

# **Reti di Calcolatori**

UniVR - Dipartimento di Informatica

**Fabio Irimie**

1° Semestre 2024/2025

# Indice

<b>1</b>	<b>Introduzione</b>	<b>2</b>
<b>2</b>	<b>Architetture di rete</b>	<b>3</b>
2.1	Reti locali . . . . .	3
2.1.1	Organizzazione del Backbone . . . . .	4
<b>3</b>	<b>Modalità di comunicazione</b>	<b>5</b>
3.1	Reti a commutazione di circuito . . . . .	5
3.1.1	Vantaggi . . . . .	6
3.1.2	Svantaggi . . . . .	7
3.2	Reti a commutazione di pacchetto . . . . .	7
3.2.1	Vantaggi . . . . .	8
3.2.2	Svantaggi . . . . .	8
<b>4</b>	<b>Ritardi di trasmissione</b>	<b>9</b>
4.1	Ordine di grandezza . . . . .	10
4.2	Strumenti per calcolare il ritardo end-to-end . . . . .	10
4.2.1	Ping . . . . .	10
4.2.2	Traceroute . . . . .	11
4.3	Quantità di dati trasferiti . . . . .	12
<b>5</b>	<b>Modello a strati</b>	<b>12</b>
5.1	Stack ISO/OSI . . . . .	13
5.2	Stack TCP/IP . . . . .	14
5.3	Entità coinvolte nella comunicazione . . . . .	15
5.3.1	Identificazione dei processi . . . . .	15
5.3.2	Ottenimento dell'IP e della porta . . . . .	16
<b>6</b>	<b>Indirizzi IP</b>	<b>17</b>
6.1	Suddivisione dei bit . . . . .	17
6.1.1	Identificazione del prefisso e del suffisso . . . . .	18
6.2	Indirizzi IP riservati . . . . .	19
6.3	Conoscere il proprio indirizzo IP . . . . .	19
6.4	Esercizi . . . . .	20
6.5	Subnetting . . . . .	21
6.5.1	Creazione delle sottoreti partendo da un blocco di indirizzi assegnato . . . . .	21
<b>7</b>	<b>Livello applicativo</b>	<b>23</b>
7.1	Esempi di applicazioni e relativi protocolli di trasporto . . . . .	24
7.2	Socket . . . . .	24
7.3	Protocolli . . . . .	25
7.3.1	Protocollo HTTP . . . . .	25

# 1 Introduzione

Il problema principale che bisogna affrontare è la comunicazione tra 2 calcolatori, cioè lo scambio di informazioni. Per far comunicare 2 calcolatori c'è bisogno di alcuni requisiti:

1. **Protocollo:** È un insieme di regole che sovraintende alla comunicazione, in cui si definiscono:

- Il formato dei messaggi
- Le azioni da intraprendere nel gestire i messaggi stessi

Questo perchè per comunicare tutti devono "parlare la stessa lingua".

2. **Architettura di rete:** Come, fisicamente, trasportare i messaggi

**Esempio 1.1.** Prendiamo ad esempio la scrittura e la spedizione delle lettere. Ci sono 2 utenti che vogliono scambiare delle lettere.

Per gestire il trasporto della lettera essa viene messa all'interno di una **busta**, che contiene informazioni su dove deve essere spedita. Una volta imbustata, va imbucata in una cassetta delle lettere da cui poi verrà prelevata e mandata alla cassetta delle lettere del secondo utente dalla **rete di distribuzione** degli uffici postali.

L'utente poi preleverà la lettera dalla cassetta delle lettere e dopo aver controllato le informazioni sulla busta, la aprirà e leggerà il contenuto.

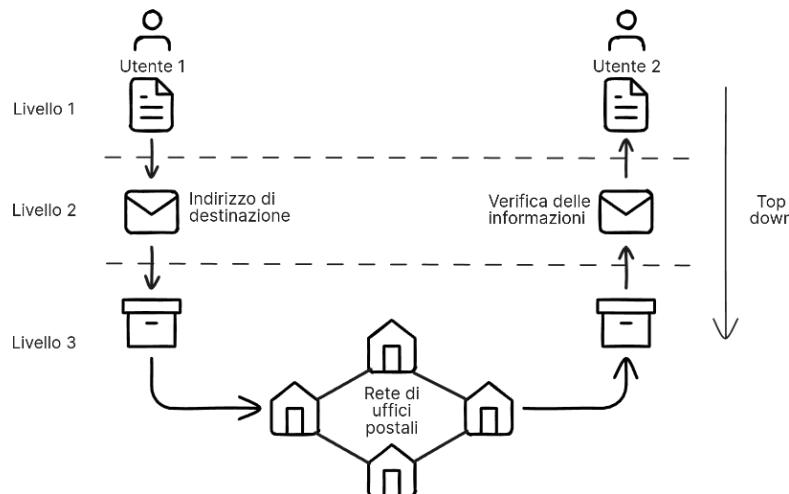


Figura 1: Esempio di comunicazione tra 2 utenti

Il **Protocollo** è il linguaggio utilizzato per comunicare tra i 2 utenti, mentre l'**Architettura di rete** è tutta quella infrastruttura che trasporta il messaggio tra i 2 utenti.

La rappresentazione dei sistemi di comunicazione di solito viene fatta nella modalità **top-down**, cioè si parte dal livello applicativo, quello più alto, fino a scendere nei livelli più bassi in cui si trova la vera e propria architettura della rete.

## 2 Architetture di rete

Di solito si fa riferimento all'architettura più utilizzata, cioè la rete **Internet**. Si possono distinguere i seguenti elementi base:

- **Calcolatori (End host)**
- **Router (Intermediate host)**
- **Collegamenti**

### 2.1 Reti locali

Le reti locali, o **LAN** (Local Area Network), sono caratterizzate da un **router di bordo** a cui sono collegati gli end host tramite **cavi fisici** o collegamenti wireless.

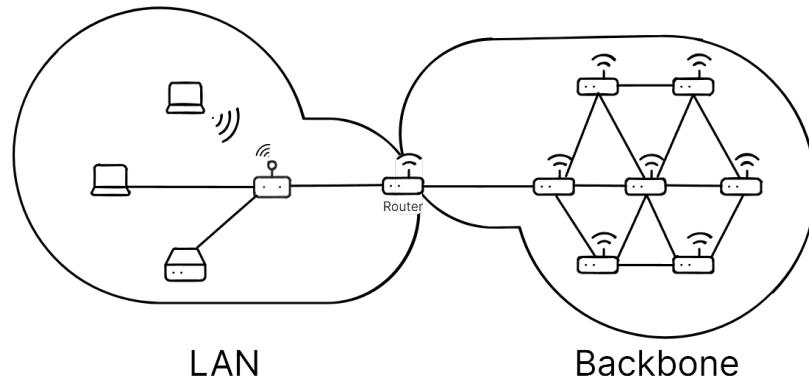


Figura 2: Rete locale

Per collegare diverse LAN tra loro esiste la **backbone**, cioè è una rete di router collegati tra di loro con una topologia gestita dal gestore della rete. Questi router sono geograficamente distribuiti su tutto il territorio.

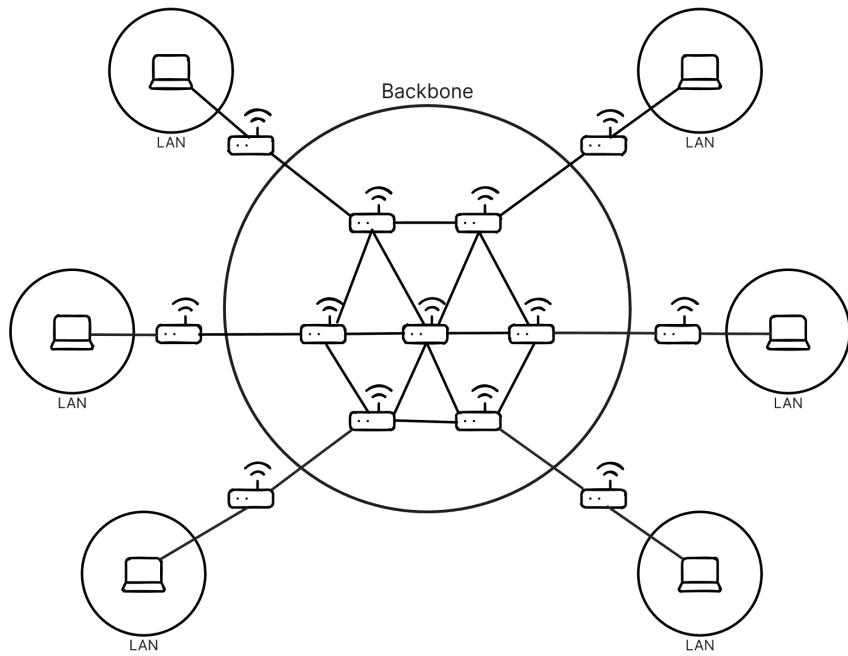


Figura 3: Intercollegamento di LAN

Nella maggior parte delle volte le connessioni tra i router all'interno della backbone sono cablate, di solito tramite fibre ottiche, soltanto in rari casi si usano connessioni wireless.

#### 2.1.1 Organizzazione del Backbone

Il backbone è composto da diverse reti che appartengono a diverse organizzazioni permettendo di creare diverse interconnessioni tra le reti. Queste organizzazioni si chiamano **Internet Service Provider** (ISP). Gli ISP hanno diversi livelli:

1. **Livello 1:** Hanno una connessione internazionale e quindi sono in grado di comunicare con tutti gli altri ISP.
2. **Livello 2:** Lavorano a livello nazionale.
3. **Livello 3:** Lavorano a livello locale.

Gli ISP di livello 1 sono collegati tra di loro per permettere la comunicazione tra ISP di livello 1 diversi. Anche gli ISP di livello più basso permettono la comunicazione tra di loro o tra gli ISP di livello superiore, tutto questo grazie ad accordi commerciali tra le varie organizzazioni.

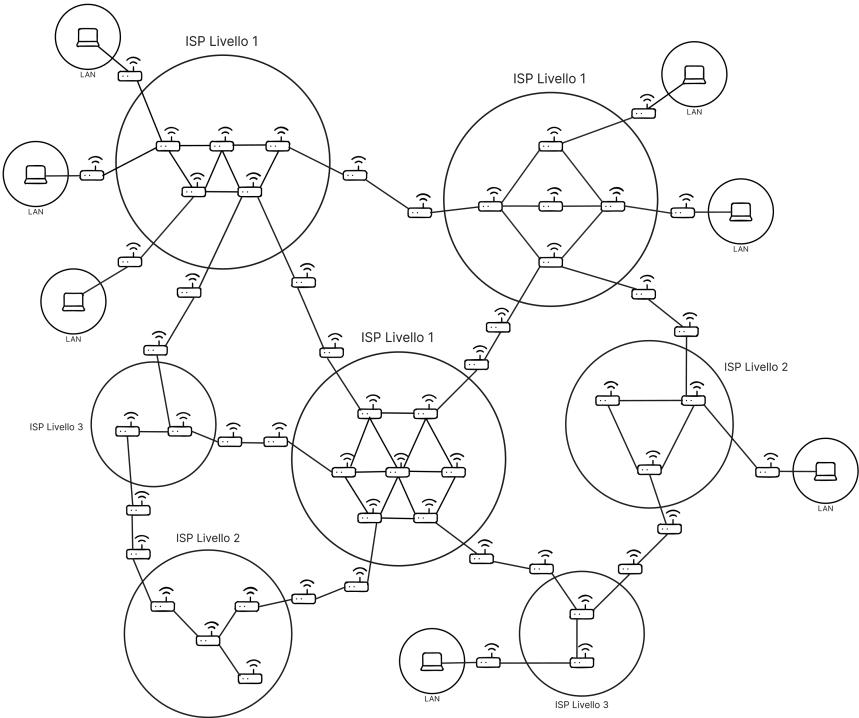


Figura 4: Livelli di ISP

**Internet** è la **Rete delle reti**, cioè è la rete che collega tutti gli ISP tra di loro ed è un organizzazione gerarchica, cioè è sufficiente creare collegamenti con un sottoinsieme di ISP operanti sul territorio per permettere il collegamento a tutta la rete.

Di conseguenza, per raggiungere un utente, in genere, si segue un percorso gerarchico. Un esempio è la rete stradale, dove per raggiungere una città si seguono le strade principali e poi si scende in quelle secondarie.

La scelta del percorso segue criteri basati su distanza e tempo.

### 3 Modalità di comunicazione

La gestione del trasporto dei messaggi è gestita dalla rete, però con che modalità trasferisco l'informazione tra 2 utenti?

#### 3.1 Reti a commutazione di circuito

È la modalità di commutazione che è stata utilizzata per la prima volta.

In questa modalità le risorse (capacità del canale di trasmissione) vengono riservate **end-to-end** per la comunicazione, cioè viene letteralmente riservato un circuito che viene utilizzato dai 2 utenti.

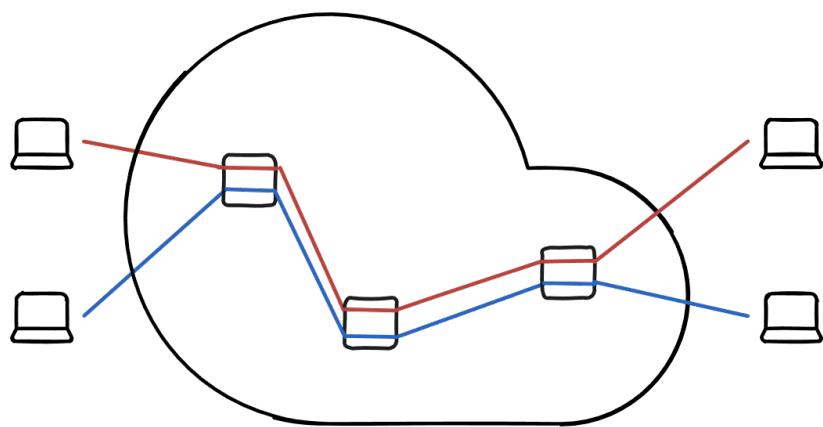


Figura 5: Comutazione di circuito

Ogni canale è completamente dedicato alla comunicazione tra i 2 utenti, quindi se più utenti vogliono comunicare tra di loro, bisogna riservare altre risorse.

### 3.1.1 Vantaggi

- Risorse dedicate
- Ritardo deterministico

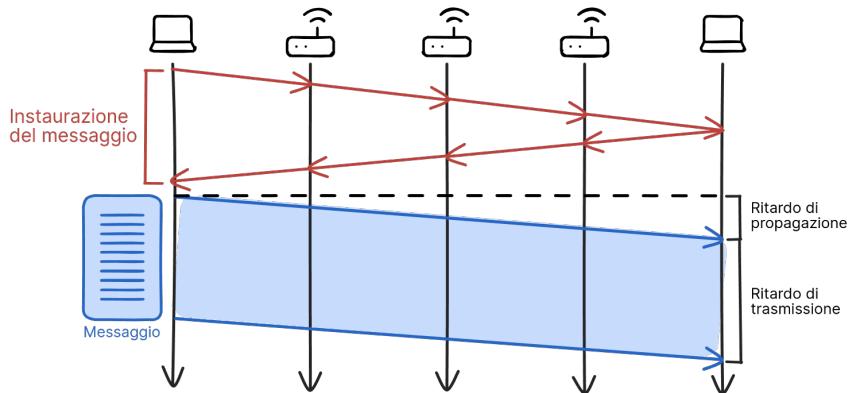


Figura 6: Ritardo

- **Ritardo di trasmissione:** Tempo necessario per trasmettere il messaggio
- **Ritardo di propagazione:** Tempo necessario per trasmettere il messaggio da un nodo all'altro

Se il messaggio è grande  $L \text{ bit}$  e il canale riservato è di  $B \text{ bit/s}$ , allora il tempo di trasmissione sarà:

$$T = \frac{L}{B}$$

Il ritardo di trasmissione e di propagazione è deterministico, perché dato il circuito di trasmissione, il tempo di trasmissione è noto.

### 3.1.2 Svantaggi

Nel corso di utilizzo sporadico si ha uno spreco di risorse, perché il circuito viene riservato per tutta la durata della comunicazione, anche se i 2 utenti non stanno comunicando.

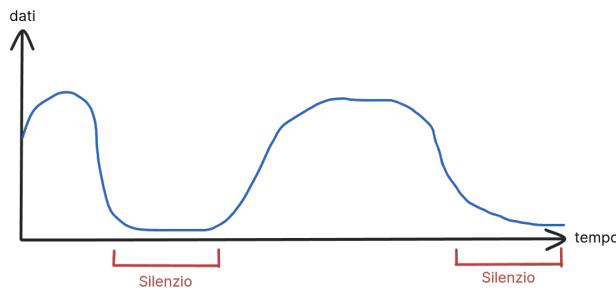


Figura 7: Spreco di risorse

## 3.2 Reti a commutazione di pacchetto

È la modalità di commutazione più utilizzata al giorno d'oggi.  
L'informazione (messaggio) viene suddivisa in **pacchetti** e ad ogni pacchetto viene aggiunto un **header** per permettere:

- La consegna del pacchetto stesso
- La ricostruzione del messaggio

Il messaggio è l'informazione da trasferire, mentre il pacchetto è una porzione del messaggio stesso.

Il messaggio prima della trasmissione viene separato in unità più piccole e a queste unità viene aggiunta un'intestazione che serve a rendere le unità indipendenti per poterle trasmettere in modo indipendente. L'intestazione permette la consegna del pacchetto perché contiene la destinazione del pacchetto e la ricostruzione del messaggio perché contiene il numero di sequenza del pacchetto.

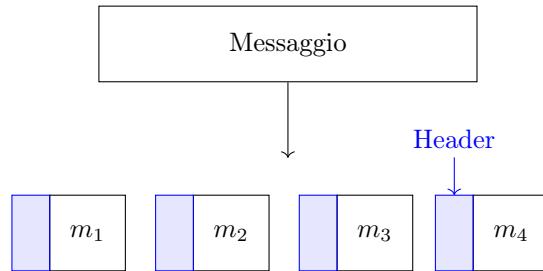


Figura 8: Messaggio e pacchetti

I pacchetti vengono salvati all'interno di un **buffer**, cioè una memoria temporanea, in attesa di essere trasmessi. Man mano che i pacchetti vengono inviati vengono accumulati nel buffer dei router e questo avviene per qualsiasi collegamento.

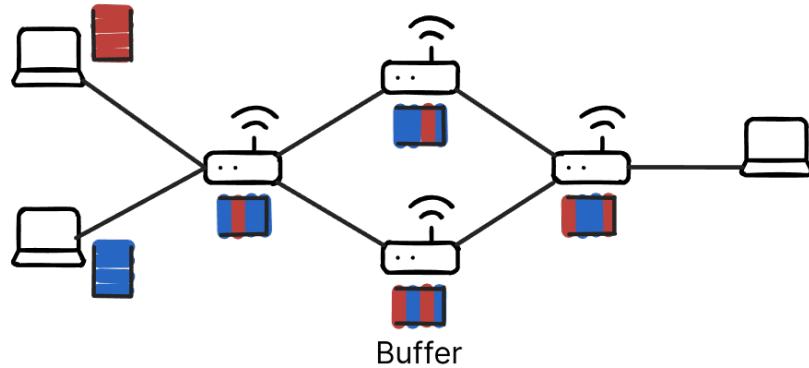


Figura 9: Rappresentazione del buffer

I pacchetti possono arrivare in ordine diverso rispetto a quello di trasmissione, però grazie all'header è possibile ricostruire il messaggio originale.

### 3.2.1 Vantaggi

- Utilizza le risorse solo quando ci sono pacchetti da trasmettere e questo viene chiamato **commutazione statistica**.
- **Multiplazione statistica**, cioè utilizzo lo stesso canale per trasmettere più pacchetti di utenti diversi.

### 3.2.2 Svantaggi

- Potenziale perdita dei pacchetti: La memoria dei buffer ha una capacità finita, quindi se il tasso di ricezione dei pacchetti è superiore al tasso di smaltimento del buffer, esso inizia a riempirsi. Le perdite aumentano la complessità di gestione della rete.

- Ritardi aumentati

I router prima di trasmettere i pacchetti, deve aspettare di ricevere tutti i pacchetti. Questo si chiama **store & forward**. Di conseguenza più aumentano i router, più aumenta il ritardo di trasmissione.

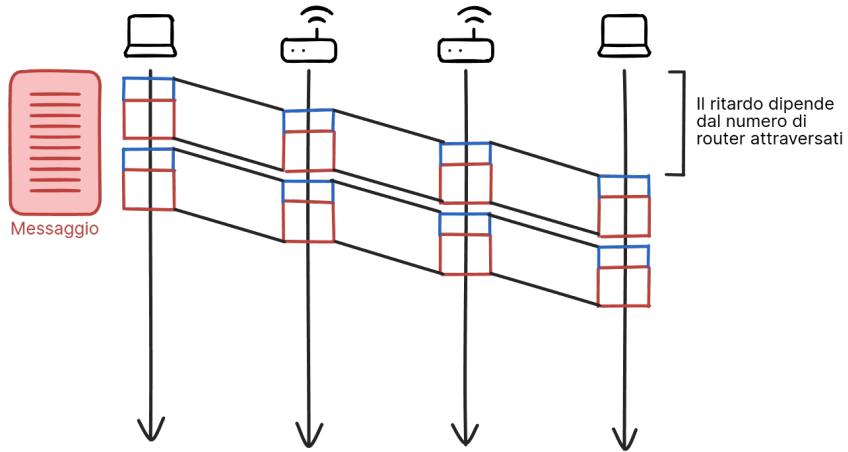


Figura 10: Ritardo di trasmissione dei pacchetti

## 4 Ritardi di trasmissione

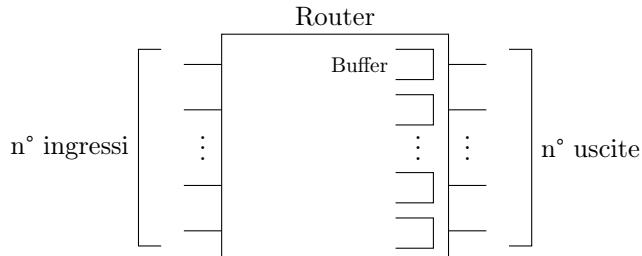


Figura 11: Struttura di un router

Su un singolo router le componenti principali del ritardo sono:

- **Ritardo di elaborazione al nodo:** Tempo necessario per elaborare il pacchetto.
- **Ritardo di accodamento:** È il tempo speso nel buffer prima che il pacchetto venga trasmesso ed è la componente principale tra tutti i ritardi.  $(\frac{L}{B})$
- **Ritardo di trasmissione:** Tempo necessario per trasmettere il pacchetto.
- **Ritardo di propagazione:** Tempo necessario per trasmettere il pacchetto da un nodo all'altro.

## 4.1 Ordine di grandezza

Dipende dalla distanza e dalla velocità di trasmissione del collegamento. La distanza si distingue in:

- **Locale:**  $< 10ms$
- **Internazionale:**  $20 - 40ms$
- **Intercontinentale:**  $> 100ms$

## 4.2 Strumenti per calcolare il ritardo end-to-end

### 4.2.1 Ping

Dati 2 utenti in 2 LAN diverse, il ping manda un pacchetto all'utente di destinazione e l'utente che lo ha mandato prima o poi riceverà un messaggio di risposta (**echo reply**). Il tempo che passa tra l'invio del pacchetto e la ricezione del messaggio di risposta è il ritardo end-to-end.

Non si può sapere se il ritardo è asimmetrico o no, cioè se il ritardo di andata è uguale al ritardo di ritorno, ma si può soltanto calcolare il ritardo totale.

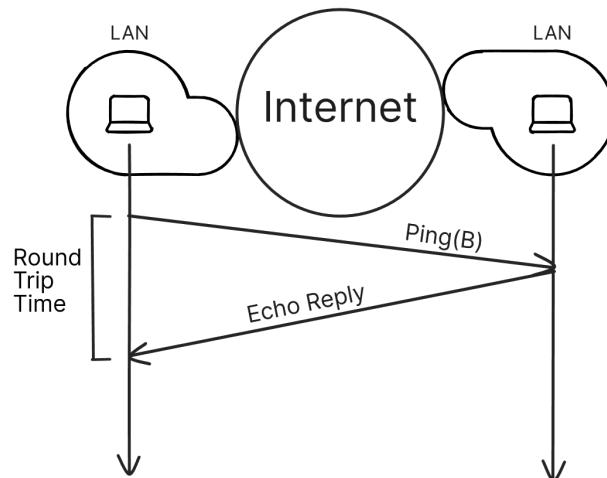


Figura 12: Ping

Se si esegue il comando `ping` da un terminale si riceve il seguente output:

```

ping www.google.com
PING www.google.com (142.251.209.4) 56(84) bytes of data.
64 bytes from mil04s50-in-f4.1e100.net (142.251.209.4): icmp_seq=1 ttl=115 time=15.2 ms
64 bytes from mil04s50-in-f4.1e100.net (142.251.209.4): icmp_seq=2 ttl=115 time=15.4 ms
64 bytes from mil04s50-in-f4.1e100.net (142.251.209.4): icmp_seq=3 ttl=115 time=15.4 ms
64 bytes from mil04s50-in-f4.1e100.net (142.251.209.4): icmp_seq=4 ttl=115 time=15.0 ms
^C
--- www.google.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3006ms
rtt min/avg/max/mdev = 14.964/15.221/15.393/0.171 ms

```

Figura 13: Output di ping

1. Viene rieseguito il ping facendo riferimento al **server fisico**
2. Tra parentesi viene rappresentato l'indirizzo che identifica il server, chiamato **indirizzo IP**
3. Alla fine c'è il tempo di risposta del server

#### 4.2.2 Traceroute

Vengono mandati 3 messaggi al primo router e si calcolano i tempi di risposta tra il primo utente e il primo router, poi viene fatta la stessa cosa con i seguenti router fino ad arrivare all'utente di destinazione.

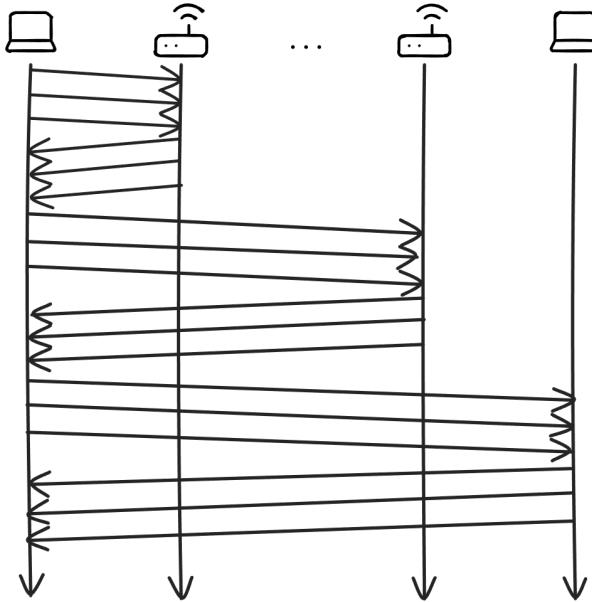


Figura 14: Traceroute

Se si esegue il comando **traceroute** da un terminale si riceve il seguente output:

```

traceroute -4 www.uv.es
traceroute to www.uv.es (147.156.200.249), 30 hops max, 60 byte packets
1 modemtim.homenet.telecomitalia.it (192.168.1.1) 7.589 ms 7.476 ms 9.427 ms
2 * * *
3 172.17.121.114 (172.17.121.114) 59.708 ms 172.17.121.116 (172.17.121.116) 59.672 ms 172.17.121.114 (172.17.121.114) 59.637 ms
4 172.17.128.136 (172.17.128.136) 59.600 ms 59.564 ms 172.17.120.134 (172.17.128.134) 59.526 ms
5 172.19.184.52 (172.19.184.52) 61.473 ms 172.19.184.50 (172.19.184.50) 62.829 ms 172.19.184.54 (172.19.184.54) 61.404 ms
6 172.19.177.62 (172.19.177.62) 61.346 ms 50.631 ms 50.538 ms
7 be200.milan26.mil.seabone.net (195.22.192.144) 56.724 ms 51.451 ms 195.22.205.116 (195.22.205.116) 51.375 ms
8 185.100.113.147 (185.100.113.147) 79.754 ms 185.100.113.145 (185.100.113.145) 81.267 ms 81.217 ms
9 nl-ams.nordu.net (88.249.209.203) 79.607 ms 67.327 ms 73.148 ms
10 213.46.175.38 (213.46.175.38) 79.312 ms 78.287 ms 79.858 ms
11 ndn-gw.mnl.lon.uk.geant.net (109.105.102.98) 78.147 ms 71.796 ms 77.514 ms
12 lag-1-0.rte.lon.uk.geant.net (62.40.98.60) 75.712 ms 78.651 ms 72.774 ms
13 lag-2-0.rte.lon2.uk.geant.net (62.40.98.65) 68.119 ms 68.000 ms 66.487 ms
14 lag-8-0.rte.par.fr.geant.net (62.40.98.167) 66.376 ms 66.318 ms 66.267 ms
15 ae4-0.rt1.bil.es.geant.net (62.40.98.222) 69.157 ms 71.086 ms 64.995 ms
16 rediris-las-geant.bil.es.geant.net (83.97.98.19) 46.398 ms 51.586 ms 53.018 ms
17 ehu-rt2.ethtrunk5.ciemat.rt2.madrid.es.rediris.es (130.206.245.5) 52.932 ms 52.877 ms 54.535 ms
18 clemat-rt2.ethtrunk2.uv.rt2.val.rediris.es (130.206.245.122) 62.924 ms 65.269 ms 65.222 ms
19 uv-pnat-router.rediris.es (130.206.211.262) 62.778 ms 58.059 ms 62.832 ms
20 quarburquartest.red.uv.es (147.156.200.150) 56.624 ms 55.719 ms 59.404 ms
21 www.uv.es (147.156.200.249) 52.868 ms 59.342 ms 52.681 ms

```

Figura 15: Output di traceroute

1. Indica il numero di router attraversati
2. Stampa i valori di ritardo dei 3 pacchetti trasmessi
3. È il nome logico del router che si sta attraversando, da questo nome si può dedurre la posizione geografica del router e l'ISP a cui appartiene
4. Gli asterischi indicano che il router è stato configurato in modo da ignorare i pacchetti e non mandare alcuna risposta, questo perchè serve soltanto a inoltrare i pacchetti ad un altro router.

### 4.3 Quantità di dati trasferiti

La quantità di informazioni che si riesce a trasmettere si misura in  $\frac{bit}{s}$  e questa informazione dipende dalla capacità di tutti i canali di trasmissione attraversati. Ogni canale avrà una dimensione diversa e si può determinare la banda totale a disposizione (**Throughput**) dal **collo di bottiglia**, cioè dalla minore capacità di trasmissione tra tutti i canali attraversati.

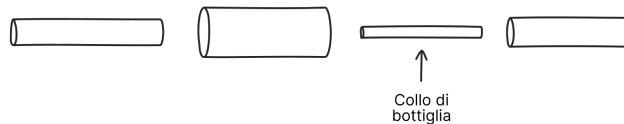
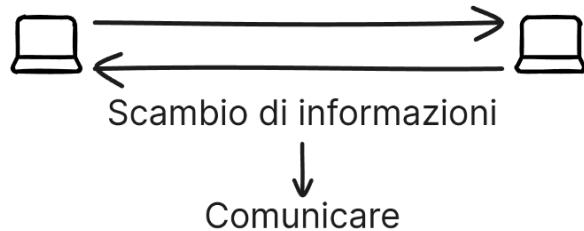


Figura 16: Throughput

## 5 Modello a strati

Il modello a strati affronta la **comunicazione tra due entità** secondo la modalità **divide et impera**.



Per scambiare informazioni le due entità devono comunicare. La comunicazione si divide in:

- **Comunicazione logica:** Gestisce le problematiche relative all'informazione

Ad esempio:

- Linguaggio utilizzato
- Come comportarsi nello scambio

- **Comunicazione Fisica:** Come trasferire i diversi bit. Il contenuto dell'informazione non è importante.

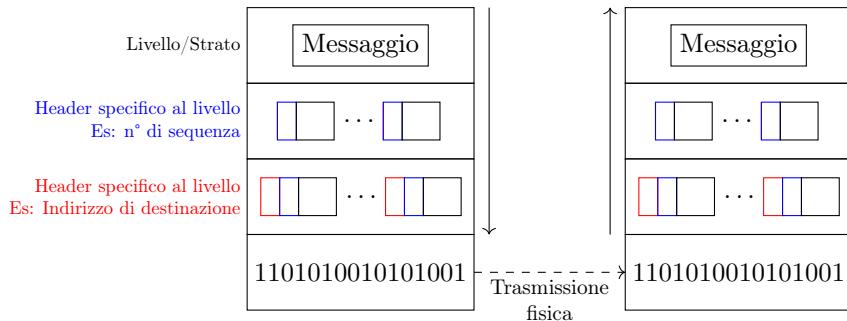


Figura 17: Comunicazione tra due entità

Ad ogni **livello** (o strato) viene elaborata parzialmente l'informazione e trasformata. Ogni livello aggiunge un **header** che contiene un'informazione specifica a quel livello, seguendo un **protocollo** (specifico per quel livello).

Ogni livello ha uno o più protocolli associati, e l'insieme dei protocolli di tutti i livelli è chiamato **stack protocollare**.

## 5.1 Stack ISO/OSI

Il modello **ISO/OSI** (International Standard Organization / Open System Interconnection) definisce dei livelli a seconda del sistema:

- **End system:**



Figura 18: Stack ISO/OSI per l'end system

- **Intermediate system:**

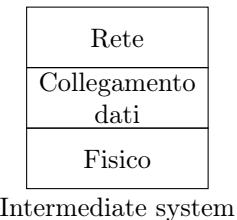


Figura 19: Stack ISO/OSI per l'intermediate system

## 5.2 Stack TCP/IP

Nella pratica (rete Internet) si utilizza lo stack protocollare **TCP/IP**:

- **End system:**



Figura 20: Stack TCP/IP per l'end system

- **Intermediate system:**

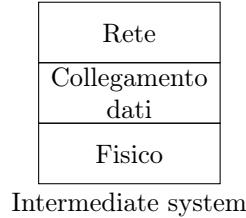


Figura 21: Stack TCP/IP per l'intermediate system

Il nome deriva dai due principali protocolli utilizzati:

- Protocollo di trasporto: **TCP** (Transport Control Protocol)
- Protocollo di rete: **IP** (Internet Protocol)

### 5.3 Entità coinvolte nella comunicazione

Su un calcolatore possono girare più applicazioni. Ogni applicazione può avere una connessione attiva, quindi ci possono essere più connessioni attive contemporaneamente.

Un applicazione può avere più istanze di comunicazione, cioè più connessioni attive contemporaneamente.

Di conseguenza **l'istanza di un'applicazione** è la vera e propria entità all'interno di una comunicazione. Quando si parla di entità si fa riferimento a uno specifico processo che gira su un calcolatore

#### 5.3.1 Identificazione dei processi

Per identificare un processo servono 2 informazioni:

1. **Indirizzo IP:** Identifica il calcolatore
2. **Porta:** Codice numerico che identifica il processo all'interno del calcolatore

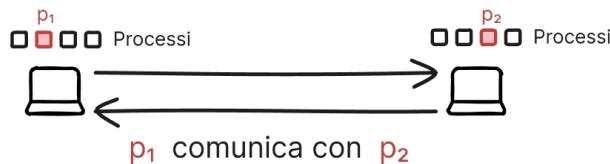


Figura 22: Comunicazione tra processi

Un flusso di comunicazione è identificato univocamente dalla tupla:

$$(IPA, IPB, Porta_A, Porta_B)$$

La porta viene assegnata ad un processo soltanto quando esso inizia a comunicare.

### 5.3.2 Ottenimento dell'IP e della porta

Queste informazioni sono contenute negli header aggiunti ad ogni livello.

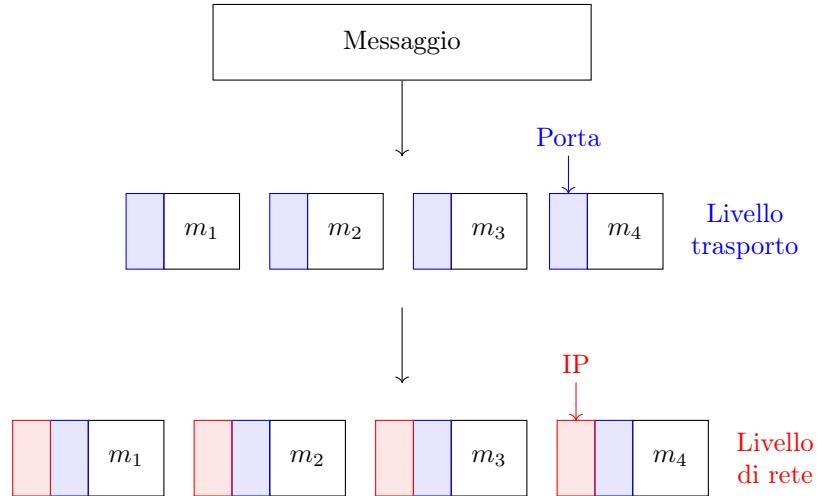


Figura 23: Porta e IP

Un pacchetto si può rappresentare nel seguente modo:

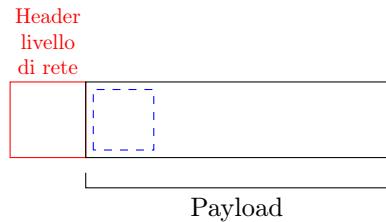
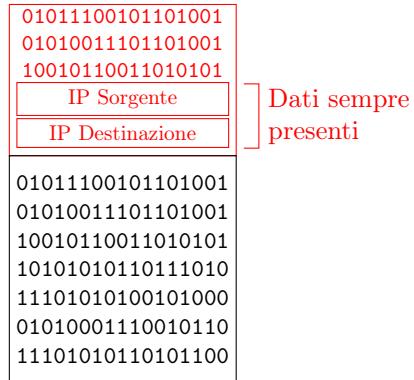


Figura 24: Rappresentazione di un pacchetto

oppure si può rappresentare anche come:



## 6 Indirizzi IP

Sono identificativi **univoci** di un’interfaccia di un host della rete Internet. Un end system può avere soltanto un’interfaccia, ma un router deve avere minimo 2 interfacce per poter permettere la comunicazione tra più entità.

**Esempio 6.1.** Un esempio di indirizzo IP è:

1001 1101 0001 1011 0001 0011 0111 1011

Per facilitare la lettura e la gestione degli indirizzi IP si usa la notazione **decimale puntata**. In questa notazione si considerano blocchi di 8 bit, di conseguenza si avranno 4 blocchi. Ogni blocco viene tradotto in un numero intero decimale compreso tra 0 e 255 e separato da un punto:

157.27.19.123

### 6.1 Suddivisione dei bit

I bit dell’indirizzo IP hanno tutti la stessa importanza?

Prendiamo ad esempio i numeri telefonici della rete fissa:

$\underbrace{0039 \ 045 \ 802}_{\text{Prefisso}} \ \underbrace{7059}_{\text{Suffisso}}$

- Il prefisso è l’identificativo di una specifica organizzazione
- Il suffisso identifica un utente specifico all’interno dell’organizzazione

Allo stesso modo in un indirizzo IP si ha **prefisso** e **suffisso**

- **Prefisso:** Identifica una rete specifica
- **Suffisso:** Identifica un’interfaccia di un host di una specifica rete

### 6.1.1 Identificazione del prefisso e del suffisso

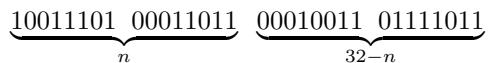
Nel caso dei numeri telefonici si inserisce una barra per separare il prefisso dal suffisso:

045/7021412

Negli indirizzi IP il numero di bit dedicati al prefisso dipende dalla dimensione della rete ed è indicato da una barra seguita da un numero.

157.27.19.123/16

Significa che i primi 16 bit dell'indirizzo IP sono dedicati al prefisso.



Quindi il numero di indirizzi si calcola come:

$$\text{indirizzi} = 2^{32-n}$$

#### Esempio 6.2.

$$n = 20 \rightarrow 2^{12} = 4096$$

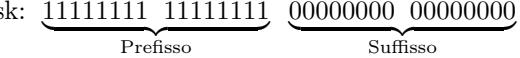
$$n = 24 \rightarrow 2^8 = 256$$

Per identificare il numero di bit del prefisso i calcolatori utilizzano una sequenza di 32 bit in cui i bit associati al prefisso sono posti a 1 e gli altri a 0, ad esempio:

/16 → 11111111 11111111 00000000 00000000

Questa sequenza è chiamata **maschera** perché è abbinata all'indirizzo IP.

#### Esempio 6.3.

IP: 10011101 00011011 00010011 01111011  
Mask: 

Facendo l'AND bit a bit tra l'IP e la maschera si ottiene l'indirizzo di rete a cui quell'indirizzo IP appartiene.

La maschera può essere rappresentata anche in:

- **Notazione decimale puntata:**

/16 → 255.255.0.0

- **Notazione esadecimale:** si divide l'IP in gruppi da 4 bit e si convertono in esadecimale:

/16 → 0xffff0000

Per capire che un numero è esadecimale, esso è prefissato da 0x.

## 6.2 Indirizzi IP riservati

Non tutti gli indirizzi IP possono essere assegnati a degli host, infatti ce ne sono alcuni riservati per delle funzioni specifiche:

1. **This host:** È l'indirizzo IP del calcolatore stesso

0.0.0.0

2. **Local broadcast** (o limited broadcast): Indirizzo IP che permette di inviare un messaggio a tutti i calcolatori della rete locale

255.255.255.255

3. **Indirizzo di rete:** I primi  $n$  bit identificano una rete, gli altri  $32 - n$  sono tutti posti a 0. Questo IP identifica la rete e nessun host può avere questo indirizzo

$\underbrace{1010100011}_{n} \underbrace{00000000000000000000}_{32-n} 0000000000000000$

4. **Directed broadcast:** È un IP che permette di mandare un messaggio a tutti gli utenti di una rete specifica, anche esterna alla rete locale

$\underbrace{1010100011}_{n} \underbrace{11111111111111111111}_{32-n} 1111$

## 6.3 Conoscere il proprio indirizzo IP

L'indirizzo ip e la dimensione del prefisso dipendono dalla rete a cui si è collegati. Per conoscere il proprio indirizzo ip si usa il comando: **ifconfig**

```
ifconfig
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
        inet6 ::1 prefixlen 128 scopeid 0x10<host>
            loop txqueuelen 1000 (Local Loopback)
            RX packets 39 bytes 5012 (4.8 KiB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 39 bytes 5012 (4.8 KiB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlp1s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    1 inet 157.27.146.164 netmask 255.255.224.0 broadcast 157.27.159.255
        inet6 2001:760:2204:222:d2:0:41:3408 prefixlen 128 scopeid 0x0<global>
        inet6 fe80::c1a6:d355:4044:f133 prefixlen 64 scopeid 0x20<link>
            ether b2:76:5a:44:c9:9e txqueuelen 1000 (Ethernet)
            RX packets 17886 bytes 2998716 (2.8 MiB)
            RX errors 0 dropped 18 overruns 0 frame 0
            TX packets 20083 bytes 35308084 (33.6 MiB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

1. **Interfaccia di rete:** Rappresenta il collegamento tra il calcolatore e la rete.

2. **Indirizzo ip**

### 3. Indirizzo broadcast

### 4. Maschera

Altre informazioni sulla rete si possono ottenere con il comando: **whois**. Questo comando si connette ad un server che contiene informazioni sugli indirizzi IP. L'output di questo comando da varie informazioni sull'IP messo come argomento, ma si può notare che la maschera è diversa da quella che viene data in output dal comando **ifconfig**.

- Per **ifconfig** il prefisso è da 20 bit, quindi  $2^{12} = 4096$  indirizzi
- per **whois** il prefisso è da 16 bit, quindi  $2^{16} = 65536$  indirizzi

Questo è dovuto al **subnetting**, cioè la creazione di sottoreti per partizionare l'organizzazione distribuita sul territorio, quindi dei 65546 indirizzi, alcuni verranno assegnati ad una sottorete e altri ad un'altra.

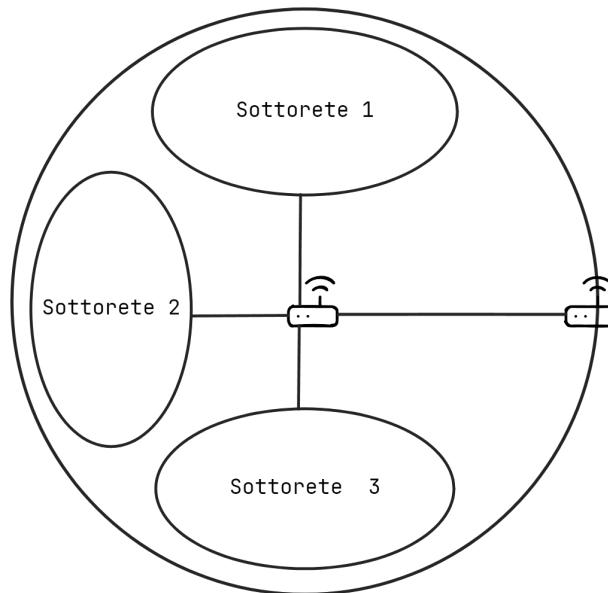


Figura 25: Subnetting

Questa suddivisione dall'esterno non viene vista e questo è il motivo per cui ipconfig mostra una maschera diversa da quella che si vede con il comando **whois**.

## 6.4 Esercizi

Conversione di IP decimale/binario

**Esercizio 6.1.** Dato il seguente IP:

1110 0111 1101 1011 1000 1011 0110 1111

la traduzione in decimale è la seguente:

231.219.139.111

**Esercizio 6.2.** Dato il seguente IP:

221.34.255.82

la traduzione in binario è la seguente:

1101 1101 0010 0010 1111 1111 0101 0010

## 6.5 Subnetting

È il processo che permette di suddividere una rete in sottoreti.

### 6.5.1 Creazione delle sottoreti partendo da un blocco di indirizzi assegnato

Un'organizzazione richiede un blocco di indirizzi in base alle proprie necessità. Prendendo come esempio l'Università di Verona, essa ha come indirizzo di rete il seguente:

157.27.0.0/16

si nota che i primi 16 bit sono dedicati al prefisso:

$\underbrace{10011101}_{\text{Prefisso}} \underbrace{00011011}_{\text{Suffisso}} \underbrace{00000000}_{\text{Prefisso}} \underbrace{00000000}_{\text{Suffisso}}$

Partendo da tale blocco si vogliono creare 2 sottoreti di pari dimensione. Per fare ciò si può prendere il primo bit del suffisso, separarlo dal suffisso e associarlo al prefisso.

$\underbrace{10011101}_{\text{Prefisso}} \underbrace{00011011}_{\text{Suffisso}} \underbrace{0}_{\text{Sottorete}} \underbrace{00000000}_{\text{Prefisso}} \underbrace{00000000}_{\text{Suffisso}}$

Il bit assegnato al prefisso può avere 2 valori, quindi si possono creare 2 sottoreti:

Sottorete 1:  $\underbrace{10011101}_{\text{Prefisso}} \underbrace{00011011}_{\text{Suffisso}} \underbrace{0}_{\text{Sottorete}} \underbrace{00000000}_{\text{Prefisso}} \underbrace{00000000}_{\text{Suffisso}}$   
Sottorete 2:  $\underbrace{10011101}_{\text{Prefisso}} \underbrace{00011011}_{\text{Suffisso}} \underbrace{1}_{\text{Sottorete}} \underbrace{00000000}_{\text{Prefisso}} \underbrace{00000000}_{\text{Suffisso}}$

In notazione decimale puntata diventa:

Sottorete 1: 157.27.0.0/17

Sottorete 2: 157.27.128.0/17

Quello che cambia è che il prefisso è diventato di 17 bit.

Da un blocco /16  $\rightarrow 2^{32-16} = 65536$  indirizzi si ottengono 2 blocchi /17  $\rightarrow 2^{32-17} = 2^{15} = 32768$  indirizzi ciascuno.

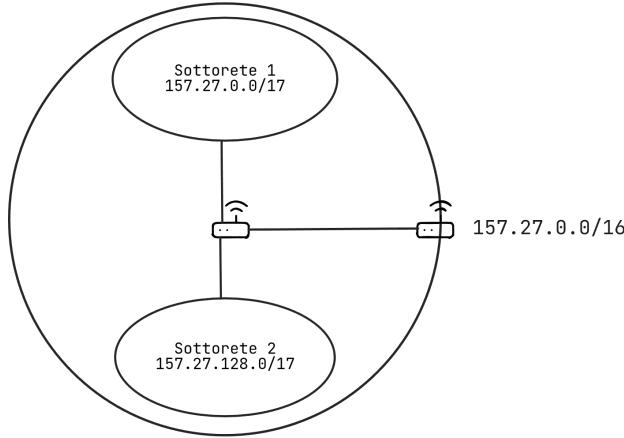


Figura 26: Risultato del subnetting

I blocchi di indirizzi si chiamano **CIDR** (Classless Inter-Domain Routing).

- **Nota storica:** In passato si usava la notazione **classful** in cui il numero di bit dedicati al prefisso era predeterminato e venivano usati i bit iniziali per distinguere i diversi casi. Ad esempio:

- **Classe A:** 8 bit di prefisso se l'IP iniziava con 0:

$$\begin{array}{ccccccccc} 0 & \underbrace{1010010} & \underbrace{00000000} & \underbrace{00000000} & \underbrace{00000000} \\ \text{Prefisso} & & & & & \text{Suffisso} \end{array}$$

- **Classe B:** 16 bit di prefisso se l'IP iniziava con 10:

$$\begin{array}{ccccccccc} 10 & \underbrace{010010} & \underbrace{10110100} & \underbrace{00000000} & \underbrace{00000000} \\ \text{Prefisso} & & & & & \text{Suffisso} \end{array}$$

- **Classe C:** 24 bit di prefisso se l'IP iniziava con 110:

$$\begin{array}{ccccccccc} 110 & \underbrace{010010} & \underbrace{10110100} & \underbrace{00000000} & \underbrace{00000000} \\ \text{Prefisso} & & & & & \text{Suffisso} \end{array}$$

L'informazione del prefisso era codificata nell'indirizzo stesso e quindi non c'era bisogno di aggiungere un'informazione all'IP come ad esempio la maschera.

Per l'alta richiesta di indirizzi, l'indirizzamento classful non era sufficiente:

Classe	N° reti	N° host
A	$2^7 \approx 16K$	$2^{24} \approx 16M$
B	$2^{14} \approx 16K$	$2^{16} \approx 65K$
C	$2^{21} \approx 2M$	$2^8 \approx 256$

Si esaurirono presto le classi A e B, quindi si passò al CIDR.

A volte assegnare delle sottoreti di uguale dimensione è uno spreco perché certe esigenze richiedono meno indirizzi e altre di più, quindi si dovrebbe suddividere la rete in sottoreti di dimensione **diversa**.

## 7 Livello applicativo

Il livello applicativo è il livello più alto dello stack protocollare:



Figura 27: Stack TCP/IP

Questo livello genera un messaggio che può essere di tipo diverso, ad esempio:

- Richiesta HTTP
- Risposta HTTP
- Email
- Il frame di un video

Quando un messaggio viene generato da un'applicazione si deve decidere quale protocollo utilizzare a livello di trasporto:

- **TCP** (Transmission Control Protocol): È un protocollo orientato alla connessione (**Connection oriented**) e affidabile. Questo protocollo dà la garanzia che il messaggio arrivi a destinazione.
- **UDP** (User Datagram Protocol): È un protocollo non orientato alla connessione (**Connectionless**) e non affidabile. Questo protocollo non dà la garanzia che il messaggio arrivi a destinazione e quindi non è detto che il messaggio arrivi.

Un servizio di trasporto **affidabile** garantisce che la consegna arrivi a destinazione.

## 7.1 Esempi di applicazioni e relativi protocolli di trasporto

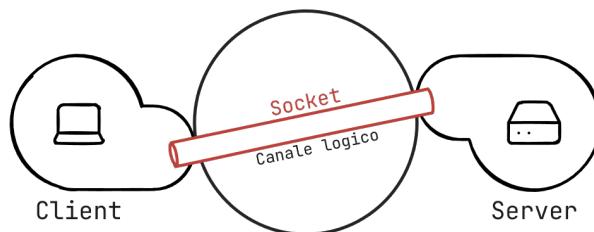
- **Web:** Un esempio di applicazione è il browser web che utilizza il protocollo HTTP per trasferire dati attraverso la rete. Quando si consegnano testo e immagini non si devono commettere errori, quindi in questi casi il protocollo HTTP si appoggia a TCP.
- **Posta elettronica:** Un’altro esempio è la posta elettronica che utilizza il protocollo SMTP per inviare la posta. Questo protocollo richiede un servizio affidabile, quindi se un pacchetto viene perso, il protocollo TCP si preoccupa di recuperarlo.
- **Audio/video:** Per la trasmissione di audio e video di solito i protocolli sono proprietari, ma in questo caso non è necessaria l'affidabilità, basta garantire la tolleranza alle perdite, quindi ci si appoggia al protocollo UDP.
- **DNS:** Per navigare su internet si usa il protocollo **DNS** (Domain Name System) che è il servizio che traduce i nomi logici nei corrispettivi indirizzi ip.

## 7.2 Socket

Quando un'applicazione deve mandare un messaggio ad un'altra applicazione essa apre un **socket** verso la destinazione. Un socket è un'interfaccia di comunicazione (astrazione software) che identifica un flusso informativo bidirezionale. Per aprire un socket bisogna specificare:

- Protocollo di trasporto
- Indirizzo IP di destinazione
- Indirizzo IP sorgente
- Porta di destinazione
- Porta di sorgente

L'apertura di un socket è effettuata soltanto da uno dei due processi.



Tra i due processi comunicanti si possono identificare due ruoli distinti:

1. **Processo Client:** È il processo che è responsabile dell'inizio della comunicazione. Il client ha un indirizzo IP dinamico, cioè cambia in base alla rete a cui è collegato.

2. **Processo Server:** È il processo che sta su un host sempre raggiungibile e sempre in ascolto. Il server ha un indirizzo IP statico.

## 7.3 Protocolli

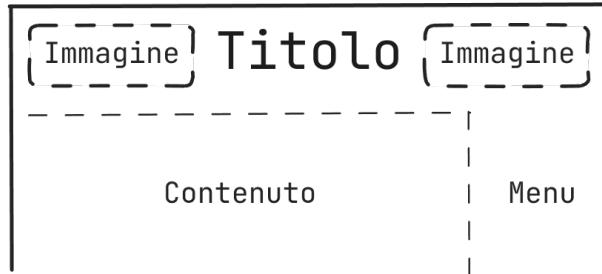
### 7.3.1 Protocollo HTTP

Il protocollo **HTTP** (HyperText Transfer Protocol) è un protocollo di livello applicativo usato per la trasmissione di documenti ipertestuali. Questo protocollo è di tipo **testuale** (i messaggi sono scritti in ASCII), determina il formato dei messaggi ed è basato sul modello **richiesta/risposta**:

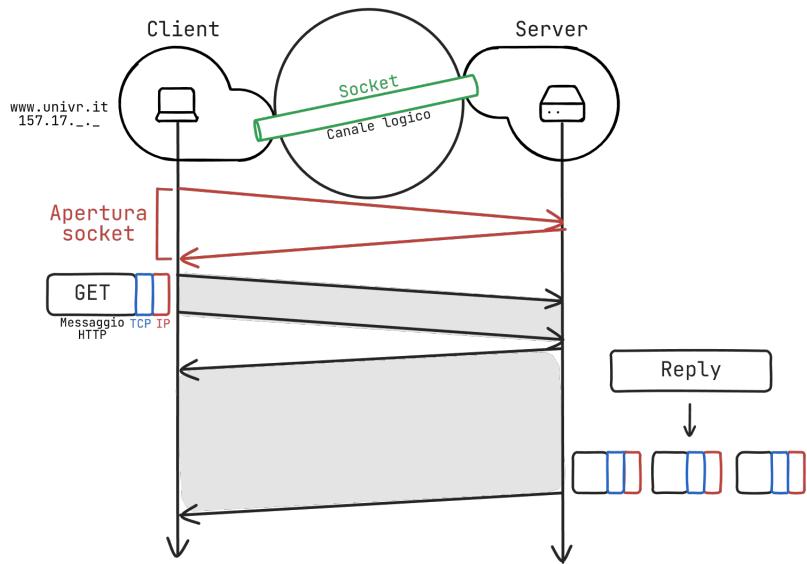
- **Richiesta:** Il client apre un socket, invia un messaggio di richiesta al server e aspetta la risposta.
- **Risposta:** Il server deve inviare un messaggio di risposta alla richiesta del client

Se la richiesta non fosse ben formata il server risponde con un messaggio di errore. Per cui il protocollo prevede sempre, ad ogni richiesta, che ci sia una risposta, positiva o negativa.

Il server contiene delle pagine web, con eventualmente altri contenuti ad esempio immagini. Il file di testo principale (`index.html`).



Un file `html` definisce sia la struttura che il contenuto di una pagina web.



La struttura delle richieste HTTP è la seguente:

1. Riga di richiesta
2. Una o più righe di intestazione

**Esempio 7.1.** Un'esempio di richiesta HTTP è:

```

1   GET /index.html HTTP/1.1 // -- Riga di richiesta
2   Host: www.univr.it      // -- Riga di intestazione
3   User-Agent: Mozilla/5.0 // (intestazione facoltativa)
4   Accept-language: en     // (intestazione facoltativo)
5

```

che corrisponde a una richiesta di una pagina web del seguente tipo:

www.univr.it / index.html  
 Server File richiesto

Per fare una richiesta si utilizzano dei **metodi**:

- **GET**: Richiede una rappresentazione di una risorsa specifica
- **POST**: Invia informazioni al server
- **HEAD**: Richiede una risposta identica a quella di una richiesta **GET** ma senza il corpo del messaggio, quindi solo con le intestazioni
- **PUT**: Carica una rappresentazione di una risorsa specifica
- **DELETE**: Cancella la risorsa specificata

La struttura delle risposte HTTP è la seguente:

1. Riga di stato:
  - Versione del protocollo
  - Codice di stato
  - Messaggio di stato
2. Una o più righe di intestazione
3. Il corpo del messaggio

**Esempio 7.2.** Un'esempio di risposta HTTP è:

```
1  HTTP/1.1 200 OK      // -- Riga di stato
2  Date: Mon, 23 May 2005 // --
3  Content-Type: text/html // | Righe di intestazione
4  Content-Length: 122    // --
5  // Riga vuota obbligatoria~~
6  <html>...</html>       // -- Corpo del messaggio
7
```

Per inviare e ricevere messaggi HTTP si può usare il comando `nc` o `netcat` e scrivere una richiesta HTTP come mostrato nell'esempio 7.1.

Altri elementi del protocollo HTTP che possono estendere il funzionamento sono i **cookies**. I cookies sono un meccanismo utilizzato dai server web per capire se è già avvenuta un'interazione con un determinato client in passato. L'IP non basta perché è dinamico.

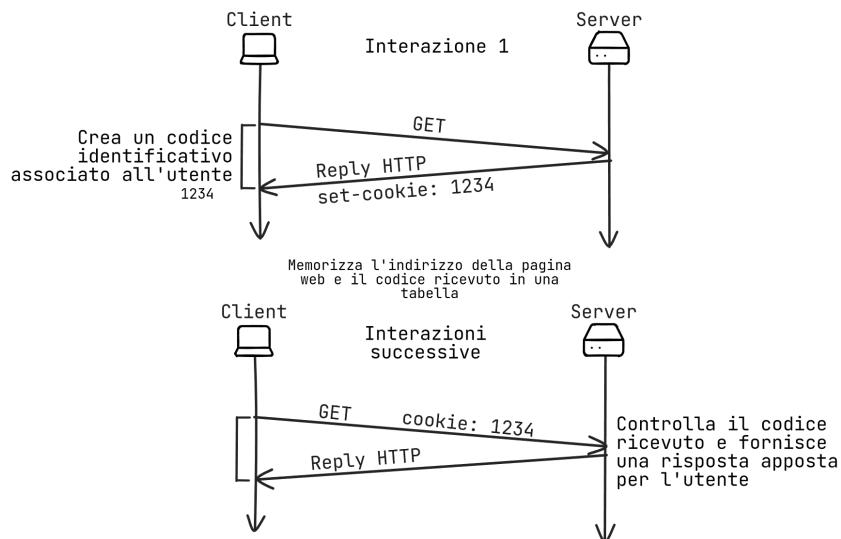


Figura 28: Esempio di cookies

Per diminuire il ritardo di accesso alle risorse si usa un meccanismo di **caching** che offre diversi vantaggi:

- Si riduce il ritardo di accesso
- Viene diminuito il carico sui server

Senza cache il ritardo che sente il secondo client è uguale al ritardo che sente il primo client e il carico è doppio. Con la cache il ritardo che sente il secondo client è minore e il carico è dimezzato perchè la cache contiene già i dati che il secondo client vuole siccome sono stati caricati dal primo client.

Una buona cache lavora con un 70% – 80% di **hit rate**.

Se il contenuto sul server di origine cambia, la cache potrebbe contenere dati obsoleti. Per risolvere questo problema, il protocollo HTTP prevede il metodo **GET condizionale** in cui una riga di intestazione della richiesta è:

```
1   ...
2   If-Modified-Since: <data>
```

Nella risposta del server (la prima con il contenuto, o le successive con "not modified") viene indicato anche il periodo di validità del contenuto: