Esercizio Trasferimento File e Multicast

Laboratorio di Programmazione di Rete A

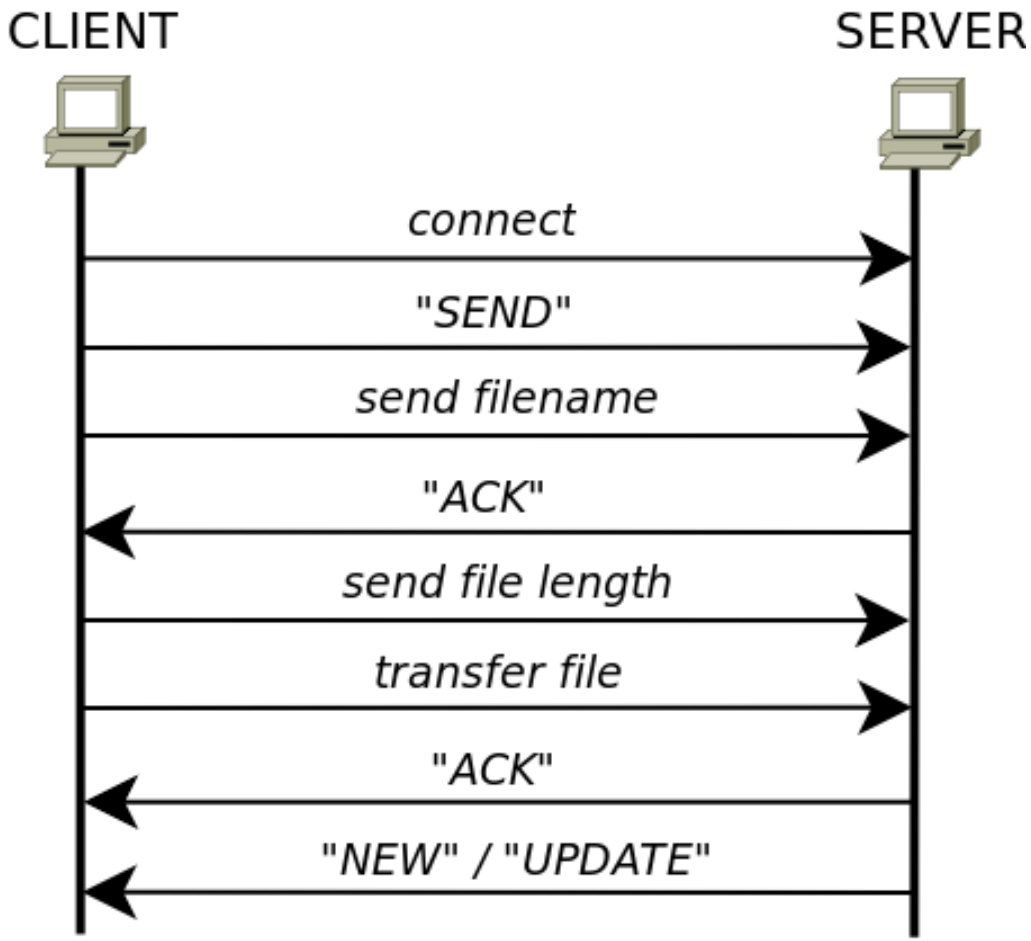
Esercitazione di Laboratorio

14/11/2007

Esercizio Trasferimento File

- Sviluppare un'applicazione distribuita che offra il servizio di trasferimento di file tra un RemoteCopyClient e un RemoteCopyServer tramite TCP.
- Il client apre una connessione verso il server. Poi richiede all'utente il nome del file da trasferire.
- Se il file esiste, il client invia al server il nome del file da trasferire, seguito dal contenuto del file, quindi torna a proporre una nuova richiesta di trasferimento all'utente e così via.
- Il server, riceve una richiesta di connessione e salva il file richiesto nella directory dell'applicazione. Alla fine del download, invia al client l'esito dell'operazione: UPDATE, se il file esisteva già ed è stato sovrascritto, oppure NEW, se è stato creato un nuovo file.

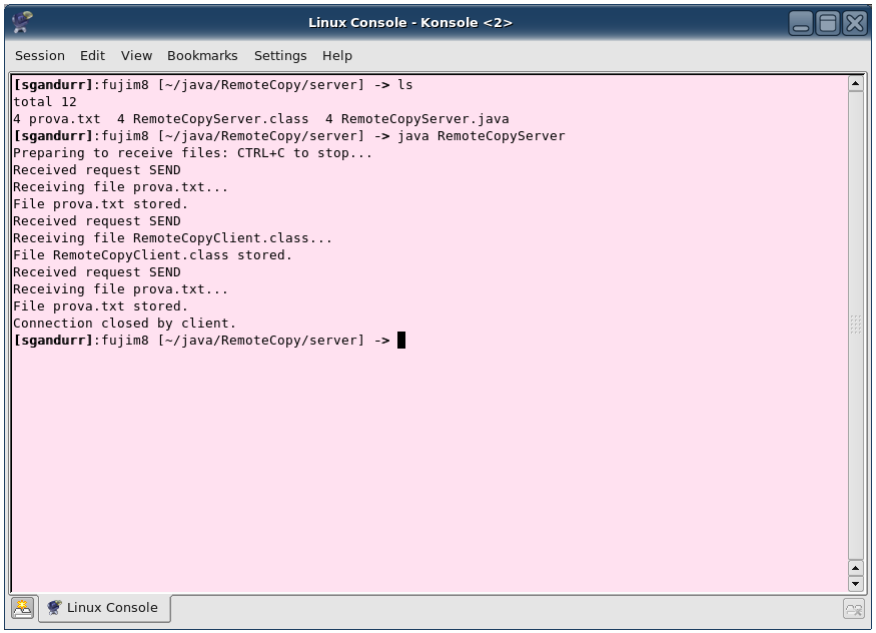
Diagramma Interazione Client/Server

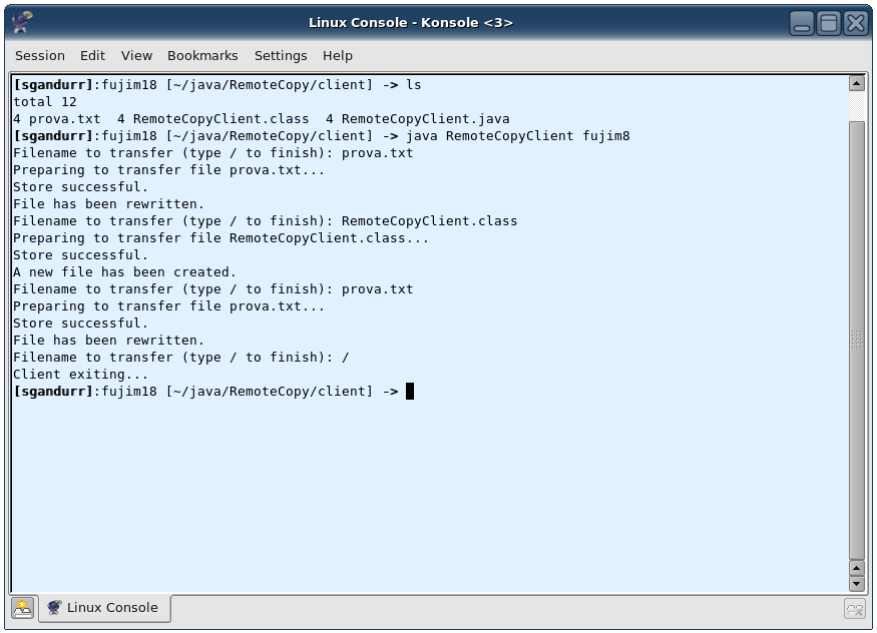


Esempio Utilizzo

- Compilare i due file `RemoteCopyClient.java` e `RemoteCopyServer.java` su due directory diverse, ad es:
  - `~/java/RemoteCopy/Client`
  - `~/java/RemoteCopy/Server`
- Supponiamo il server stia su `fujim8`, il client su `fujim18`. Connettersi in ssh ai due host.
- Nella directory del server, eseguire `java RemoteCopyServer`.
- Nella directory del client, eseguire `java RemoteCopyClient fujim8`.
- Dal client, digitare il nome dei file da trasferire, uno alla volta.
- Per terminare il client digitare "/".
- Per terminare il server digitare "CTRL + C".

Esempio Utilizzo





Il Server

- Versione semplificata senza thread. Solo un client alla volta può inviare file.
- Il codice del server si compone di due cicli: uno per accettare le connessioni, uno per ricevere i files.

```
Socket clientSocket=null;
while(true)
{...
   try
   {
      clientSocket = serverSocket.accept();
   ...
   }
   ...
   boolean finished = false
   while(!finished)
   {/*riceve i files*/
   }/*finito di ricevere i file da un client*/
   ...
}/*chiusura server*/
```

Il Client

- Il client chiede all'utente, all'interno di un ciclo while, il nome dei file da trasferire.
- Ogni volta che invia un file, lo invia a blocchi (chunks).

```
public void sendInChunks(DataInputStream input, int totalSize,
                         int chunksize) throws IOException
{
   //copy bytes from input to output in chunks
   byte bytes[] = new byte[chunksize];
   int writecount;
   while(totalSize > 0)
   {
      writecount = (totalSize >= chunksize ? chunksize : totalSize);
      input.read(bytes, 0, writecount);
      outputStreamSocket.write(bytes, 0,  writecount);
      totalSize = totalSize - writecount;
   }
   outputStreamSocket.flush();
}
```

Estensione del Client e del Server

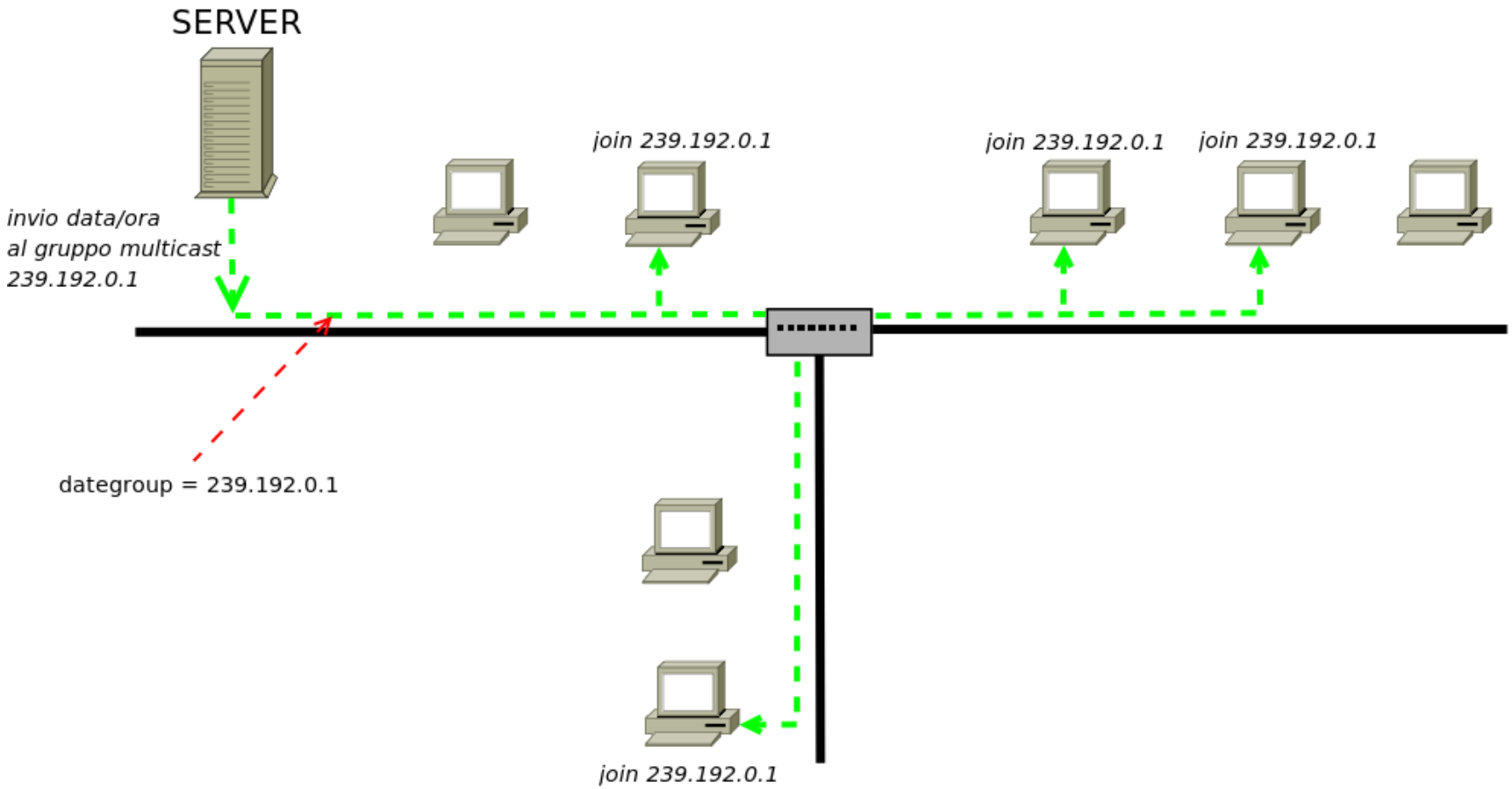
- Provare ad implementare come ulteriore tipo di richiesta la RECEIVE.
- Il server, dopo aver ricevuto una richiesta “RECEIVE”, aspetta il nome del file, e invia al client il file corrispondente. Il client invia un “ACK” a trasferimento avvenuto.
- Implementare la versione multithreaded del server: all’interno del primo ciclo `while` ogni volta che una connessione viene accettata, il server trasferisce il controllo e la gestione della connessione ad un thread (di un pool), e ritorna ad accettare altre connessioni.

```
Socket clientSocket=null;
while(true)
{...
   try
   {
      clientSocket = serverSocket.accept();
   ...
   }
   /*il socket viene passato ad un thread*/
   ...
}/*chiusura server*/
```

Esercizio Server Data-Ora in Multicast

- Definire un server TimeServer, che invia su un gruppo di multicast dategroup, ad intervalli regolari, la data e l'ora. L'attesa tra un invio ed il successivo può essere implementata tramite il metodo `sleep()`. L'indirizzo IP del gruppo dategroup è introdotto da riga di comando.
- Definire un client TimeClient che si unisce al gruppo dategroup e riceve, per dieci volte consecutive, la data e l'ora e le visualizza, quindi termina.

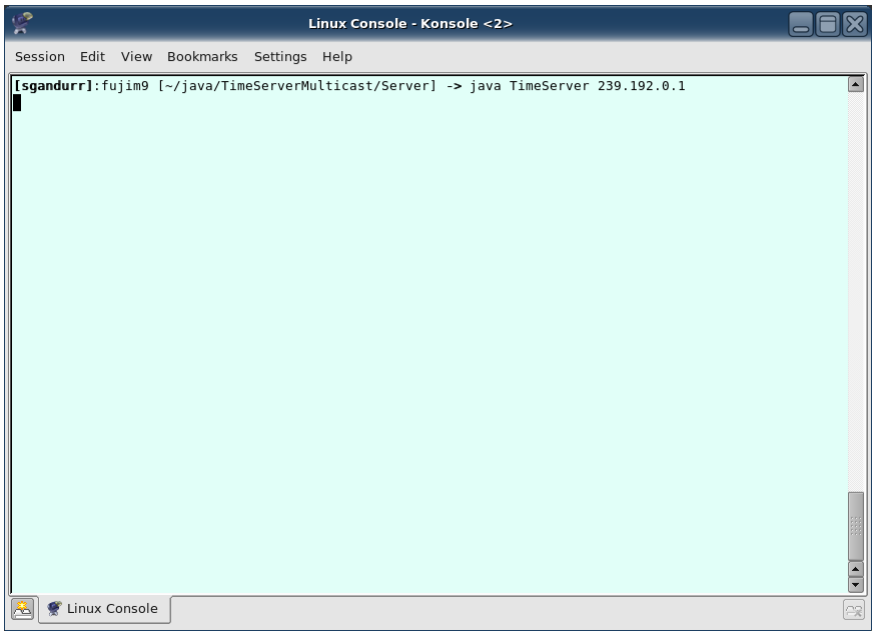
Esempio Multicast

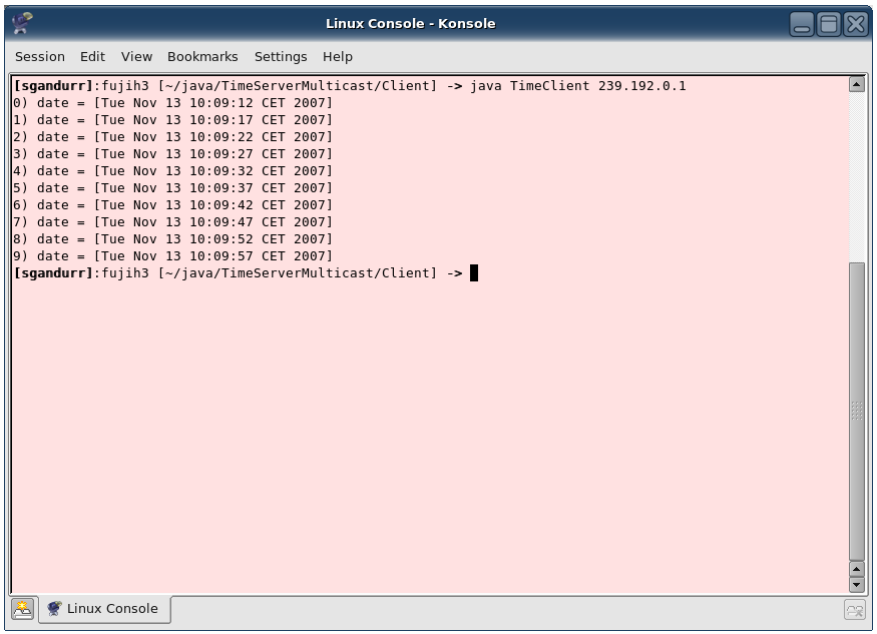


Esempio Utilizzo

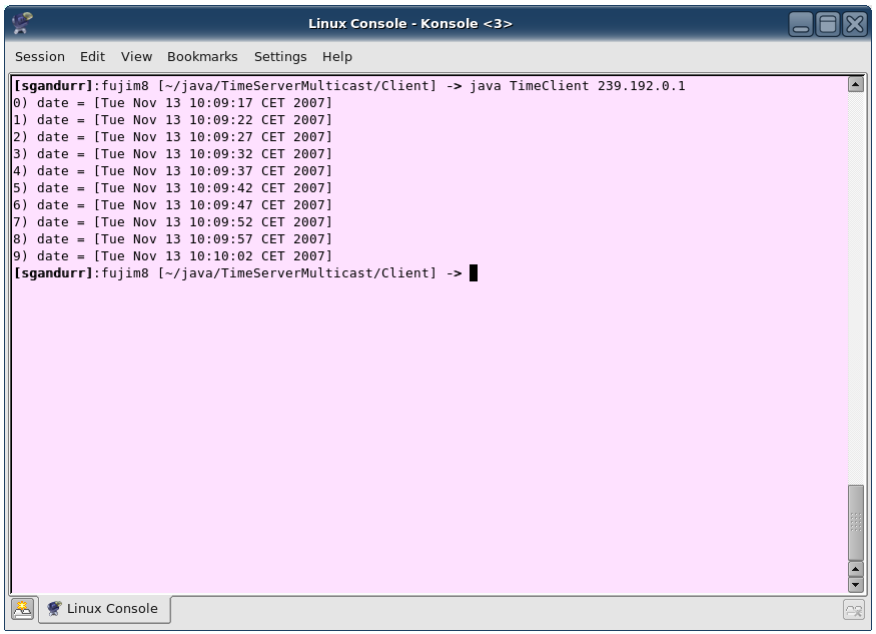
- Compilare i due file `TimeClient.java` e `TimeServer.java` su due directory diverse, ad es:
  - `~/java/TimeServerMulticast/Client`
  - `~/java/TimeServerMulticast/Server`
- Supponiamo il server stia su `fujim9`, e tre client su `fujih3`, `fujim8` e `fujii5`. Connettersi in ssh a tutti gli host.
- Nella directory del server, eseguire `java TimeServer 239.192.0.1`.
- Nella directory di ogni client, eseguire `java TimeClient 239.192.0.1`.
- Per terminare il server digitare “CTRL + C”.

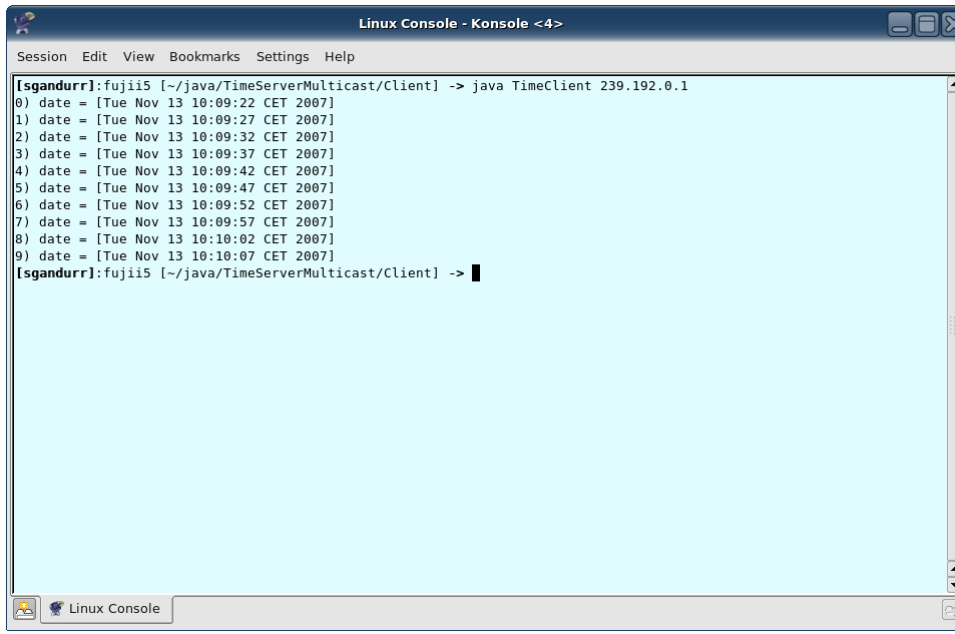
Esempio Utilizzo





Esempio Utilizzo





Il Server

- Il server usa un `DatagramSocket` per inviare pacchetti in multicast.
- In alternativa, il server può utilizzare un `MulticastSocket`.
- Il tipo di socket utilizzato dal server non è importante: le infomazioni di multicast sono contenute nel `DatagramPacket` e nel socket `MulticastSocket` usato dai client.

```
DatagramSocket socket=  new DatagramSocket();
```

oppure:

```
MulticastSocket socket = new MulticastSocket();
```

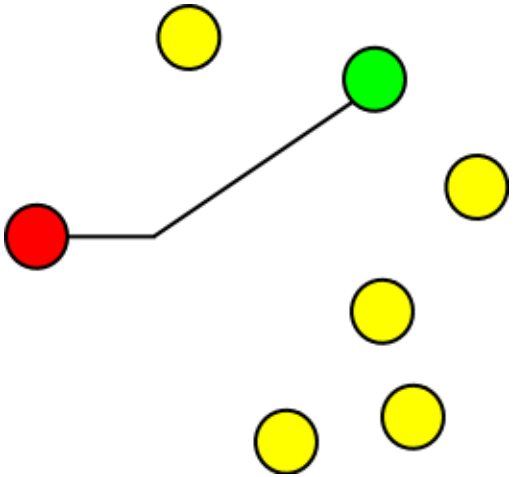
Il Client

Il client semplicemente crea un MulticastSocket, effettua la join, e riceve datagrammi UDP come se fosse un DatagramSocket normale:

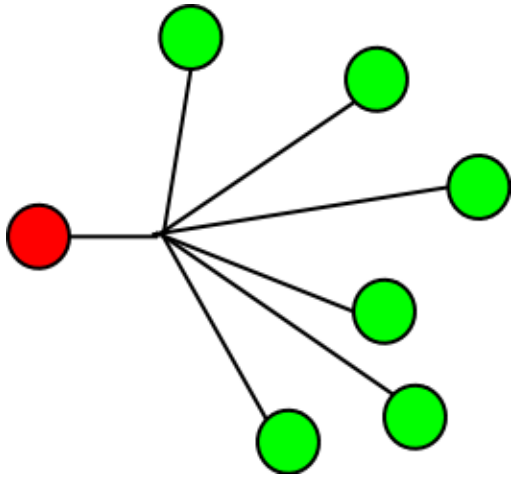
```
..
MulticastSocket socket = new MulticastSocket(port);
InetAddress address = InetAddress.getByName(multicastAddress);
socket.joinGroup(address);
...
for (int i = 0; i < 10; i++)
{
   byte[] buf = new byte[256];
   DatagramPacket packet = new DatagramPacket(buf, buf.length);
   socket.receive(packet);
}
...
socket.leaveGroup(address);
```

Unicast/Broadcast

Unicast: una singola destinazione

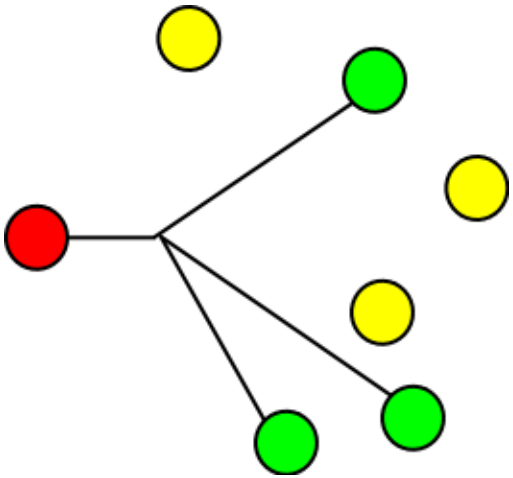


Broadcast: tutti gli host della rete

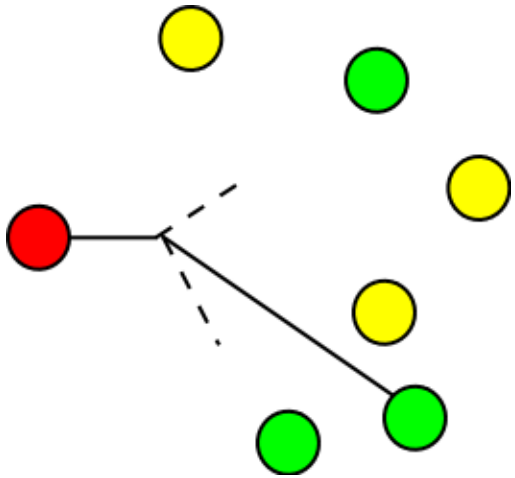


Multicast/Anycast

Multicast: un gruppo di host



Anycast: un host tra un insieme



Broadcast

- Alcune tecnologie hardware hanno dei meccanismi per inviare pacchetti contemporaneamente ad un insieme di destinazioni.
- Le tecnologie hardware mettono a disposizione un indirizzo broadcast.
- La scheda di rete riconosce sia l'indirizzo hardware locale che l'indirizzo broadcast e accetta frame che hanno come destinazione o l'uno o l'altro indirizzo.

Broadcast su Ethernet

- Nel caso di Ethernet, la consegna broadcast avviene inviando un solo pacchetto. In altri casi, i dispositivi di commutazione della rete devono inoltrare il pacchetto attraverso più collegamenti.
- Gli indirizzi hardware di Ethernet sono composti da 48 bit: l'indirizzo composto da tutti 1 è usato per richiedere una trasmissione broadcast.

Multicast

- Alcune tecnologie hardware permettono anche di effettuare una consegna multipunto: trasmissione di tipo multicast.
- A differenza del broadcast, essa permette ad ogni sistema di scegliere se vuole partecipare ad un determinato indirizzo di multicast.
- A livello hardware, l'interfaccia si mette in ascolto su un indirizzo hardware di multicast.

Multicast

- Gli indirizzi unicast e broadcast identificano o un computer o un insieme di computer sullo stesso segmento fisico: l'inoltro (il routing) dei pacchetti dipende dalla topologia di rete.
- Un indirizzo multicast, invece, identifica un insieme arbitrario di ricevitori: il pacchetto va inoltrato a tutti i segmenti.

Multicast su Ethernet

- Nel caso di Ethernet, metà degli indirizzi sono indirizzi di multicast!
- Il bit meno significativo del byte più significativo identifica se l'indirizzo è di multicast:
  - $01.00.00.00.00.00_{16} \rightarrow$ bit di multicast
- All'avvio, la scheda è in ascolto sull'indirizzo MAC locale e quello di broadcast.
- Il driver della scheda permette di configurare il dispositivo per avere uno o più indirizzi di multicast.
- Una volta configurata, la scheda accetta qualsiasi pacchetto inviato all'indirizzo locale, a quello broadcast o a quelli multicast su cui è stata fatta la join.

Multicast IP

- La trasmissione Multicast IP permette di inviare un pacchetto ad un sottoinsieme di computer.
- Il sottoinsieme di computer si può estendere anche a reti fisiche diverse.
- Il sottoinsieme di computer prende il nome di gruppo multicast.

Caratteristiche

- Indirizzi di classe D: i primi quattro bit dell'indirizzo IP multicast sono settati a 1110. Gli altri 28 specificano un particolare gruppo multicast. Gl'indirizzi multicast sono nell'intervallo [224.0.0.0-239.255.255.255].
- Appartenenza dinamica ai gruppi: un host può unirsi e lasciare un gruppo dinamicamente.
- Utilizzo dell'hardware: se supporta la trasmissione multicast ok, altrimenti si mappano gli IP su indirizzi broadcast/unicast.
- Utilizzo di router multicast per inviare il pacchetto su reti fisiche diverse.

Indirizzi IP Multicast

- L'indirizzo 224.0.0.0 è riservato e non può essere assegnato ad alcun gruppo.
- Gli indirizzi 224.0.0.1-255 sono destinati ai protocolli di multicast. Il router non inoltra i pacchetti inviati a questi IP.
  - 224.0.0.1: tutti gli host della rete.
  - 224.0.0.2: tutti i router della rete.
- Il blocco di indirizzi 239.192.0.0/14 è riservato per essere utilizzato nell'ambito di un'organizzazione (RFC 2365, par. 6.2). L'intervallo completo è: da 239.192.0.0 a 239.195.255.255. Usate questi.

Tradurre da IP Multicast a Multicast Ethernet

Lo standard IP specifica come tradurre un IP multicast in indirizzo multicast di Ethernet.

Traduzione

Per tradurre un IP multicast nel corrispondente multicast Ethernet, si copiano i 23 bit meno significativi dell'indirizzo IP nei 23 bit meno significativi dell'indirizzo Ethernet speciale $01.00.5E.00.00.00_{16}$.

Collisione di Indirizzi

- La traduzione non è univoca: più indirizzi multicast IP corrispondono allo stesso indirizzo multicast Ethernet.
- 28 bit di multicast $\rightarrow$ 23 bit copiati.
- Un host può ricevere datagrammi multicast non destinati a lui!
- Il software IP controlla che il pacchetto che gli viene inoltrato dal driver del dispositivo sia effettivamente destinato a quell'host.

Router Multicast

- I router multicast inoltrano i datagrammi multicast tra le reti.
- L'host deve semplicemente inviare un datagramma ad un router multicast. L'host non ha bisogno di conoscere l'IP di un router multicast.
- Infatti, i router multicast ascoltano tutte le trasmissioni multicast IP. Se necessario, inoltrano il datagramma su un'altra rete.

IGMP

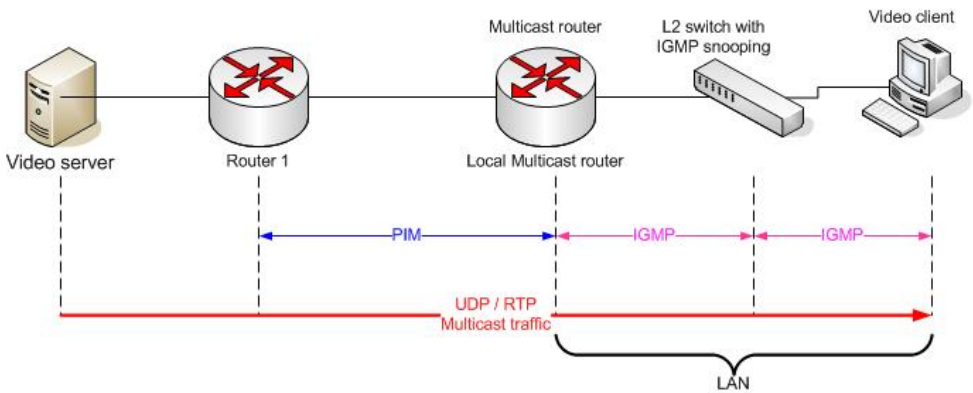
- IGMP = Internet Group Management Protocol.
- I router multicast e gli host che usano il multicast devono usare IGMP, per comunicare informazioni sull'appartenenza a gruppi.

Due fasi (semplificate):

1. Un host per unirsi ad un gruppo invia un messaggio IGMP all'indirizzo multicast del gruppo. I router ricevono il messaggio e ricalcolano l'instradamento multicast al gruppo propagando le informazioni relative ad altri router.
2. I router interrogano periodicamente la rete locale per stabilire se ci sono ancora host di un certo gruppo. Se nessun host di quel gruppo risponde, il router non annuncia più il gruppo multicast ad altri router.

Routing

- IGMP è usato tra gli host e i router.
- Protocol Independent Multicast (PIM) è utilizzato tra i router per scambiarsi informazioni di multicast.
- L'insieme dei percorsi da una sorgente a tutti i membri del gruppo è detta forward tree (albero di inoltro).
- Ogni router multicast è un nodo, la sorgente di un datagram la radice, l'ultimo router del percorso una foglia.
- Ciascuna entry in una tabella di routing multicast è una coppia: (gruppo di multicast, sorgente), dove sorgente è un prefisso di rete.



IGMP Snooping

- Di default, uno switch (livello 2) inoltra un frame Ethernet con indirizzo multicast a tutte le reti connesse allo switch.
- Ma gli switch sono nati proprio per limitare il traffico: invio del frame solo alla porta destinataria.
- Allora, grazie alla tecnica IGMP Snooping uno switch può esaminare informazioni di livello 3 nei pacchetti IGMP scambiati tra gli host e i router per limitare il traffico.
- Se lo switch rileva che un host ha effettuato una join per un particolare gruppo di multicast, allora aggiunge nella tabella di multicast l'associazione [porta dello switch dell'host / gruppo di multicast]
- Se lo switch intercetta un messaggio di uscita da un gruppo da parte di un host, rimuove dalla tabella l'entry relativa alla [porta dell'host / gruppo di multicast].