Multicasting with Sockets

Lesson 8

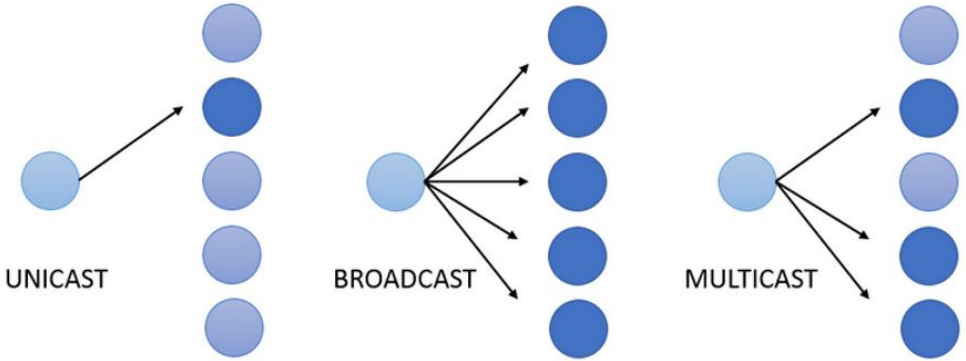
by
Yoshan Wijesuriya

Objectives

- Understand the concept of multicasting
- Understand multicast networking concepts
- Using MulticastSocket to create multicast senders and receivers

Introduction

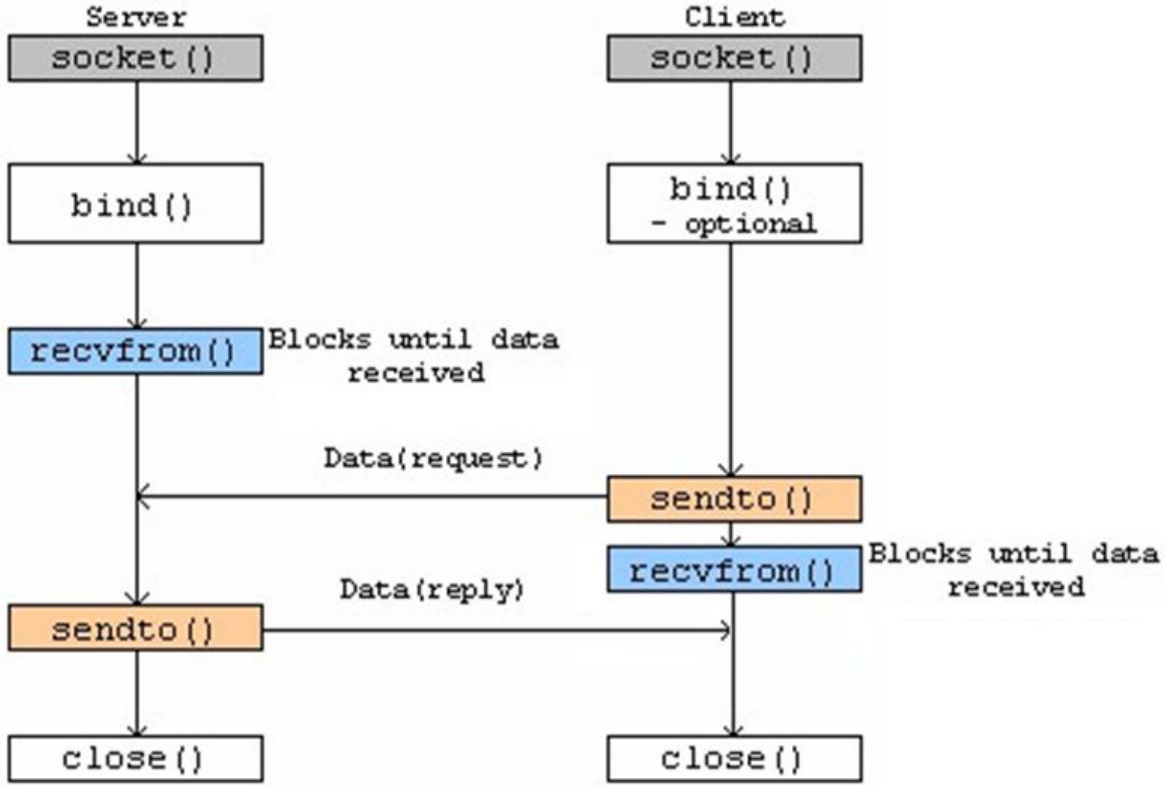
- Unicast
  - One sender to one receiver
  - Web browsing, email
- Broadcast
  - One sender to all receivers in a network
  - ARP requests, DHCP discovery, TV
- Multicast
  - One sender to multiple selected receivers
  - Live streaming, online gaming



Multicast Networking Concepts

- Class D IP Addresses:
  - Range from 224.0.0.0 to 239.255.255.255.
  - Reserved exclusively for multicast traffic.
- Multicast Groups:
  - Identified by a multicast IP address.
  - Hosts can join or leave groups dynamically.
  - Multiple groups can exist simultaneously.
- Internet Group Management Protocol (IGMP):
  - Manages multicast group memberships on IPv4 networks.
  - Allows hosts to communicate group membership to local routers.
- Multicast Routing Protocols:
  - Protocol Independent Multicast (PIM)
  - Distance Vector Multicast Routing Protocol (DVMRP)
  - Multicast Forwarding

Recap: UDP Client-Server Flow



Multicasting in Java

- `java.net.MulticastSocket` Class:
  - Subclass of DatagramSocket.
  - Supports sending and receiving IP multicast packets.
  - Enables joining and leaving multicast groups.
- https://download.java.net/java/early_access/panama/docs/api/java.base/java/net/MulticastSocket.html

```
MulticastSocket socket = new MulticastSocket(5000); // Bind to port 5000
```

MulticastSocket

- Joining a Multicast Group

```
MulticastSocket socket = new MulticastSocket(5000); // Bind to port 5000


InetAddress group = InetAddress.getByName("230.0.0.0");
NetworkInterface netIf = NetworkInterface.getByInetAddress(InetAddress.getLocalHost());
SocketAddress groupAddress = new InetSocketAddress(group, 5000);

socket.joinGroup(groupAddress, netIf);
```

- InetAddress group: Obtains the multicast group address.
- NetworkInterface netIf: Retrieves the network interface to use.
- You can specify the interface by name, e.g., NetworkInterface.getByName("eth0").
- Alternatively, get the default interface based on the local host address.
- SocketAddress groupAddress: Combines the group address and port into a SocketAddress.
- socket.joinGroup(groupAddress, netIf): Joins the multicast group on the specified network interface.

MulticastSocket

- Leaving a Multicast Group

```
socket.leaveGroup(groupAddress, netIf);
```

- Sending Data to a Multicast Group

```
String message = "Hello Multicast Group";
byte[] buffer = message.getBytes();

DatagramPacket packet = new DatagramPacket(buffer, buffer.length, groupAddress);
socket.send(packet);
```

- Setting Time-to-Live (TTL)

```
socket.setTimeToLive(64);
```

- Controlling Loopback Mode

```
socket.setOption(StandardSocketOptions.IP_MULTICAST_LOOP, false);
```

Multicast Sender and Receiver

- Code example

Questions?

- Any questions?
- Comments?
- Concerns?
- Ideas to share?

Thank You!