

Corso di Laurea in Ingegneria Informatica



**Corso di Reti di Calcolatori
(a.a. 2010/11)**

Roberto Canonico (roberto.canonico@unina.it)

Giorgio Ventre (giorgio.ventre@unina.it)

**Router NAT
Applicazioni P2P**

10 novembre 2010

**I lucidi presentati al corso sono uno strumento didattico
che NON sostituisce i testi indicati nel programma del corso**

Nota di copyright per le slide COMICS



Nota di Copyright

Questo insieme di trasparenze è stato ideato e realizzato dai ricercatori del Gruppo di Ricerca COMICS del Dipartimento di Informatica e Sistemistica dell'Università di Napoli Federico II. Esse possono essere impiegate liberamente per fini didattici esclusivamente senza fini di lucro, a meno di un esplicito consenso scritto degli Autori. Nell'uso dovranno essere esplicitamente riportati la fonte e gli Autori. Gli Autori non sono responsabili per eventuali imprecisioni contenute in tali trasparenze né per eventuali problemi, danni o malfunzionamenti derivanti dal loro uso o applicazione.

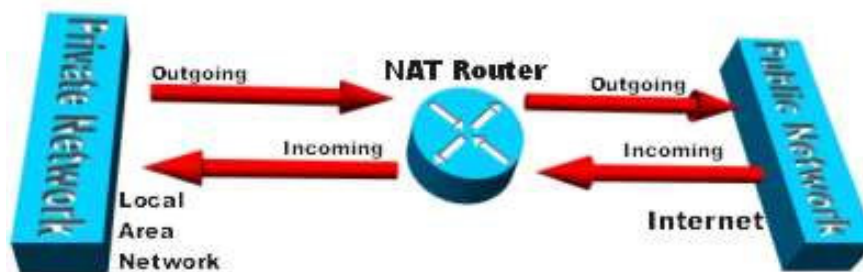
Autori:

Simon Pietro Romano, Antonio Pescapè, Stefano Avallone,
Marcello Esposito, Roberto Canonico, Giorgio Ventre

NAT: Network Address Translation



- Network Address Translation (RFC 1631) consente ad un dispositivo di agire come intermediario tra Internet (rete pubblica) e una rete privata
- In questo modo, un unico indirizzo IP può rappresentare un intero gruppo di computer

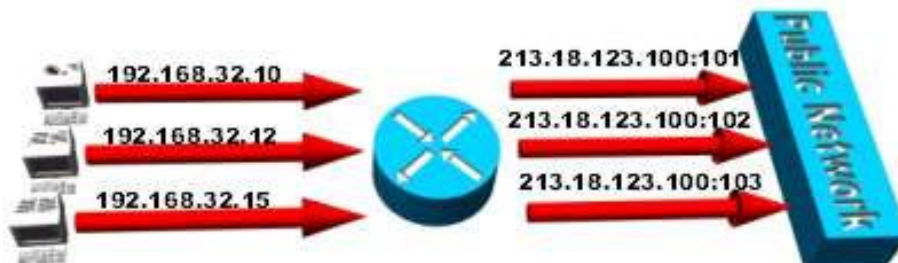


3

NAT



- L'uso più comune del NAT è quello di mappare un insieme di indirizzi privati su di un unico indirizzo pubblico, utilizzando differenti porti per mantenere traccia delle diverse connessioni



4

NAT



- Quando il router riceve un pacchetto inviato da un computer della rete privata ad un computer esterno, salva in una tabella l'indirizzo e il porto del mittente, oltre ai nuovi valori che esso assegna
- Tale tabella viene consultata anche quando il router riceve un pacchetto dal computer destinazione

Source Computer	Source Computer's IP Address	Source Computer's Port	NAT Router's IP Address	NAT Router's Assigned Port Number
A	192.168.32.10	400	215.37.32.203	1
B	192.168.32.13	50	215.37.32.203	2
C	192.168.32.15	3750	215.37.32.203	3
D	192.168.32.18	206	215.37.32.203	4

5

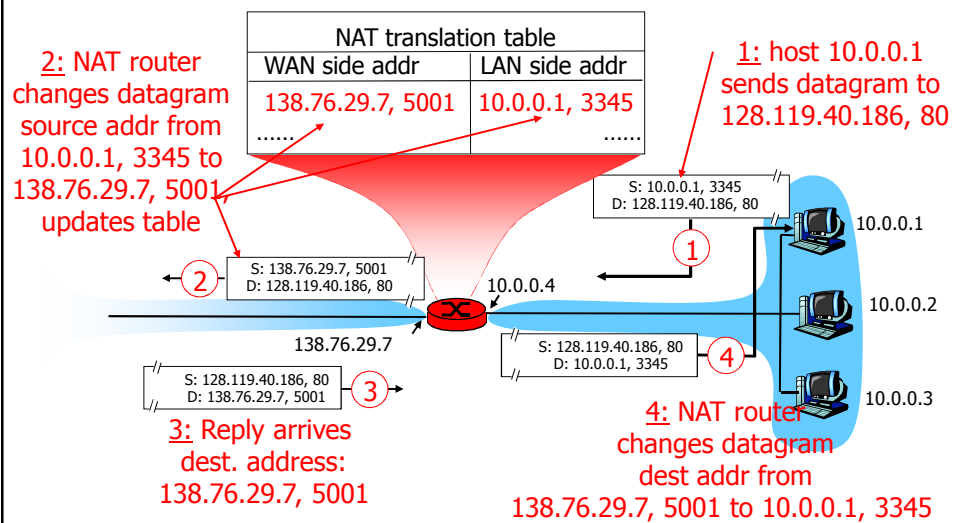
NAT: Network Address Translation



Implementation: NAT router must:

- *outgoing datagrams: replace* (source IP address, port #) of every outgoing datagram to (NAT IP address, new port #)
 - . . . remote clients/servers will respond using (NAT IP address, new port #) as destination addr.
- *remember (in NAT translation table)* every (source IP address, port #) to (NAT IP address, new port #) translation pair
- *incoming datagrams: replace* (NAT IP address, new port #) in dest fields of every incoming datagram with corresponding (source IP address, port #) stored in NAT table

NAT: un esempio



7

NAT: Network Address Translation



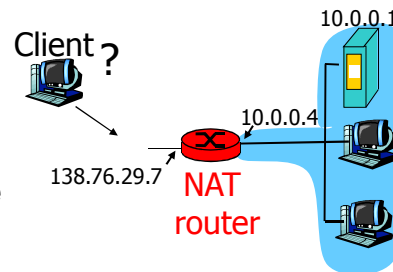
- 16-bit port-number field:
 - 60,000 simultaneous connections with a single LAN-side address!
- NAT is controversial:
 - routers should only process up to layer 3
 - violates end-to-end argument
 - NAT possibility must be taken into account by app designers, eg, P2P applications
 - address shortage should instead be solved by IPv6

Network Layer

4-8

NAT traversal problem

- client wants to connect to server with address 10.0.0.1
 - server address 10.0.0.1 local to LAN (client can't use it as destination addr)
 - only one externally visible NATted address: 138.76.29.7
- solution 1: statically configure NAT to forward incoming connection requests at given port to server
 - e.g., (138.76.29.7, port 2500) always forwarded to 10.0.0.1 port 25000

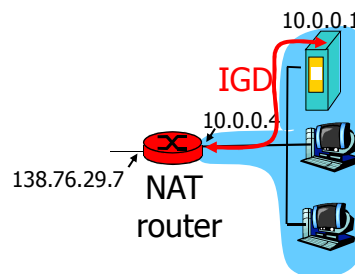


Network Layer

4-9

NAT traversal problem

- solution 2: Universal Plug and Play (UPnP) Internet Gateway Device (IGD) Protocol. Allows NATted host to:
 - ❖ learn public IP address (138.76.29.7)
 - ❖ add/remove port mappings (with lease times)
- i.e., automate static NAT port map configuration



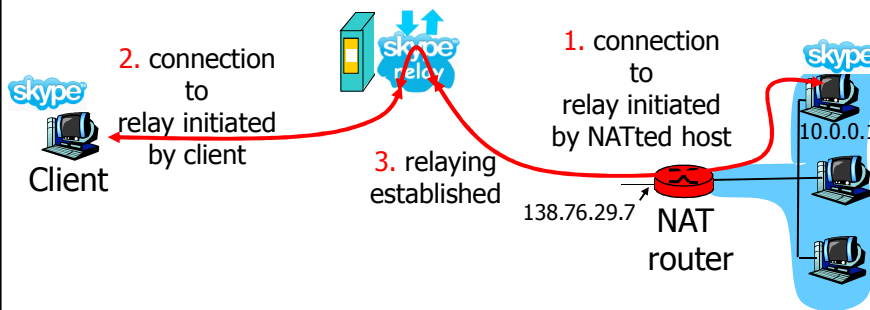
Network Layer

4-10



NAT traversal problem

- solution 3: relaying (used in Skype)
 - NATed client establishes connection to relay
 - External client connects to relay
 - relay bridges packets between to connections



Network Layer

4-11



Chapter 4: Network Layer

- 4.1 Introduction
- 4.2 Virtual circuit and datagram networks
- 4.3 What's inside a router
- **4.4 IP: Internet Protocol**
 - Datagram format
 - IPv4 addressing
 - **ICMP**
 - IPv6
- 4.5 Routing algorithms
 - Link state
 - Distance Vector
 - Hierarchical routing
- 4.6 Routing in the Internet
 - RIP
 - OSPF
 - BGP
- 4.7 Broadcast and multicast routing

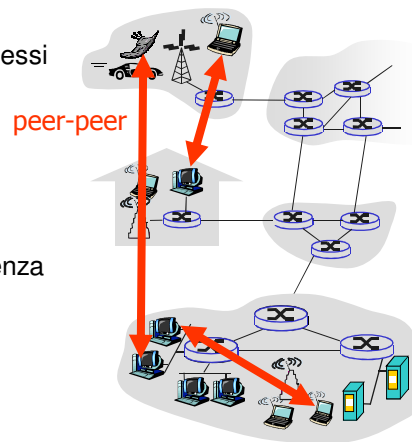
Network Layer

4-12

Applicazioni p2p



- P2P: un modello alternativo al client/server
- Non esistono server sempre connessi (always-on server)
- Non esiste una differenziazione funzionale tra client e server
- End-system (peer) comunicano direttamente
- I peer sono connessi ad intermittenza e cambiano il proprio IP

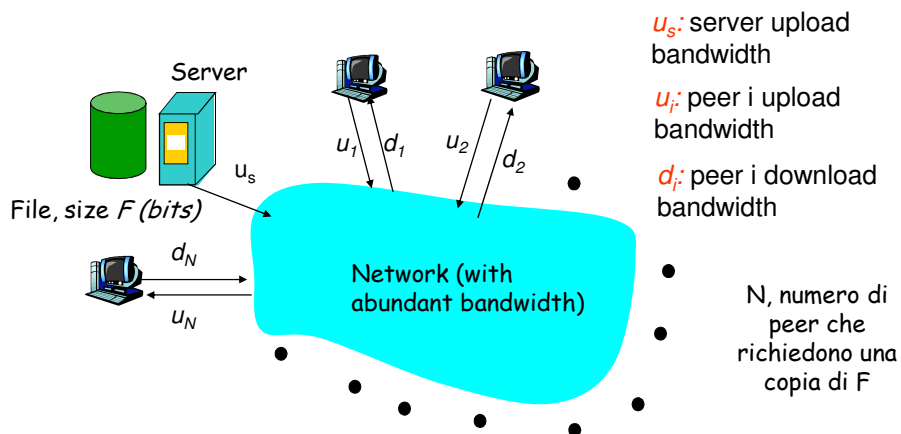


13

File Distribution: Server-Client vs P2P



Quanto tempo serve per distribuire un file da un server ad N peer (tempo di distribuzione, D)?

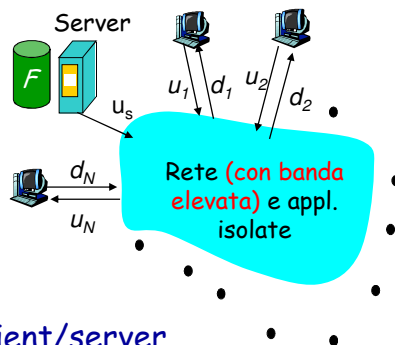


14

File distribution: caso "server-client"



- Il server invia in sequenza N copie del file:
 - tempo richiesto: NF/u_s
- Il client i impiega un tempo pari a F/d_i per il download



Approccio client/server

$$\text{Tempo per distribuire } F \text{ bit a } N \text{ client} = D_{cs} = \max \{ NF/u_s, F/\min(d_i) \}$$

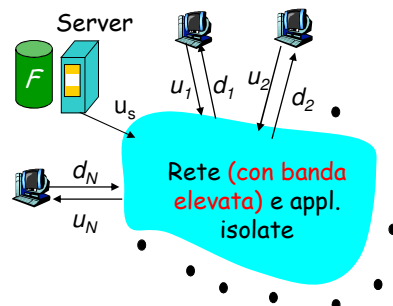
per N elevato, il termine NF/u_s è dominante: incremento lineare al crescere di N

15

File distribution: caso P2P



- Dipende dal numero di peer coinvolti...
- Il server deve inviare almeno una copia. Tempo richiesto: F/u_s
- Il client i impiega un tempo pari a F/d_i per il download (F/d_{\min})
- NF bit totali devono essere 'scaricati': velocità massima di upload: $u_s + \sum u_i$



Approccio p2p

$$D_{P2P} = \max \{ F/u_s, F/\min(d_i), NF/(u_s + \sum u_i) \}$$

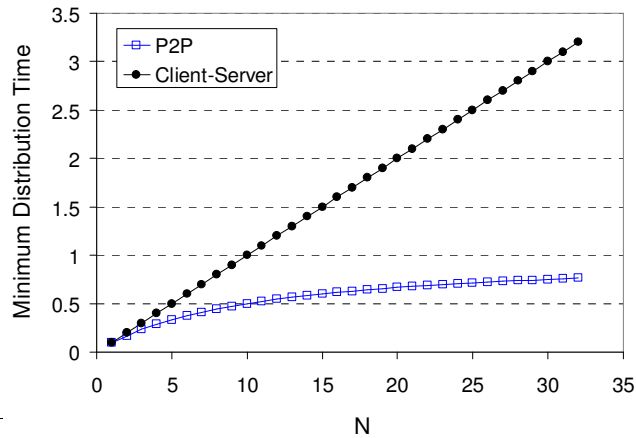
16

Server-client vs P2P: confronto di prestazioni



Ipotesi:

- Tutti i peer hanno lo stesso rate di upload = u
- $F/u = 1$ ora, $u_s = 10u$, $d_{\min} \geq u_s$



17

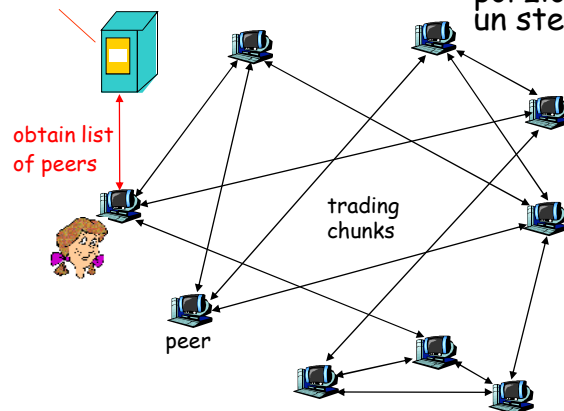
File distribution: BitTorrent



- P2P file distribution

tracker: tiene traccia dei peer che compongono un "torrente"

torrente: gruppo di peer che si scambiano porzioni (chunk) di un stesso file

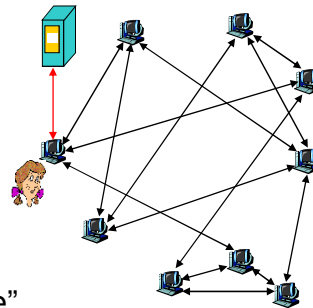


18

BitTorrent (1/3)



- Il file è diviso in *chunk* di 256KB.
- Quando un peer si aggiunge ad un “torrente”:
 - si registra presso il tracker per avere la lista dei peer e si connette ad un sottoinsieme di tali peer (“neighbors”)
 - non possiede chunk, ma ne accumulerà nel tempo
- Durante il download, il peer esegue l’upload di chunk verso altri peer
- I peer possono attivarsi e disattivarsi dinamicamente
- Una volta scaricato l’intero file, il peer può (egoisticamente) abbandonare, o (altruisticamente) rimanere nel “torrente”



19

BitTorrent (2/3)



1) Prelievo di chunk

- Peer differenti possiedono differenti sottoinsiemi di chunk del file
- Periodicamente, un peer chiede a tutti i ‘neighbor’ la lista dei chunk in loro possesso
- Tale peer invia richieste per i propri chunk mancanti
 - tecnica ‘*rarest first*’

2) Invio di chunks: ‘tit-for-tat’

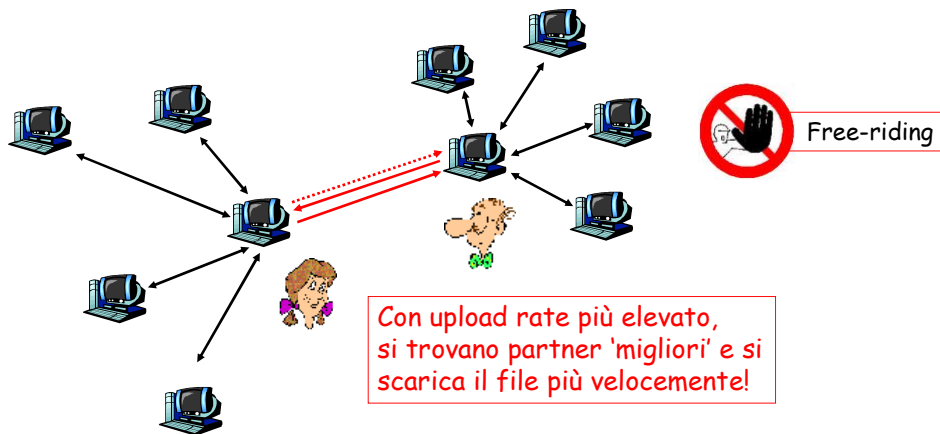
- L’idea di fondo è quella di dare priorità a chi fornisce dati al rate più alto.
- Un peer invia chunk ai 4 neighbor attualmente più veloci (che gli inviano chunk al rate più elevato)
 - i ‘top 4’ sono ricalcolati ogni 10 secondi
- Ogni 30 secondi: si seleziona in maniera casuale un altro peer, e si inizia ad inviargli chunk
 - il peer appena scelto può essere aggiunto ai ‘top 4’
 - “optimistically unchoke”

20

BitTorrent: tit-for-tat



- (1) Alice effettua l' "unchoke" ottimistico di Bob
- (2) Alice diventa uno dei 'top-four provider' di Bob, il quale 'ricambia'
- (3) Bob diventa uno dei 'top-four provider' di Alice



21

P2P file sharing

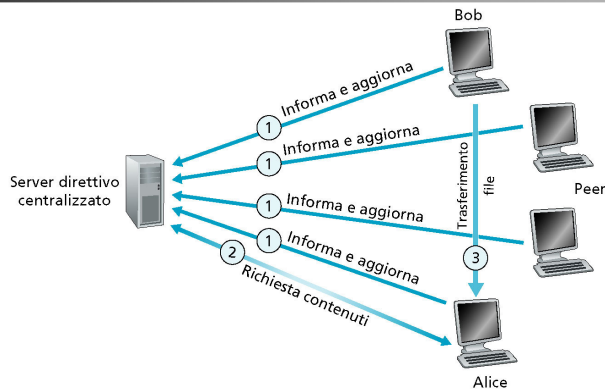


- Alice è connessa ad Internet e ha lanciato la sua applicazione di file sharing P2P sul suo PC
- Alice non è perennemente connessa ad Internet, non ha un hostname, l'IP cambia ad ogni connessione
- Alice chiede "Hey Jude"
- L'applicazione mostra tutti gli altri peer che hanno una copia di Hey Jude
- Alice sceglie uno dei peer, Bob
- Il file è trasferito (copiato) dal PC di Bob a quello di Alice usando HTTP
- Mentre Alice effettua il download, altri peer possono prendere file da Alice.
- Alice è quindi sia un client che un server

Tutti i peer sono anche server: architettura altamente scalabile

22

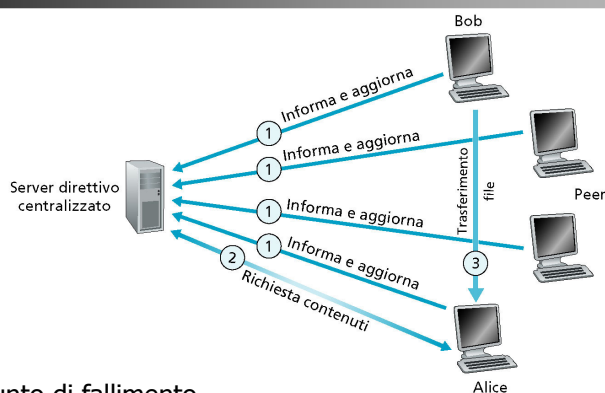
P2P con directory centralizzata:



- Quando un peer si connette alla rete si collega ad un server centralizzato fornendo:
 - Il proprio indirizzo IP
 - Il nome degli oggetti resi disponibili per la condivisione
- In server in questo modo raccoglie le info sui peer attivi e le aggiorna dinamicamente

23

P2P con directory centralizzata:



- Singolo punto di fallimento
- Collo di bottiglia per le prestazioni
- Violazione del diritto di autore
- **Il trasferimento dei file è decentralizzato, ma la localizzazione dei contenuti è pesantemente centralizzata !!!**

24

P2P con directory decentralizzata:



gnutella.com

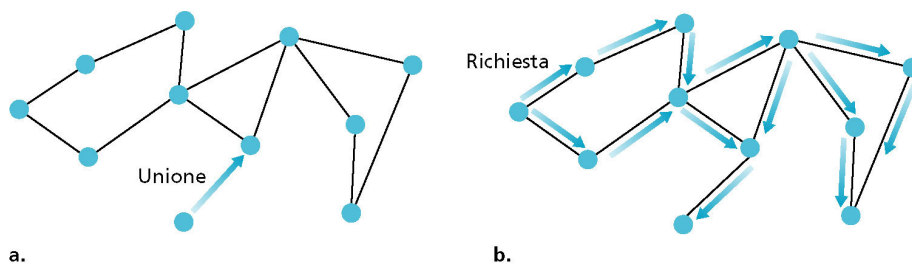
- Architettura completamente distribuita (senza server centralizzati)
- Si realizza un'architettura di rete sovrapposta (overlay network, fatta da connessioni TCP in corso)
- L'overlay network ha una struttura paritetica.
- Nonostante la rete possa avere centinaia di migliaia di partecipanti, ogni peer è connesso al max a 10 altri peer nella overlay.
- Due problematiche:
 - come viene costruita e gestita la rete di peer
 - come un peer localizza un contenuto

25

P2P con directory decentralizzata:



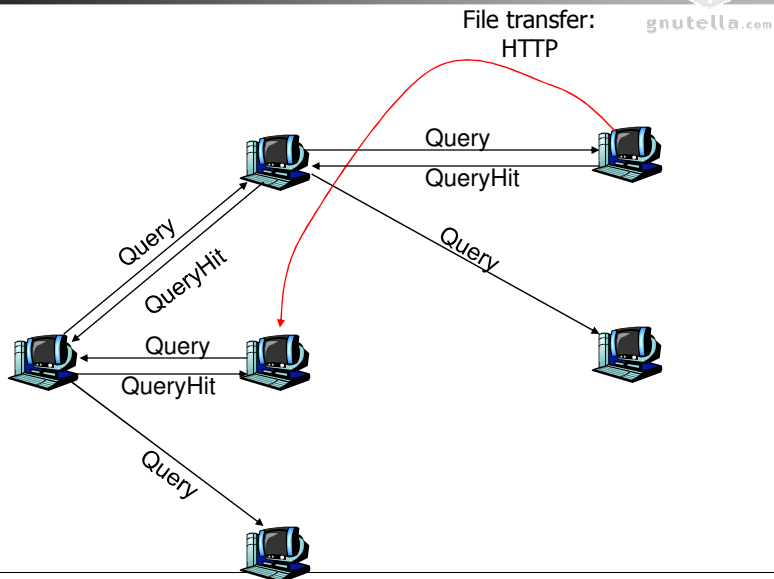
gnutella.com



- I *peer*, una volta unitisi alla rete, inviano richieste mediante la tecnica del *flooding* (inondazione), query flooding
 - Gnutella Query e QueryHit
- Query flooding a raggio limitato
 - scope del messaggio, 7 ad esempio
 - pro e contro

26

P2P con directory decentralizzata:



27

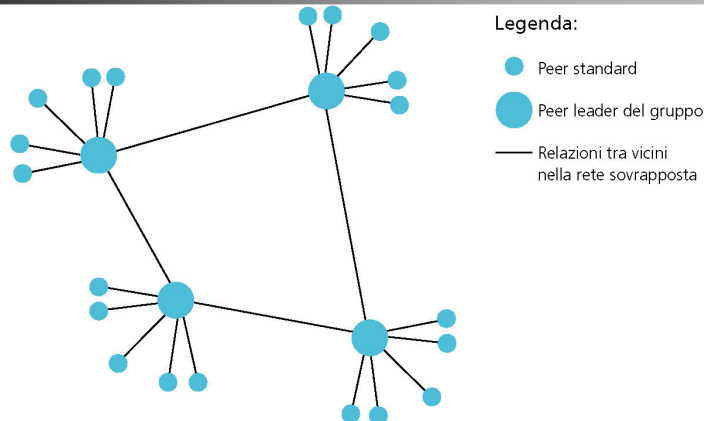
P2P con directory decentralizzata:



1. Il peer X deve trovare altri peer già parte della overlay: mantiene una lista di IP o contatta un sito Gnutella contenente la lista
2. Dopo l'accesso alla lista, X tenta di impostare una connessione TCP con i peer della lista; quando si connette a Y si ferma
3. X spedisce a Y un ping Gnutella; Y lo inoltra finchè il contatore non si azzerà
4. Tutti i peer che ricevono un messaggio ping, rispondono con un pong: esso contiene l'indirizzo di chi ha inviato il pong, il numero di file in condivisione, la dimensione totale
5. Quando X riceve i messaggi di pong, avendo l'IP, può impostare una connessione TCP con alcuni di essi...
6. Ci possono essere più fasi di bootstrap in parallelo

28

P2P con directory decentralizzata:



- Sfrutta le caratteristiche positive di Napster e Gnutella
- Ogni peer è associato ad un group leader (mini hub) che è esso stesso un peer
- Un group leader memorizza le informazioni in condivisione dei "figli"
- Ogni group leader è in grado di interrogare altri group leader
 - query flooding applicato alla rete dei group leader

29

P2P con directory decentralizzata:



- Fase di bootstrap: un peer che si connette deve essere associato ad un group leader o deve essere designato group leader
- L'overlay è costituita da connessioni TCP tra peer e group leader e tra coppie di group leader
- Ogni file possiede un identificatore hash e un descrittore
- I peer (client) spediscono le query al proprio group leader; quest'ultimo risponde per ogni richiesta con l'indirizzo IP del detentore della risorsa, l'hash, e dei metadati associati alla risorsa. Il group leader inoltra sia le richieste sia le eventuali risposte da parte di altri group leader
- Il peer (client) seleziona la risorsa file per il download e invia una richiesta HTTP al detentore della risorsa usando l'hash come identificatore
- Tecniche per migliorare le prestazioni
 - Accodamento delle richieste e limitazione del numero degli upload simultanei
 - Priorità di incentivo
 - Downloading parallelo di parti dello stesso file da più utenti

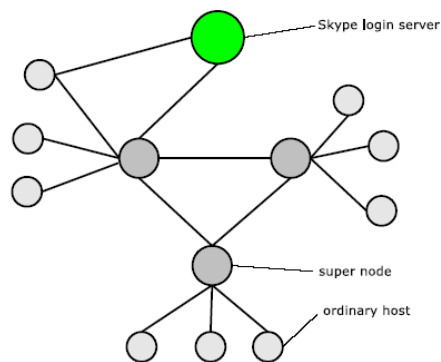
30



- Skype è una applicazione P2P VoIP sviluppata da KaZaa nel 2003
 - Supporta anche instant messaging e conferencing.
- Il protocollo è proprietario
- Skype usa una rete overlay, con tre tipi di host
 - Host ordinari, Skype users
 - Super nodi, Skype users con sufficiente CPU, memoria, banda...
 - Server di login per l'autenticazione



- Ciascun client Skype mantiene una lista di indirizzi IP di super nodi conosciuti. Inizialmente questa lista è vuota.
- Un client Skype si connette alla rete attraverso un super nodo
- I super nodi sono responsabili della localizzazione degli utenti, del routing delle chiamate, del mantenimento delle informazioni circa gli host connessi alla rete Skype



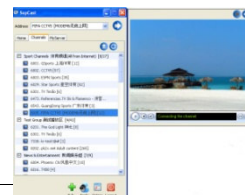
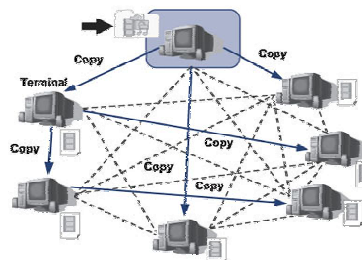


- Connessione ai super nodi:
 - Primo login
 - Alla prima esecuzione dopo l'installazione un client Skype comunica con il server Skype (skype.com)
 - Durante la comunicazione, la cache dell'host è popolata di 7 indirizzi IP di super nodi da usare per il bootstrap
 - A questo punto l'host può contattare uno di essi per il join
 - Selezionato il super nodo per il join, parte la fase di autenticazione con user name e password con il server Skype
 - L'host viene periodicamente aggiornato con indirizzi IP di nuovi super nodi
 - Login successivi
 - Per i login successivi un client sceglie uno degli indirizzi dei super nodi e stabilisce la connessione



Keywords:

- *Chunk*
- *Peer-list*
- *Buffer-map*
- Ogni peer è sia un viewer che un server.
- Alcuni viewer ricevono il flusso direttamente dalla sorgente originale (server IPTv); altri lo ricevono attraverso altri peer.



P2P: approfondimenti



- DHT (Distributed Hash Table)
- p2p di seconda generazione
- ...
- **Ai corsi di Reti di Calcolatori II e Applicazioni Telematiche**