SOCKETS UDP

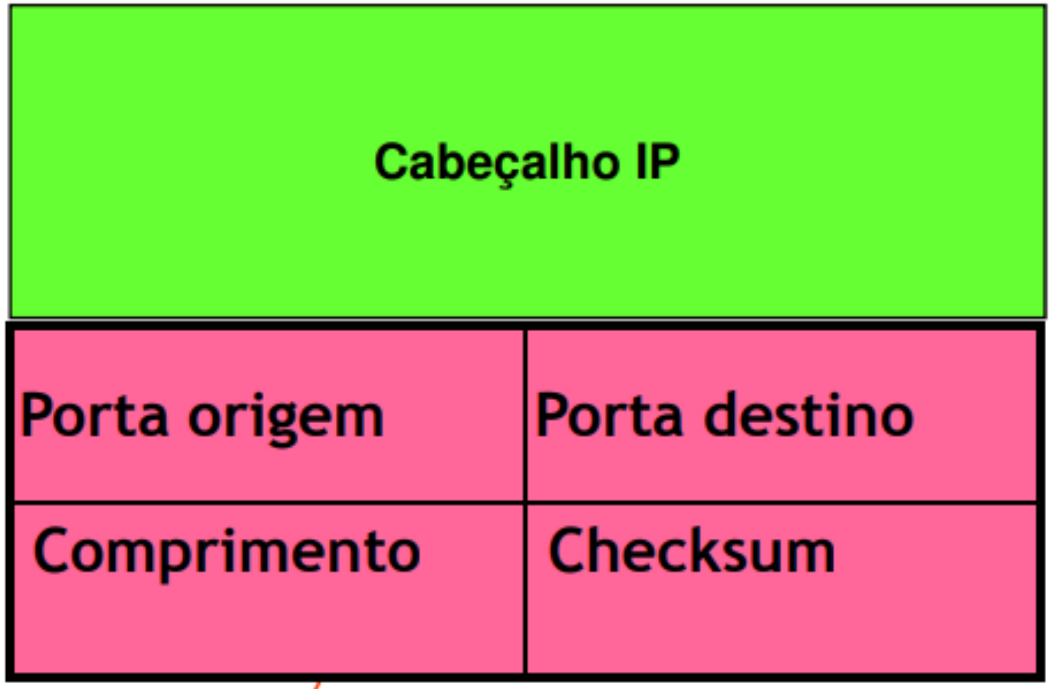
Prof. DSc Marcelo Lisboa

Sistemas Distribuídos

SOCKETS UDP: Características

- **Sockets UDP: canal não-confiável**
  - Não garante entrega dos datagramas
  - Pode entregar datagramas duplicados
  - Não garante ordem de entrega dos datagramas
  - Não tem estado de conexão (escuta, estabelecida)

Datagrama UDP



ENDEREÇO IP

Cabeçalho UDP: PORTAS

Datagrama

Mensagem auto-contida

Tamanho máximo: limitado pelo protocolo IP v4

2^{16} bytes (cabeçalhos + conteúdo) = 65.536 bytes

SOCKETS UDP: Comandos Básicos

- Criar socket
  - **DatagramSocket s = new DatagramSocket(6789);**
- Receber um datagrama
  - **s.receive(req);**
- Enviar um datagrama
  - **s.send(resp);**
- Fechar um socket
  - **s.close();**
- Montar um datagrama para receber mensagem
  - **new DatagramPacket(buffer, buffer.length);**
- Montar um datagrama para ser enviado
  - **new DatagramPacket(msg, msg.length, inet, porta);**
  - ***Buffer e msg são byte[]***

SOCKETS UDP: Servidor

```
import java.net.*;
import java.io.*;

// cria um socket UDP
DatagramSocket s = new DatagramSocket(6789);
byte[] buffer = new byte[1000];
System.out.println("*** Servidor aguardando request");

// cria datagrama para receber request do cliente
DatagramPacket r = new DatagramPacket(buffer, buffer.length);
s.receive(r);
System.out.println("*** Request recebido de: " + r.getAddress());

// envia resposta
DatagramPacket resp = new DatagramPacket(r.getData(), r.getLength(),
                          r.getAddress(), r.getPort());
s.send(resp);
s.close();
```

Servidor de “um-tiro”. Ao receber uma conexão de um cliente, retorna a resposta e encerra a execução.

SOCKETS UDP: Cliente

```
import java.net.*;
import java.io.*;

// cria um socket UDP
s = new DatagramSocket();
System.out.println("* Socket criado na porta: " + s.getLocalPort());
byte[] m = args[0].getBytes(); // transforma arg em bytes

InetAddress serv = InetAddress.getByName(args[1]);
int porta = 6789;
DatagramPacket req = new DatagramPacket(m, args[0].length(), serv, porta);

// envia datagrama contendo a mensagem m
s.send(req);

byte[] buffer = new byte[1000];
DatagramPacket resp = new DatagramPacket(buffer, buffer.length);
s.setSoTimeout(10000); // timeout em ms

// recebe resposta do servidor - fica em wait ateh chegada
s.receive(resp);
System.out.println("* Resposta do servidor:" + new String(resp.getData()));

// fecha socket
s.close();
```

SOCKETS UDP: Esquema Cliente-Servidor

- **Cliente**
  1. **Criar socket:** um socket pode ser utilizado para enviar datagramas para qualquer socket servidor
  2. **Montar datagrama** com <servidor:porta> de destino <servidor:porta> de origem
  3. **Enviar datagrama**
  4. **Bloqueia num receive**
     - ...
     - ...
     - ...
  5. **Recebe a resposta**
  6. **Trata a resposta**
  7. **Volta ao item 2**

- **Servidor**
  1. **Aguarda num receive**
     - ...
     - ...
     - ...
     - ...
     - ...
     - ...
     - ...
     - ...
     - ...
  2. **Recebe a mensagem**
  3. **Processa a mensagem**
  4. **Envia resposta ao cliente**
  5. **Volta ao item 1**

SOCKETS UDP: Exercícios

- Baseando-se no código dos slides anteriores, fazer um servidor que atenda aos clientes invertendo a string recebida ou fazendo uma modificação qualquer na mensagem recebida (versão 1)

- Alterar o servidor (ex. 1) para tratar solicitações dos clientes de forma concorrente. Cada vez que uma mensagem é recebida, o servidor cria uma thread para atendê-la e volta a esperar por nova solicitação de cliente