Laboratório 1 - Redes de Computadores

Emanoel Johannes Cardim Lazaro, Gabriel Pinheiro

17 de setembro de 2019

Universidade de Brasília - Instituto de Ciências Exatas
Departamento de Ciência da Computação - CIC 116726 - Informática e Sociedade
2019.2 - Turma B - Professora Dr. Priscila Solis Mendez
Prédio CIC/EST - Campus Universitário Darcy Ribeiro
Asa Norte 70919-970 Brasília, DF
emanoeljohannes@hotmail.com

Resumo

O relatório descreve a implementação de servidor web/client, com arquitetura TCP, feito com sockets na linguagem Python.

Palavras-chave: *TCP. Sockets. Web. Server. Client*

1 Introdução

O relatório a seguir relata o primeiro laboratório da disciplina de Redes de Computadores, no qual consiste em programar um software do tipo cliente para acessar os serviços de um servidor. O tipo de servidor usado foi o Web.

2 Fundamentação Téorica

Nosso projeto tem como objetivo demonstrar uma conexão entre cliente e servidor. Tal conexão foi feita através da API de sockets do Python.

Através dos sockets, dois processos podem se comunicar, estando ou não na mesma rede. Quando nosso navegador carrega uma página web, ou quando integramos um sistema a um banco de dados, o socket é um dos grandes responsáveis pela conexão.

Considerando a arquitetura TCP/IP, os sockets estão entre a camada de aplicação e a camada de transporte. Estando nessa posição, os sockets fazem um papel de interface, em que são os pontos finais entre as conexões.

3 Ambiente Experimental e Análise de Resultados

Descrição do cenário: Para realizar o laborátorio, de forma a fazer uma conexão web, utilizamos a API de sockets do Python. Para provar essa conexão estabelecida, nosso projeto faz a comunicação entre um arquivo que gera um servidor, que fica ouvindo uma porta de nossa rede, e outro arquivo de interface, que seria o cliente, que envia mensagens para essa porta, e o servidor que está conectado, resgata essa mensagem que foi enviada e retorna ela ao cliente. Esse retorno pode ser visto tanta na interface em sí, quanto no prompt de comando a qual o servidor foi iniciado. O server também identifica qual foi o IP que se conectou, e se a conexão foi encerrada ou não, assim como a porta a qual ele estava ouvindo, no caso, a porta 80, com ip de localhost, graças ao servidor Apache ligado, através do Xampp.

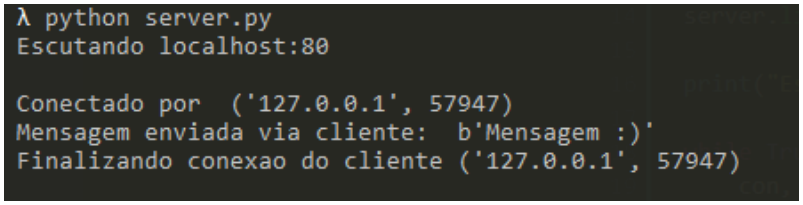

Figura 1: Exemplo de mensagens vindas do servidor, assim o cliente envia uma mensagem, e se desconecta

O servidor ficará escutando infinitamente, e suporta quantas conexões desejarmos, através do metódo listen(), disponível ao criarmos o socket TCP no Python.

Sobre o cliente, criamos uma interface utilizando a biblioteca Tkinter do Python, que nos permitiu criar uma interface simples, mas funcional, capaz de enviar uma mensagem a porta 80, e exibir de volta a mensagem, dessa vez vinda do servidor.

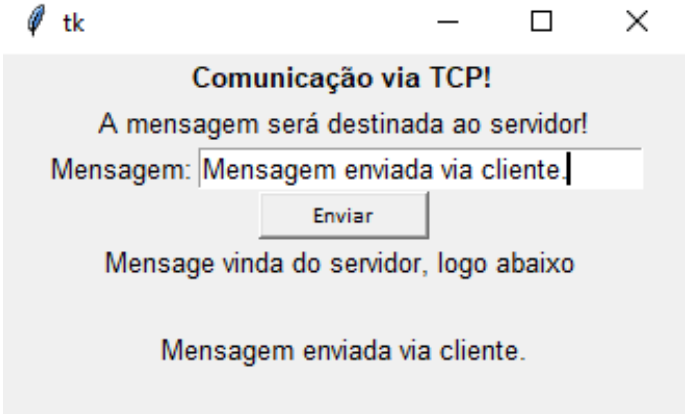

Figura 2: A mensagem "Mensagem enviada via cliente"printada na última linha da interface, veio do servidor, retornando a própria mensagem do cliente como resposta, afim de provar uma conexão estabelecida

Análise dos resultados: O resultado foi como o esperado, com um servidor ouvindo através da porta, e um cliente enviando dados a esse servidor, com este retornando as mensagens, provando uma conexão efetivamente estabelecida. Um dos desafios, foi fazer o servidor resgatar essa mensagem, e enviar novamente ao cliente.

Clique aqui para ter acesso ao vídeo demonstrativo e explicativo de nosso código.

4 Conclusão

Obtivemos um bom conhecimento prático sobre sockets, e sobre como fazer uma conexão real, fazendo uma espécie de chat com o servidor, que retorna a própria mensagem enviada via cliente. Este conceito, aplicado com maior número de clientes, pode ter uma grande utilidade.