Programação de *sockets*

Roberto Sadao Yokoyama

UTFPR-CP

Agosto, 2016

Prática de hoje:[1]

[1] Baseado no exemplo do livro: Kurose[1]

Objetivo

Aprender a criar aplicação cliente-servidor que se comunica usando sockets

- Socket: Uma interface criada pela aplicação e controlada pelo SO (uma "porta") na qual o processo da aplicação pode enviar e receber mensagens para/de outro processo da aplicação
- API socket
  - introduzida no BSD4.1 UNIX em 1981
  - criada, usada e liberada explicitamente pelas aplicações
  - paradigma cliente-servidor
  - dois tipos de serviços de transporte por meio da API socket: UDP e TCP

Fundamentos de programação de socket

- Servidor deve estar "rodando"antes que o cliente possa lhe enviar algo
- Servidor deve ter um socket (porta) pelo qual recebe e envia segmentos
- Da mesma forma, o cliente precisa de um socket
- Socket é identificado localmente com um número de porta
- Cliente precisa saber o endereço IP do servidor e o número de porta do socket

Programação de socket com UDP

UDP: sem "conexão"entre cliente e servidor

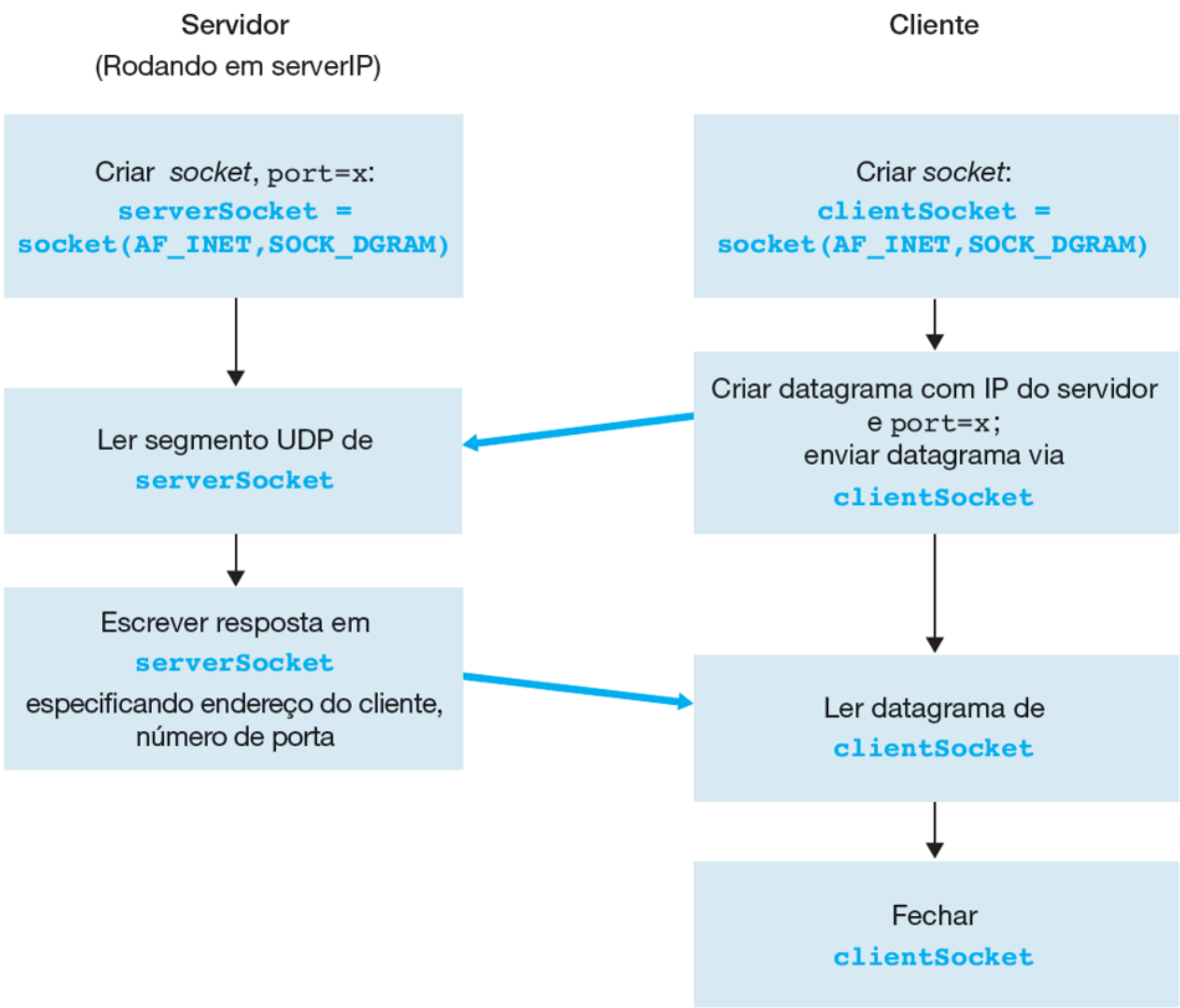
- sem “handshaking”
- Emissor conecta de forma explícita endereço IP e porta do destino a cada segmento
- SO conecta endereço IP e porta do socket emissor a cada segmento
- Servidor pode extrair endereço IP, porta do emissor a partir do segmento recebido
- Cliente precisa saber o endereço IP do servidor e o número de porta do socket
- Ponto de vista da aplicação: UDP oferece transferência não confiável de grupos de bytes ("datagramas") entre cliente e servidor

Programação de socket com UDP

Exemplo

- **cliente**:
  - usuário digita linha de texto
  - programa cliente envia linha ao servidor
- **servidor**:
  - servidor recebe linha de texto
  - coloca todas as letras em maiúsculas
  - envia linha modificada ao cliente
- **cliente**:
  - recebe linha de texto
  - apresenta

Interação de socket cliente/servidor: UDP



Código cliente

```
from socket import *
servidor='127.0.0.1'
porta=9090

socketCliente = socket(AF_INET, SOCK_DGRAM)

mensagem=raw_input("Escreva uma mensagem:")

socketCliente.sendto(mensagem,(servidor,porta))

mensagemModificada, enderecoServidor =
socketCliente.recvfrom(2048)

print mensagemModificada
```

Código servidor

```
from socket import *

porta=9090

socketServidor = socket(AF_INET,SOCK_DGRAM)

socketServidor.bind(('',porta))

print "Servidor estah pronto"

while 1:
    mensagemRecebida, enderecoCliente =
    socketServidor.recvfrom(2048)
    mensagemModificada = mensagemRecebida.upper()
    socketServidor.sendto(mensagemModificada,enderecoCliente)
```

Atividade

a. Modifique/implemente a aplicação (cliente-servidor) do exemplo para utilizar o protocolo de comunicação TCP.

b. Acrescente à cifra de césar na aplicação. O servidor deve gerar uma chave k e transmitir para cliente. O cliente deve encriptar a mensagem, m, usando k, $E(k, m)$ e enviar ao servidor. O servidor decripta a mensagem, D(E(k,m)), converte a mensagem em letras maiúsculas, M, encripta $E(k, M)$ e devolve ao cliente. O cliente decripta a mensagem e exibe na tela.

c. Modifique o servidor de maneira que aceite múltiplos clientes conectados simultaneamente.

Referências

[1] J. Kurose and R. Keith. *Redes de Computadores e a Internet – Uma Abordagem Top–Down*. Pearson Education 6ed., 2013.