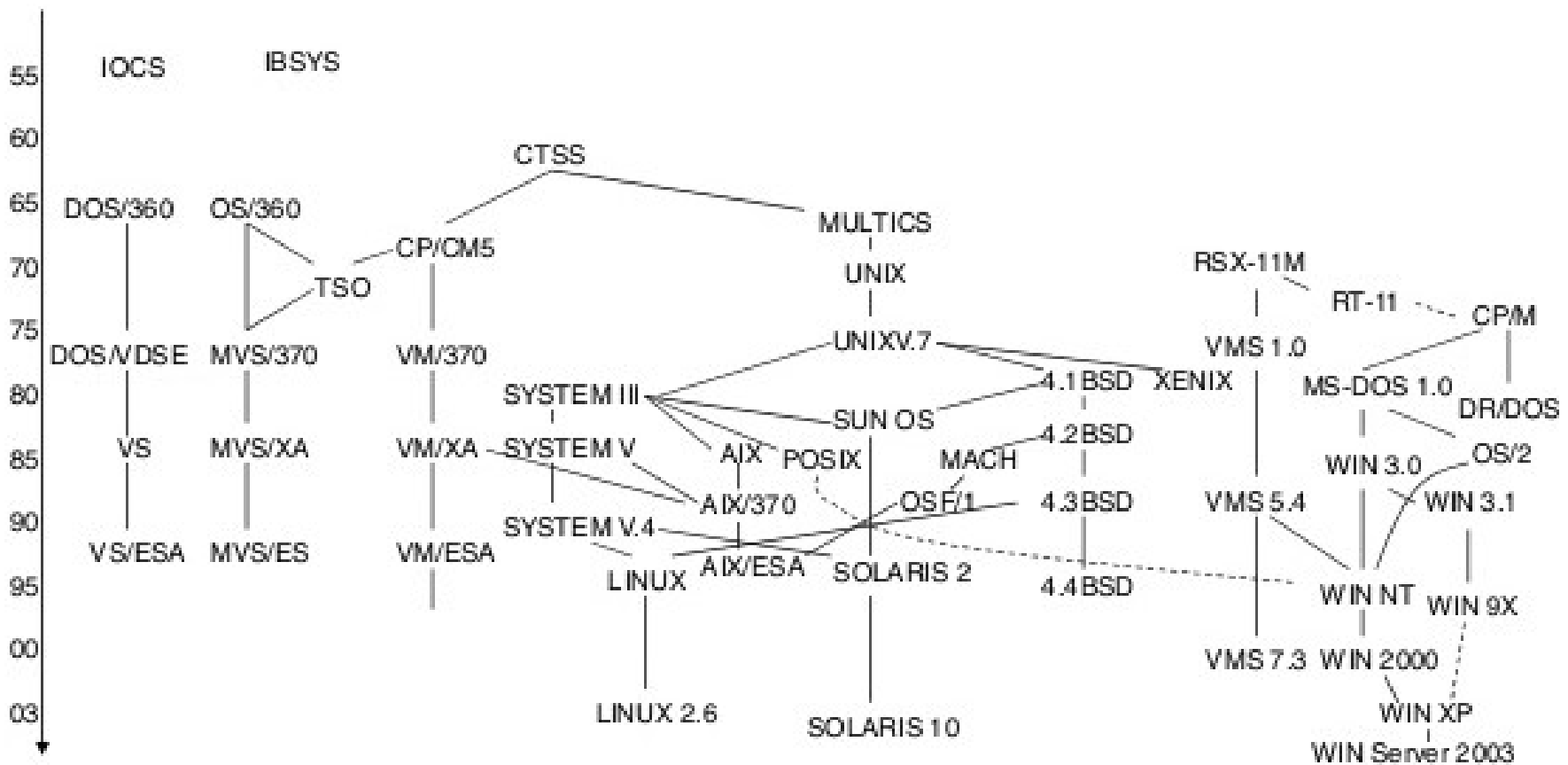
Sistema Operativo: Time-sharing

- E’ in buona sostanza l’**estensione del concetto di multiprogrammazione**
- Il tempo di esecuzione della CPU viene diviso in intervalli (**quanti**). Allo scadere di un quanto l’esecuzione del job corrente viene interrotta, anche se non deve fare i/O, e si passa ad un altro processo (job) . Processi in generale appartenenti ad utenti diversi
- Il **context-switch (passaggio)** e’ molto rapido e trasparente all;utente che ha l’**impressione che vengano eseguiti molti programmi in parallelo**
- La presenza di **piu’ utenti** rende necessario l'inserimento di meccanismi di protezione della memoria e del file-system
- Un job viene caricato dal disco alla memoria, e viceversa (**swapping**)

Sistema Operativo: Storia

- **CTSS (Compatible Time-Sharing System) 1965**: viene introdotto il concetto di multi programmazione ed il concetto di time-sharing
- **Multics 1965**: introduce il concetto di processo
- **Unix 1970**: derivato di Mutics e CTSS inzialmente sviluppato ai Bell-labs . Inizialmente sviluppato su due architetture specifiche, **poi sviluppato in C** (inizialmente tutto in assembly).

Sistemi Operativi: Storia



TERMINOLOGIA UN SUNTO

Sistema Operativo

- Tipicamente in un sistema informatico ci sono **(molti) processi concorrenti e “poche” risorse**, ad esempio la CPU. Un sistema operativo deve **gestire l'allocazione delle risorse**.
- Esiste un termine di sistemi operativi diffuso per tale allocazione noto come **Scheduling**
- A seconda del tipo di sistema operativo, gli obiettivi dello scheduling dei processi possono essere diversi.
- Indipendentemente dal tipo di sistema, al momento di **allocare le risorse deve essere prevista una certa equità**, la politica dichiarata deve essere applicata e l'utilizzo complessivo del sistema deve essere **bilanciato**.
- Gli obiettivi che devono essere raggiunti caratterizzano il tipo di sistema operativo.

Sistema Operativo: Multiprogrammato

- In un sistema operativo **multiprogrammazione** ci sono uno o **più programmi (processi) residenti nella memoria** principale del computer pronti per essere eseguiti. Solo un programma alla volta potra' usare la CPU, gli altri sono in attesa del loro turno.
- Se un dato programma (processo) chiede un'operazione di I/O mentre viene eseguita l''operazione in questione l'uso della CPU e' passato ad un altro processo
- **Il programma in esecuzione continua fino a quando non restituisce volontariamente la CPU o quando si blocca per svolgere funzioni di I/O**.
- Come potete vedere, l'obiettivo è molto chiaro: **i processi in attesa di IO non dovrebbero bloccare altri processi** che a loro volta sprecano il tempo della CPU. L'idea è di mantenere la CPU occupata finché ci sono processi pronti per essere eseguiti.

Sistema Operativo: Multiprogrammato

- Affinché tale sistema funzioni correttamente, **il sistema operativo deve essere in grado di caricare più programmi in partizioni separate della memoria** principale e **fornire la protezione** richiesta onde evitare che un processo acceda alla partizione di memoria dell'altro.
- Quando si hanno più programmi in memoria l'OS deve gestire la frammentazione, di fatto quando i programmi entrano o escono (swapping) dalla memoria principale.
- Un altro problema che deve essere gestito è che **programmi di grandi dimensioni potrebbero non poter essere caricati in una sola volta in memoria, questo aspetto e' generalmente gestito mediante il concetto di Virtual Memory**. Nei moderni sistemi operativi i programmi sono suddivisi in blocchi di dimensioni uguali chiamati pagine
- il termine multi-programmazione è un termine **vecchio** perché nei moderni sistemi operativi l'intero programma non è caricato completamente nella memoria principale

Sistema Operativo: Multitasking

- Il **multitasking** ha lo stesso significato della multiprogrammazione in senso generale poiché entrambi si riferiscono al fatto di **avere più (programmi, processi, attività, thread) in esecuzione allo stesso tempo**.
- Multitasking è il termine utilizzato nei moderni sistemi operativi quando più attività condividono una risorsa comune (CPU e memoria). In qualsiasi momento la CPU sta eseguendo un'attività solo mentre altre attività attendono il proprio turno.
- **L'illusione del parallelismo** si ottiene quando la CPU viene riassegnata a un'altra attività (context switch). Esistono poche differenze principali tra multitasking e multiprogrammazione.
- Un **task in un sistema operativo multitasking non è necessariamente un intero programma applicativo** (i programmi nei moderni sistemi operativi sono suddivisi in **pagine logiche**).
- **Il task può anche riferirsi a un thread** di esecuzione quando un processo è suddiviso in attività secondarie.
- Il task non usa la CPU fino a quando non termina come nel vecchio modello multiprogrammazione, **ma piuttosto ha una quantità precisa del tempo della CPU chiamato quantum**.
- **Per iper-semplificare il multitasking e la multiprogrammazione si riferiscono ad un concetto simile (condivisione del tempo di CPU) il primo viene utilizzato nei sistemi operativi moderni mentre lil secondo veniva utilizzato in sistemi operativi precedenti**.

Sistema Operativo: Time-Sharing

- Chiaramente in un sistema a processore singolo, l'esecuzione parallela è un'illusione. Solo una singola istruzione di un singolo processo (task) alla volta puo' essere eseguita dalla CPU
- **Il tempo della CPU viene condiviso tra i processi (tasks)**. I sistemi operativi multiprogrammazione e multitasking non sono altro che sistemi di condivisione del tempo.
- Nella multiprogrammazione, anche se la CPU è condivisa tra i programmi, ma non è l'esempio ottimale di time sharing, infatti un programma continua a funzionare finché non si blocca (termina o operazione di I/O)
- **In un sistema multitasking (sistema operativo moderno) la quantità totale del tempo della CPU viene divisa in quanti temporali, ed ogni processo o task usa un numero equo di quanti temporali**

Sistema Operativo: Multiutente

- Un sistema single user ma multitasking permette ad un singolo utente di accedere a computer ma il singolo utente potra’ avviare piu’ applicazioni (task)

- Multiutente: piu’ utenti potranno acceder contemporaneamente al sistema ed condividerne le risorse , ed il sistema si occupera’ di schedulare l’uso delle risorse fra i vari utenti e quindi processi

Sistema Operativo: Multiprocesso

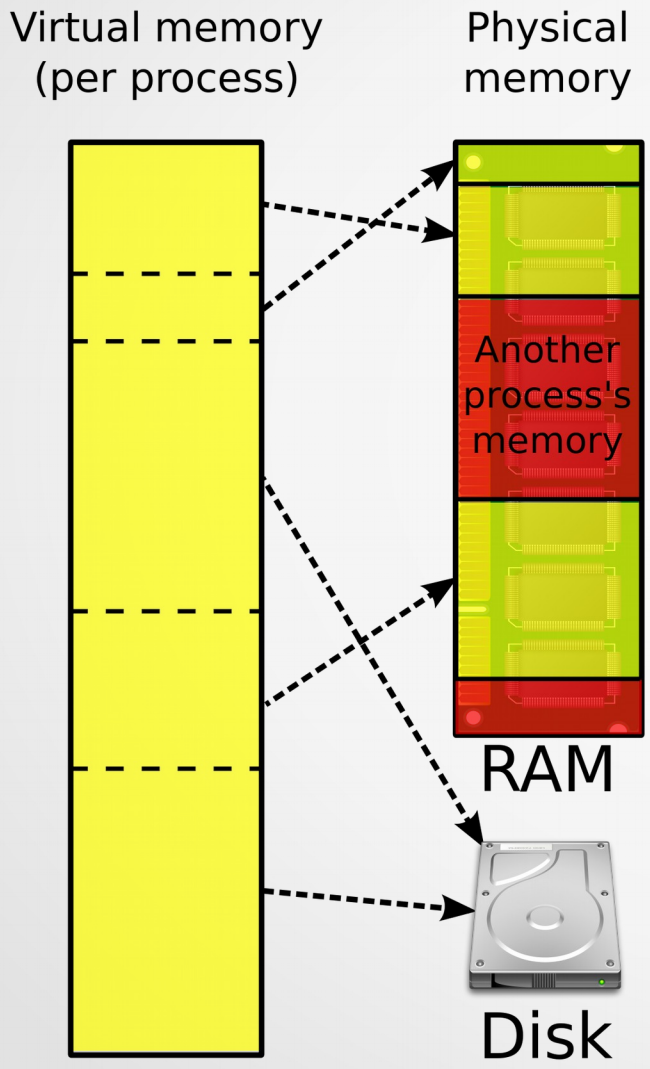
- Il **multiprocesso** a volte si riferisce all'esecuzione di più processi (programmi) allo stesso tempo. Questa dedizioni puo' creare confusione, avendo appena visto il concetto di multiprogrammato e multitasking
- **Il concetto di multiprocesso "generalmente" si riferisce effettivamente alle unità CPU anziché ai processi in esecuzione. Cioe' e' l'hardware sottostante che fornisce più di un processore.**
- Esistono molte varianti sullo schema di base, ad esempio avere più core su un singolo die o più die in un unico package o più package in un unico sistema.
- In sintesi, il multiprocesso si riferisce essenzialmente all'hardware sottostante (più CPU, core)

Sistema Operativo: Multithreading

- Il **multi threading** è un modello di esecuzione che consente a un **singolo processo di avere più segmenti di codice (thread) eseguiti** contemporaneamente nel contesto di quel processo.
- È possibile pensare ai thread come processi secondari che condividono le risorse del processo padre ma che vengono eseguite indipendentemente. **Più thread di un singolo processo possono condividere la CPU in un singolo sistema CPU o (puramente) eseguire in parallelo in un sistema multicore ad esempio**.
- Un sistema multitasking può avere processi multi-thread in cui diversi processi condividono la CPU e allo stesso tempo ognuno ha i propri thread.
- Ad esempio una **GUI in cui vuoi emettere un comando che richiede tempi lunghi di esecuzione (un calcolo matematico complesso). A meno che non si esegua questo comando in un thread di esecuzione separato, non sarà possibile interagire con la GUI dell'applicazione principale**

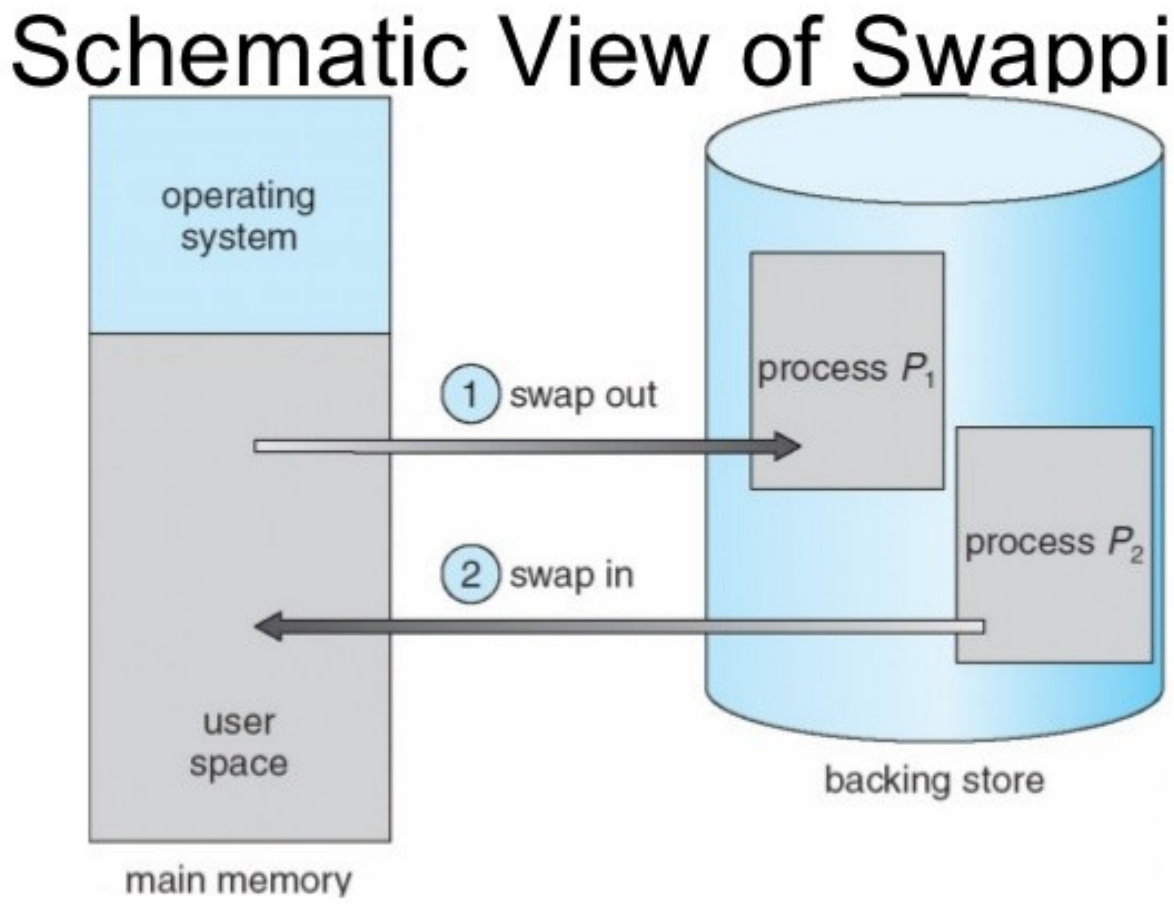
DUE PICCOLI APPROFODNIMENTI

Virtual Memeory



La memoria virtuale (Virtual Memory) è una tecnica di gestione della memoria che fornisce una "astrazione idealizzata delle risorse di archiviazione effettivamente disponibili su una data macchina" che quindi "crea l'illusione per gli utenti (processi, programmi, applicazioni)di un grande spazio di memoria. "

Swapping


Schematic View of Swapping
operating system
user space
main memory
1 swap out
2 swap in
process P_1
process P_2
backing store

IN BREVE COMPONENTI FONDAMENTALI DI UN SISTEMA OPERATIVO

Sistema Operativo: gestione dei processi

- Un processo e' in pratica **un programma in esecuzione , che quindi utilizza risorse**
- l'OS deve essere in grado di:
  - **Creare e terminare i processi** stessi
  - **Gestire la comunicazione fra i processi** stessi
  - **Sospendere e riattivare i processi**

Sistema Operativo: gestione della memoria principale

- La memoria principale in pratica e' un'array di byte indirizzabili singolarmente, che puo' essere condiviso fra CPU e dispositivi di I/O
- l'OS deve essere in grado di:
  - **Tenere traccia ad esempio di quale aree di memoria sono utilizzate e da chi** (da quale processo)
  - **Decidere ad esempio per quali processi allocare una data area di memoria** quando e' libera
  - In ultima analisi **allocare a liberare lo spazio in memoria**

Sistema Operativo: gestione della memoria secondaria

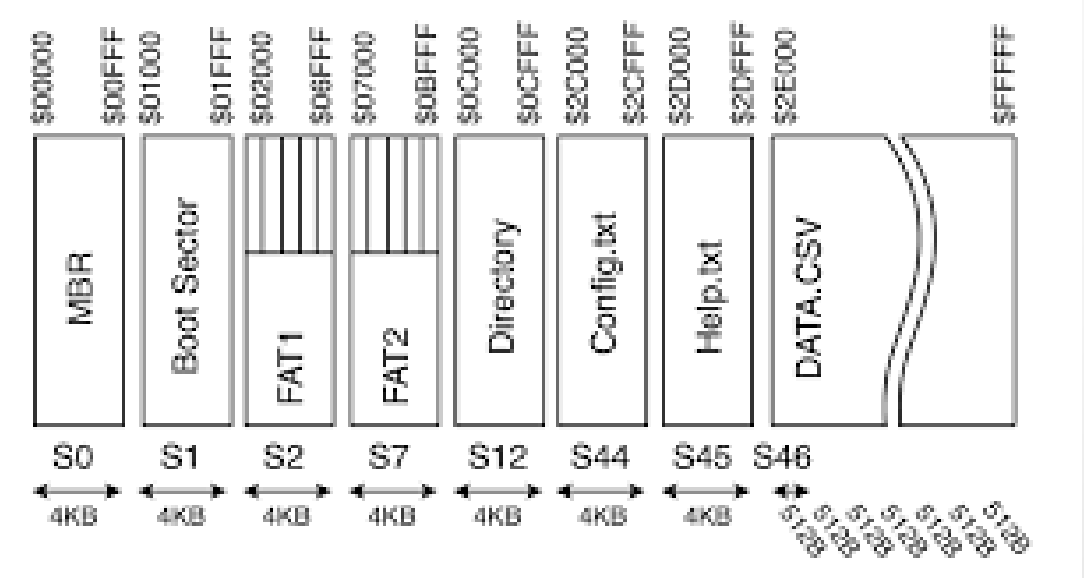
- I calcolatori sappiamo sono dotati di una **memoria secondaria** non volatile e di grandi dimensioni (gradi rispetto alla memoria principale che e' volatile)
- L'OS deve poter gestire le seguenti attivita':
  - **Allocare lo spazio richiesto e liberarlo** quando non piu' utilizzato
  - **Gestire lo scheduling di accesso ai dispositivi** (ad esempio hard-disk o CD/DVD o chiavette USB)

Sistema Operativo: gestione del filesystem

- Un **file** (archivio) e' l'astrazione informatica del concetto di contenitore di informazioni, indipendentemente dal dispositivo sul quale viene memorizzato
- Un **filesystem** e' sostituito da molti **files** (una **directory** contiene riferimenti ad altri files), L'OS deve poter gestire le seguenti attivita':
  - **Creare e cancellare files e directory,** manipolarli
  - **Codificare il filesystem** sulla memoria secondaria
  - Esempi, ext3, ext4, NTFS, FAT32 ...

Sistema Operativo: gestione del filesystem

- Il **filesystem** : deve gestire l'allocazione dello spazio sul disco, mantenere ad esempio un indice di dove sono memorizzati i dati relativi ad un dato file . Mantenere le caratteristiche di un file ad esempio i dati di accessi, i permessi, i nomi etc etc.



Sistema Operativo: gestione dell'I/O

- L'OS deve gestire l'I/O e quindi l'interazione con i vari dispositivi hardware:
  - Un interfaccia comune per la gestione dei vari d**evice driver**
  - Avere diversi **driver (kernel module)** per i diversi dispositivi , quindi componenti specifiche per le varie componenti hardware del sistema
  - Un sistema di **buffering e caching** delle informazioni da e verso i vari dispositivi

Sistema Operativo: protezione

- L’OS deve implementare un **meccanismo di protezione software**. Quindi gestire e controllare l’accesso dei vari programmi (processi) alle risorse condivise del sistema (ad esempio la memoria)
  - Distinguere fra uso autorizzato o meno
  - Fornire meccanismi di base per attuare la protezione

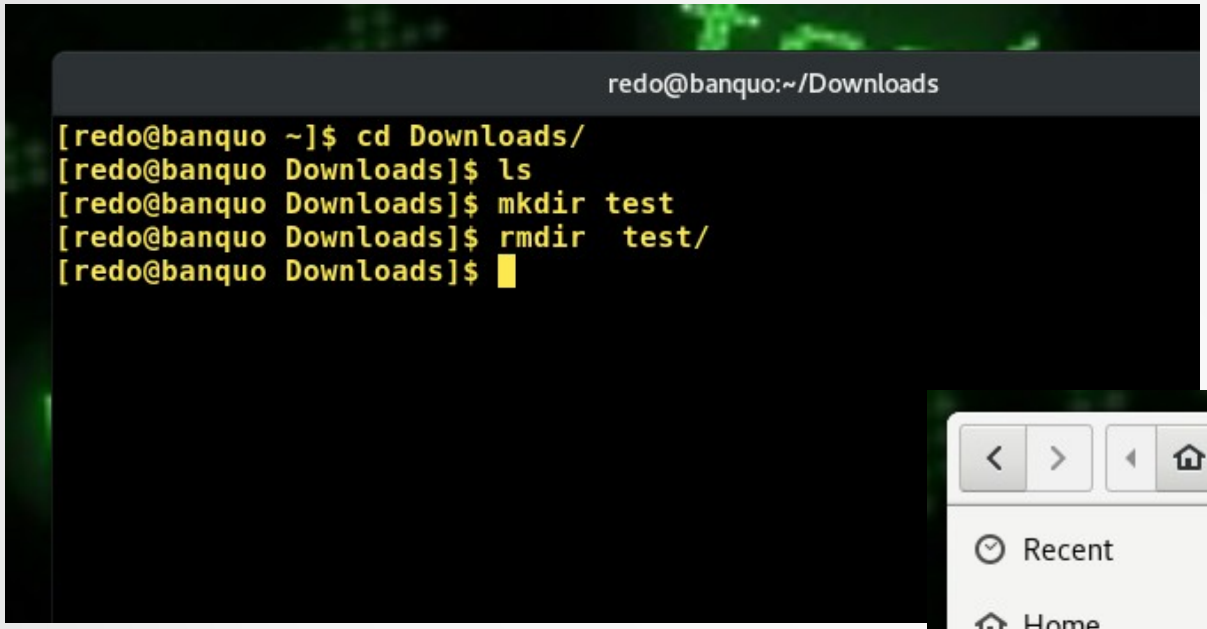
Sistema Operativo: networking

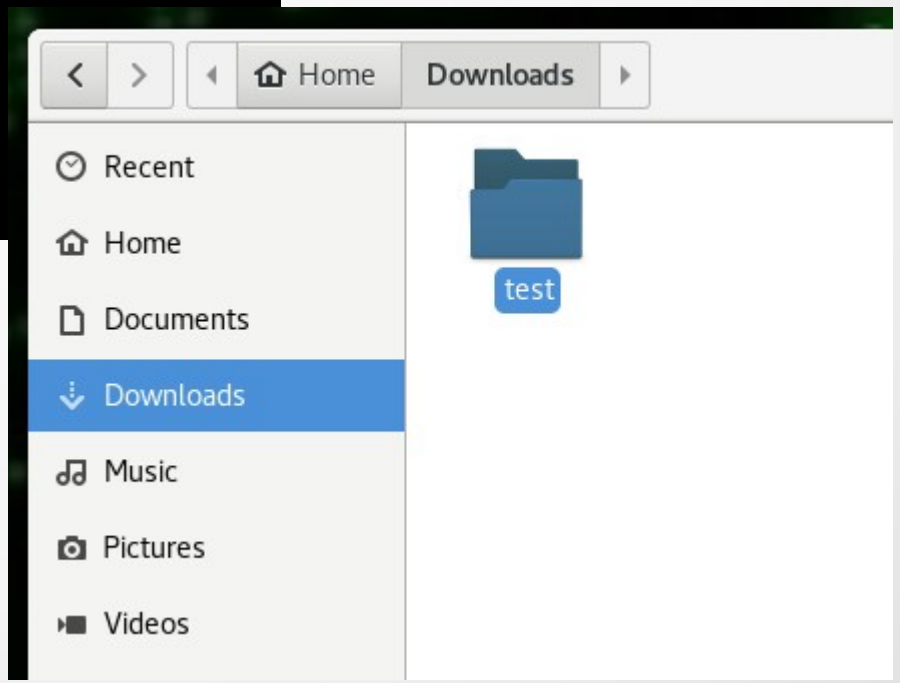
- Oramai una componente essenziale di un OS e’ appunto il networking, q uidni la capacita’ di far comunicare due o piyu’ elaboratori (processi) e di condividere risorse.
- Un’OS fornisce i protocolli di comunicazione di base come **TCP/IP , UDP**
- **Oltre che servizi di comunicazione ad alto livello** come ad esempio filesystem condivisi (SMB, NFS) e molti altri.

Sistema Operativo: interprete dei comandi

- I sistemi operativi forniscono un’interfaccia utente:
  - Avviare o terminare un programma
  - Interagire con le componenti base del sistema operativo , come il filesystem
- Possiamo avere essenzialmente due tipi:
  - Grafica, e quindi icone e finestre
  - Testuale, linea di comando

Sistema Operativo: interprete dei comandi





CAMBIO DI PROSPETTIVA

Sistema Operativo: prospettiva del programmatore

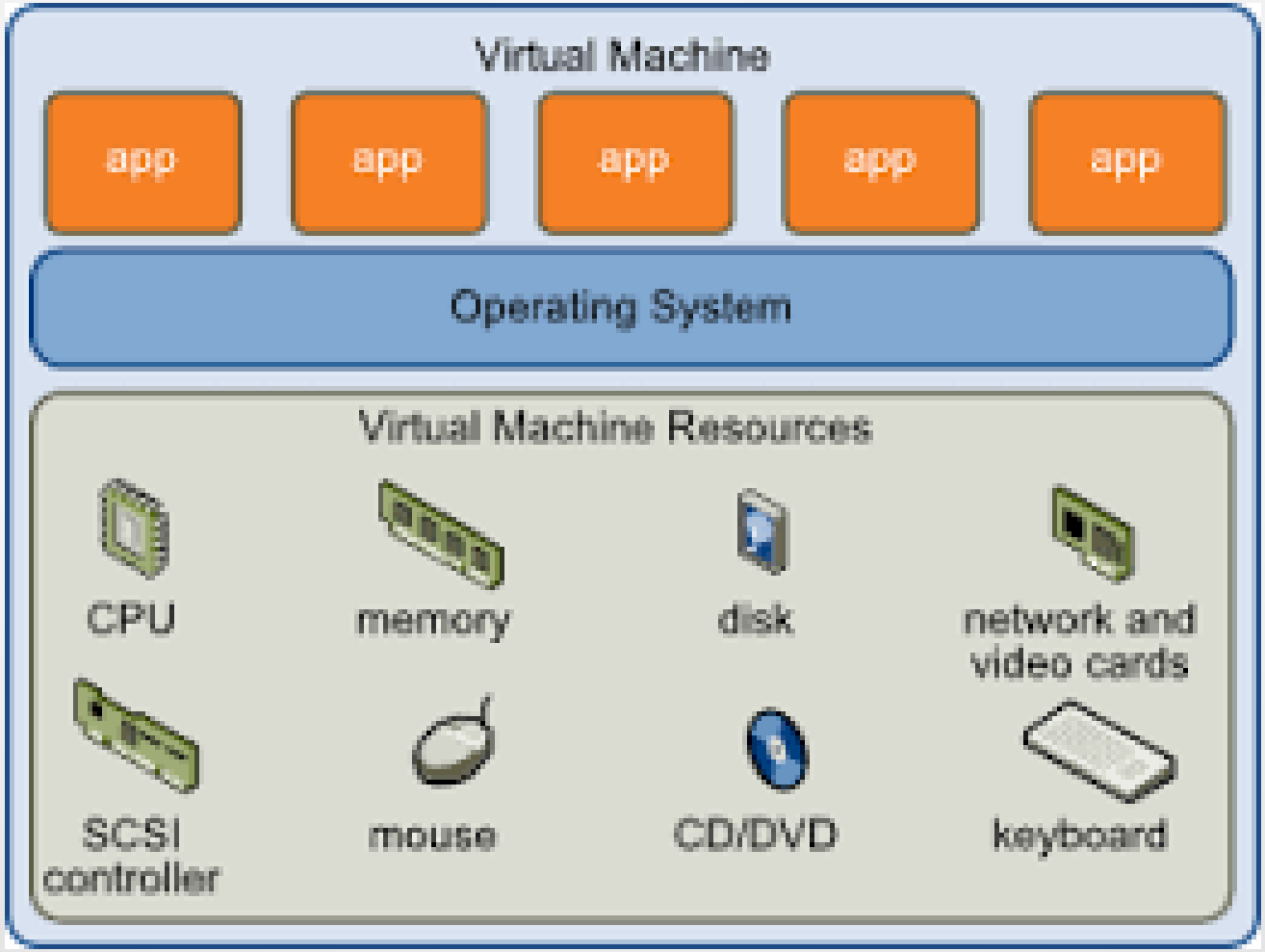
- La System Call (chiamate di sistema) ad esempio:

UNIX	WIN32	UNIX	WIN32
fork	CreateProcess	mkdir	CreateDirectory
waitpid	WaitForSingleObject	rmdir	RemoveDirectory
execve	-	link	-
exit	ExitProcess	unlink	DeleteFile
open	CreateFile	mount	-
close	CloseHandle	umount	-
read	ReadFile	chdir	SetCurrentDirectory
write	WriteFile	chmod	-
lseek	SetFilePointer	kill	-
stat	GetFileAttributesEx	time	GetLocalTime

VMs, DOCKER e Cloud Computing

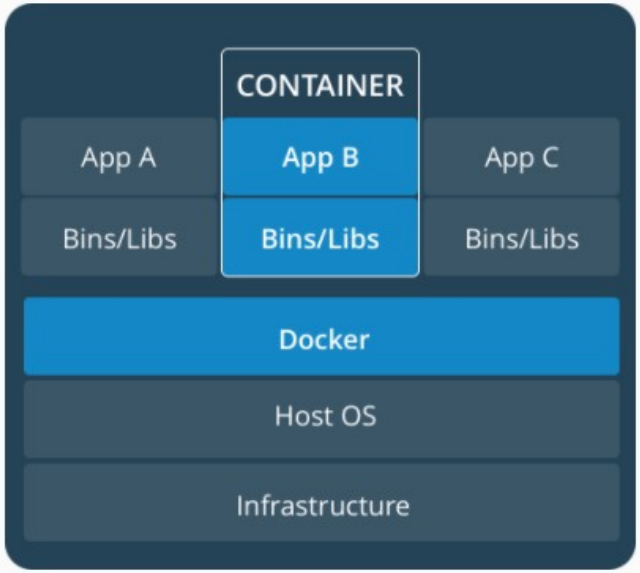
Cloud computing

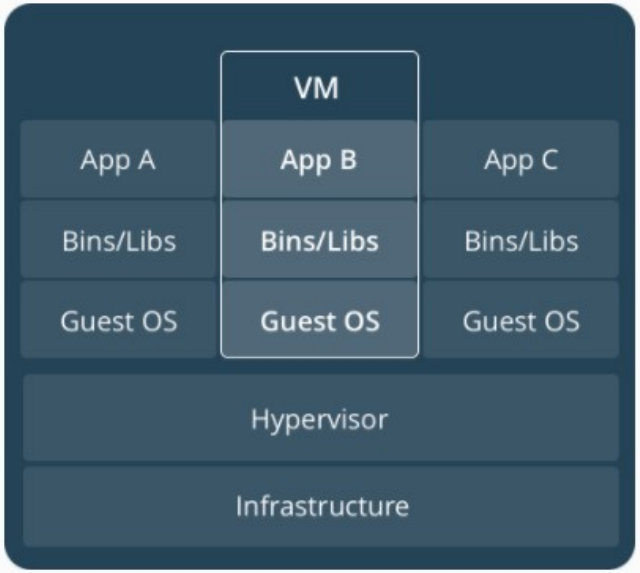
- Virtual Machine



Cloud computing

- Docker vs VM





CONTAINERS

Containers are an abstraction at the app layer that packages code and dependencies together. Multiple containers can run on the same machine and share the OS kernel with other containers, each running as isolated processes in user space. Containers take up less space than VMs (container images are typically tens of MBs in size), and start almost instantly.

VIRTUAL MACHINES

Virtual machines (VMs) are an abstraction of physical hardware turning one server into many servers. The hypervisor allows multiple VMs to run on a single machine. Each VM includes a full copy of an operating system, one or more apps, necessary binaries and libraries - taking up tens of GBs. VMs can also be slow to boot.

Cloud computing

- Infrastruttura virtuale