Università degli Studi di Trieste

Corso di Laurea Magistrale in
INGEGNERIA CLINICA

UNIVERSITÀ
DEGLI STUDI
DI TRIESTE

Dipartimento di
Ingegneria e Architettura

# Reti di calcolatori: Livello Sessione, Presentazione e Applicazione

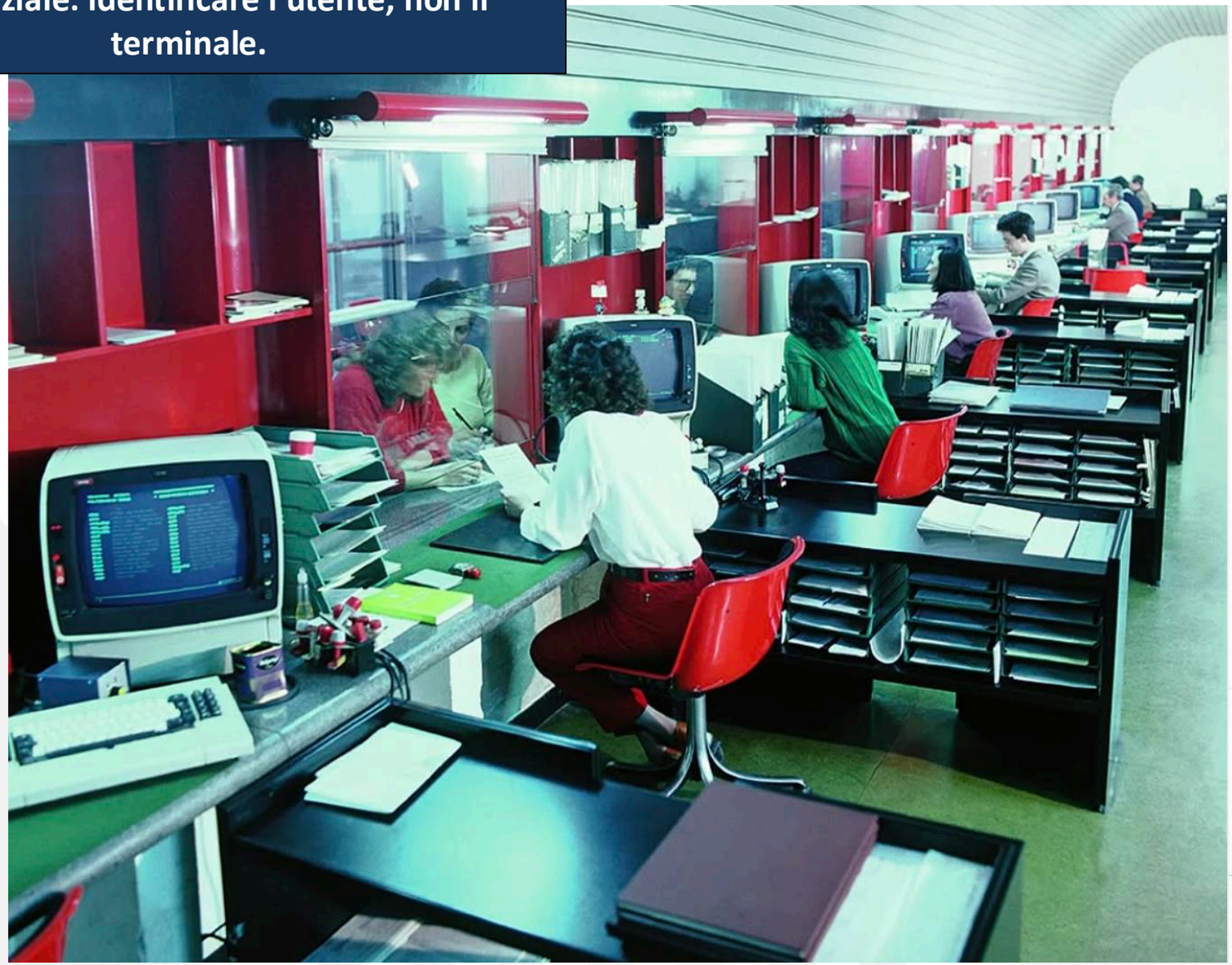
## Corso di Informatica Medica

**Docente: Aleksandar Miladinović**

*Questa presentazione è stata realizzata, in parte o interamente, basandosi sulle slide fornite dal Prof. Francesco Brun.*

# Livello 5 — Sessione (L5)

**Idea iniziale: identificare l’utente, non il terminale.**



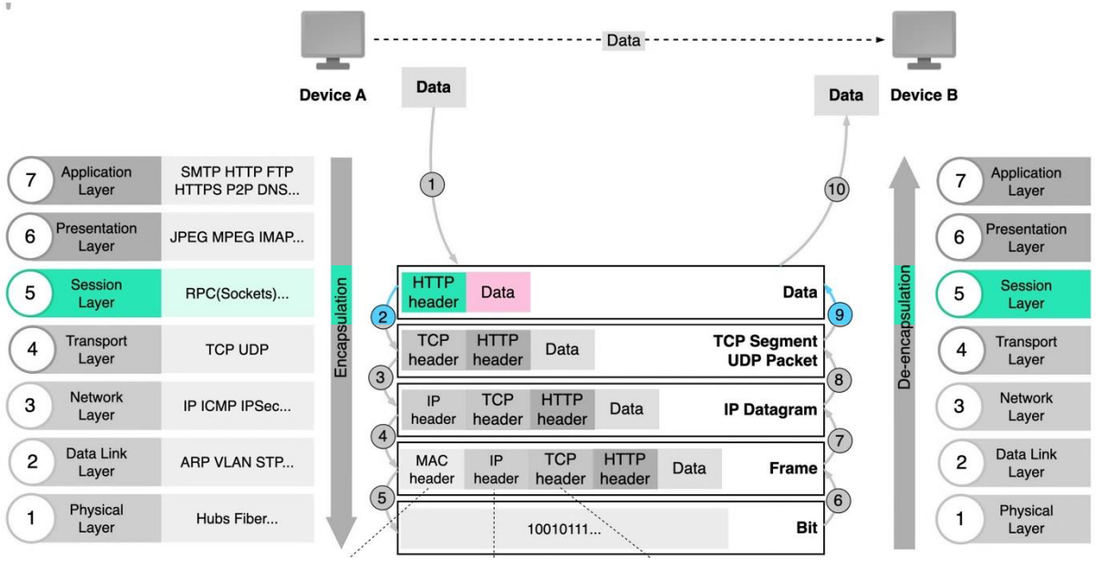
Politecnico di Torino, https://40.csi.it/portfolio-items/informatica-innova-pa/

# L5 - Obiettivi



- Capire cos’è una sessione e perché serve.
- Collegare il modello OSI al mondo TCP/IP (dove L5 è “assorbito” dall’app).
- Vedere esempi pratici: Telnet, cookie e sessioni web.
- Conoscere limiti, rischi e buone pratiche.

# L5 e OSI ISO




- OSI: L1…L4 (rete, trasporto) si occupano di **consegna** e **integrità** dei dati.
- L5 **Sessione**: gestione del **dialogo logico** tra due applicazioni.
- TCP/IP: L5–L7 sono spesso **fusi** dentro i protocolli applicativi e nelle logiche dell’app.

# Cos’è una sessione



- Un **contesto logico** che identifica un utente/processo durante un dialogo prolungato.
- Tiene traccia di **chi sei** (es. user ID), **stato** e **permessi**.
- Non dipende dalla posizione fisica (terminale 3 o 4): ti identifica come **utente**.

# Perché serve una sessione



- Le reti sottostanti sono **best effort**: i pacchetti possono perdersi/riordinarsi.
- Le applicazioni hanno bisogno di **continuità** (login, carrello, preferenze).
- La sessione collega **più scambi** in un’unica conversazione coerente.

# Cosa fa (funzioni tipiche)

UNIVERSITÀ DEGLI STUDI DI TRIESTE

- **Autenticazione** (sei davvero tu?) e **autorizzazione** (cosa puoi fare).
- **Dialog control** (chi parla quando; turn-taking nelle app che lo richiedono).
- **Sincronizzazione** e **checkpoint** (riprendere dopo interruzioni).
- **Rilevamento/recupero** da errori a livello applicativo.

# Sessioni sul Web (HTTP è stateless)

UNIVERSITÀ DEGLI STUDI DI TRIESTE

- HTTP **non** tiene stato: ogni richiesta è indipendente.
- Il **server** crea lo stato di sessione con un **identificatore** associato all'utente.
- Due strategie pratiche:
- **Cookie** con **Session ID** (server-side state).
- Token (es. portatori di stato lato client) → *accenno, non dettagliare*.

# Cookie e Session ID

- **Cookie**: piccola stringa salvata dal browser che riporta un **ID univoco**.
- Al ritorno sul sito, il browser invia il cookie → il server riconosce **la stessa sessione**.
- Effetti pratici: persistenza del **login**, del **carrello**, **tracking** prezzi/visite.

- Layer 5 distinguishes between **user sessions**
  - Identifies a user independent from L2, L3, or L4 addresses

| | |
|---|---|
| 7 | Application |
| 6 | Presentation |
| 5 | **Session** |
| 4 | Transport |
| 3 | Network |
| 2 | Data Link |
| 1 | Physical |



**HTTP** — **HTTP Cookies**: Arbitrary text strings that store user-specific information

# Sessione ≠ protocollo unico



- Non esiste un **protocollo universale** di L5: le **app** decidono **come** gestire la sessione.
- Nel web: cookie/session-ID; in ambienti remoti moderni: **SSH** (autenticazione, canali, keepalive).
- Nei sistemi multiutente storici: **user ID** e permessi a livello OS.

# Sicurezza (cenni)

UNIVERSITÀ DEGLI STUDI DI TRIESTE

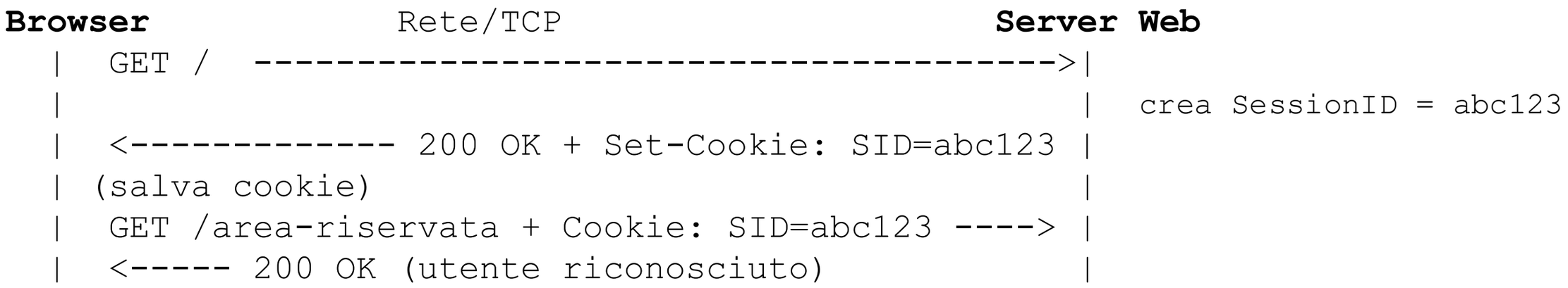
- Rischi: **furto di cookie**, **session hijacking**, **fixation**.
- Mitigazioni: **HTTPS** (cifratura L7), cookie **HttpOnly/Secure**, scadenze brevi, rigenerazione ID.
- La sessione **identifica** l'utente, ma la **cifratura** non è compito di L5 (avviene a L7).

# Limiti e realtà TCP/IP



- Nella pratica Internet, L5 è **logica applicativa**.
- Pro e contro: **flessibilità** (ogni app fa la sua sessione) vs **eterogeneità** (niente standard unico).
- Impatto didattico: capiamo **il concetto** a L5, vediamo **implementazioni** concrete a L7.

# Mini diagramma — Sessione web con cookie



# Cosa ricordare



- La sessione è un **contenitore di stato** per conversazioni applicative.
- Sul Web si realizza con **cookie/session-ID** perché HTTP è **stateless**.
- In TCP/IP reale, le logiche di L5 sono **dentro l'app** (L7).
- Attenzione a **sicurezza** e **privacy** quando gestisci le sessioni.

# Glossario essenziale



- **Sessione**: contesto che lega più scambi dello stesso utente/processo.
- **Session ID**: identificatore univoco della sessione (spesso via cookie).
- **Dialog control**: regole su “chi parla quando” in un dialogo strutturato.
- **Checkpoint**: punto di ripresa per evitare di ricominciare da zero dopo un’interruzione.
- **Stateless vs stateful**: protocollo senza stato (HTTP) vs con stato (logica di sessione dell’app).

# Livello 6 — Presentazione (L6)

# Livello 6 – Presentazione



- Capire **a cosa serve** il livello di presentazione.
- Collegare **codifica, serializzazione, compressione, cifratura** al percorso dei dati.
- Vedere **errori tipici** (mismatch di codifica) e come evitarli.

# Dove si colloca

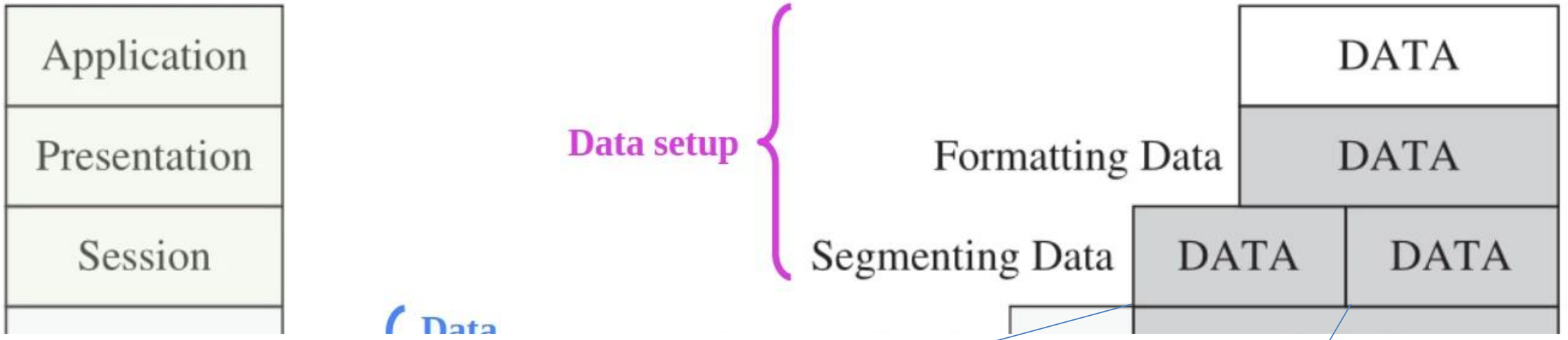


OSI: tra **Sessione (L5)** e **Applicazione (L7)**.
Scopo: **rappresentare** i dati in modo comprensibile alle applicazioni su entrambe le parti.
Nella pratica TCP/IP moderno: molte funzioni di L6 sono **incorporate in L7**.

# Che cosa fa



**Codifica dei caratteri** (es. ASCII, UTF-8).
**Trasformazione/serializzazione** di strutture: oggetti → byte (JSON, XML, binario).
**Compressione** per ridurre dimensioni (gzip/deflate).
**Cifratura**/firma per riservatezza e integrità (spesso gestite da protocolli applicativi o TLS).




```
01000111010001001010100001000000010111011100110110100101101101000
```

# Livello 6 – Presentazione



```
01000111010001001010100001000000010111101110011011010010110110100
```

Base 64

```
010001 110100 010101 010100 001000 000010 111101 110011 011010 010110 110100
```

# Livello 6 – Presentazione



```
010001110100010010101000010000000010111101110011011010010110110100
```

6 bit (Base 64)

```
010001 110100 010101 010100 001000 000010 111101 110011 011010 010110 110100
```

4bit (esadecimale)

```
0100 0111 0100 0101 0101 0100 0010 0000 0010 1111 0111 0011 0110 1001 0110 1101
```

# Livello 6 – Presentazione

UNIVERSITÀ DEGLI STUDI DI TRIESTE

```
010001110100010010101000010000000010111101110011011010010110110100
```

6 bit (Base 64)

```
010001 110100 010101 010100 001000 000010 111101 110011 011010 010110 110100
```

4bit (esadecimale)

```
0100 0111 0100 0101 0101 0100 0010 0000 0010 1111 0111 0011 0110 1001 0110 1101
```

32 bit

```
01000111010001010101010000100000  00101111011100110110100101101101
```

# Livello 6 – Presentazione

UNIVERSITÀ DEGLI STUDI DI TRIESTE

```
01000111010001001010100001000000010111101110011011010010110110100
```

6 bit (Base 64)

```
010001 110100 010101 010100 001000 000010 111101 110011 011010 010110 110100
```

4bit (esadecimale)

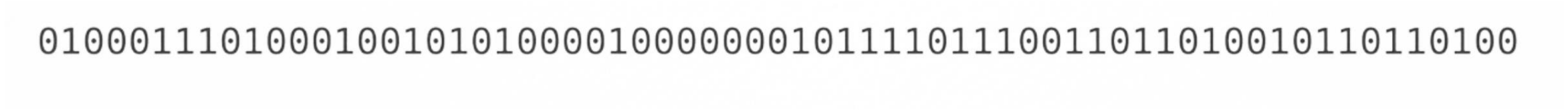
```
0100 0111 0100 0101 0101 0100 0010 0000 0010 1111 0111 0011 0110 1001 0110 1101
```

32 bit (int)

1,195,725,856 796,092,781

```
01000111010001010101010000100000  00101111011100110110100101101101
```

# Livello 6 – Presentazione




6 bit (Base 64)

010001 110100 010101 010100 001000 000010 111101 110011 011010 010110 110100

4bit (esadecimale)

0100 0111 0100 0101 0101 0100 0010 0000 0010 1111 0111 0011 0110 1001 0110 1101

32 bit (int)

1,195,725,856 796,092,781

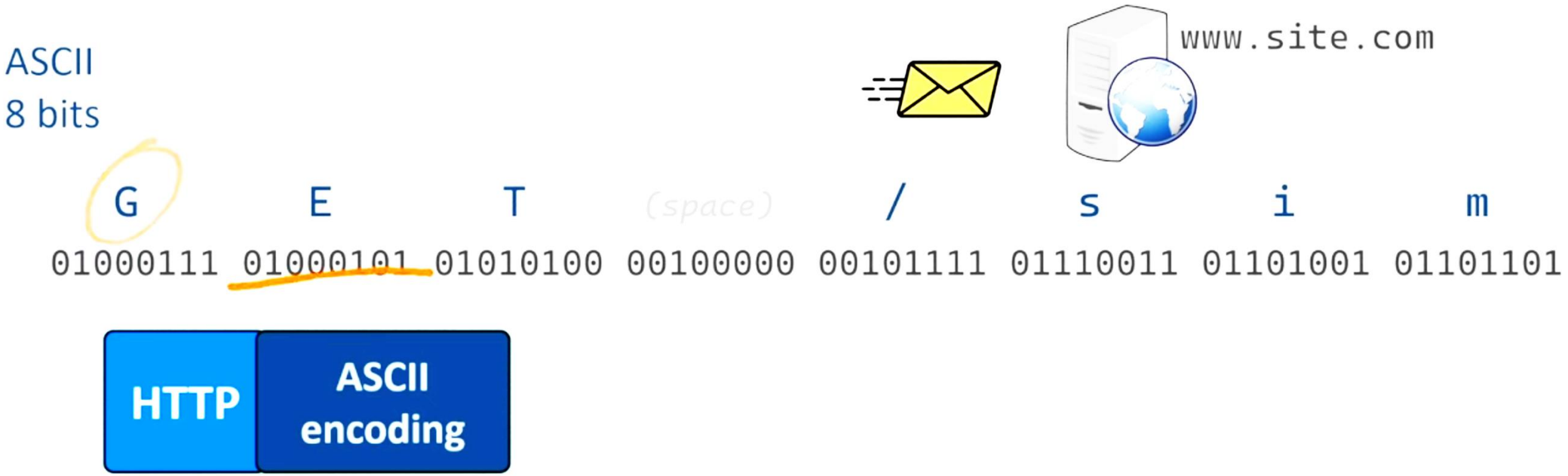
01000111010001010101010000100000 00101111011100110110100101101101

5,135,603,447,297,698,157

64bit (int)

0100011101000101010101000010000000010111011100110110100101101101

# Livello 6 – Presentazione



# Codifica dei caratteri



**ASCII**: 7/8 bit, base storica; “A” = 65 (0x41).
**UTF-8**: estende a tutti i caratteri Unicode; lunghezza variabile (1–4 byte).
**ISO-8859-1/Latin-1**: codifiche storiche europee.
Se mittente e destinatario **non concordano** la codifica, si ottiene testo illeggibile.

| dec | hex | oct | char | dec | hex | oct | char | dec | hex | oct | char | dec | hex | oct | char |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 000 | NULL | 32 | 20 | 040 | space | 64 | 40 | 100 | @ | 96 | 60 | 140 | ` |
| 1 | 1 | 001 | SOH | 33 | 21 | 041 | ! | 65 | 41 | 101 | A | 97 | 61 | 141 | a |
| 2 | 2 | 002 | STX | 34 | 22 | 042 | " | 66 | 42 | 102 | B | 98 | 62 | 142 | b |
| 3 | 3 | 003 | ETX | 35 | 23 | 043 | # | 67 | 43 | 103 | C | 99 | 63 | 143 | c |
| 4 | 4 | 004 | EOT | 36 | 24 | 044 | $ | 68 | 44 | 104 | D | 100 | 64 | 144 | d |
| 5 | 5 | 005 | ENQ | 37 | 25 | 045 | % | 69 | 45 | 105 | E | 101 | 65 | 145 | e |
| 6 | 6 | 006 | ACK | 38 | 26 | 046 | & | 70 | 46 | 106 | F | 102 | 66 | 146 | f |
| 7 | 7 | 007 | BEL | 39 | 27 | 047 | ' | 71 | 47 | 107 | G | 103 | 67 | 147 | g |
| 8 | 8 | 010 | BS | 40 | 28 | 050 | ( | 72 | 48 | 110 | H | 104 | 68 | 150 | h |
| 9 | 9 | 011 | TAB | 41 | 29 | 051 | ) | 73 | 49 | 111 | I | 105 | 69 | 151 | i |
| 10 | a | 012 | LF | 42 | 2a | 052 | * | 74 | 4a | 112 | J | 106 | 6a | 152 | j |
| 11 | b | 013 | VT | 43 | 2b | 053 | + | 75 | 4b | 113 | K | 107 | 6b | 153 | k |
| 12 | c | 014 | FF | 44 | 2c | 054 | , | 76 | 4c | 114 | L | 108 | 6c | 154 | l |
| 13 | d | 015 | CR | 45 | 2d | 055 | - | 77 | 4d | 115 | M | 109 | 6d | 155 | m |
| 14 | e | 016 | SO | 46 | 2e | 056 | . | 78 | 4e | 116 | N | 110 | 6e | 156 | n |
| 15 | f | 017 | SI | 47 | 2f | 057 | / | 79 | 4f | 117 | O | 111 | 6f | 157 | o |
| 16 | 10 | 020 | DLE | 48 | 30 | 060 | 0 | 80 | 50 | 120 | P | 112 | 70 | 160 | p |
| 17 | 11 | 021 | DC1 | 49 | 31 | 061 | 1 | 81 | 51 | 121 | Q | 113 | 71 | 161 | q |
| 18 | 12 | 022 | DC2 | 50 | 32 | 062 | 2 | 82 | 52 | 122 | R | 114 | 72 | 162 | r |
| 19 | 13 | 023 | DC3 | 51 | 33 | 063 | 3 | 83 | 53 | 123 | S | 115 | 73 | 163 | s |
| 20 | 14 | 024 | DC4 | 52 | 34 | 064 | 4 | 84 | 54 | 124 | T | 116 | 74 | 164 | t |
| 21 | 15 | 025 | NAK | 53 | 35 | 065 | 5 | 85 | 55 | 125 | U | 117 | 75 | 165 | u |
| 22 | 16 | 026 | SYN | 54 | 36 | 066 | 6 | 86 | 56 | 126 | V | 118 | 76 | 166 | v |
| 23 | 17 | 027 | ETB | 55 | 37 | 067 | 7 | 87 | 57 | 127 | W | 119 | 77 | 167 | w |
| 24 | 18 | 030 | CAN | 56 | 38 | 070 | 8 | 88 | 58 | 130 | X | 120 | 78 | 170 | x |
| 25 | 19 | 031 | EM | 57 | 39 | 071 | 9 | 89 | 59 | 131 | Y | 121 | 79 | 171 | y |
| 26 | 1a | 032 | SUB | 58 | 3a | 072 | : | 90 | 5a | 132 | Z | 122 | 7a | 172 | z |
| 27 | 1b | 033 | ESC | 59 | 3b | 073 | ; | 91 | 5b | 133 | [ | 123 | 7b | 173 | { |
| 28 | 1c | 034 | FS | 60 | 3c | 074 | < | 92 | 5c | 134 | \ | 124 | 7c | 174 | \| |
| 29 | 1d | 035 | GS | 61 | 3d | 075 | = | 93 | 5d | 135 | ] | 125 | 7d | 175 | } |
| 30 | 1e | 036 | RS | 62 | 3e | 076 | > | 94 | 5e | 136 | ^ | 126 | 7e | 176 | ~ |
| 31 | 1f | 037 | US | 63 | 3f | 077 | ? | 95 | 5f | 137 | _ | 127 | 7f | 177 | DEL |

# Esempio rapido (ASCII/UTF-8)



```
Testo: "GET /"
Byte (hex): 47 45 54 20 2F
"G"=0x47, "E"=0x45, "T"=0x54, " "=0x20, "/"=0x2F
```

Con UTF-8 caratteri accentati occupano più byte:

```
"è" → 0xC3 0xA8
```

# Mismatch di codifica (errore tipico)



Mittente invia **UTF-8**, destinatario interpreta **Latin-1** → compaiono **simboli strani** (mojibake).

Casi reali: log, vecchi sistemi, interfacce tra apparati medicali diversi.

Prevenzione: **dichiarare** la codifica (es. header Content-Type: text/html; charset=UTF-8).

# Serializzazione dei dati



- **Testo**: JSON, XML, CSV — leggibili, interoperabili, più pesanti.
- **Binario**: Protobuf, ASN.1, MessagePack — compatti, richiedono schema/decoder.
- Scelta pratica: dipende da **interoperabilità**, **prestazioni**, **vincoli legacy**.

# Numeri e formati binari



- **Endianness**: ordine dei byte per interi (big vs little endian).
- **Float**: IEEE-754 (attenzione alla precisione).
- È necessario **definire dimensioni** (int16/int32) e rappresentazione, non solo “numero”.

# Compressione



- Riduce traffico e tempi su link lenti: es. **gzip/deflate**.
- Effetti collaterali: CPU per comprimere/decomprimere, possibili ritardi su stream piccoli.
- Nel web: negoziazione via header (es. Content-Encoding, Accept-Encoding).

# Cifratura (collocazione pratica)

UNIVERSITÀ DEGLI STUDI DI TRIESTE

- Obiettivo: **riservatezza** + **integrità** + **autenticità**.
- Oggi si usa **TLS** (HTTPS) tra L7 e L4: concettualmente è “funzione di presentazione”, **implementata** come strato a parte.
- Accorgimenti: versioni TLS aggiornate, suite robuste, gestione certificati.

# Come le app “negoziano” la presentazione

UNIVERSITÀ DEGLI STUDI DI TRIESTE

- Il **protocollo applicativo** decide: header, campi, meta-info.
- Esempio HTTP:

```
Content-Type (tipo e charset)
Accept / Accept-Language / Accept-Encoding
```

- Morale: L7 **specifica le regole**, L6 **realizza** codifica/compressione/cifratura.

# Mini-pipeline



**[App Dati]**
↓ (serializzazione: JSON/XML/binario)
**[Presentazione: codifica UTF-8, compressione gzip, cifratura/TLS]**
↓
**[Trasporto/Rete/Link: TCP/IP/Ethernet]**
↓
**[Presentazione: decifratura, decompressione, decodifica]**
↓
**[App Dati]**

# Errori che abbiamo visto e soluzioni



- Testo illeggibile → **charset dichiarato** e coerente end-to-end.
- Numeri errati → specificare **endianness** e **dimensione**.
- File “pesanti” → valutare **compressione** mirata.
- Dati sensibili → **HTTPS/TLS** obbligatorio, gestione sicura dei certificati.

# Cosa ricordare



- La presentazione rende i **byte significativi** per l'applicazione.
- Codifica/serializzazione corretta = **interoperabilità**.
- Compressione/cifratura migliorano **efficienza** e **sicurezza**.
- Nel mondo reale, queste scelte si vedono e si impostano **a L7.**

# Glossario essenziale



- **Codifica**: mappare caratteri in byte (UTF-8, ASCII…).
- **Serializzazione**: trasformare strutture in byte (JSON/XML/binario).
- **Endianness**: ordine dei byte per rappresentare numeri.
- **Compressione**: ridurre dimensione dei dati (gzip).
- **Cifratura/TLS**: proteggere confidenzialità e integrità in transito.

# Livello 7 — Applicazione (L7)

# Obiettivi

UNIVERSITÀ DEGLI STUDI DI TRIESTE

- Capire cosa fa il **livello applicazione**.
- Collegare **URL/URI, DNS, HTTP/HTTPS** e modello **client–server**.
- Vedere lo **scambio richiesta/risposta**, **header**, **codici**, **cookie**.
- Mettere in relazione con **L5 (sessione)** e **L6 (presentazione)**.

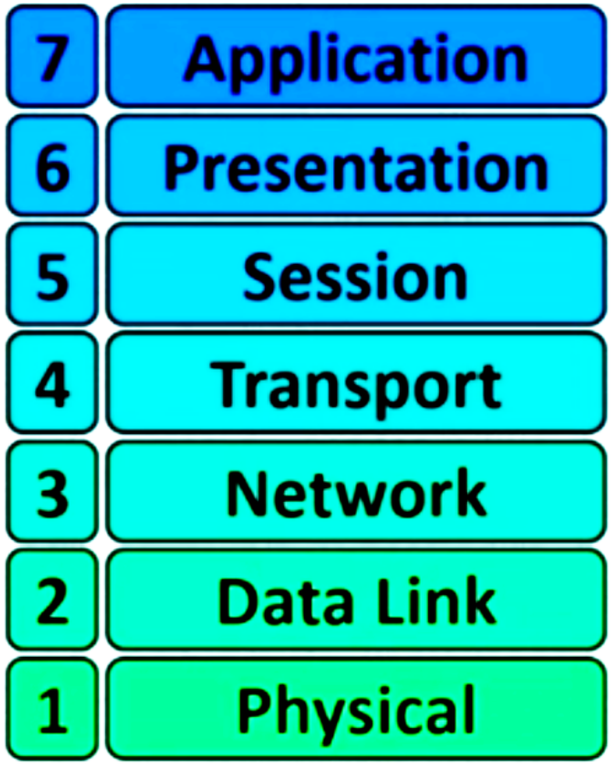
# Livello 7 – Applicazione



- E’ il livello dei servizi finali all’utente
- Qui sono definiti i protocolli che agiscono da interfaccia "uomo/calcolatore"

- Ad esempio:
  - ✓ terminale virtuale
  - ✓ trasferimento file
  - ✓ posta elettronica
  - ✓ web

# Livello 7 – Applicazione

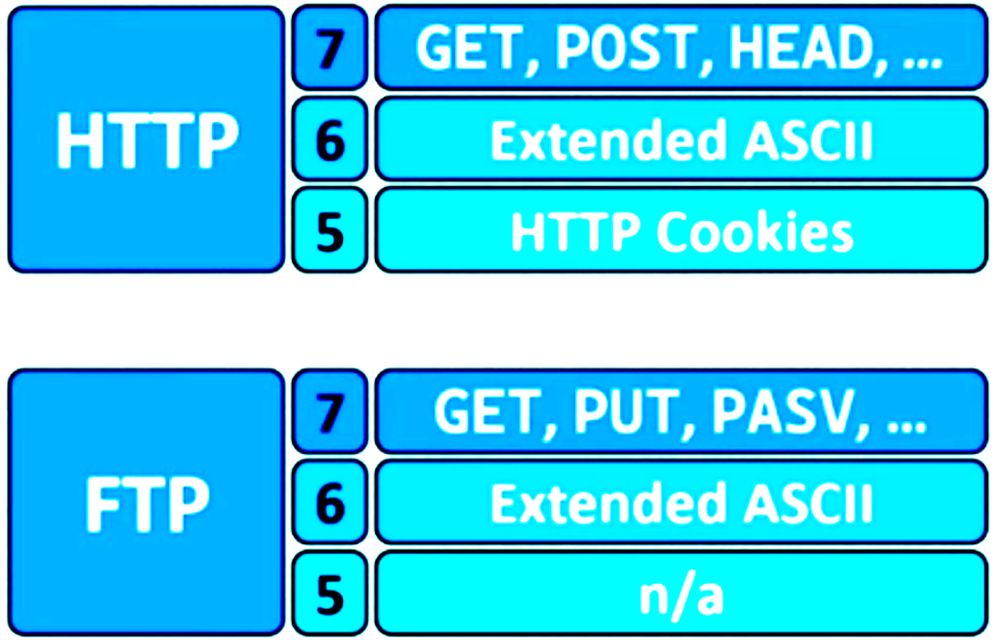



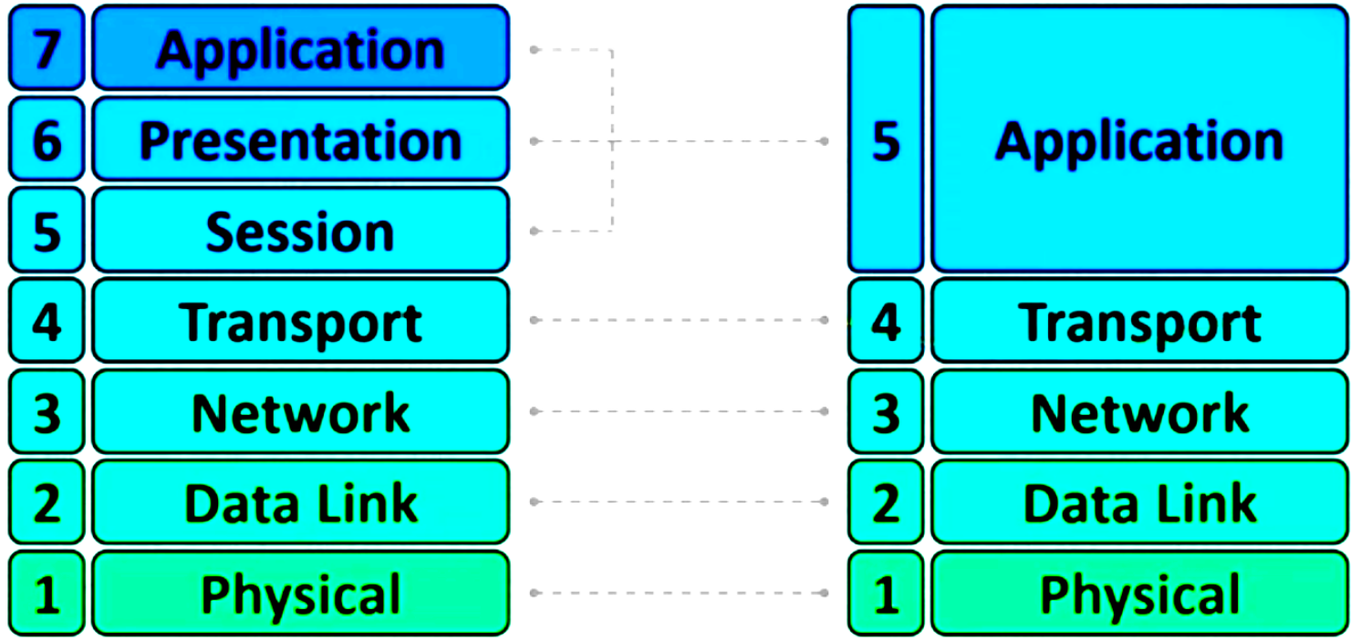

OSI Model

# Livello 7 – Applicazione

# Livello 7 – Applicazione






**OSI Model**
Seven Layer

**TCP/IP**
Five Layer

# Collocazione nel modello



- OSI: L7 è dove vivono i **protocolli applicativi** (HTTP, DNS, SMTP, …).
- TCP/IP: spesso si parla genericamente di “**application layer**”.
- Le scelte di **stato**, **formati**, **sicurezza** sono decise qui.

# Cos’è un protocollo applicativo



- Insieme di **regole e messaggi** che definiscono come un’app parla con un’altra.
- Esempi: **HTTP/HTTPS** (web), **DNS** (nomi ↔ IP), **SMTP/IMAP** (mail), **NTP** (ora), **SSH** (remoto).

# Modello client–server



- **Server**: fornisce un servizio (web, mail, DNS).
- **Client**: consuma il servizio (browser, app).
- Mappatura su porte note (esempi): HTTP 80, HTTPS 443, DNS 53, SMTP 25.

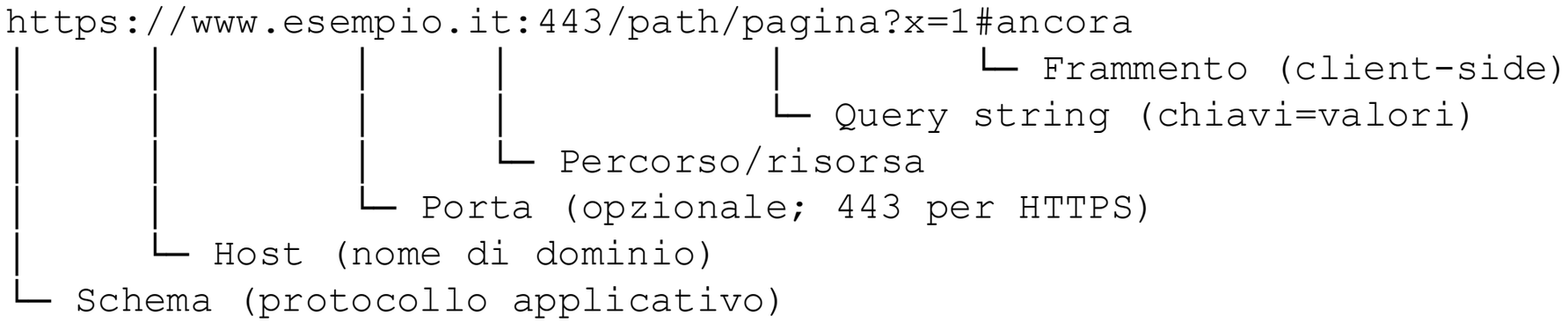
# Il modello client/server

- I servizi Internet si fondano sull'idea di un fornitore e un consumatore
  - ✓ Il fornitore del servizio è il servente (**server**)
  - ✓ Il consumatore del servizio è il cliente (**client**)

- Appositamente distribuiti in Internet ci sono svariati server
- Grosse organizzazioni hanno una **server farm** (opportunamente protetta)
- Si tratta di (insiemi di) calcolatori programmati per rispondere a richieste via rete

- I termini server e client possono indicare:
  - ✓ Sia l'hardware del calcolatore destinato a offrire/ricevere il servizio
  - ✓ Il software in esecuzione su quel calcolatore
- Il server è quindi un concetto di **livello 7 ISO/OSI**

# URL/URI (struttura)



# Porte e servizi (ripasso mirato)



- Le **porte** di L4 identificano l'**applicazione** su un host (HTTP 80, HTTPS 443, …).
- Una connessione tipica: **client porta effimera → server porta nota**.
- Più app sullo stesso host = **porte diverse**.

# HTTP in 1 slide

UNIVERSITÀ DEGLI STUDI DI TRIESTE

- **Stateless**: ogni richiesta è indipendente (stato gestito dall'app).
- **Metodi**: GET (leggi), POST (crea), PUT/PATCH (aggiorna), DELETE (cancella), HEAD/OPTIONS.
- **Risposte** con **codici di stato** (2xx OK, 3xx redirect, 4xx client error, 5xx server error).

```
Richiesta:
GET / HTTP/1.1
Host: www.esempio.it
Accept: text/html
Accept-Encoding: gzip
Cookie: SID=abc123

Risposta:
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8
Content-Encoding: gzip
Set-Cookie: SID=abc123; HttpOnly; Secure
```

**Header fondamentali**
**Host** (obbligatorio in HTTP/1.1), **Content-Type**, **Content-Length**.
**Accept / Accept-Language / Accept-Encoding** (negoziazione).
**Cache-Control / ETag / Last-Modified** (caching).
**Cookie / Set-Cookie** (identità di sessione lato app).

# Codici di stato (rapido)



- **200** OK, **201** Created, **204** No Content
- **301/302** Redirect, **304** Not Modified
- **400** Bad Request, **401/403** Auth/Forbidden, **404** Not Found
- **500** Server Error, **503** Service Unavailable

# HTTPS (TLS sotto il cofano)



- **Cifratura** del canale con **TLS** (fra L7 e L4).
- Autenticazione del **server** con certificato.
- Evita lettura/manomissione (sniffing/MITM).

# Sessione web (collegamento a L5)

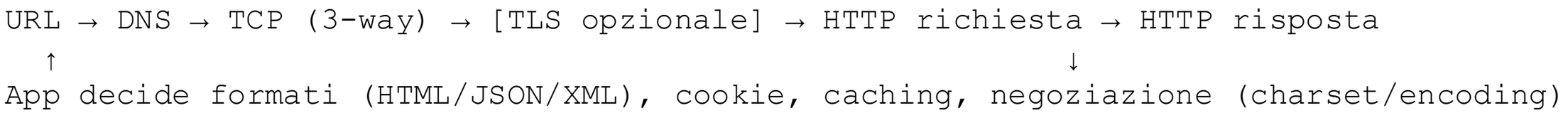


HTTP è senza stato → le app usano **cookie** con **Session ID**.

Il server associa **SID** → **stato utente** (login, carrello, preferenze).

Rischi: furto cookie, hijacking → mitigazioni con **Secure/HttpOnly**, scadenze, rigenerazione.

# Pipeline completa (riassunto L7→L1)




Browser legge **URL**.
Risolve **DNS**.
Apre **TCP** verso porta del server (80/443).
(Se HTTPS) negozia **TLS**.
Invia **GET** con header.
Riceve **200 OK** con **Content-Type/Encoding**, eventuali **Set-Cookie**.
Rendering o parsing del contenuto.

# Altri protocolli applicativi (cenni)



- **SMTP/IMAP/POP3** (posta), **FTP/SFTP** (file), **SSH** (remoto), **NTP** (ora).
- Ogni protocollo definisce i **messaggi** e il **flusso** per il suo dominio.

# Cenno storico (utile a capire)



- **Telnet**: sessione testuale “manuale” (non sicura): utile per **didattica**.
- **BBS**: precursori del web, scambio testuale, codifiche **ASCII** → ponte con L6.

# Telnet



- **Cos’è Telnet**
  - Protocollo di rete usato per l’**accesso remoto** a dispositivi o server.
  - Permette di aprire una **sessione interattiva** in modalità testuale su un host remoto.
  - Utilizza **TCP porta 23**.

# Telnet

UNIVERSITÀ DEGLI STUDI DI TRIESTE

- **Cos'è Telnet**
  - Protocollo di rete usato per l'**accesso remoto** a dispositivi o server.
  - Permette di aprire una **sessione interattiva** in modalità testuale su un host remoto.
  - Utilizza **TCP porta 23**

```
telnet 132.163.96.1 13 (telnet time.nist.gov 13)
```

1. Connette al **server dell'ora ufficiale (NIST)**.
2. Porta **13** → servizio **Daytime Protocol (RFC 867)**.
3. Il server invia **data e ora correnti** in formato testo e chiude la conessione

```
60977 25-10-29 12:39:28 05 0 0 520.1 UTC(NIST) *
Connection closed by foreign host.
```

```
60977 25-10-29 12:39:28 05 0 0 520.1 UTC(NIST) *
Connection closed by foreign host.
```

UNIVERSITÀ DEGLI STUDI DI TRIESTE

| Campo | Esempio | Significato |
|---|---|---|
| 60977 | | MJD (Modified Julian Date) – conteggio continuo dei giorni dal 17 novembre 1858. |
| 25-10-29 | | Data nel formato AA-MM-GG (anno-mese-giorno). |
| 12:39:28 | | Ora UTC nel formato hh:mm:ss. |
| 05 | | Indicatore ora legale (DST): 00 = non attiva, 50 = attiva, 99 = sconosciuta. |
| 0 | | Indicatore di secondo intercalare (leap second): 0 = nessuno, 1 = +1 secondo, 2 = −1 secondo. |
| 0 | | Codice di stato: 0 = tempo valido, valori >0 = errore o problema. |
| 751.1 | | Scarto temporale in millisecondi rispetto all'orologio di riferimento NIST. |
| UTC(NIST) | | Indica che il tempo è fornito dal servizio UTC del NIST. |
| * | | Stato sincronizzazione: * = sincronizzato, # = non sincronizzato. |

# NTP



- **Cos’è Telnet**
  - Protocollo di rete usato per l’**accesso remoto** a dispositivi o server.
  - Permette di aprire una **sessione interattiva** in modalità testuale su un host remoto.
  - Utilizza **TCP porta 23**.

# Telnet (bbs.retrocampus.com)

UNIVERSITÀ DEGLI STUDI DI TRIESTE

```
(base) aleksandarmiladinovic@MacBook-Air-M1 ~ % telnet 178.79.152.19
Trying 178.79.152.19...
Connected to li271-19.members.linode.com.
Escape character is '^]'.

RETROCAMPUS BBS
---------------

SUPPORTED SYSTEMS:

1- CBM PETSCII  W/ECHO   40X25 (6510)
2- APPLE-1/II   NOECHO   40X24 (6502)
3- APPLE-1/II   W/ECHO   40X24 (6503)
4- PURE ASCII   W/ECHO   80X24 (8000)
T- PURE ASCII   W/ECHO   72X24
5- PURE ASCII   NOECHO   80X24 (8001)
6- DOS CP 437   W/ECHO   80X24 (8088)
7- DOS CP 437   NOECHO   80X24 (8089)
8- TELNET LINUX W/ECHO   80X24 (8086)
9- VIC-20 ASCII W/ECHO   22X23 (6561)
0- OLIVETTI M10 W/ECHO   40X15 (8085)
M- MINITEL      W/ECHO   40X24 (1651)
P- PRESTEL      W/ECHO   40X24 (6499)

PLEASE SELECT A SYSTEM
PRESS ENTER TO CLOSE  CONNECTION

>
```

# BBS



Un BBS (o Bulletin Board System) è un sistema telematico che consente a computer remoti di accedere a un elaboratore centrale per condividere o prelevare risorse.

Il sistema è stato sviluppato negli anni 1970 e ha costituito il fulcro delle prime comunicazioni telematiche amatoriali, dando vita alla telematica di base.

Tra le novità consentite dai sistemi BBS, le principali furono la messaggistica e il file sharing centralizzato

# World wide web

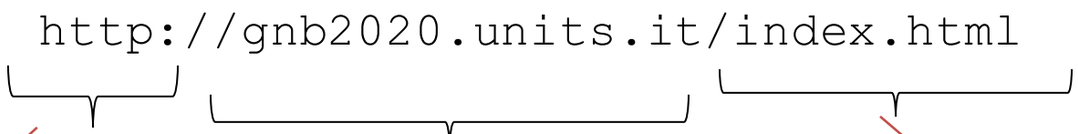
- Il servizio Internet più utilizzato oggi è il **world wide web** (**web** o **www**)
- Si tratta di testo e immagini (**ipertesti**) messi a disposizione in forma di "pagine"
- Le pagine sono collegate tra loro mediante **link** (creando la "ragnatela")

- Si fonda su diverse "tecnologie":
  - ✓ Il protocollo **HTTP** (*Hyper Text Transfer Protocol*)
  - ✓ Un linguaggio per le pagine: **HTML** (*Hyper Text Markup Language*)
  - ✓ Il concetto di **URL** (*Uniform Resource Locator*) per reperire le pagine

- Un apposito software detto **browser** consente la "navigazione" tra pagine
- La navigazione di norma inizia mediante digitazione di un URL

# URL

- Ad esempio, l'URL:




**Protocollo di livello applicazione**: di norma HTTP, HTTPS o FTP. Se non si specifica nulla il browser aggiunge http://

**Nome web server:** può essere l'indirizzo TCP (indirizzo IP:porta) del web server su cui risiede la risorsa ma anche un alias come units.it

**Pathname della risorsa:** è il percorso del file (in questo caso il file HTML index.html) all'interno del file system del web server (in questo caso nella directory «radice»).

- Nel mondo "giornalistico" si parla di visitare "l'indirizzo" web

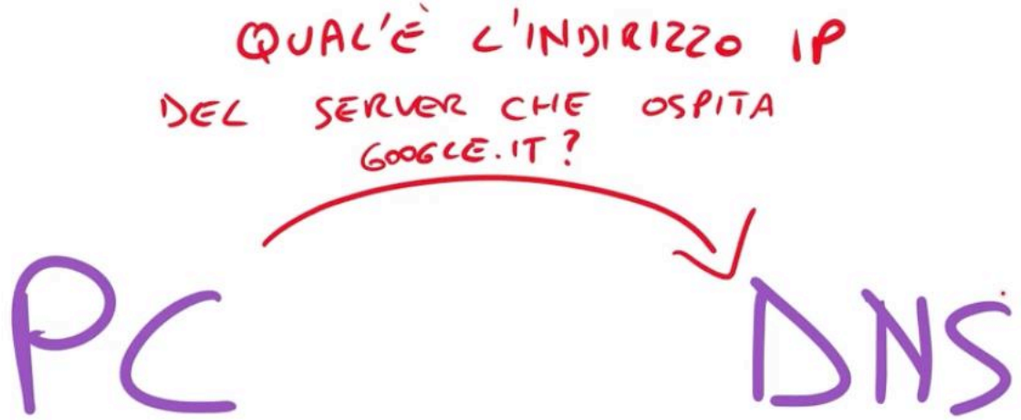
# DNS (*Domain Name System*)

UNIVERSITÀ DEGLI STUDI DI TRIESTE

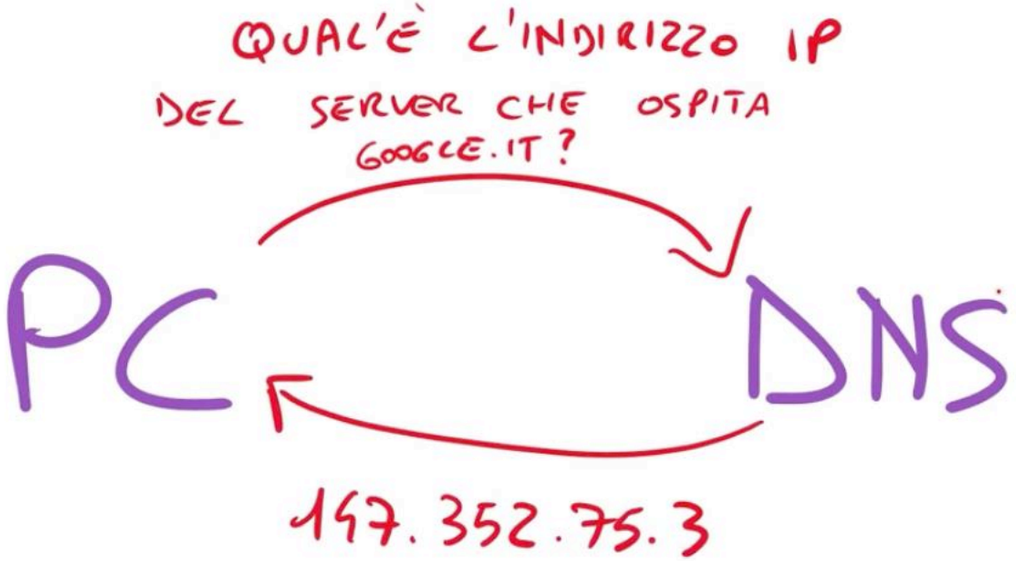
- E’ il sistema che **risolve nomi mnemonici in indirizzi IP** e viceversa
- (Il web senza DNS sarebbe come un cellulare senza rubrica…)
- E’ realizzato tramite un database distribuito costituito dai server DNS

- Un’organizzazione compra un blocco di indirizzi IP pubblici
- Quasi sempre compra anche un **dominio** ovvero un nome mnemonico

- Un dominio (es units.it) è composto da
  - ✓ nome
  - ✓ un'estensione legata (in teoria) alla nazione dell’organizzazione
- L’insieme di nome.estensione dev’essere univoco al mondo
- Se ne occupa un’autorità coordinata a livello mondiale

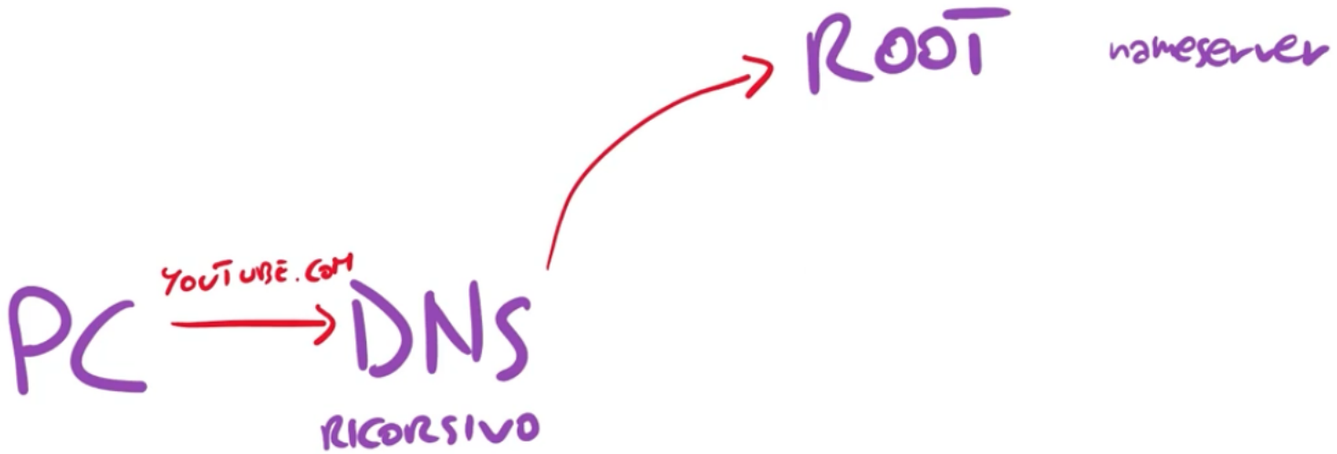



**Rappresentazione grafica tratta dal video:** https://youtu.be/p7PbB9MnKko?si=_vSjnfTdug3CIjKs




**Rappresentazione grafica tratta dal video:** https://youtu.be/p7PbB9MnKko?si=_vSjnfTdug3CIjKs

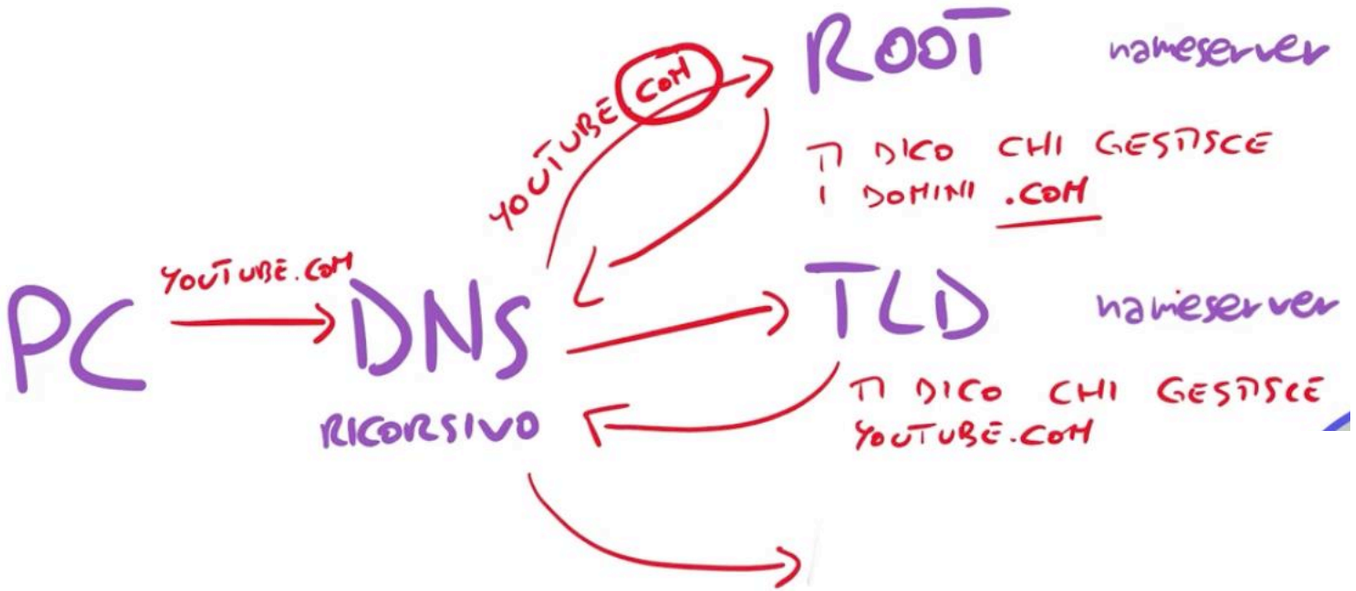

**Rappresentazione grafica tratta dal video:** https://youtu.be/p7PbB9MnKko?si=_vSjnfTdug3CIjKs



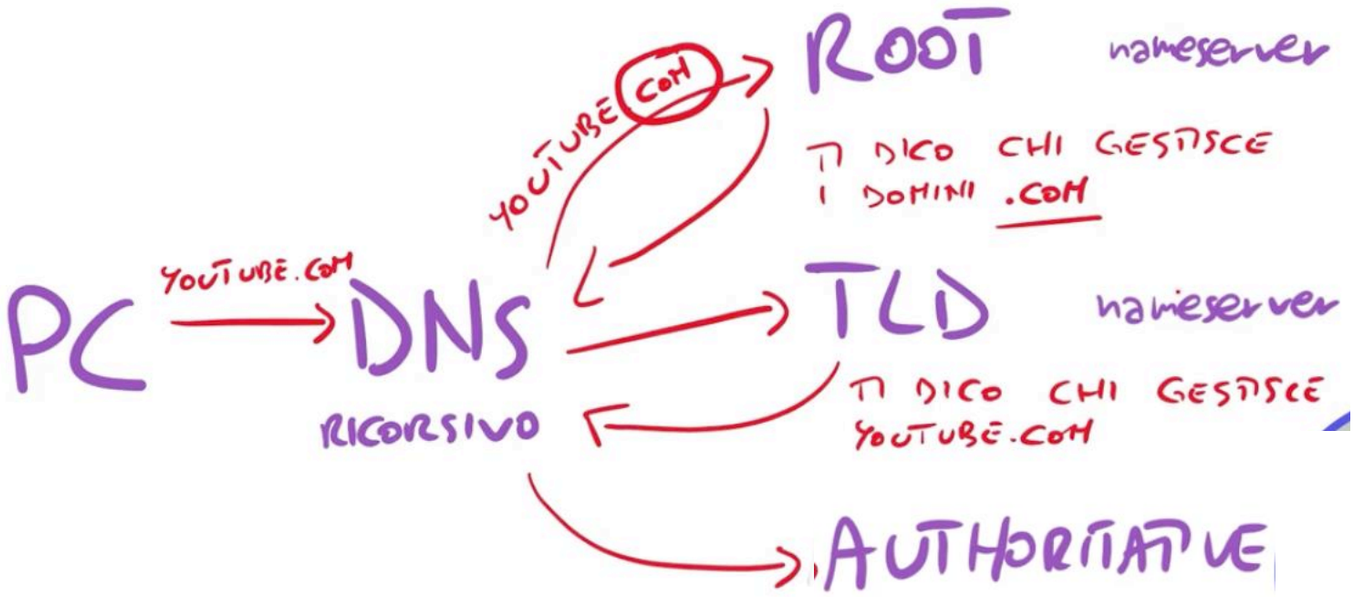

**Rappresentazione grafica tratta dal video:** https://youtu.be/p7PbB9MnKko?si=_vSjnfTdug3CIjKs


**Rappresentazione grafica tratta dal video:** https://youtu.be/p7PbB9MnKko?si=_vSjnfTdug3CIjKs

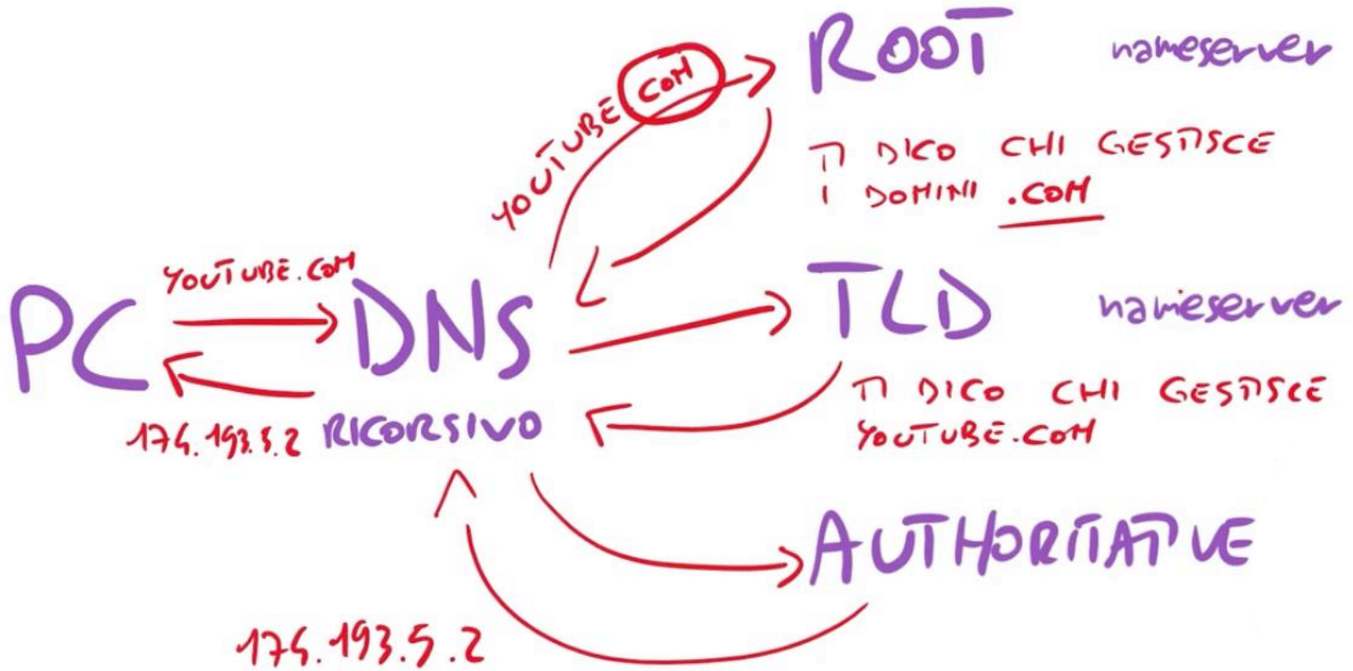

**Rappresentazione grafica tratta dal video:** https://youtu.be/p7PbB9MnKko?si=_vSjnfTdug3CIjKs



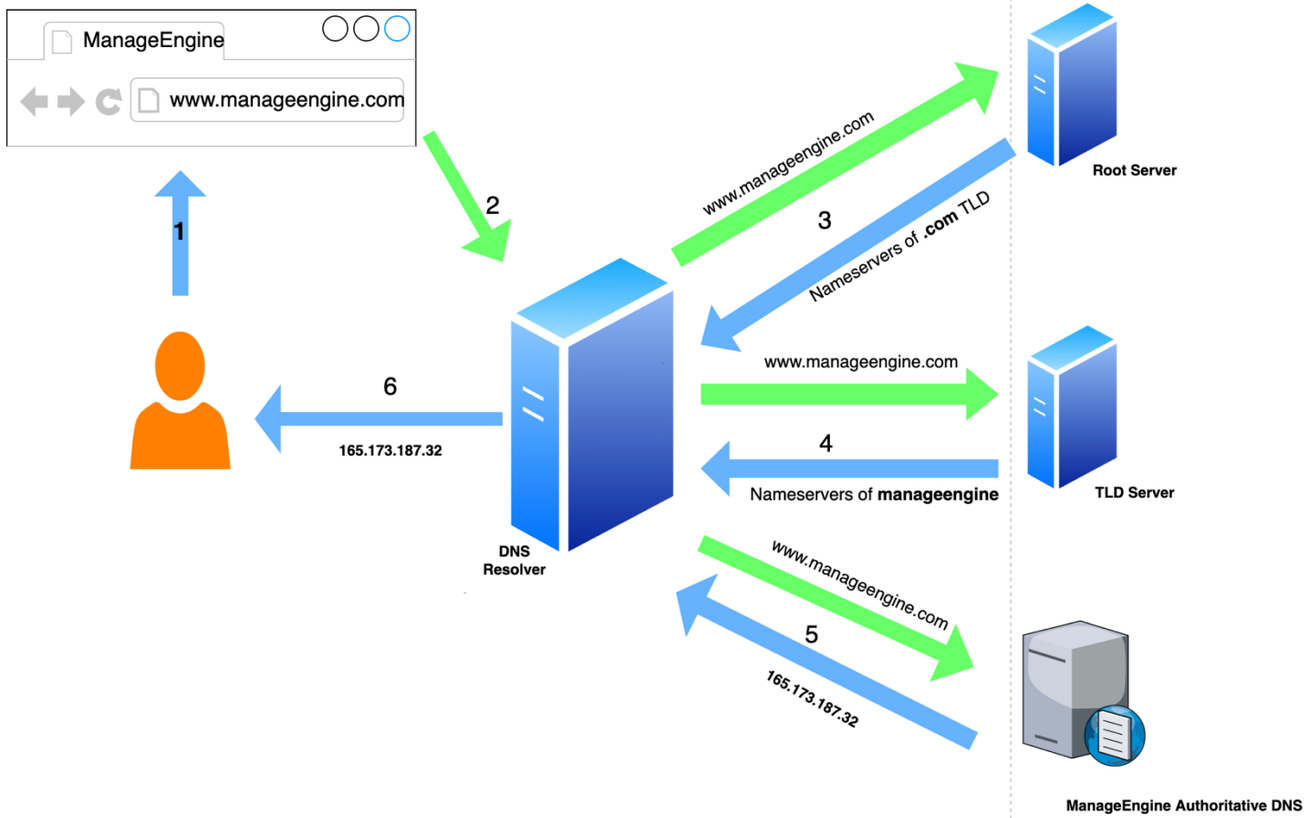

**Rappresentazione grafica tratta dal video:** https://youtu.be/p7PbB9MnKko?si=_vSjnfTdug3CIjKs




**Rappresentazione grafica tratta dal video:** https://youtu.be/p7PbB9MnKko?si=_vSjnfTdug3CIjKs




**Rappresentazione grafica tratta dal video:** https://youtu.be/p7PbB9MnKko?si=_vSjnfTdug3CIjKs


ManageEngine
www.manageengine.com
1
2
3
4
5
6
www.manageengine.com
Nameservers of .com TLD
www.manageengine.com
Nameservers of manageengine
www.manageengine.com
165.173.187.32
165.173.187.32
DNS Resolver
Root Server
TLD Server
ManageEngine Authoritative DNS

# Estensioni DNS

UNIVERSITÀ DEGLI STUDI DI TRIESTE

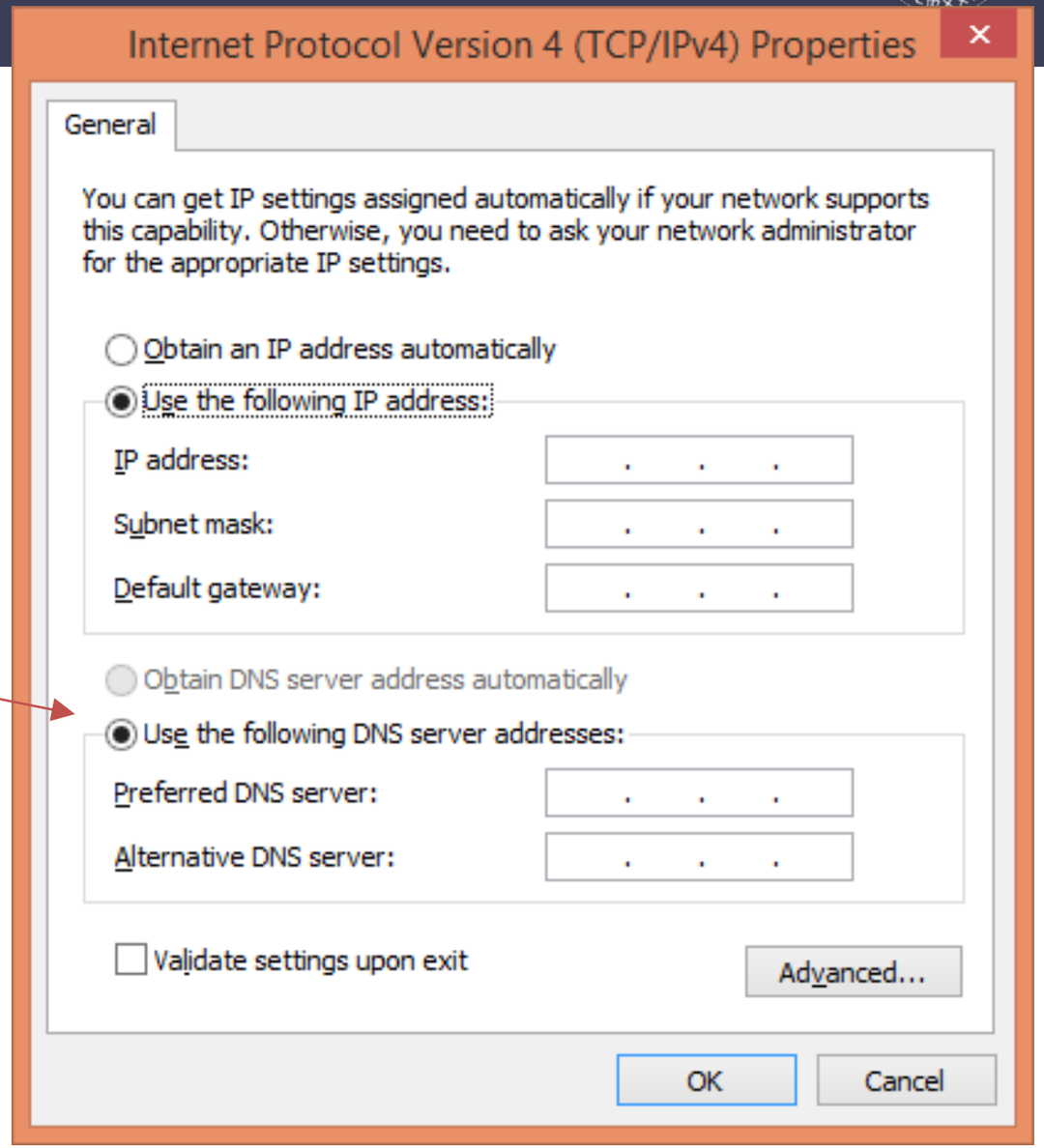
- L'estensione del dominio era stata pensata per identificare la nazione:
  - ✓ .it per l'Italia
  - ✓ .fr per Francia
  - ✓ …
- Negli USA hanno pensato diverse estensioni:
  - ✓ .com per organizzazioni commerciali
  - ✓ .org per organizzazione non lucrative
  - ✓ .gov per organizzazioni governative
  - ✓ .edu per le università

- Oggi c'è maggiore libertà ed esistono numerosi estensioni (.net, .tv)

# Configurazione DNS

- Ogni nodo necessita di sapere chi contattare per risolvere i nomi di dominio
- Oltre a IP, subnet mask e default gateway è comune configurare il DNS

- Per semplicità questi dati possono anche essere forniti in automatico (tipico di configurazioni DHCP)



# Posta elettronica

- Oggi è comune usare la posta elettronica attraverso il web
- In realtà è possibile configurare anche un **client di posta** (es. MS Outlook)
- Un client di posta fa uso dei seguenti protocolli:
  - ✓ **SMTP** (*Simple Mail Transfer Protocol*) per spedire un messaggio
  - ✓ **POP** (*Post Office Protocol*) oppure
  - ✓ **IMAP** (*Internet Message Access Protocol*) per ricevere
- L'invenzione di SMTP non fu brillante per vari motivi tra cui:

  "*Each character is transmitted as a 8-bit byte with the high-order bit cleared to zero*"

  ovvero era stato pensato per mandare mail solo in ASCII e senza allegati
- Far evolvere il protocollo e superare questi limiti non fu facile