



POLITECNICO  
MILANO 1863

# Fondamenti di TELECOMUNICAZIONI

Prof. Marco Mezzavilla

# Lezione 9 – Livello Applicativo 2

# INDICE

## 9. LIVELLO APPLICATIVO 2

1. **Applicazioni e architetture**
2. **Web browsing**
3. **Posta elettronica**
4. **Risoluzione di nomi simbolici e caching**
5. **Streaming e peer-to-peer**

Parte I (ieri)

Parte II (oggi)

# RISOLUZIONE DI NOMI SIMBOLICI

01

# Domain Name System (DNS)

- ❑ Gli indirizzi IP (32 bit) sono poco adatti ad essere usati dagli applicativi
- ❑ E' più comodo utilizzare indirizzi simbolici:
  - ❑ E' meglio www.google.com o 74.125.206.99?
- ❑ Occorre una mappatura fra **indirizzi IP** (usati dalle macchine di rete) ed i **nomi simbolici** (usati dagli esseri umani)



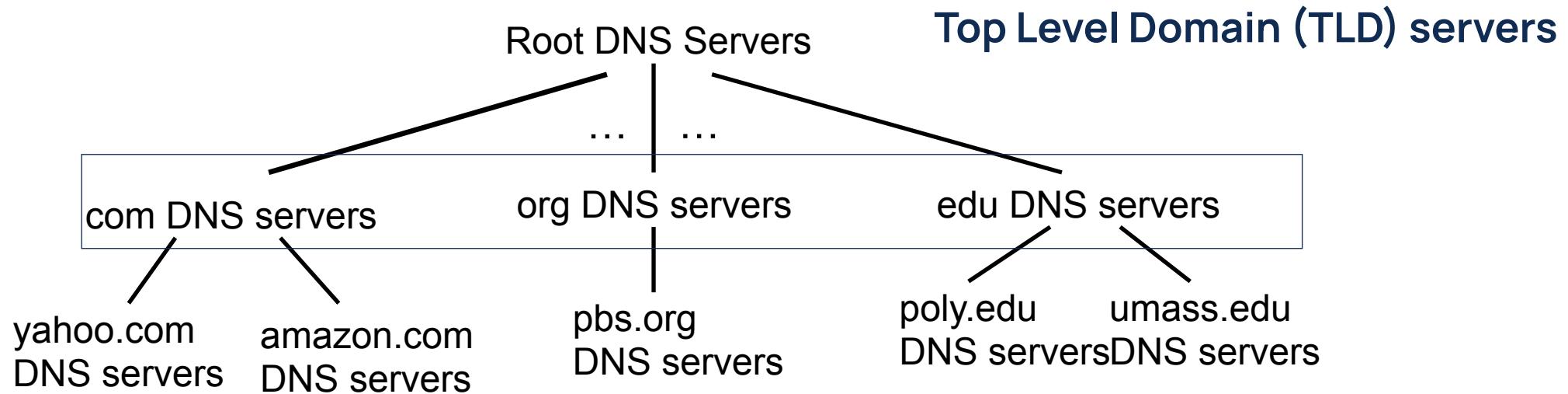
# Domain Name System (DNS)

## □ Ingredienti

- Database distribuito costituito da molti name servers con organizzazione gerarchica
- Protocollo applicativo tra name server e host per risolvere nomi simbolici (tradurre nomi simbolici in indirizzi IP)

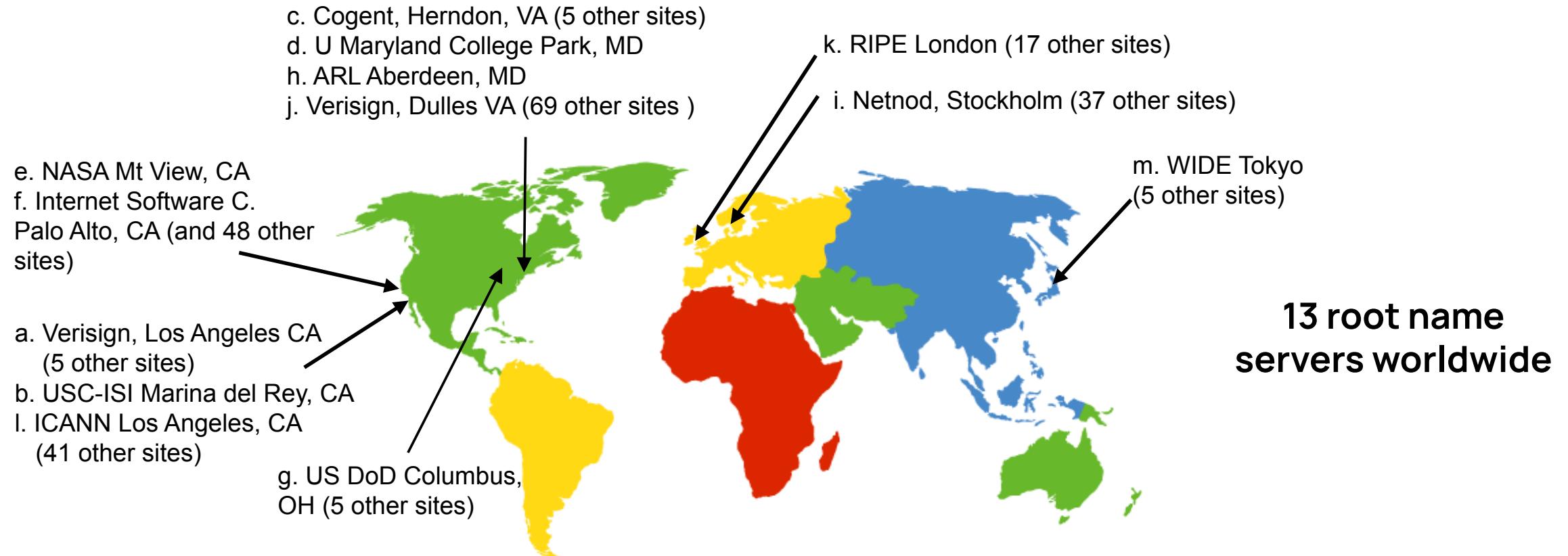


# Database distribuito e gerarchico



- Ogni livello nella gerarchia ha diversa “profondità” di informazione
- Esempio: un utente vuole l’IP di www.google.com
- Root name server sanno come “trovare” i name server di che gestiscono i domini .com
- I name server .com sanno come trovare i name server che gestiscono il dominio google.com
- I name server google.com sanno risolvere il nome simbolico www.google.com

# Root name server (NS)



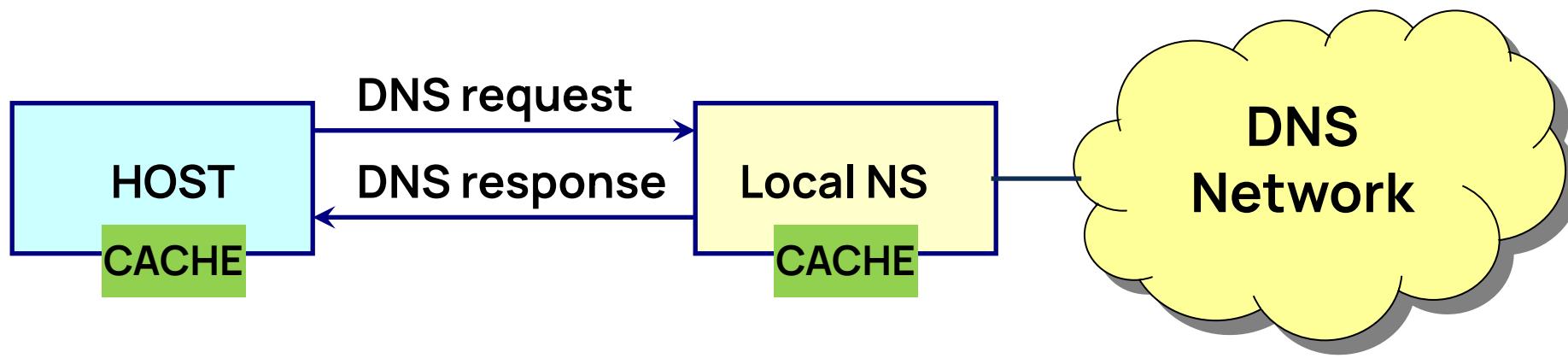
# Altri tipi di NS

## □ Local Name Servers

- Ogni ISP (residenziale, università, compagnia) ha un NameServer (o server DNS) locale
- Direttamente collegati agli host
- Tutte le volte che un host deve risolvere un indirizzo simbolico contatta il Local Name Server
- Il Local Name Server (eventualmente) contatta i Root Name Server nella gerarchia

# Come ottenere un mappaggio

1. **Richiesta del sito:** Quando scrivi un nome di dominio (es. **google.com**) nel browser, il tuo computer non sa ancora qual è l'indirizzo IP di quel sito. Ha bisogno di chiedere ai **server DNS**.
2. **Cache locale:** Prima di fare una richiesta online, il tuo computer controlla se ha già l'indirizzo IP di quel sito memorizzato in una **cache** (una specie di "memoria temporanea"). Se l'ha visitato di recente, potrebbe avere già l'indirizzo e non deve chiedere a nessuno.
3. **Richiesta al DNS:** Se l'indirizzo non è nella cache, il tuo computer invia una richiesta a un **server DNS** (di solito fornito dal tuo provider Internet, o LNS visto nella slide precedente). Questo server DNS cerca di trovare l'indirizzo IP del sito richiesto.
4. **Ricerca DNS:**
  1. Se il server DNS del provider non conosce già l'indirizzo, **chiederà ad altri server DNS**.
  2. Potrebbe chiedere prima ai **root DNS** (che sono come "punti di partenza" per trovare altri server DNS) e poi ai **DNS autoritativi**, che sono i server che gestiscono effettivamente il dominio, come **google.com**.
5. **Ritorno dell'indirizzo IP:** Una volta trovato l'indirizzo IP di **google.com**, il server DNS lo invia al tuo computer.
6. **Connessione al sito:** Ora che il tuo computer ha l'indirizzo IP, può **connettersi direttamente al server di Google** e mostrarti il sito web.



# Come aggiungere un dominio alla “rete” DNS

- ❑ Una nuova startup I-Like-Networking vuole registrare il dominio I-Like-Networking.com (supponendo che il dominio sia disponibile)
- ❑ I-Like-Networking registra il dominio presso un registrar di domini (ad esempio, GoDaddy).
- ❑ I-Like-Networking specifica i server DNS che gestiranno il dominio, i quali possono essere esterni o configurati da I-Like-Networking.
- ❑ Il DNS registrar comunica il record al TLD .com, che inserisce un delegation record per I-Like-Networking.com che punta ai server DNS specificati (ad esempio, dns1.I-Like-Networking.com).
- ❑ Dopo aver configurato server e record DNS, ci vorrà del tempo perché queste modifiche si diffondano nella rete DNS globale. Questo processo è noto come **propagazione del DNS** e può richiedere da pochi minuti a 48 ore.

# Caching

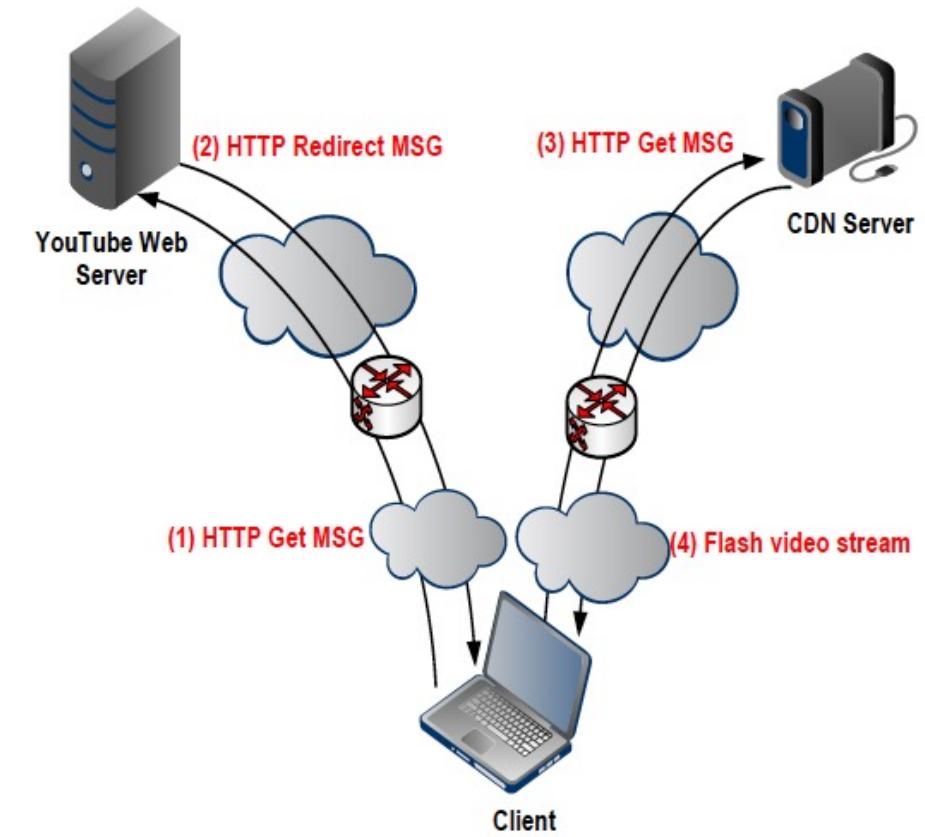
- ❑ Un server, dopo aver reperito una mappatura può memorizzarla temporaneamente (cache)
- ❑ All'arrivo di una nuova richiesta può fornire l'informazione senza risalire al *server autoritativo*
- ❑ Il **TimeToLive** (TTL) indica per quanto tempo una certa mappatura può essere conservata nella cache
- ❑ Il TTL viene impostato dal *server autoritativo*
- ❑ I TLD (e.g., *.com*, *.org*, *.it*) server sono generalmente “memorizzati” nei Local Name Servers
- ❑ I server non-authoritative usano il TTL per decidere un time-out

# STREAMING, P2P

02

# HTTP e video streaming: YouTube

- Architettura di YouTube si basa su:
  - HTTP per la distribuzione di contenuti video
    - **DASH**: Dynamic Adaptive Streaming over HTTP
      - Il video è disponibile in diverse **qualità** (ad esempio 240p, 480p, 720p, 1080p, 4K)
      - Il **player di YouTube** monitora la tua connessione Internet e fornisce una qualità consona
    - **Buffering**: il processo mediante il quale il player di YouTube scarica e memorizza alcuni segmenti video in anticipo rispetto alla posizione corrente di visualizzazione. Il buffer ti permette di vedere il video senza interruzioni, anche se ci sono brevi cali di velocità nella tua connessione Internet.
  - **Content Delivery Networks (CDN)**, cioè una rete di server distribuiti in tutto il mondo. I server CDN “avvicinano” fisicamente i video agli utenti, in modo che il video venga trasmesso dal server più vicino, riducendo il tempo di caricamento e migliorando la qualità.



# P2P file sharing

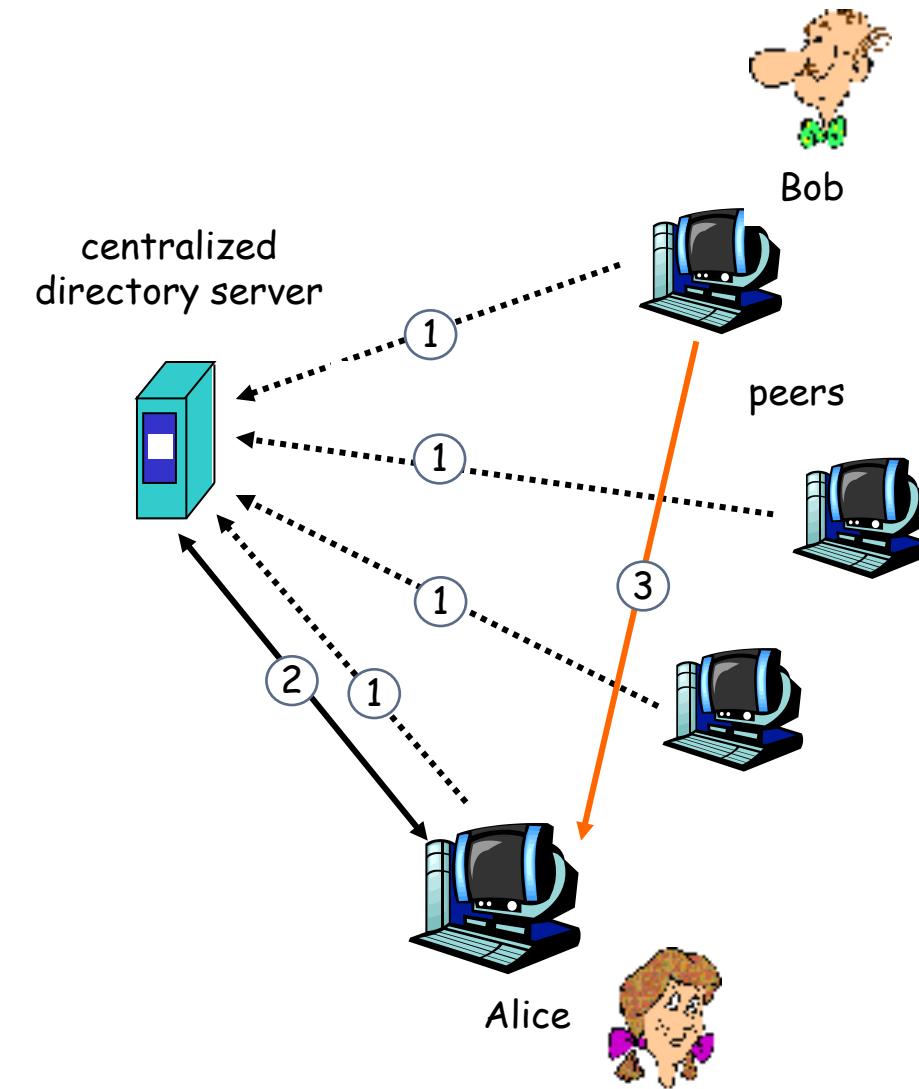
**L'applicazione P2P è sia client che server.**

- Gli utenti utilizzano il software P2P sul proprio PC
- Si collegano in modo intermittente a Internet
- Se un utente cerca un file l'applicazione trova altri utenti che lo hanno
- L'utente sceglie da chi scaricarlo
- Il file è scaricato usando un protocollo come HTTP
- Altri utenti potranno (in seguito o in contemporanea) scaricare il file dall'utente

# P2P: directory centralizzata

## □ Meccanismo di “Napster”

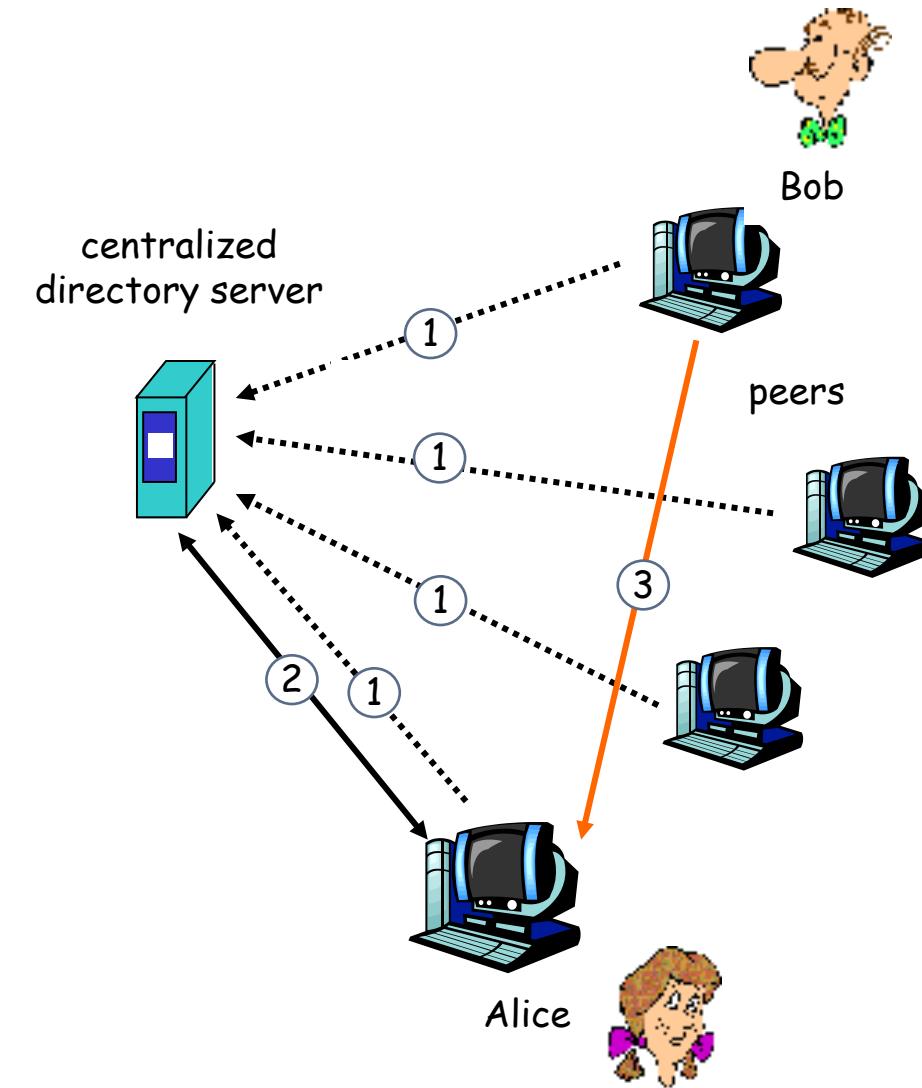
1. Quando i peer si connettono, informano il server centrale:
  - Indirizzo IP
  - File condivisi
2. Il peer interroga il server centrale per uno specifico file
3. Il file viene scaricato direttamente



# P2P: directory centralizzata

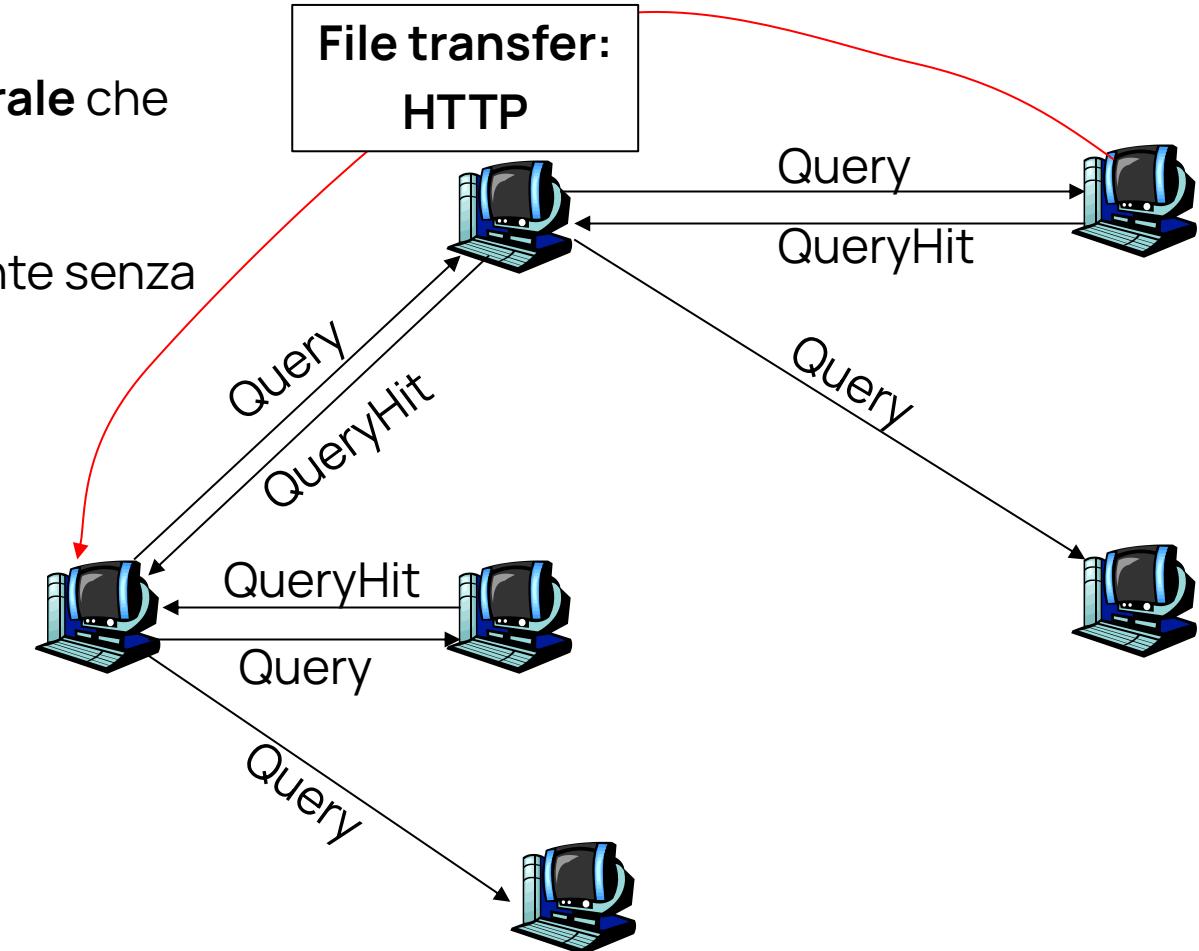
- Problemi dell'architettura centralizzata
  - Se il server si rompe il sistema si blocca
  - Il server è un collo di bottiglia per il sistema
  - Chi gestisce il server può essere accusato di infrangere le regole sul copyright

Il trasferimento file è **distribuito**, ma la ricerca dei contenuti è fortemente **centralizzata**



# P2P completamente distribuita

- Nel modello **P2P distribuito**, non esiste un server centrale che coordina o gestisce le connessioni tra i peer.
- Tutti i peer sono uguali tra loro e comunicano direttamente senza la necessità di una parte centralizzata.
- I peer si autogestiscono e trovano altri peer attraverso meccanismi distribuiti.
- **Ricerca di un file**
  - I messaggi di richiesta vengono diffusi in rete
  - I peer inoltrano le richieste fino a una certa distanza
  - Le risposte vengono inviate sul cammino opposto



# P2P completamente distribuita

## Accesso alla rete

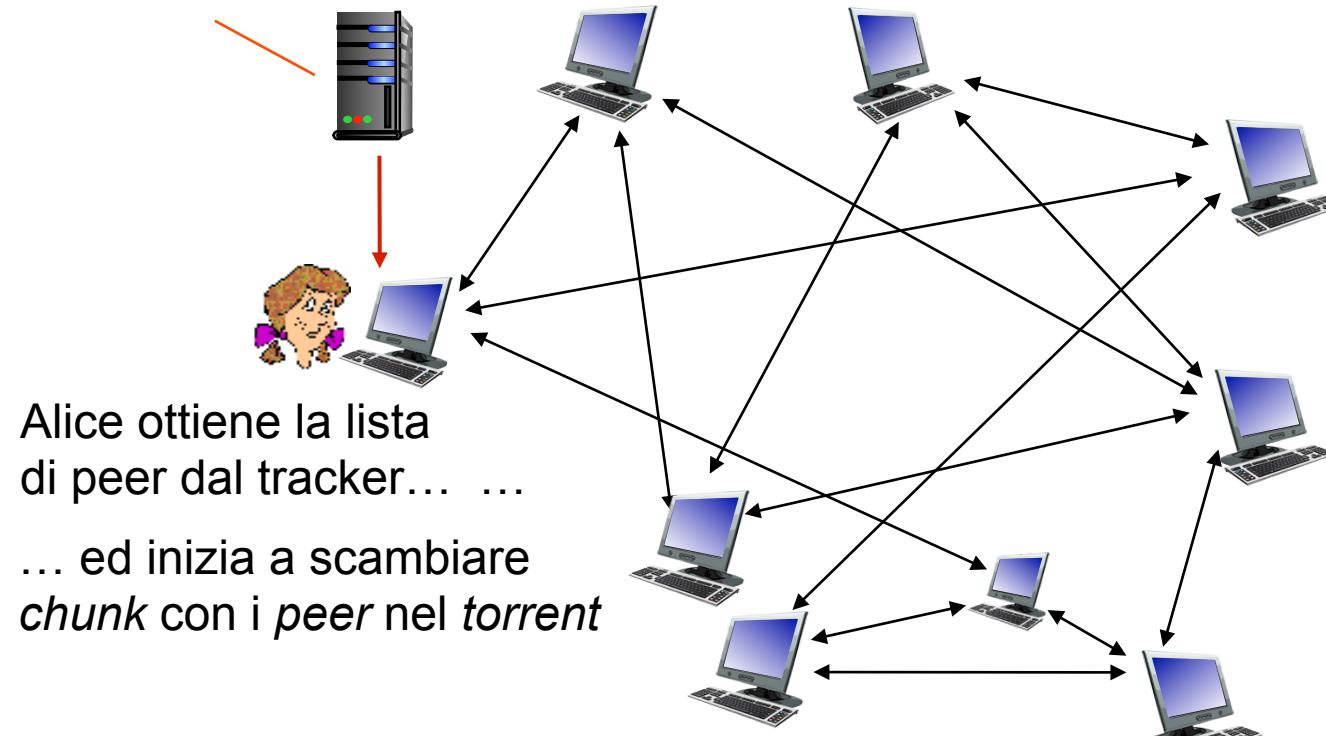
- Per iniziare il processo di accesso il peer X deve trovare almeno un altro peer; la ricerca si basa su liste note
- X scandisce la lista fino a che un peer Y risponde
- X invia un messaggio di Ping a Y;
- Y inoltra il Ping nella rete di overlay.
- Tutti i peer che ricevono il Ping rispondono a X con un messaggio di Pong
- X riceve molti messaggi di Pong e può scegliere a chi connettersi aumentando il numero dei suoi vicini nella rete di overlay

# BitTorrent

- I file sono divisi in *chunk* di 256 kbyte

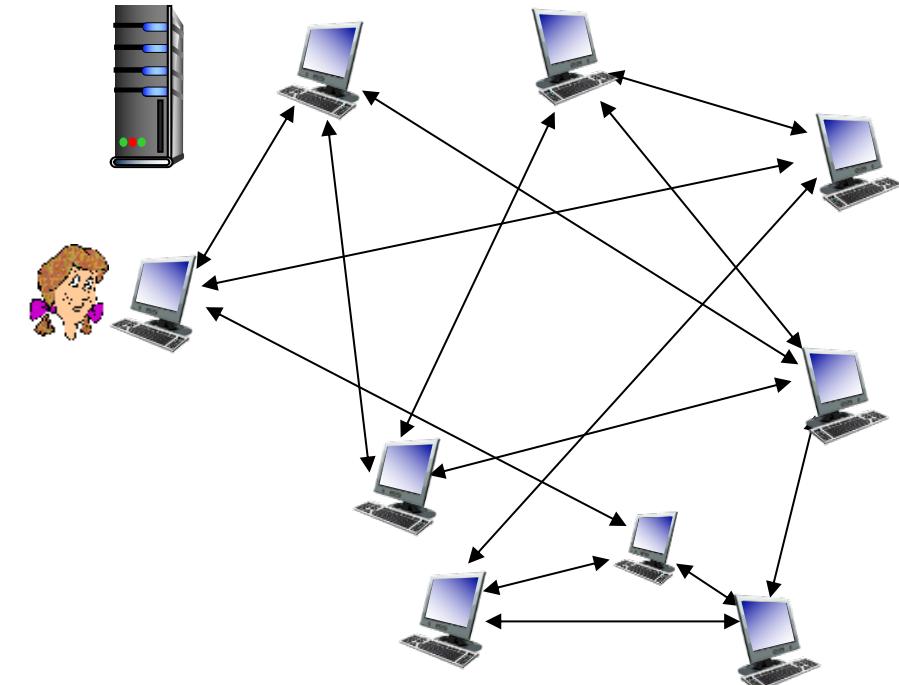
**tracker:** tiene traccia dei peer che partecipano ad un torrent

**torrent:** gruppo di peer che si scambiano chunk di un file



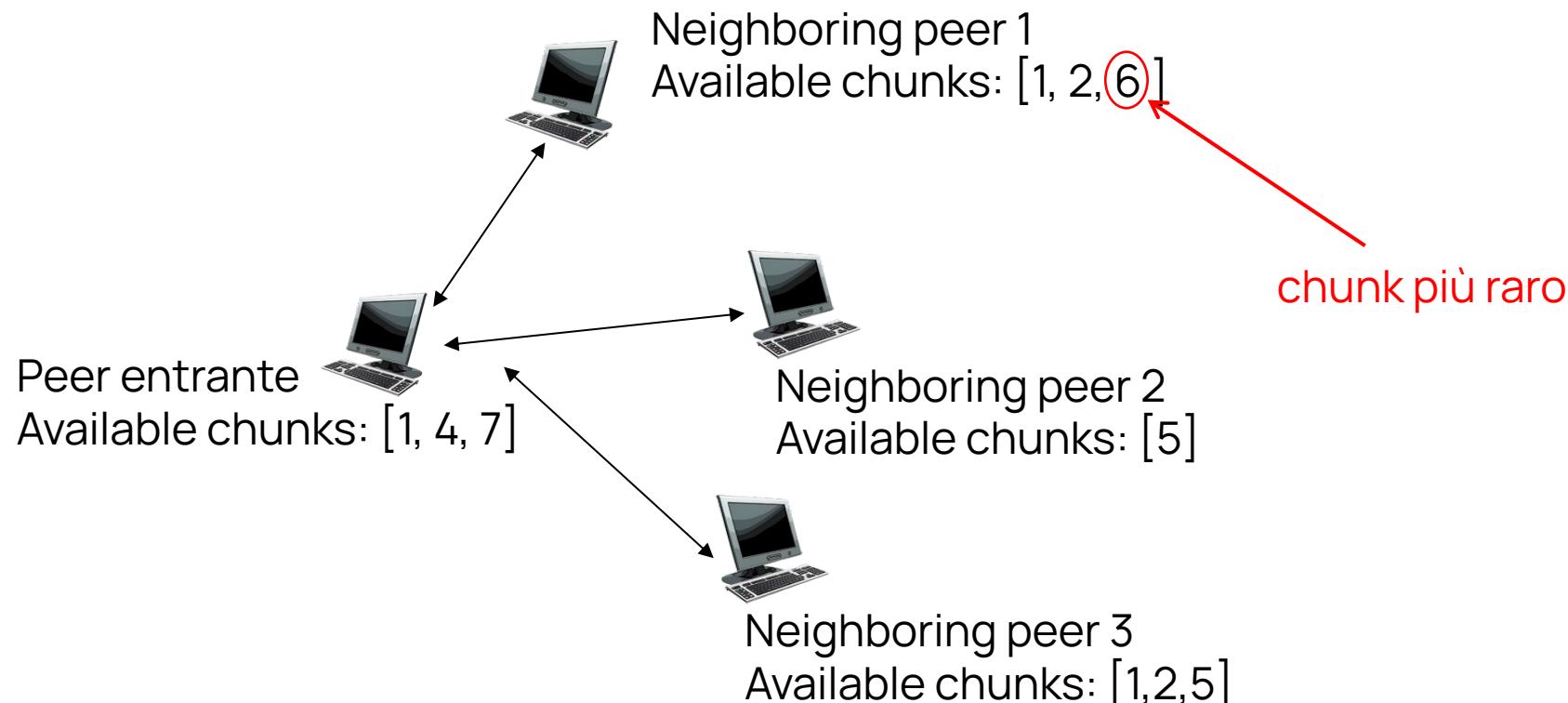
# BitTorrent – entrare nel torrent

- ❑ I peer che entrano in un torrent si registrano presso un tracker per ottenere una lista di peer “attivi”
- ❑ Il tracker invia una lista di peer attivi su un torrent (indirizzi IP)
- ❑ Il peer entrante stabilisce connessioni TCP con un sottoinsieme dei peer nella lista (neighboring peers)
- ❑ I neighboring peers inviano al peer entrante la lista dei chunk disponibili
- ❑ Il peer entrante sceglie quale chunk scaricare e da quale peer scaricare



# Meccanismo di richiesta dei *chunks*

- ❑ Il principio del **Rarest First**
  - ❑ Il peer entrante, tra tutti i chunk mancanti, scarica prima i chunk più rari nelle liste di chunk ricevute da tutti i neighboring peer



# Fondamenti di TELECOMUNICAZIONI

Prof. Marco Mezzavilla  
[marco.mezzavilla@polimi.it](mailto:marco.mezzavilla@polimi.it)