LẬP TRÌNH MẠNG

UDP Socket

hungdn@ptit.edu.vn

Nội dung

- Giới thiệu UDP
- Làm việc với UDP Socket
- Kỹ thuật lập trình với UDP Socket

1. Giới thiệu

Giao thức UDP

	Các tầng OSI	Họ giao thức TCP	TCP/IP Stack				
7	Tầng ứng dụng	Tầng ứng dụng	HTTP	FTP	SMTP	RIP	DNS
6	Tầng trình diễn						
5	Tầng phiên						
4	Tầng giao vận	Tầng giao vận	TCP			UDP	
3	Tầng mạng	Tầng Internet	ICMP,IP, IGMP				
2	Tầng liên kết dữ liệu	Tầng mạng	Ethernet, ATM, Frame Relay,..				
1	Tầng vật lý						

- **Giao thức User Datagram Protocol gửi các gói tin độc lập (datagram) không cần đảm bảo nhận và theo thứ tự**
- **Một số thuật ngữ**
  - Packet vs Datagram
  - MTU
  - Port
  - TTL (Time To Live)

Datagram

- Datagram
  - Một thông điệp tự cấu trúc và độc lập, được gửi thông qua mạng sử dụng giao thức UDP
  - Mang đầy đủ thông tin nguồn, đích nhưng không đảm bảo: đến đích, đến đúng thời điểm, đúng thứ tự và nội dung
- Header của UDP có 4 trường, 16 bit

Table :UDP datagram

Bit	0-15	16-31
0	Source port	Destination port
32	Length	Checksum
...	Data	

2. Làm việc với với UDP Socket

Lớp DatagramPacket

- Lớp DatagramPacket : cho phép tạo gói tin truyền thông với giao thức UDP

```
public final class DatagramPacket
```

- Các thể hiện (gói tin) của lớp này chứa 4 thành phần quan trọng
  - Địa chỉ
  - Dữ liệu
  - Kích thước gói tin
  - Số hiệu cổng

Lớp DatagramPacket

- Khởi tạo gói tin nhận từ mạng

```
public DatagramPacket(byte[] inBuffer, int length)
```

- Tham số
  - inBuffer: Bộ đệm nhập, chứa dữ liệu của gói tin nhận
  - length: kích cỡ của dữ liệu của gói tin nhận, nó thường được xác định bằng lệnh: *Length= buffer.length*
- Ví dụ

```
byte[] inBuff=new byte[512];//bộ đệm nhập
DatagramPacket inData=new DatagramPacket(inBuf, inBuff.length);
```

Lớp DatagramPacket (Cont.)

- Khởi tạo gói tin gửi từ mạng

```
public DatagramPacket(byte[] outBuffer , int length,
                                    InetAddress destination, int port)
```

- Tham số
  - `outBuffer`: Bộ đệm xuất chưa dữ liệu của gói tin gửi
  - `length`: kích cỡ dữ liệu của gói tin gửi tính theo số byte và thường bằng `outBuffer.length`.
  - `destination`: Địa chỉ nơi nhận gói tin
  - `port`: Số hiệu cổng đích, nơi nhận gói tin

Lớp DatagramPacket (Cont.)

- Ex khởi tạo gói tin gửi từ mạng

```
String s="Hello World!";
//Bộ đệm xuất và gán dữ liệu cho bộ đệm xuất
byte[] outBuff=s.getBytes();
//Địa chỉ đích
InetAddress addrDest=InetAddress.getByName("localhost");
//Số cổng đích
int portDest=3456;
//Tạo gói tin gửi
 DatagramPacket outData=new DatagramPacket (outBuff, outBuff.length,
                                            addrDest, portDest);
```

Lớp DatagramPacket (Cont.)

- Các phương thức
  - *public InetAddress getAddress():* Phương thức này trả về đối tượng InetAddress của máy trạm từ xa chứa trong gói tin nhận.
  - *public int getPort():* Trả về số hiệu cổng của máy trạm từ xa chứa trong gói tin.
  - *public byte[] getData()*: Trả về dữ liệu chứa trong gói tin dưới dạng mảng byte.
  - *public int getLength():* Trả về kích cỡ của dữ liệu chưa trong gói tin tính theo số byte
- Tương ứng với các phương thức get, lớp DatagramPacket có 4 phương thức set tương ứng để thiết lập giá trị cho gói tin

Lớp DatagramSocket

- Lớp DatagramSocket cho phép tạo ra đối tượng socket truyền thông với giao thức UDP. Socket này cho phép gửi/nhận gói tin DatagramPacket

```
public class DatagramSocket
```

- Khởi tạo

```
public DatagramSocket() throws SocketException
public DatagramSocket(int port) throws SocketException
```

  - Cho phép tạo socket với số cổng xác định hoặc không xác định(anonymous, thường được sử dụng ở client)
  - Nếu tạo socket không thành công, nó ném trả về ngoại lệ *SocketException*

Ex: *UDPPortScanner*

```
public class UDPPortScanner {
	public static void main(String[] args) {
		for (int port = 1024; port <= 65535; port++) {
		try {
			DatagramSocket server = new DatagramSocket(port);
			server.close();
		} catch (SocketException ex) {
		System.out.println("There is a server on port " + port + ".");
		} // end try
		} // end for
	}
}
```

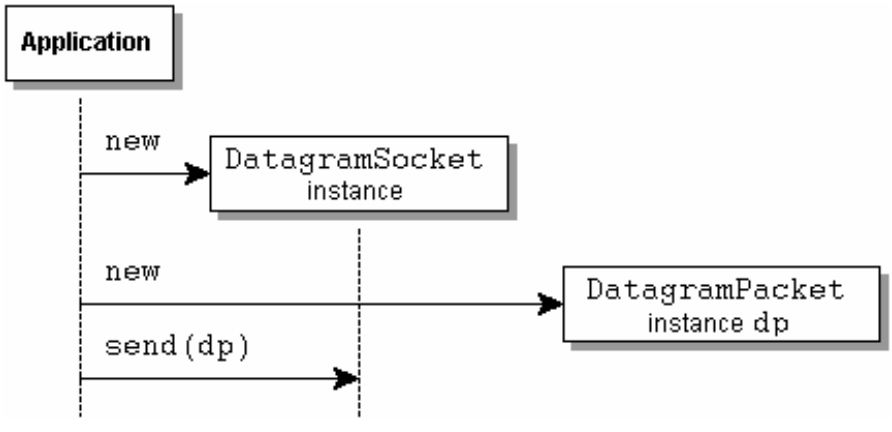
Lớp DatagramSocket (Cont.)

- Phương thức *public void send(DatagramPacket dp) throws IOException*
  - *Cho phép gửi gói tin UDP qua mạng*
  - *Ví dụ chương trình nhận chuỗi từ bàn phím, tạo gói tin và gửi tới Server*
- Phương thức *public void receive(DatagramPacket dp) throws IOException*
  - *Cho phép nhận gói tin UDP qua mạng*
  - *Ví dụ chương trình tạo đối tượng DatagramSocket với số cổng xác định, nghe và nhận gói dữ liệu gửi đến, hiển thị nội dung gói tin; địa chỉ;*
- *...*

Lớp DatagramSocket (Cont.)

Một số phương thức của lớp DatagramSocket	
void	bind(SocketAddress addr) Gắn kết DatagramSocket với địa chỉ và số cổng cụ thể
void	connect(InetAddress address,int port) Kết nối socket với địa chỉ máy trạm từ xa
void	connect(SocketAddress addr) Kết nối socket với địa chỉ socket từ xa.
void	disconnect() Huỷ bỏ kết nối
boolean	isBound() Trả về trạng thái kết nối của socket.
boolean	isClosed() Kiểm tra socket đã đóng hay chưa
boolean	isConnected() Kiểm tra trạng thái kết nối
void	*Close*() Phương thức đóng socket

3. Kỹ thuật lập trình truyền thông với UDP



Phía Server

1. Tạo đối tượng DatagramSocket với số cổng xác định

```
DatagramSocket myServer = new DatagramSocket(port);
```

2. Khai báo bộ đệm nhập/xuất hay inBuffer/outBuffer dạng mảng kiểu byte

```
byte[] receiveData  =  new byte[1024];
```

3. Khai báo đối tượng DatagramPackage để gửi/**nhận**

```
DatagramPacket  receivePacket  =  new DatagramPacket  (receiveData,
                                                  receiveData.length);
```

Phía Server (Cont.)

4. Thực hiện gửi/nhận gói tin với phương thức send() / receive()

```
myServer.receive(receivePacket);
```

5. Đóng socket, giải phóng các tài nguyên khác, kết thúc chương trình nếu cần, không quay về bước 3

Phía Client

1. Tạo đối tượng DatagramSocket với số cổng nào đó hoặc xác định

```
DatagramSocket myClient = new DatagramSocket();
```

2. Khai báo bộ đệm xuất/nhập hay outBuffer/inBuffer dạng mảng kiểu byte

```
byte[] sendData = new byte[1024];
sendData = "hello world".getBytes(); // trường hợp gửi dữ liệu
```

3. Khai báo đối tượng DatagramPackage để **gửi**/nhận

```
InetAddress IPAddress = InetAddress.getByName("localhost");
DatagramPacket sendPacket = new DatagramPacket (sendData,
                    sendData.length, IPAddress, port); // số cổng
```

Phía Client (Cont.)

4. Thực hiện gửi/nhận gói tin với phương thức send() / receive()

```
myClient.receive(sendPacket);
```

5. Đóng socket, giải phóng các tài nguyên khác, kết thúc chương trình nếu cần, không quay về bước 3

Ex: Xử lý đảo chuỗi sử dụng UDP

- **ReverseString**
  - **reverse()**
- **UDPServer**
  - **openServer()**
  - **listening()**
- **UDPClient**
  - **connect()**
  - **send(String str)**
  - **receive()**
  - **close()**
- **UDPServerRun**
  - **main()**
- **UDPClientRun**
  - **main()**

Ex: ReverseString

```
public class ReverseString {
	private String str;
	public ReverseString() { }
	public ReverseString(String str) {
		this.str = str; }
	public   void   reverse(){
		String   tmp   ="";
		for(int  i=str.length()   -  1;   i  >=0   ;i--)
			tmp   +=  str.substring(i,   i+1);
		this.str   =  tmp;
	}
	public String getStr() {
		return str;
	}
	public void setStr(String str) {
		this.str = str;
	}
}
```

Ex : UDPServer

```
public class UDPServer {
	DatagramSocket myServer = null;
	String input;
	int port = 1107;

	public void openServer() {
		try {
			myServer = new DatagramSocket(port);
		} catch (SocketException e) {
			System.out.println(e);
		}

	}
	...
}
```

- Lớp UDPServer
- Phương thức openServer khởi tạo một **DatagramSocket**

Ex: UDPServer (Cont.)

```
public void listening() {
	byte[] receiveData = new byte[1024];
	byte[] sendData = new byte[1024];
	while (true) {
		try {
			//	Nhan du lieu
			DatagramPacket receivePacket = new DatagramPacket(receiveData,
									receiveData.length);
			myServer.receive(receivePacket);
			input = new String(receivePacket.getData());
			//	Xu li du lieu
			ReverseString obj = new ReverseString(input);
			obj.reverse();
			// Dong goi thong tin du lieu can tra lai
....
```

Ex: UDPServer (Cont.)

```
                        InetAddress IPAddress = receivePacket.getAddress();
                        int port = receivePacket.getPort();
                        sendData = obj.getStr().getBytes();
                        DatagramPacket sendPacket = new DatagramPacket(sendData,
                                                sendData.length, IPAddress, port);
                        // Gui du lieu ve client
                        myServer.send(sendPacket);
                } catch (SocketException e) {
                        System.out.println(e);
                } catch (IOException e) {
                        System.out.println(e);
                }

        }
}
```

Ex: UDPClient

```
public class UDPClient {
//khai   bao  socket  cho   client,  cong  gui  va   nhan du  lieu
        DatagramSocket client = null;
        int port = 1107;

//Tao  ket   noi
        public void connect() {
                try {
                        client = new DatagramSocket(); //không cần khai báo port
                } catch (SocketException e) {
                        System.err.println(e);
                }
        }

// continue
```

Ex: UDPClient (Cont.)

```
public void send(String str) {
        if (client != null) {
                byte[] sendData = new byte[1024]; // bo dem gui dl
                try {
                        InetAddress IPAddress = InetAddress.getByName("localhost");
                        sendData = str.getBytes();
                        DatagramPacket sendPacket = new
                        DatagramPacket(sendData, sendData.length, IPAddress, port);
                        client.send(sendPacket);
                } catch (SocketException e) {
                        System.err.println(e);
                } catch (IOException e) {
                        System.err.println(e);
                }
        }
 }
}
// continue
```

Ex: UDPClient (Cont.)

```
//nhan   du  lieu   tra  ve  tu   server
public String receive() {
        if (client != null) {
                byte[] receiveData = new byte[1024]; // bo dem nhan dl
                try {
                        DatagramPacket receivePacket = new
                                        DatagramPacket(receiveData, receiveData.length);
                        client.receive(receivePacket);
                        return new String(receivePacket.getData());
                } catch (SocketException e) {
                        System.err.println(e);
                } catch (IOException e) {
                        System.err.println(e);
                }
        }
        return null;
}
// continue
```

Ex: UDPClient (Cont.)

```
// dong ket noi
public void close() {
	if (client != null) {
		try {
			client.close();
		} catch (Exception e) {
			System.err.println(e);
		}
	}
}
```

Ex: Run

- **Server**

```
public class UDPServerRun {
	public static void main(String[] args) {
		UDPServer server = new UDPServer();
		server.openServer();
		server.listening();
	}
}
```

- **Client**

```
public class UDPClientRun {
	public static void main(String[] args) {
		UDPClient client = new UDPClient();
		client.connect();
		client.send("hello world!");
		System.out.println(client.receive());
	}
}
```

Demo

- **DatagramPackage**
- **DatagramSocket**
- **InetAddress**
- **ReverseString**
  - `reverse()`
- **UDPServer**
  - `openServer()`
  - `listening()`
- **UDPClient**
  - `connect()`
  - `send(String str)`
  - `receive()`
  - `close()`
- **UDPServerRun**
  - `main()`
- **UDPClientRun**
  - `main()`

Exercise

- **UDP**
  - Quét cổng dịch vụ sử dụng giao thức UDP
  - Giao tiếp lấy thời gian server về client
- **Cài đặt máy tính cá nhân đơn giản theo mô hình client/server với giao thức TCP/IP**
  - Client gửi yêu cầu về phép toán cần thực hiện add/sub/mul/div ...
  - Server tiếp nhận và xử lý trả về kết quả
  - Sử dụng luồng đối tượng
- **Case study**
  - Cài đặt và mở rộng ứng dụng Login từ xa sử dụng TCP/IP

Exercise

- **Cài đặt các bài tập ở phần TCP Socket sang sử dụng UDP Socket**
- **Case study**
  - Cài đặt và mở rộng ứng dụng Login từ xa hỗ trợ đa luồng song song

Tổng kết

- **Giao thức UDP**
- **DatagramPacket**
- **DatagramSocket**
- **Chủ đề**
  - Mô phỏng DNS (Domain Name System),
  - Ứng dụng streaming media, Voice over IP
  - Trivial File Transfer Protocol (TFTP)
  - Video conference

Case study: Login sử dụng UDP Socket

Model

View

Controller

Q & A