Capitolo 2
Livello di applicazione

Reti di calcolatori e Internet: Un approccio top-down

4ª edizione
Jim Kurose, Keith Ross

Pearson Paravia Bruno Mondadori Spa
©2008

All material copyright 1996-2007
J.F Kurose and K.W. Ross, All Rights Reserved

Capitolo 2: Livello di applicazione

- 2.1 Principi delle applicazioni di rete
- 2.2 Web e HTTP
- 2.4 Posta elettronica
  - SMTP, POP3, IMAP
- 2.5 DNS
- 2.6 Applicazioni P2P

Capitolo 2: Livello di applicazione

Obiettivi:

- Fornire i concetti base e gli aspetti implementativi dei protocolli delle applicazioni di rete
  - modelli di servizio del livello di trasporto
  - paradigma client-server
  - paradigma peer-to-peer
- Apprendere informazioni sui protocolli esaminando quelli delle più diffuse applicazioni di rete
  - HTTP
  - SMTP / POP3 / IMAP
  - DNS

Alcune diffuse applicazioni di rete

- Posta elettronica
- Web
- Messaggistica istantanea
- Autenticazione in un calcolatore remoto
- Condivisione di file P2P
- Giochi multiutente via rete
- Streaming di video-clip memorizzati
- Telefonia via Internet
- Videoconferenza in tempo reale
- Grid computing

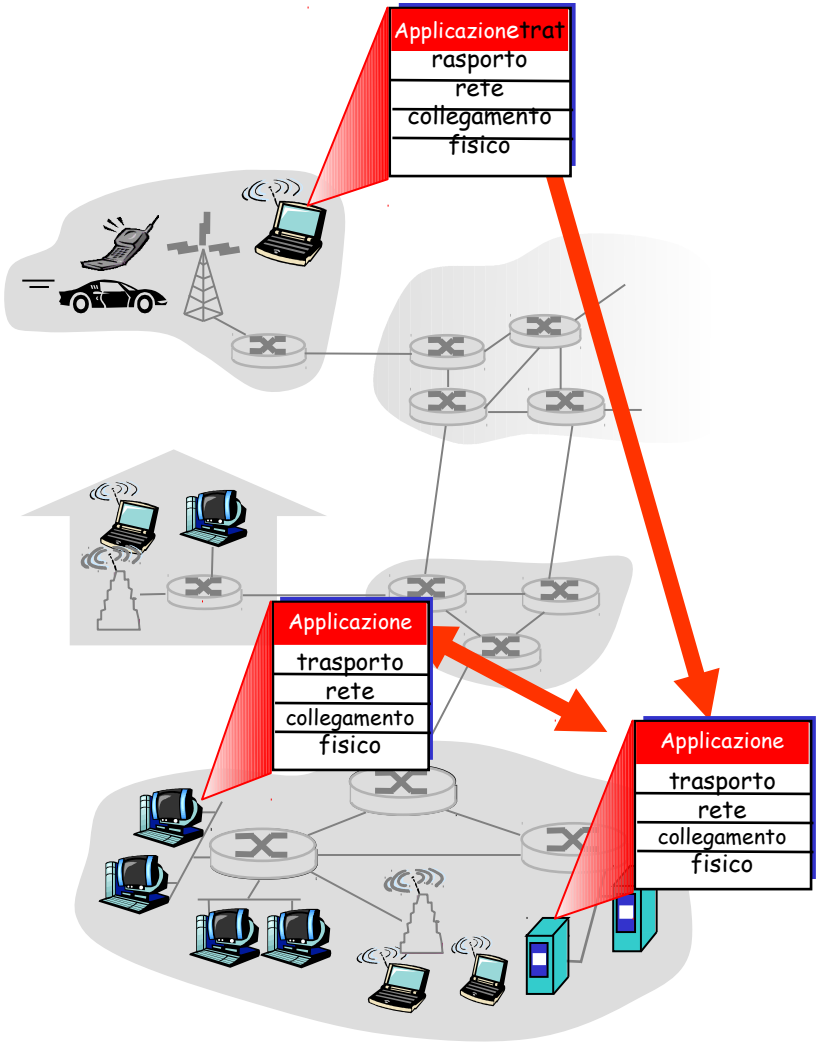
Creare un'applicazione di rete

Scrivere programmi che

- girano su sistemi terminali diversi
- comunicano attraverso la rete
- Ad es. il software di un server Web comunica con il software di un browser

software in grado di funzionare su più macchine

- non occorre predisporre programmi per i dispositivi del nucleo della rete, quali router o commutatori Ethernet



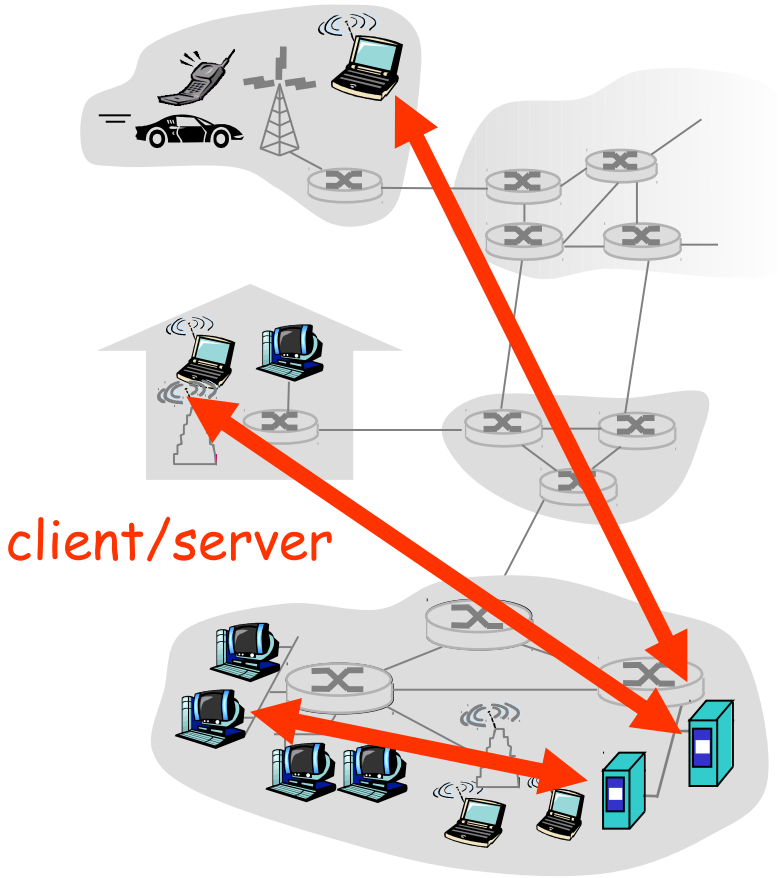
Capitolo 2: Livello di applicazione

- 2.1 Principi delle applicazioni di rete
- 2.2 Web e HTTP
- 2.4 Posta elettronica
  - SMTP, POP3, IMAP
- 2.5 DNS
- 2.6 Applicazioni P2P

Architetture delle applicazioni di rete

- Client-server
- Peer-to-peer (P2P)
- Architetture ibride (client-server e P2P)

Architettura client-server



server:

- host sempre attivo
- indirizzo IP fisso
- server farm per creare un potente server virtuale

client:

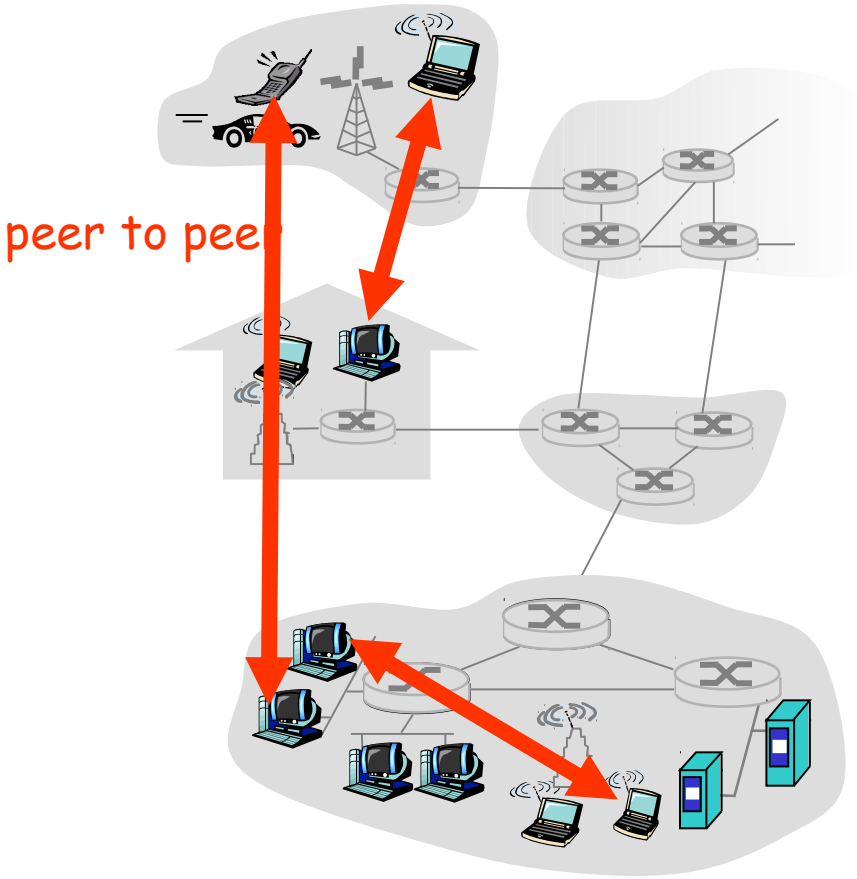
- comunica con il server
- può contattare il server in qualunque momento
- può avere indirizzi IP dinamici
- non comunica direttamente con gli altri client

Architettura P2P pura

- non c'è un server sempre attivo
- coppie arbitrarie di host (peer) comunicano direttamente tra loro
- i peer non devono necessariamente essere sempre attivi, e cambiano indirizzo IP

Facilmente scalabile

Difficile da gestire



Ibridi (client-server e P2P)

Skype

- Applicazione P2P di Voice over IP
- Server centralizzato: ricerca indirizzi della parte remota
- Connessione client-client: diretta (non attraverso il server)

Messaggistica istantanea

- La chat tra due utenti è del tipo P2P
- Individuazione della presenza/location centralizzata:
  - l'utente registra il suo indirizzo IP sul server centrale quando è disponibile online
  - l'utente contatta il server centrale per conoscere gli indirizzi IP dei suoi amici

Processi comunicanti

Processo: programma in esecuzione su di un host.

- All'interno dello stesso host, due processi comunicano utilizzando schemi interprocesso (definiti dal Sistema Operativo).
- processi su host differenti comunicano attraverso lo scambio di messaggi

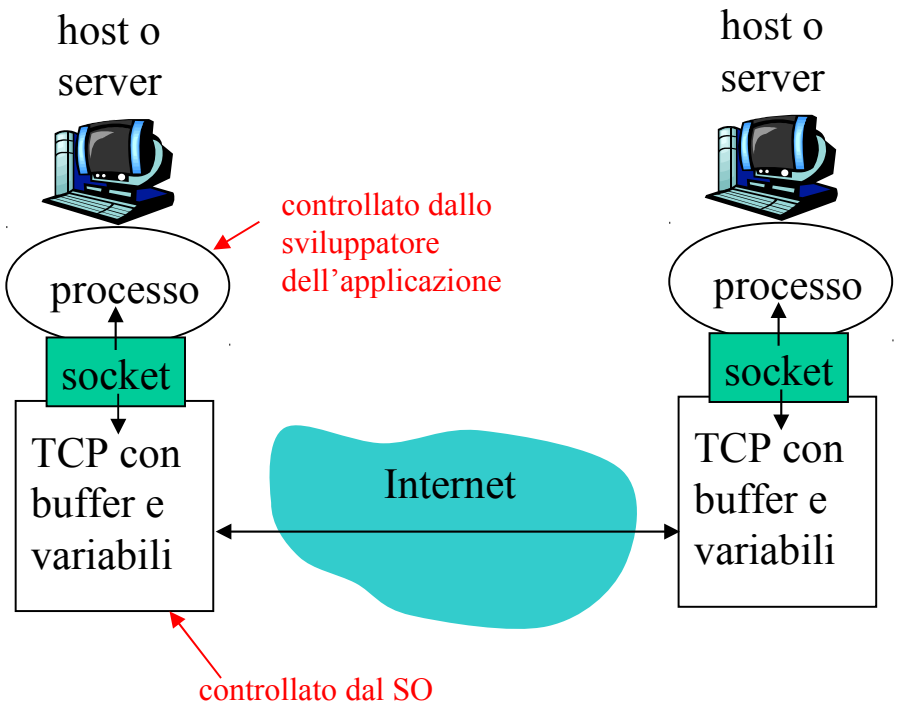
Processo client: processo che dà inizio alla comunicazione

Processo server : processo che attende di essere contattato

- Nota: le applicazioni con architetture P2P hanno processi client e processi server

Socket

- un processo invia/riceve messaggi a/da la sua socket
- una socket è analoga a una porta
  - un processo che vuole inviare un messaggio, lo fa uscire dalla propria "porta" (socket)
  - il processo presuppone l'esistenza di un'infrastruttura esterna che trasporterà il messaggio attraverso la rete fino alla "porta" del processo di destinazione



Processi di indirizzamento

- Affinché un processo su un host invii un messaggio a un processo su un altro host, il mittente deve identificare il processo destinatario.
- Un host ha un indirizzo IP univoco a 32 bit
- D: È sufficiente conoscere l'indirizzo IP dell'host su cui è in esecuzione il processo per identificare il processo stesso?
- Risposta: No, sullo stesso host possono essere in esecuzione molti processi.
- L'identificatore comprende sia l'indirizzo IP che i numeri di porta associati al processo in esecuzione su un host.
- Esempi di numeri di porta:
  - HTTP server: 80
  - Mail server: 25
- Per inviare un messaggio HTTP al server gaia.cs.umass.edu:
  - Indirizzo IP: 128.119.245.12
  - Numero di porta: 80

Protocollo a livello di applicazione

- Tipi di messaggi scambiati, ad esempio messaggi di richiesta e di risposta
- Sintassi dei tipi di messaggio: quali sono i campi nel messaggio e come sono descritti
- Semantica dei campi, ovvero significato delle informazioni nei campi
- Regole per determinare quando e come un processo invia e risponde ai messaggi

Protocolli di pubblico dominio:

- Definiti nelle RFC
- Consentono l'interoperabilità
- Ad esempio, HTTP, SMTP

Protocolli proprietari:

- Ad esempio, Skype

Quale servizio di trasporto richiede un'applicazione?

Perdita di dati

- alcune applicazioni (ad esempio, audio) possono tollerare qualche perdita
- altre applicazioni (ad esempio, trasferimento di file, telnet) richiedono un trasferimento dati affidabile al 100%

Temporizzazione

- alcune applicazioni (ad esempio, telefonia Internet, giochi interattivi) per essere "realistiche" richiedono piccoli ritardi

Throughput

- alcune applicazioni (ad esempio, quelle multimediali) per essere "efficaci" richiedono un'ampiezza di banda minima
- altre applicazioni ("le applicazioni elastiche") utilizzano l'ampiezza di banda che si rende disponibile

Sicurezza

- Cifratura, integrità dei dati, ...

Requisiti del servizio di trasporto di alcune applicazioni comuni

Applicazione	Tolleranza alla perdita di dati	Throughput	Sensibilità al tempo
Trasferimento file	No	Variabile	No
Posta elettronica	No	Variabile	No
Documenti Web	No	Variabile	No
Audio/video in tempo reale	Sì	Audio: da 5 Kbps a 1 Mbps Video: da 10 Kbps a 5 Mbps	Sì, centinaia di ms
Audio/video memorizzati	Sì	Come sopra	Sì, pochi secondi
Giochi interattivi	Sì	Fino a pochi Kbps	Sì, centinaia di ms
Messaggistica istantanea	No	Variabile	Sì e no

Servizi dei protocolli di trasporto Internet

Servizio di TCP:

- *orientato alla connessione:* è richiesto un setup fra i processi client e server
- *trasporto affidabile* fra i processi d'invio e di ricezione
- *controllo di flusso:* il mittente non vuole sovraccaricare il destinatario
- *controllo della congestione:* "strozza" il processo d'invio quando le rete è sovraccaricata
- *non offre:* temporizzazione, garanzie su un'ampiezza di banda minima, sicurezza

Servizio di UDP:

- trasferimento dati inaffidabile fra i processi d'invio e di ricezione
- non offre: setup della connessione, affidabilità, controllo di flusso, controllo della congestione, temporizzazione né ampiezza di banda minima e sicurezza

D: Perché preoccuparsi? Perché esiste UDP?

Applicazioni Internet: protocollo a livello applicazione e protocollo di trasporto

Applicazione	Protocollo a livello applicazione	Protocollo di trasporto sottostante
Posta elettronica	SMTP [RFC 2821]	TCP
Accesso a terminali remoti	Telnet [RFC 854]	TCP
Web	HTTP [RFC 2616]	TCP
Trasferimento file	FTP [RFC 959]	TCP
Multimedia in streaming	HTTP (es. YouTube) RTP [RFC 1889]	TCP o UDP
Telefonia Internet	SIP, RTP, proprietario (es. Skype)	Tipicamente UDP

Capitolo 2: Livello di applicazione

- 2.1 Principi delle applicazioni di rete
  - Architetture delle applicazioni
  - Servizi richiesti dalle applicazioni
- 2.2 Web e HTTP
- 2.4 Posta elettronica
  - SMTP, POP3, IMAP
- 2.5 DNS
- 2.6 Applicazioni P2P

Web e HTTP

- Terminologia
- Una pagina web è costituita da oggetti
- Un oggetto può essere un file HTML, un'immagine JPEG, un'applet Java, un file audio, …
- Una pagina web è formata da un file base HTML che include diversi oggetti referenziati
- Ogni oggetto è referenziato da un URL
- Esempio di URL:

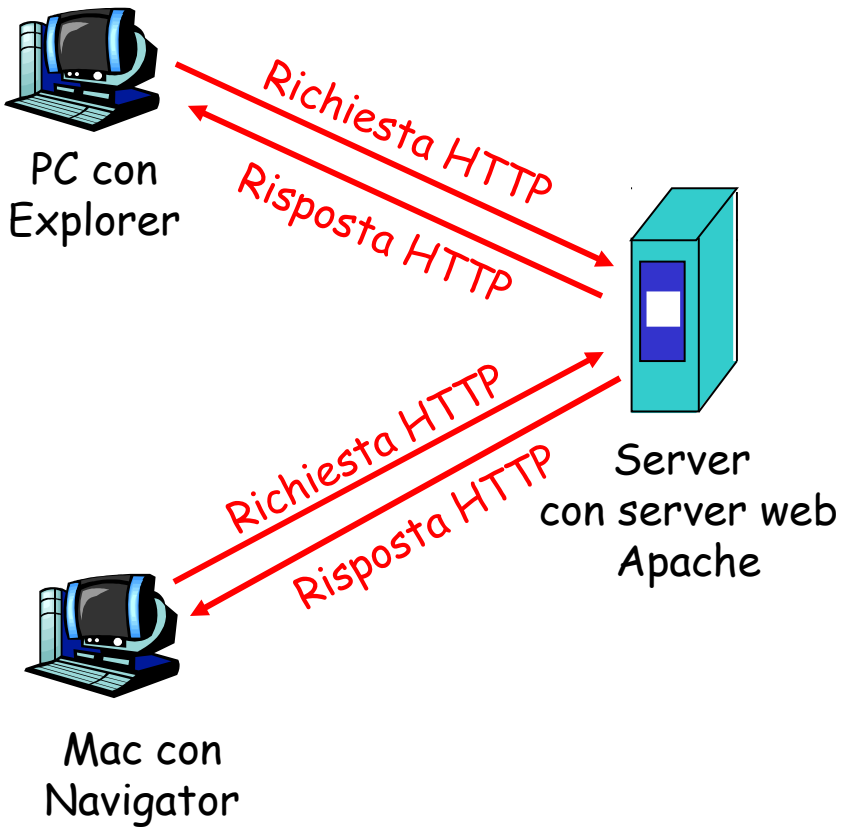
`www.someschool.edu` / `someDept/pic.gif`

nome dell'host — nome del percorso

Panoramica su HTTP

HTTP: hypertext transfer protocol

- Protocollo a livello di applicazione del Web
- Modello client/server
  - *client:* il browser che richiede, riceve, "visualizza" gli oggetti del Web
  - *server:* il server web invia oggetti in risposta a una richiesta



Panoramica su HTTP (continua)

Usa TCP:

- Il client inizializza la connessione TCP (crea una socket) con il server, la porta 80
- Il server accetta la connessione TCP dal client
- Messaggi HTTP scambiati fra browser (client HTTP) e server web (server HTTP)
- Connessione TCP chiusa

HTTP è un protocollo "senza stato" (stateless)

- Il server non mantiene informazioni sulle richieste fatte dal client

nota

I protocolli che mantengono lo "stato" sono complessi!

- La storia passata (stato) deve essere memorizzata
- Se il server e/o il client si bloccano, le loro viste dello "stato" potrebbero essere contrastanti e dovrebbero essere riconciliate

Interazione utente-server: i cookie

Molti dei più importanti siti web usano i cookie

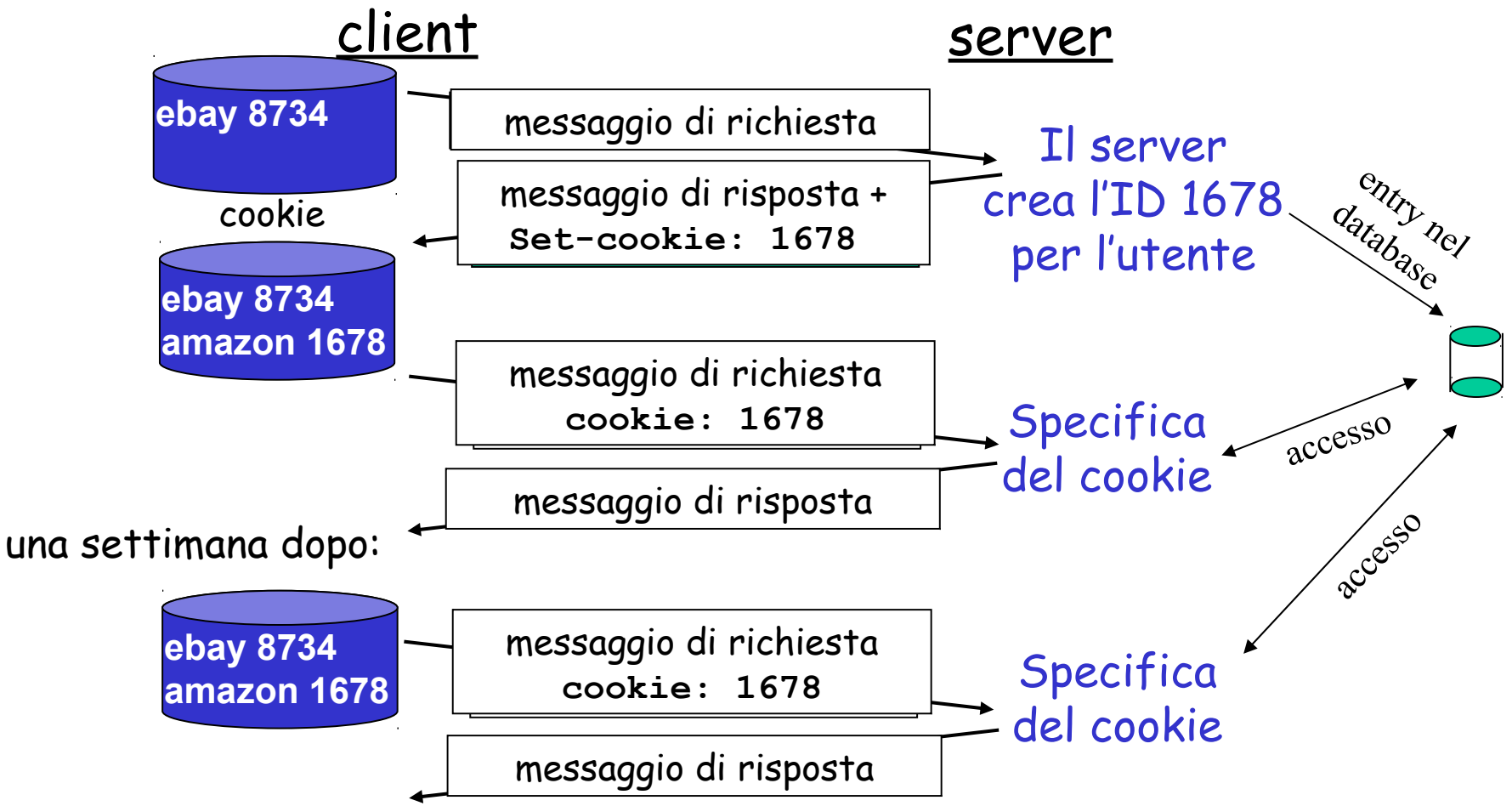
Quattro componenti:

1) Una riga di intestazione nel messaggio di *risposta* HTTP
2) Una riga di intestazione nel messaggio di *richiesta* HTTP
3) Un file cookie mantenuto sul sistema terminale dell'utente e gestito dal browser dell'utente
4) Un database sul sito

Esempio:

- Susan accede sempre a Internet dallo stesso PC
- Visita per la prima volta un particolare sito di commercio elettronico
- Quando la richiesta HTTP iniziale giunge al sito, il sito crea un identificativo unico (ID) e una *entry* nel database per ID

Cookie (continua)



Cookie (continua)

Cosa possono contenere i cookie:

- autorizzazione
- carta per acquisti
- raccomandazioni
- stato della sessione dell'utente (e-mail)

Lo "stato"

- Mantengono lo stato del mittente e del ricevente per più transazioni
- I messaggi http trasportano lo stato

nota

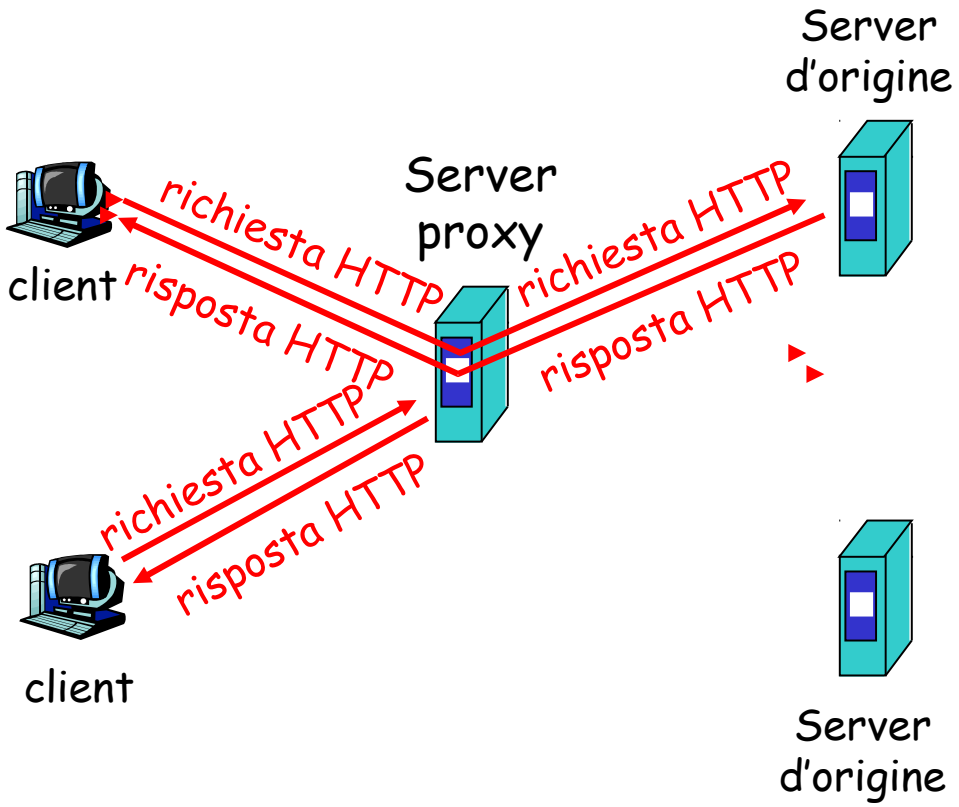
Cookie e privacy:

- i cookie permettono ai siti di imparare molte cose sugli utenti
- l'utente può fornire al sito il nome e l'indirizzo e-mail

Cache web (server proxy)

Obiettivo: soddisfare la richiesta del client senza coinvolgere il server d'origine

- L'utente configura il browser: accesso al Web tramite la cache
- Il browser trasmette tutte le richieste HTTP alla cache
  - oggetto nella cache: la cache fornisce l'oggetto
  - altrimenti la cache richiede l'oggetto al server d'origine e poi lo inoltra al client



Cache web (continua)

- La cache opera come client e come server
- Tipicamente la cache è installata da un ISP (università, aziende o ISP residenziali)

Perché il caching web?

- Riduce i tempi di risposta alle richieste dei client.
- Riduce il traffico sul collegamento di accesso a Internet.
- Internet arricchita di cache consente ai provider "scadenti" di fornire dati con efficacia (ma così fa anche la condivisione di file P2P)

Esempio di caching

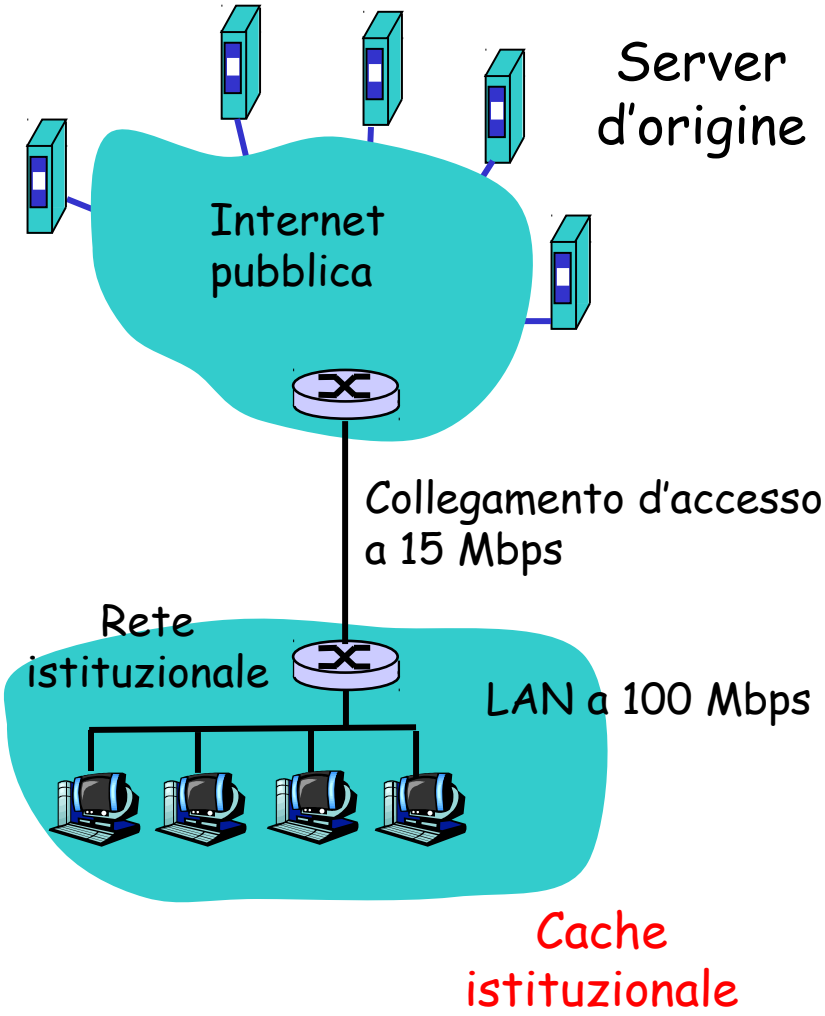
Ipotesi

- Dimensione media di un oggetto = 1Mb (Megabit)
- Frequenza media di richieste dai browser istituzionali ai server d'origine = 15/sec
- Ritardo dal router istituzionale a qualsiasi server d'origine e ritorno al router = 2 sec

Conseguenze

- utilizzazione sulla LAN = 15%
- utilizzazione sul collegamento d'accesso = 100%
- ritardo totale = ritardo di Internet + ritardo di accesso + ritardo della LAN

= 2 sec + minuti + millisecondi



Esempio di caching (continua)

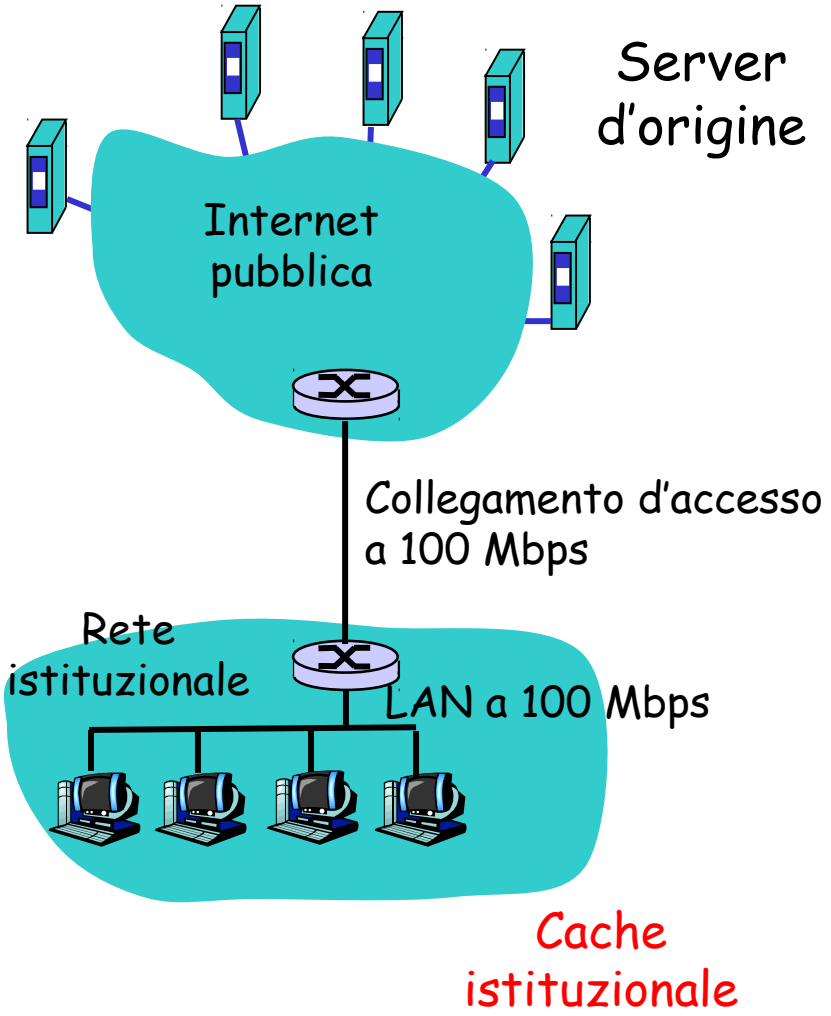
Soluzione possibile

- aumentare l'ampiezza di banda del collegamento d'accesso a 100 Mbps, per esempio

Conseguenze

- utilizzo sulla LAN = 15%
- utilizzo sul collegamento d'accesso = 15%
- ritardo totale = ritardo di Internet + ritardo di accesso + ritardo della LAN

  = 2 sec + msecs + msecs
- l'aggiornamento spesso è molto costoso



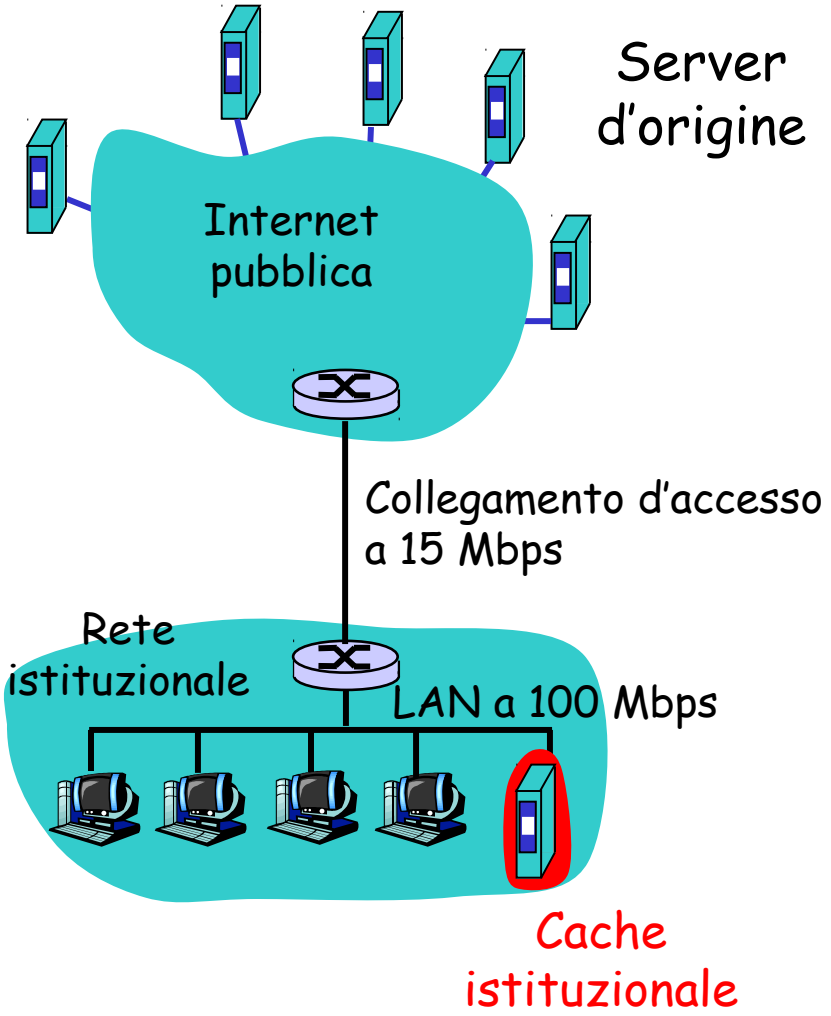
Esempio di caching (continua)

Soluzione possibile: installare la cache

- supponiamo una percentuale di successo (*hit rate*) pari a 0,4

Conseguenze

- il 40% delle richieste sarà soddisfatto quasi immediatamente
- il 60% delle richieste sarà soddisfatto dal server d'origine
- l'utilizzazione del collegamento d'accesso si è ridotta al 60%, determinando ritardi trascurabili (circa 10 msec)
- ritardo totale medio = ritardo di Internet + ritardo di accesso + ritardo della LAN =
  0,6*(2,01) sec + millisecondi < 1,3 sec



Capitolo 2: Livello di applicazione

- 2.1 Principi delle applicazioni di rete
- 2.2 Web e HTTP
- 2.4 Posta elettronica
  - SMTP, POP3, IMAP
- 2.5 DNS
- 2.6 Applicazioni P2P

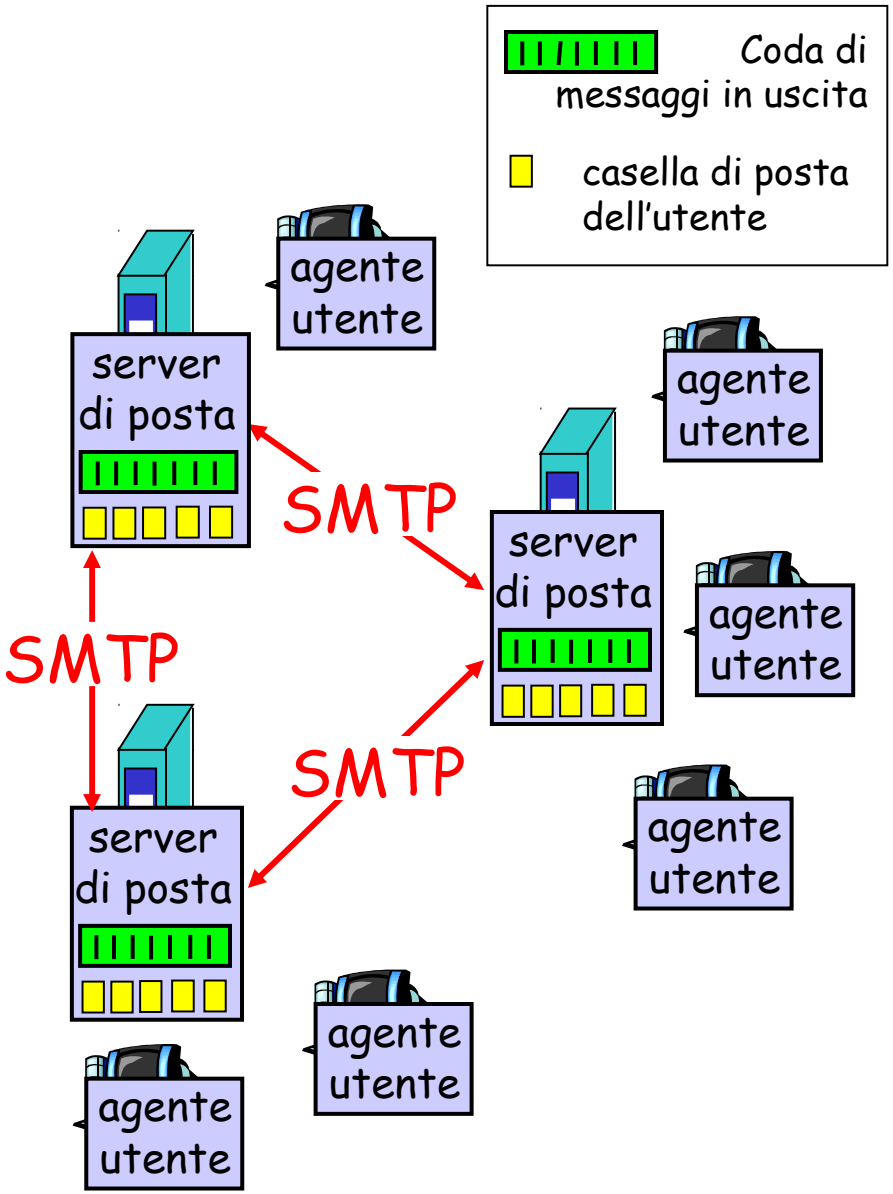
Posta elettronica

Tre componenti principali:

- agente utente
- server di posta
- simple mail transfer protocol: SMTP

Agente utente

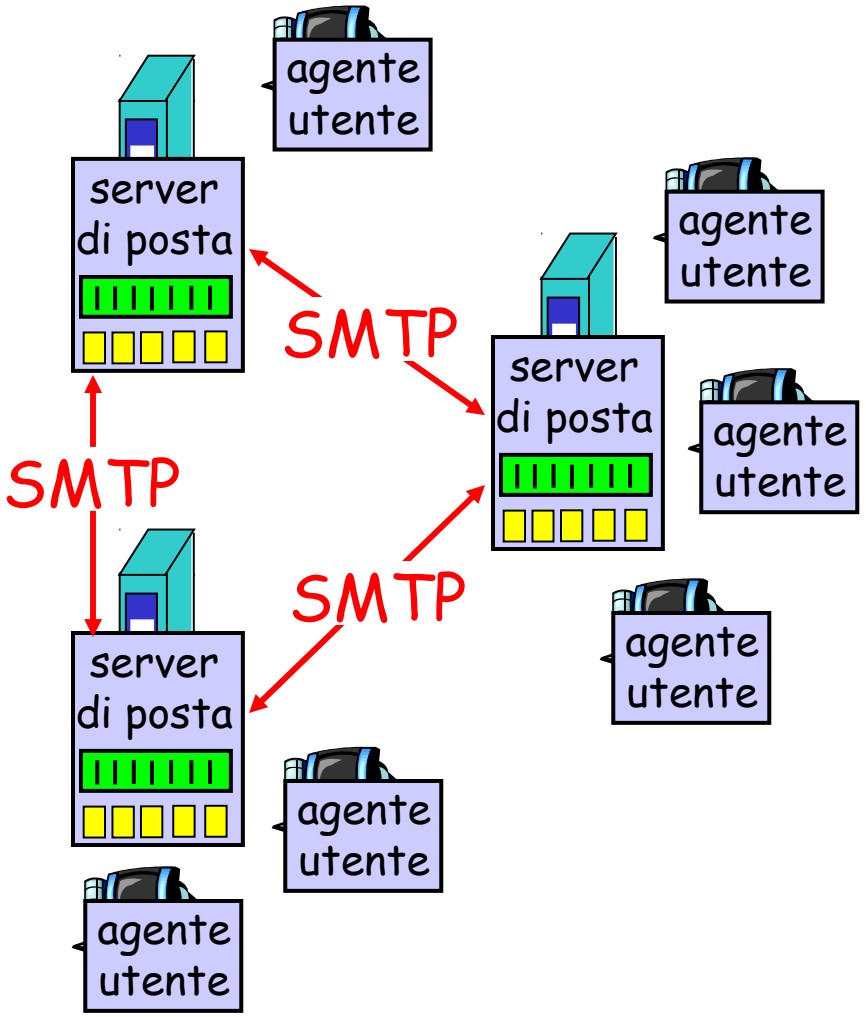
- detto anche "mail reader"
- composizione, editing, lettura dei messaggi di posta elettronica
- esempi: Eudora, Outlook, elm, Mozilla, Thunderbird
- i messaggi in uscita o in arrivo sono memorizzati sul server



Posta elettronica: server di posta

Server di posta

- Casella di posta (*mailbox*) contiene i messaggi in arrivo per l'utente
- Coda di messaggi da trasmettere
- Protocollo SMTP tra server di posta per inviare messaggi di posta elettronica
  - client: server di posta trasmittente
  - "server": server di posta ricevente

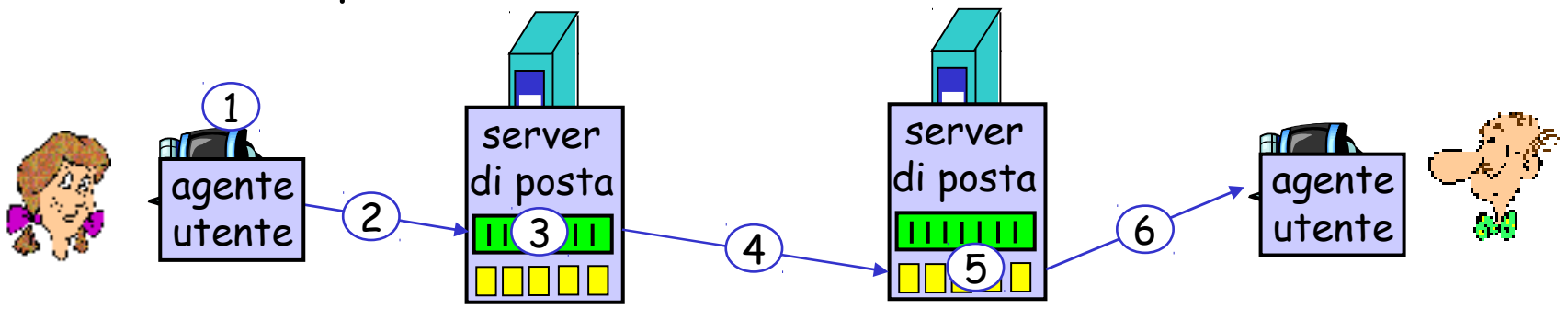


Posta elettronica: SMTP [RFC 2821]

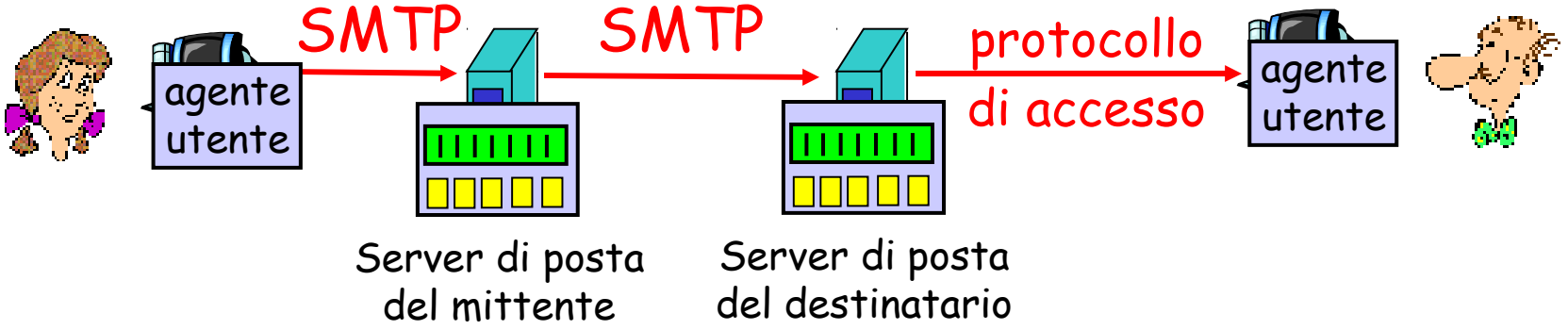
- usa TCP per trasferire in modo affidabile i messaggi di posta elettronica dal client al server, porta 25
- trasferimento diretto: il server trasmittente al server ricevente

Scenario: Alice invia un messaggio a Roberto

1) Alice usa il suo agente utente per comporre il messaggio da inviare "a" `rob@someschool.edu`
2) L'agente utente di Alice invia un messaggio al server di posta di Alice; il messaggio è posto nella coda di messaggi
3) Il lato client di SMTP apre una connessione TCP con il server di posta di Roberto
4) Il client SMTP invia il messaggio di Alice sulla connessione TCP
5) Il server di posta di Roberto pone il messaggio nella casella di posta di Roberto
6) Roberto invoca il suo agente utente per leggere il messaggio



Protocolli di accesso alla posta



- SMTP: consegna/memorizzazione sul server del destinatario
- Protocollo di accesso alla posta: ottenere i messaggi dal server
  - POP: Post Office Protocol [RFC 1939]
    - autorizzazione (agente <--> server) e download
  - IMAP: Internet Mail Access Protocol [RFC 1730]
    - più funzioni (più complesse)
    - manipolazione di messaggi memorizzati sul server
  - HTTP: gmail, Hotmail , Yahoo! Mail, ecc.

Protocollo POP3

Fase di autorizzazione

- Comandi del client:
  - `user:` dichiara il nome dell'utente
  - `pass:` password
- Risposte del server
  - `+OK`
  - `-ERR`

Fase di transazione, client:

- `list:` elenca i numeri dei messaggi
- `retr:` ottiene i messaggi in base al numero
- `dele:` cancella
- `quit`

```
S: +OK POP3 server ready
C: user rob
S: +OK
C: pass hungry
S: +OK user successfully logged on
```

```
C: list
S: 1 498
S: 2 912
S: .
C: retr 1
S: <message 1 contents>
S: .
C: dele 1
C: retr 2
S: <message 1 contents>
S: .
C: dele 2
C: quit
S: +OK POP3 server signing off
```

POP3 (altro) e IMAP

Ancora su POP3

- Il precedente esempio usa la modalità "scarica e cancella"
- Roberto non può rileggere le e-mail se cambia client
- Modalità "scarica e mantieni": copia i messaggi su più client
- POP3 è un protocollo senza stato tra le varie sessioni

IMAP

- Mantiene tutti i messaggi in un unico posto: il server
- Consente all'utente di organizzare i messaggi in cartelle
- IMAP conserva lo stato dell'utente tra le varie sessioni:
  - I nomi delle cartelle e l'associazione tra identificatori dei messaggi e nomi delle cartelle

Capitolo 2: Livello di applicazione

- 2.1 Principi delle applicazioni di rete
- 2.2 Web e HTTP
- 2.4 Posta elettronica
  - SMTP, POP3, IMAP
- 2.5 DNS
- 2.6 Applicazioni P2P

DNS: Domain Name System

Persone: molti identificatori:

- nome, codice fiscale, numero della carta d'identità

Host e router di Internet:

- indirizzo IP (32 bit) - usato per indirizzare i datagrammi
- "nome", ad esempio, www.yahoo.com – usato dagli esseri umani

D: Come associare un indirizzo IP a un nome?

Domain Name System:

- *Database distribuito* implementato in una gerarchia di *server DNS*
- *Protocollo a livello di applicazione* che consente agli host, ai router e ai server DNS di comunicare per *risolvere* i nomi (tradurre indirizzi/nomi)
  - Si noti: funzioni critiche di Internet implementate come protocollo a livello di applicazione
  - complessità nelle parti periferiche della rete

DNS

Servizi DNS

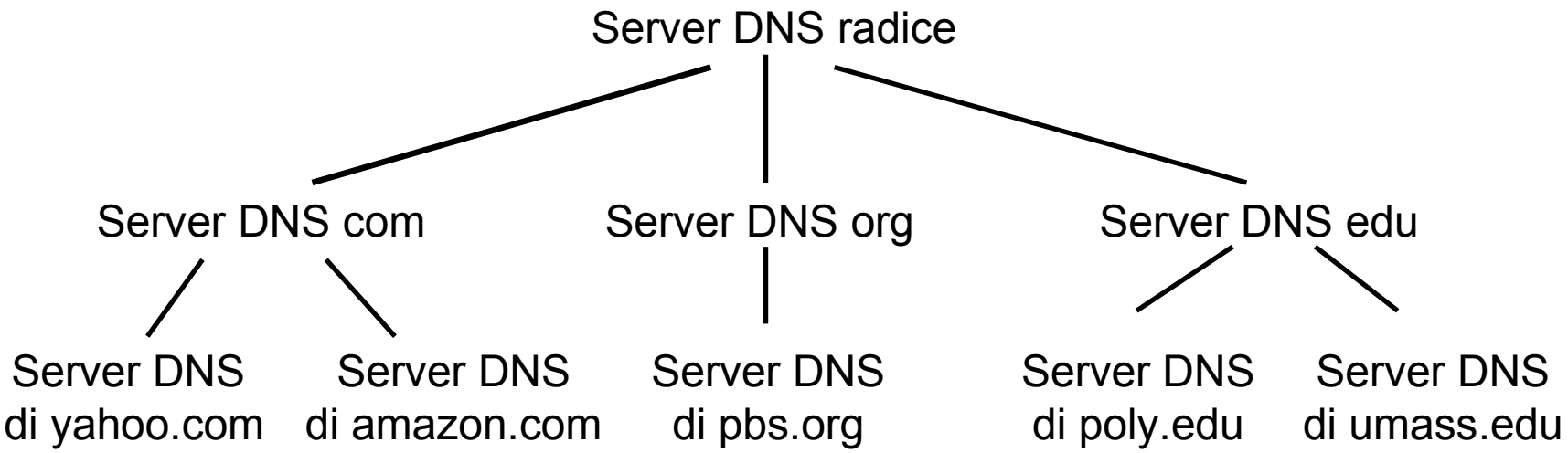
- Traduzione degli hostname in indirizzi IP
- Host aliasing
  - un host può avere più nomi
- Mail server aliasing
- Distribuzione locale
  - server web replicati: insieme di indirizzi IP per un nome canonico

Perché non centralizzare DNS?

- singolo punto di guasto
- volume di traffico
- database centralizzato distante
- manutenzione

Un database centralizzato su un singolo server DNS non è *scalabile* !

Database distribuiti e gerarchici

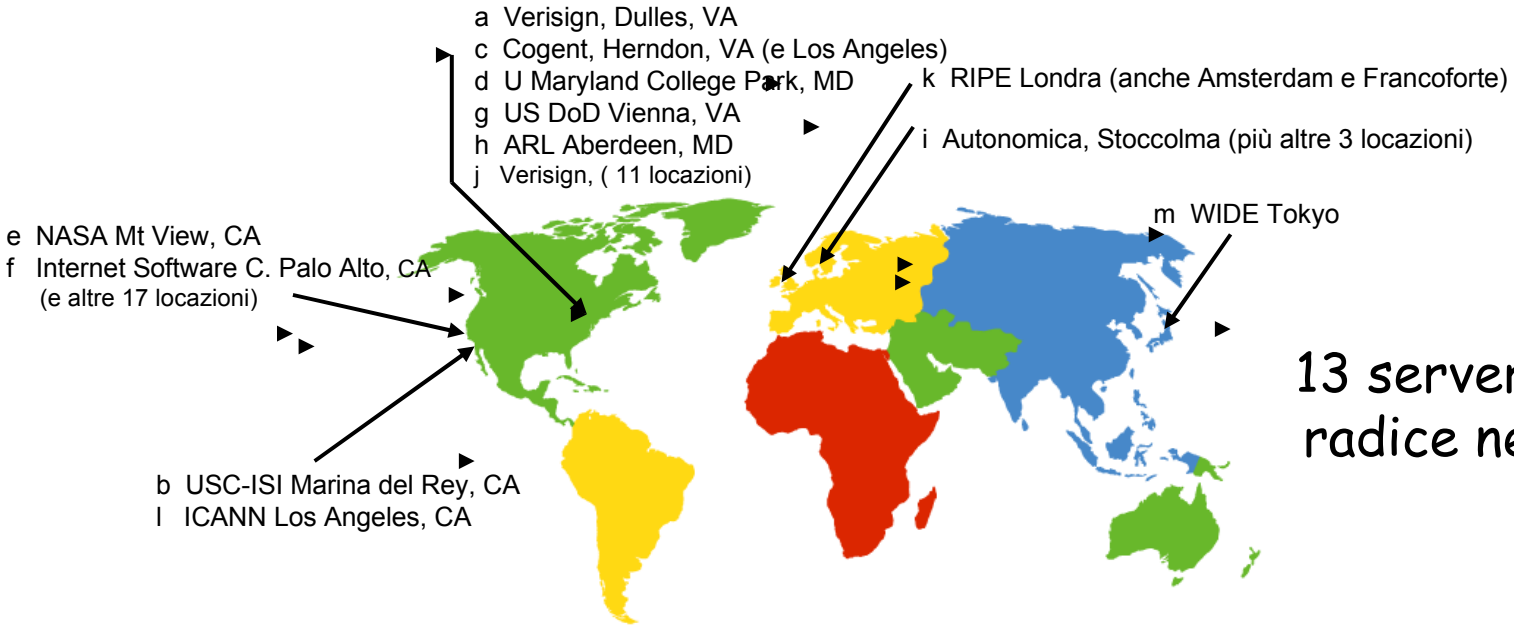


Il client vuole l'IP di www.amazon.com; 1ª approssimazione:

- Il client interroga il server radice per trovare il server DNS com
- Il client interroga il server DNS com per ottenere il server DNS amazon.com
- Il client interroga il server DNS amazon.com per ottenere l'indirizzo IP di www.amazon.com

DNS: server DNS radice

- contattato da un server DNS locale che non può tradurre il nome
- server DNS radice:
  - contatta un server DNS autorizzato se non conosce la mappatura
  - ottiene la mappatura
  - restituisce la mappatura al server DNS locale



13 server DNS radice nel mondo

Server TLD e server di competenza

- Server TLD (top-level domain): si occupano dei domini com, org, net, edu, ecc. e di tutti i domini locali di alto livello, quali uk, fr, ca e jp.
  - Network Solutions gestisce i server TLD per il dominio com
  - Educause gestisce quelli per il dominio edu

- Server di competenza (*authoritative server*): ogni organizzazione dotata di host Internet pubblicamente accessibili (quali i server web e i server di posta) deve fornire i record DNS di pubblico dominio che mappano i nomi di tali host in indirizzi IP.
  - possono essere mantenuti dall'organizzazione o dal service provider

Server DNS locale

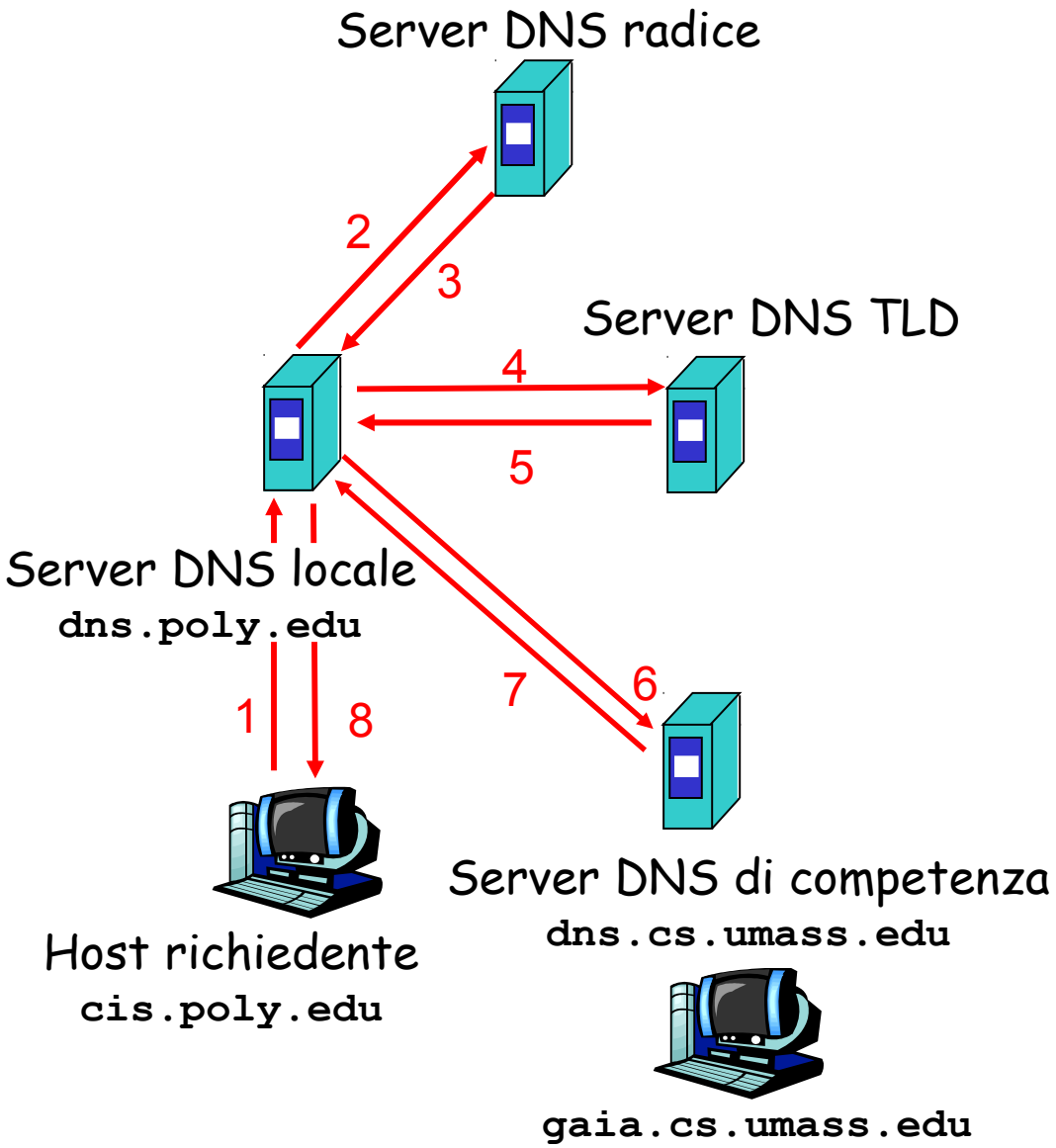
- Non appartiene strettamente alla gerarchia dei server
- Ciascun ISP (università, società, ISP residenziale) ha un server DNS locale.
  - detto anche "default name server"
- Quando un host effettua una richiesta DNS, la query viene inviata al suo server DNS locale
  - il server DNS locale opera da proxy e inoltra la query in una gerarchia di server DNS

Esempio

- L'host cis.poly.edu vuole l'indirizzo IP di gaia.cs.umass.edu

Query iterativa:

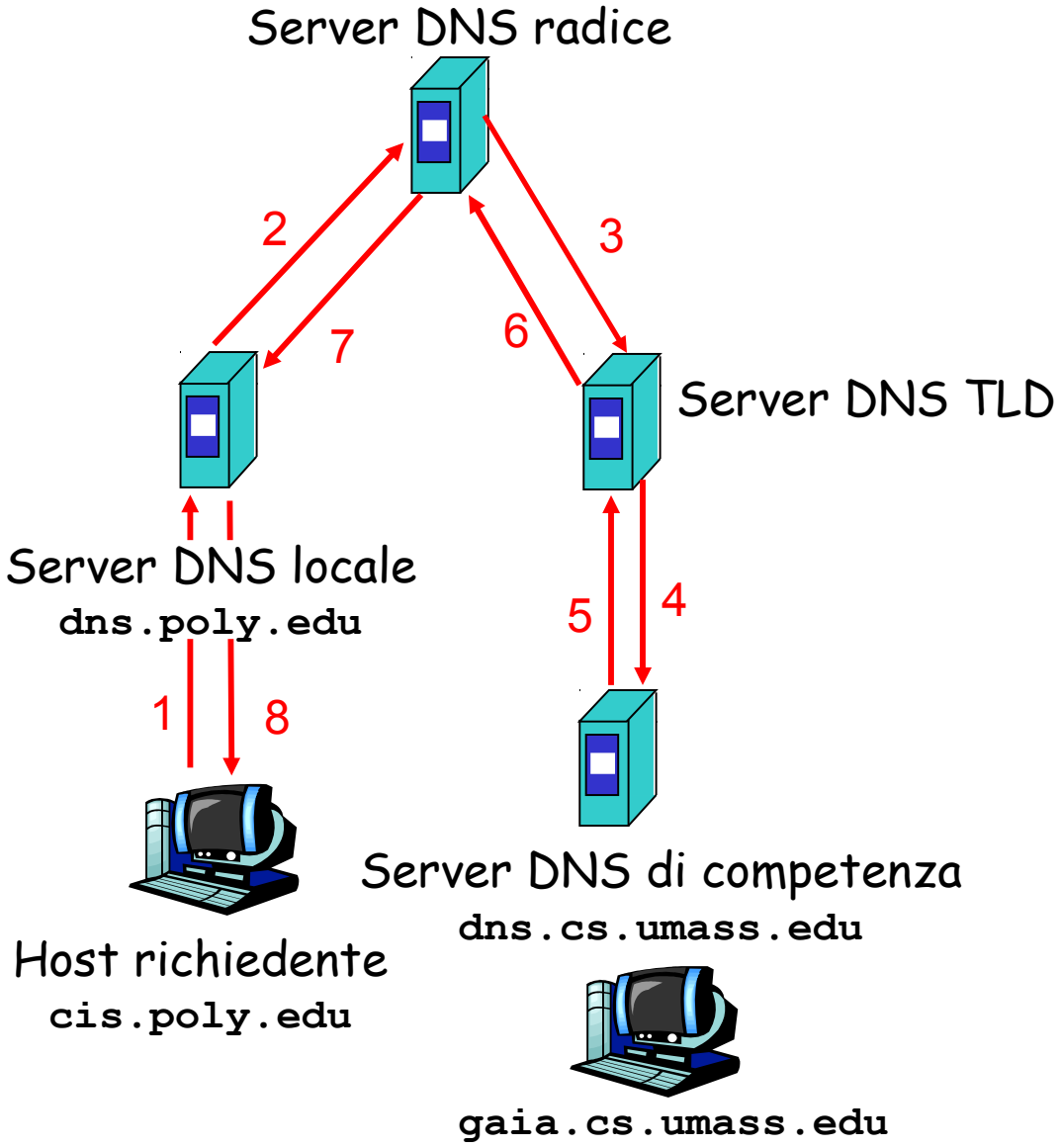
- Il server contattato risponde con il nome del server da contattare
- "Io non conosco questo nome, ma puoi chiederlo a questo server".



Esempio

Query ricorsiva:

- Affida il compito di tradurre il nome al server DNS contattato
- Compito difficile?



DNS: caching e aggiornamento dei record

- Una volta che un server DNS impara la mappatura, la mette nella *cache*
  - le informazioni nella cache vengono invalidate (spariscono) dopo un certo periodo di tempo
  - tipicamente un server DNS locale memorizza nella cache gli indirizzi IP dei server TLD
    - quindi i server DNS radice non vengono visitati spesso
- I meccanismi di aggiornamento/notifica sono progettati da IETF
  - RFC 2136
  - http://www.ietf.org/html.charters/dnsind-charter.html

Record DNS

DNS: database distribuito che memorizza i record di risorsa (RR)

Formato RR: **(name, value, type, ttl)**

- Type=A
  - **name** è il nome dell'host
  - **value** è l'indirizzo IP
- Type=NS
  - **name** è il dominio (ad esempio foo.com)
  - **value** è il nome dell'host del server di competenza di questo dominio
- Type=CNAME
  - **name** è il nome alias di qualche nome "canonico" (nome vero)

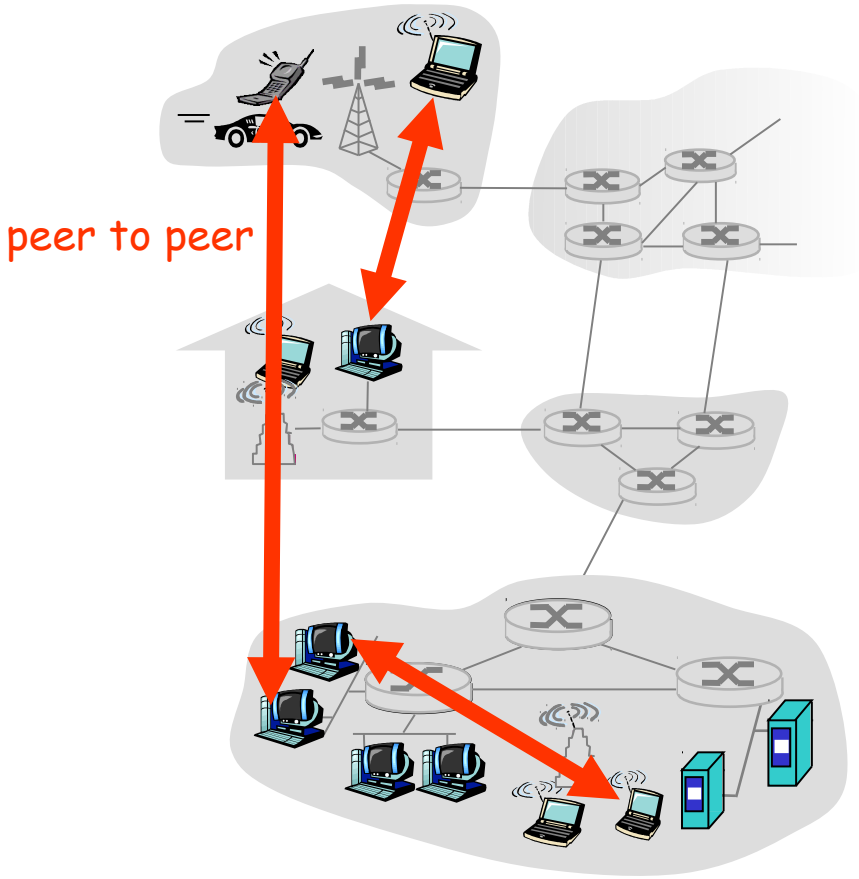
    `www.ibm.com` è in realtà `servereast.backup2.ibm.com`
  - **value** è il nome canonico
- Type=MX
  - **value** è il nome del server di posta associato a **name**

Capitolo 2: Livello di applicazione

- 2.1 Principi delle applicazioni di rete
- 2.2 Web e HTTP
- 2.4 Posta elettronica
  - SMTP, POP3, IMAP
- 2.5 DNS
- 2.6 Condivisione di file P2P

Architettura P2P pura

- non c'è un server sempre attivo
- coppie arbitrarie di host (peer) comunicano direttamente tra loro
- i peer non devono necessariamente essere sempre attivi, e cambiano indirizzo IP
- Tre argomenti chiave:
  - Distribuzione di file
  - Ricerca informazioni
  - Caso di studio: Skype

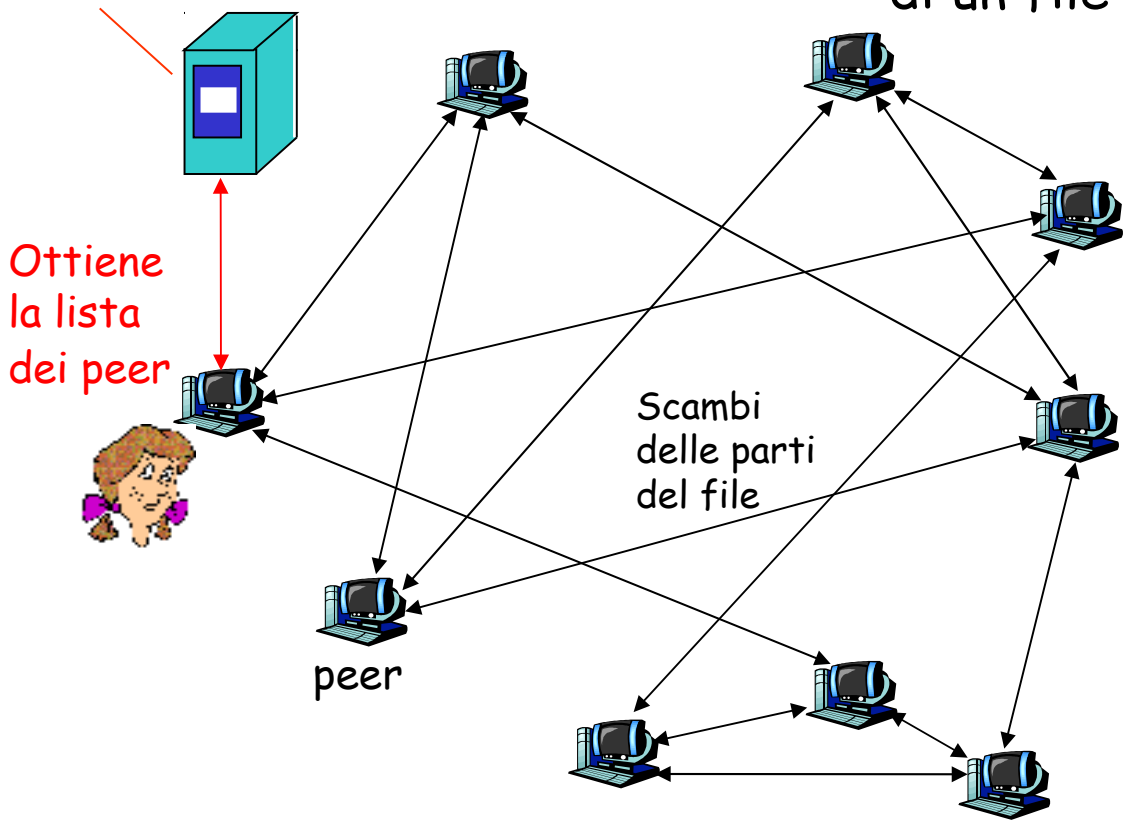


Distribuzione di file: BitTorrent

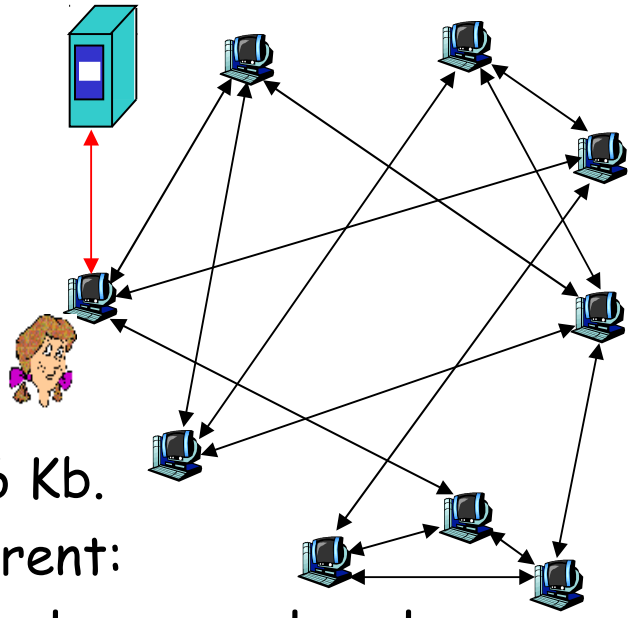
- Distribuzione di file P2P

tracker: tiene traccia dei peer che partecipano

torrent: gruppo di peer che si scambiano parti di un file



BitTorrent (1)



- Il file viene diviso in parti (*chunk*) da 256 Kb.
- Quando un peer entra a far parte del torrent:
  - non possiede nessuna parte del file, ma le accumula col passare del tempo
  - si registra presso il tracker per avere la lista dei peer, e si collega ad un sottoinsieme di peer vicini ("neighbors")
- Mentre effettua il download, il peer carica le sue parti su altri peer.
- I peer possono entrare e uscire a piacimento dal torrent
- Una volta ottenuto l'intero file, il peer può lasciare il torrent (egoisticamente) o (altruisticamente) rimanere collegato.

BitTorrent (2)

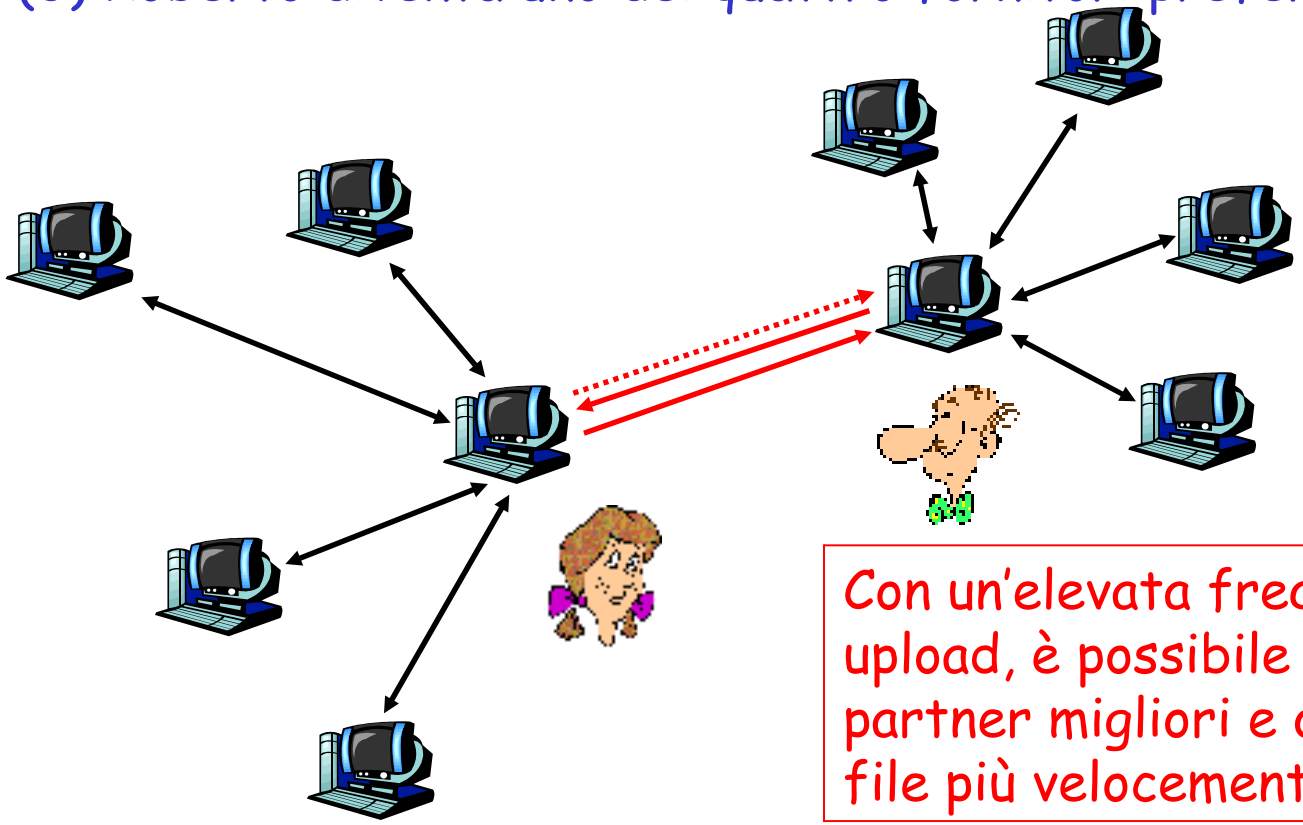
- In un dato istante, peer diversi hanno differenti sottoinsiemi del file
- periodicamente, un peer (Alice) chiede a ciascun vicino la lista dei chunk che possiede
- Alice invia le richieste per le sue parti mancanti:
  - Adotta la tecnica del *rarest first*
- Alice invia le sue parti a quattro vicini, quelli che attualmente le stanno inviando i propri chunk alla frequenza più alta
  - i 4 favoriti vengono rivalutati ogni 10 secondi
- ogni 30 secondi seleziona casualmente un altro peer, e inizia a inviargli chunk
  - Il peer appena scelto può entrare a far parte dei top 4
  - A parte i "top 4" e il "nuovo entrato", gli altri peer sono "soffocati", cioè non ricevono nulla.

BitTorrent: occhio per occhio

(1) Alice casualmente sceglie Roberto

(2) Alice diventa uno dei quattro fornitori preferiti di Roberto; Roberto ricambia

(3) Roberto diventa uno dei quattro fornitori preferiti di Alice



Con un'elevata frequenza di upload, è possibile trovare i partner migliori e ottenere il file più velocemente!

P2P: ricerca di informazioni

Indice nei sistemi P2P: corrispondenza tra le informazioni e la loro posizione negli host

File sharing (es. e-mule)

- L'indice tiene traccia dinamicamente della posizione dei file che i peer condividono.
- I peer comunicano all'indice ciò che possiedono.
- I peer consultano l'indice per determinare dove trovare i file.

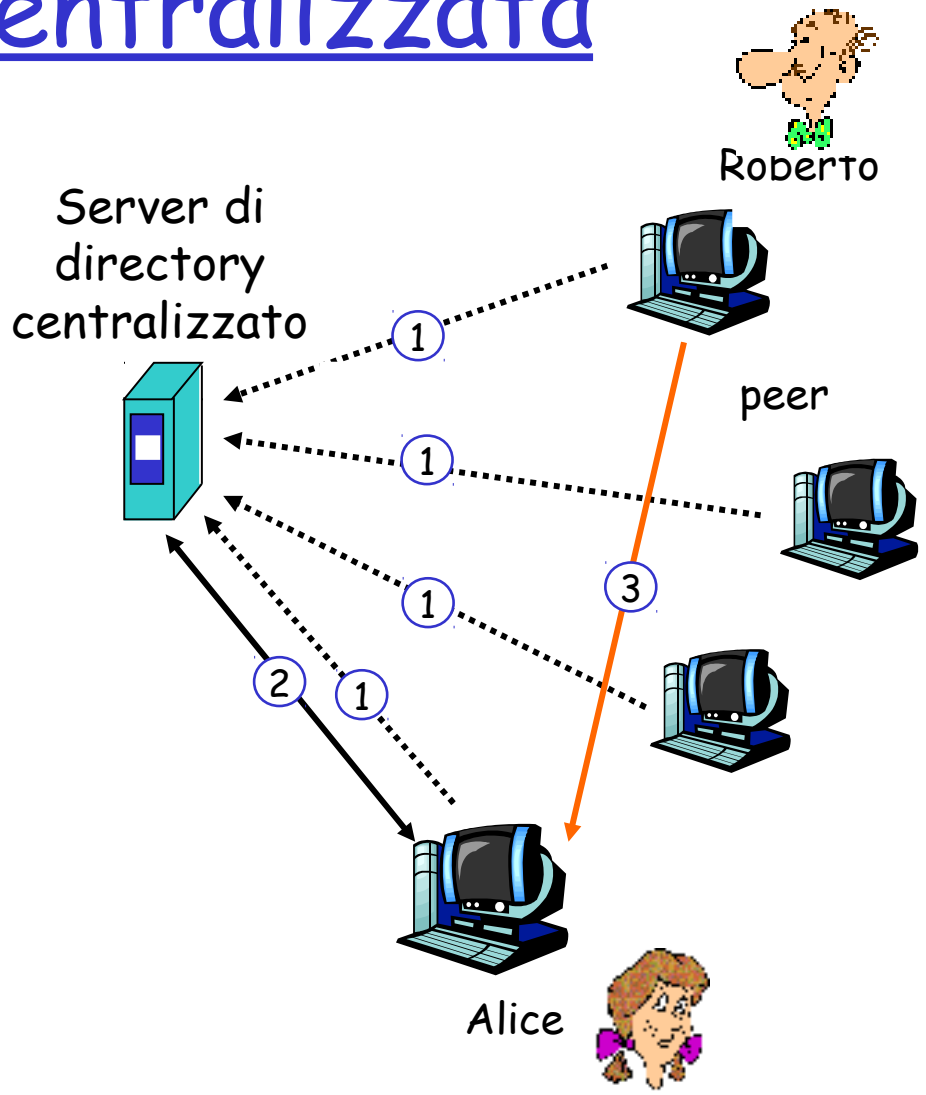
Messaggeria istantanea

- L'indice crea la corrispondenza tra utenti e posizione.
- Quando l'utente lancia l'applicazione, informa l'indice della sua posizione
- I peer consultano l'indice per determinare l'indirizzo IP dell'utente.

P2P: directory centralizzata

Progetto originale di "Napster"

1) quando il peer si collega, informa il server centrale:
   - indirizzo IP
   - contenuto
2) Alice cerca la canzone "Hey Jude"
3) Alice richiede il file a Roberto



P2P: problemi con la directory centralizzata

- Unico punto di guasto
- Collo di bottiglia per le prestazioni
- Violazione del diritto d'autore

Il trasferimento dei file è distribuito, ma il processo di localizzazione è fortemente centralizzato

Query flooding

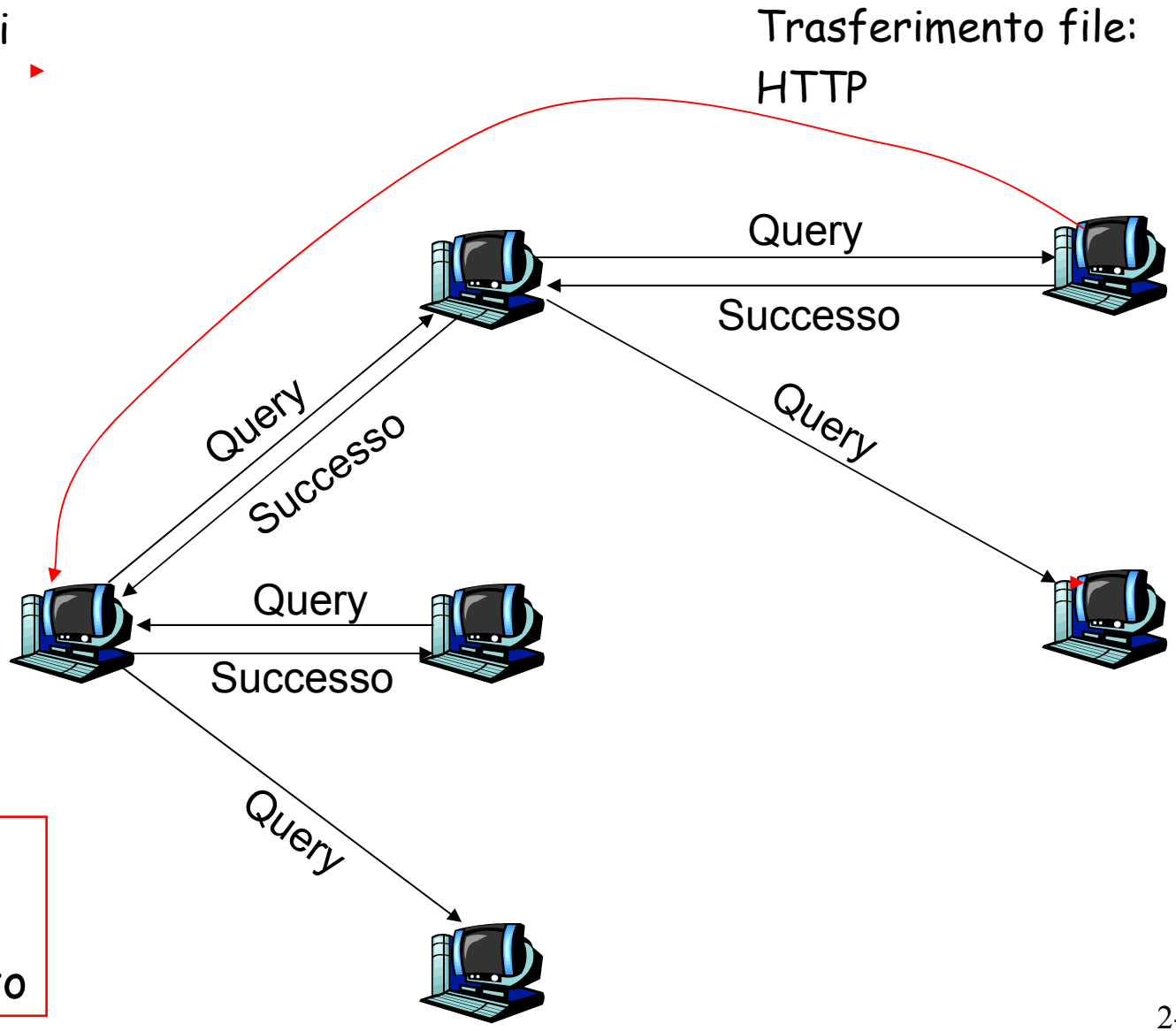
- Completamente distribuito
  - nessun server centrale
- Protocollo di pubblico dominio usato da Gnutella
- Ciascun peer indicizza i file che rende disponibili per la condivisione (e nessun altro)

Rete di copertura: grafo

- Arco tra i peer X e Y se c'è una connessione TCP
- Tutti i peer attivi e gli archi formano la rete di copertura
- Un arco è un collegamento virtuale e *non* fisico
- Un dato peer sarà solitamente connesso con meno di 10 peer vicini nella rete di copertura

Query flooding

- Il messaggio di richiesta è trasmesso sulle connessioni TCP esistenti
- Il peer inoltra il messaggio di richiesta
- Il messaggio di successo è trasmesso sul percorso inverso



Scalabilità:
query flooding
a raggio limitato

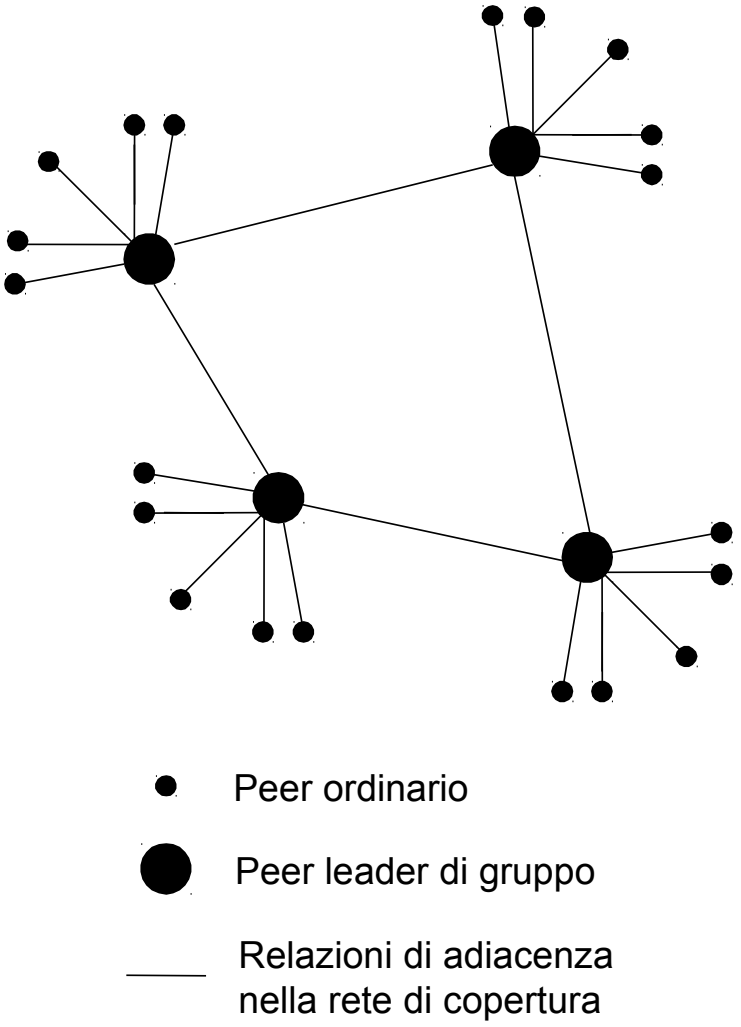
Gnutella: unione di peer

1. Per unire il peer Alice alla rete, bisogna trovare qualche altro peer della rete Gnutella: usate la lista dei peer candidati
2. Alice tenta in sequenza di impostare una connessione TCP con i peer della lista finché non stabilisce una connessione con Roberto
3. Flooding: Alice invia un messaggio Ping a Roberto; Roberto inoltra il messaggio Ping ai suoi vicini (che a loro volta lo inviano ai propri vicini…)
4. Tutti i peer che ricevono il messaggio Ping rispondono ad Alice con un messaggio Pong
5. Alice riceve molti messaggi Pong. Quindi può impostare delle connessioni TCP addizionali

Distacco dei peer: consultate il problema alla fine del capitolo!

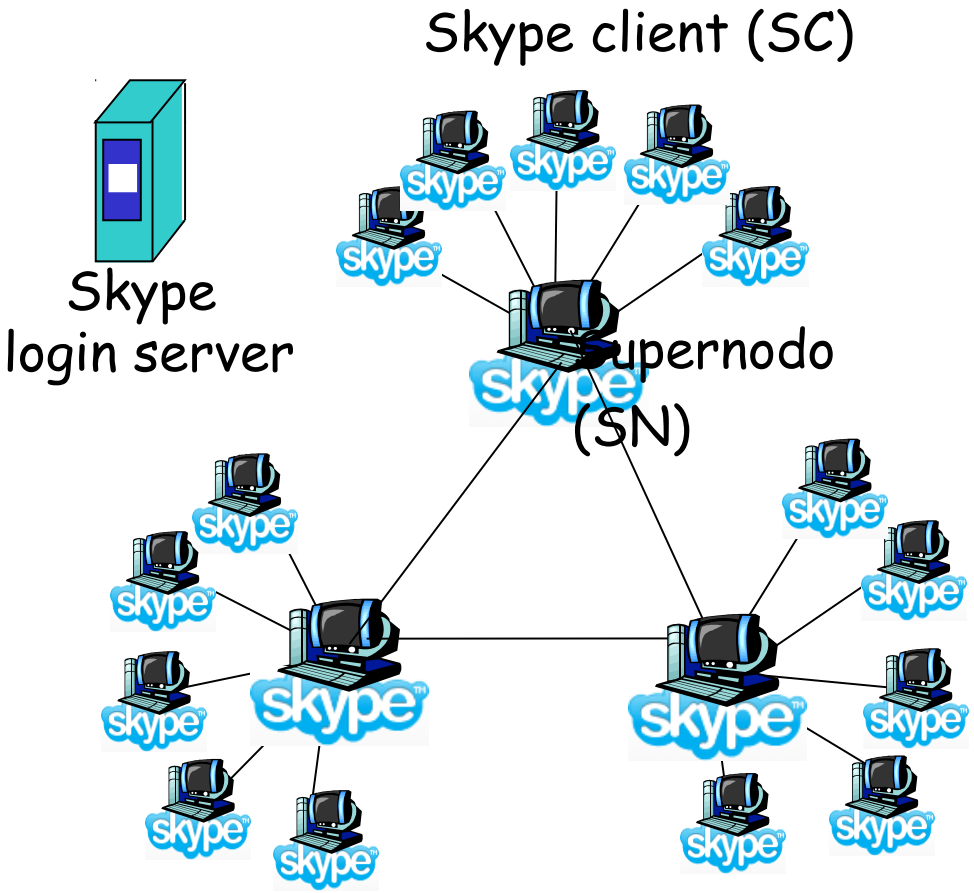
Copertura gerarchica

- La copertura gerarchica combina le migliori caratteristiche di indice centalizzato e query flooding
- Ogni peer è un leader di gruppo o è assegnato a un leader di gruppo
  - Connessione TCP tra peer e il suo leader di gruppo
  - Connessioni TCP tra qualche coppia di leader di gruppo
- Il leader di gruppo tiene traccia del contenuto di tutti i suoi figli.



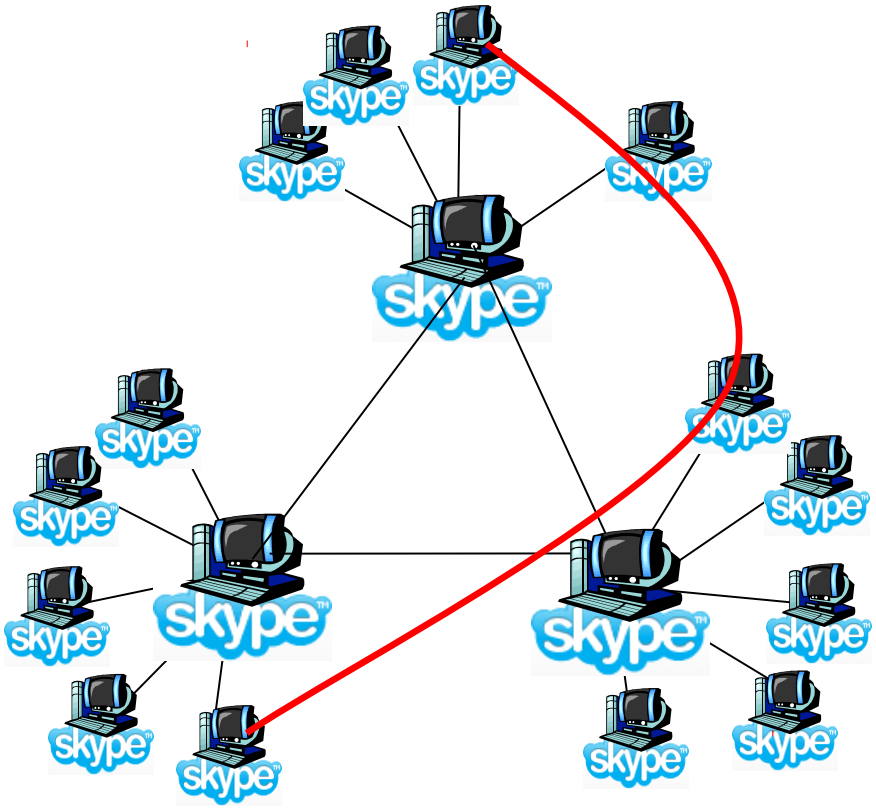
Caso di studio P2P: Skype

- intrinsecamente P2P: coppie di utenti comunicano tra loro
- Protocollo proprietario (dedotto mediante reverse engineering)
- Copertura gerarchica con i supernodi
- L'indice crea corrispondenza tra nomi utente e indirizzi IP



Peer e relay

- Si pone un problema quando sia Alice che Roberto hanno NAT.
  - NAT evita che un host al di fuori della rete domestica crei una connessione con un host all'interno di questa
- Soluzione
  - Usando il supernodo di Alice e Roberto, si sceglie un relay
  - Ciascun peer inizia la sessione con il relay.
  - I peer ora comunicano con NAT attraverso il relay



Capitolo 2: Riassunto

Lo studio delle applicazioni di rete adesso è completo!

- Architetture delle applicazioni
  - client-server
  - P2P
  - ibride
- Requisiti dei servizi delle applicazioni:
  - affidabilità, ampiezza di banda, ritardo
- Modello di servizio di trasporto di Internet
  - orientato alle connessioni, affidabile: TCP
  - inaffidabile, datagrammi: UDP
- Protocolli specifici:
  - HTTP
  - SMTP, POP, IMAP
  - DNS
  - P2P: BitTorrent, Skype

Riassunto

Molto importante: conoscere i *protocolli*

- Tipico scambio di messaggi di richiesta/risposta:
  - il client richiede informazioni o servizi
  - il server risponde con dati e codici di stato
- Formati dei messaggi:
  - intestazioni: campi che forniscono informazioni sui dati
  - dati: informazioni da comunicare
- Controllo o messaggi di dati
  - in banda, fuori banda
- Architettura centralizzata o decentralizzata
- Protocollo senza stato o con stato
- Trasferimento di messaggi affidabile o inaffidabile
- "Complessità nelle parti periferiche della rete"