Redes de Computadores I - LAB#3

Danielo Gonçalves Gomes
Manuel Gonçalves da Silva Neto
Liuz Guerreiro Palácio

Agenda

- Sockets UDP e TCP em Python3
- Atividades

Materiais e Métodos

- Conhecimentos necessários:
  - Chapters 2 e 3 do **Foundations of Python Network Programming**
  - https://www.webnots.com/what-is-http/ (HTTP request/response)
  - https://www.youtube.com/watch?v=bTThyxVy7Sk (Python Network)
  - https://pythonprogramming.net/python-sockets/ (Sockets intro)
  - https://pythonprogramming.net/python-port-scanner-sockets/ (Port Scan)
  - https://www.youtube.com/watch?v=SFERo-OjfdE (Intro - Português)

Materiais e Métodos

- Extras para consulta
  - https://panda.ime.usp.br/aulasPython/static/aulasPython/index.html
  - http://docs.python.org/3/library/
  - https://www.youtube.com/watch?v=Rc4JQWowG5I (Jupyter - Video)
  - https://www.youtube.com/watch?v=a1P_9fGrfnU (Spyder3 - Video)

Atividades

- Introdução a sockets – TCP
  - Requisições HTTP via sockets
  - Simulando um portScan simples com o método connect() do socket.

Atividades

- *Criando servidores sockets TCP*
  - Importar socket
  - Criar variáveis para armazenar o endereço e porta do server
  - Instanciar a classe socket, com o parâmeto socket.SOCK_STREAM
  - Realizar o bind() e o listen() na classe instanciada
  - Criar um loop infinito para manipular as conexões dos clientes
  - Utilizar o método .accept() para aceitar conexões
  - Utilizar os métodos .send() e .recv(BUFFER) para enviar ou receber msg
  - Fechar as conexões de cada cliente com .close()

Atividades

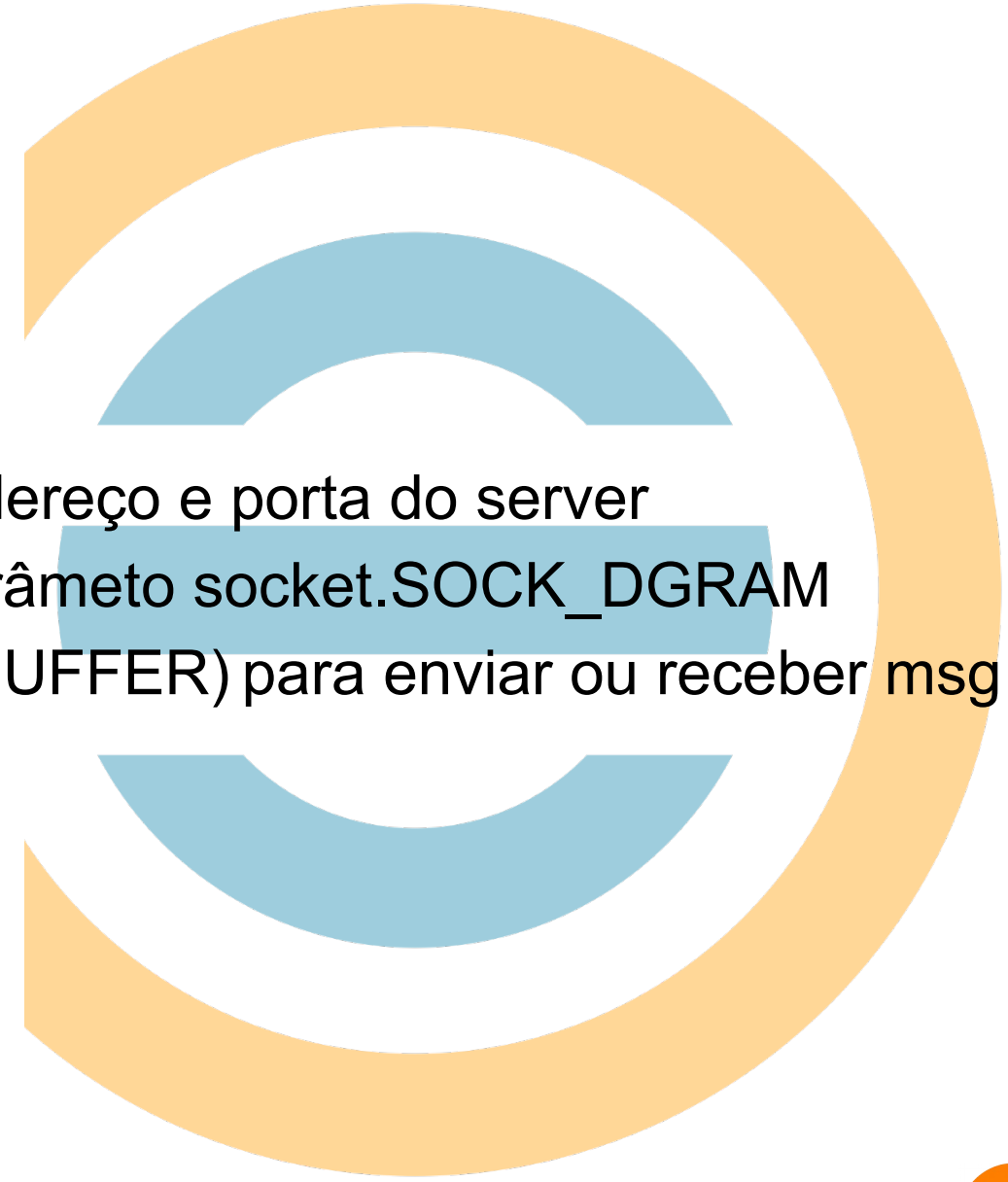
- *Criando servidores sockets UDP*
  - Importar socket
  - Criar variáveis para armazenar o endereço e porta do server
  - Instanciar a classe socket, com o parâmeto socket.SOCK_DGRAM
  - Realizar o bind() na classe instanciada
  - Criar um loop infinito para manipular as conexões dos clientes
  - Utilizar os métodos .send() e .recv(BUFFER) para enviar ou receber msg

Atividades

- *Criando clientes sockets TCP*
  - Importar socket
  - Criar variáveis para armazenar o endereço e porta do server
  - Instanciar a classe socket, com o parâmeto socket.SOCK_STREAM
  - Realizar o .connect() na classe instanciada
  - Utilizar os métodos .send() e .recv(BUFFER) para enviar ou receber msg
  - Fechar as conexões com .close()

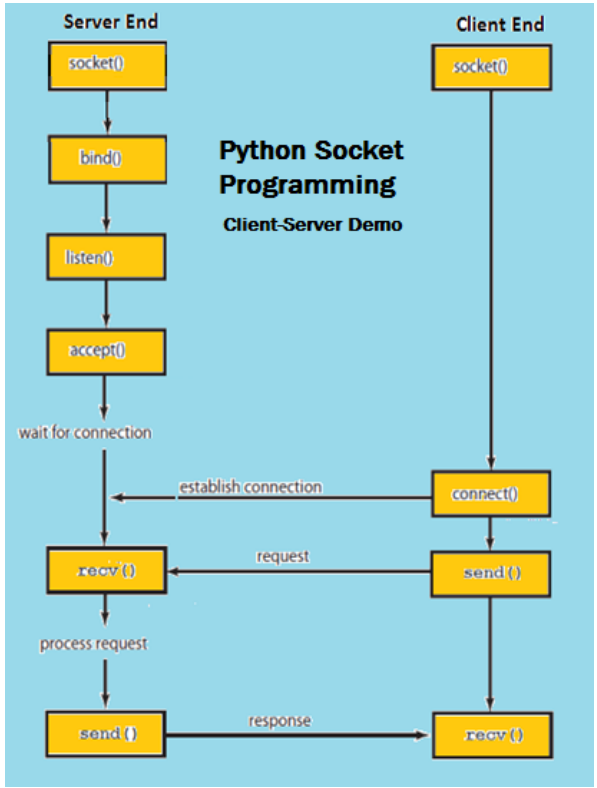
Atividades

- *Criando clientes sockets UDP*
  - Importar socket
  - Criar variáveis para armazenar o endereço e porta do server
  - Instanciar a classe socket, com o parâmeto socket.SOCK_DGRAM
  - Utilizar os métodos .send() e .recv(BUFFER) para enviar ou receber msg
  - Encerrar o socket com .close()



Atividades

- *WorkFlow sockets TCP*



Atividades

- *Exemplo de Server TCP*

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Mon Apr  9 09:36:59 2018

@author: manuel
Servidor simples com sockets operando no localhost, porta 5000
"""

import socket
HOST = '127.0.0.1'              # Endereco IP do Servidor
PORT = 5000            # Porta que o Servidor ira rodar
tcp = socket.socket(socket.AF_INET, socket.SOCK_STREAM) #cria o socket TCP
orig = (HOST, PORT) #Tupla contendo as configs
tcp.bind(orig) #no server usamos bind, no client usamos .connect()
tcp.listen() #Habilita o aceite de conexoes
print("Servidor aguardando conexões na porta %s, para encerrar pressione CTRL+C" % PORT)
while True:
    #aceita conexoes e armazena informacoes sobre quem conectou
    con, cliente = tcp.accept() #Aguarda novas conexoes
    print('Concetado por', cliente)
    mensagem  =  con.recv(2048)
    print("Mensagem recebida: %s" % mensagem)
    con.send("Sua mensagem foi processada com sucesso no servidor".encode('utf-8'))
    print('Finalizando conexao do cliente', cliente)
    con.close()
```

Atividades

- *Exemplo de cliente TCP*

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Mon Apr  9 09:45:52 2018

@author: manuel
Exemplo de cliente TCP acessando um server no localhost, porta 5000
"""

import socket
HOST = '127.0.0.1'     # Endereco IP do Servidor onde vou conectar
PORT = 5000            # Porta que o Servidor esta rodando
tcp = socket.socket(socket.AF_INET, socket.SOCK_STREAM) #socket tcp
dest = (HOST, PORT)
tcp.connect(dest) #no client utilizamos connect
print("Cliente conectado ao servidor %s na porta %s" % (HOST, PORT))
msg = input("Digite algo e pressione ENTER: ")
tcp.send(msg.encode('utf-8'))
recebida = tcp.recv(2048)
print("dados recebidos de retorno: ", recebida)
tcp.close()
```

Atividades

- *Exemplo de Server UDP*

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Mon Apr  9 10:15:22 2018

@author: manuel
Servidor socket UDP rodando  em todos os endereços disponiveis, porta 5001
"""

import socket
UDPSock = socket.socket(socket.AF_INET,socket.SOCK_DGRAM)

# Listen on port 5001
listen_addr = ("",5001) #utiliza todos os endereços disponiveis no PC
UDPSock.bind(listen_addr) #utilizamos somente o bind, sem connect ou accept

# Recebe dados de qualquer fonte em buffers de 2048
print("Servidor UDP rodando na porta  5001, para encerrar pressione CTRL+C...")
while True:
        data,addr = UDPSock.recvfrom(2048)
        print("Dados <%s> recebidos de: %s" % (data.strip(), addr))
        UDPSock.sendto("Mensagem processada pelo servidor...".encode('utf-8'),addr)
```

Atividades

- *Exemplo de cliente UDP*

```
"""
Created on Mon Apr  9 10:31:49 2018

@author: manuel
Cliente socket UDP acessando um server no localhost, porta 5001
"""

import socket
UDPSock = socket.socket(socket.AF_INET,socket.SOCK_DGRAM)

dados  = input("Digite uma frase para enviar ao servidor UDP: ")

# Simply set up a target address and port ...
addr = ("localhost",5001)
# ... and send data out to it!
print("Enviando dados para o servidor via UDP...")
UDPSock.sendto(dados.encode('utf-8'),addr) #apenas enviamos direto

#recebendo uma resposta com um buffer de 2048
msg, server = UDPSock.recvfrom(2048)
print("Servidor retornou: %s" % msg)
UDPSock.close()
```

Dicas

- *Ampliar os exemplos, mesclar com uso de listas e outras funcionalidades Python;*
- *Ler o material e assistir todos os vídeos dos links recomendados;*
- *Tentar compreender o problema de cada questão , e não somente criar o código fonte;*
- *Entrar em contato com antecedência hábil para realização de pesquisas adicionais e leituras complementares que serão indicadas.*

UNIVERSIDADE FEDERAL DO CEARÁ

GREAT
GRUPO DE REDES DE COMPUTADORES
ENGENHARIA DE SOFTWARE
E SISTEMAS

Obrigado