Atividade – AV2

Questão 01: Servidor Web

Nesta tarefa, você desenvolverá um servidor Web simples em Python, capaz de processar apenas uma requisição. Seu servidor Web (i) criará um socket de conexão quando contatado por um cliente (navegador); (ii) receberá a requisição HTTP dessa conexão; (iii) analisará a requisição para determinar o arquivo específico sendo requisitado; (iv) obterá o arquivo requisitado do sistema de arquivo do servidor; (v) criará uma mensagem de resposta HTTP consistindo no arquivo requisitado precedido por linhas de cabeçalho; e (vi) enviará a resposta pela conexão TCP ao navegador requisitante. Se um navegador requisitar um arquivo que não está presente no seu servidor, seu servidor deverá retornar uma mensagem de erro "404 Not Found".

A seguir, é oferecido o código estrutural para o seu servidor. Sua tarefa é concluir o código, rodar seu servidor e depois testá-lo enviando requisições de navegadores rodando em hospedeiros diferentes. Se você rodar seu servidor em um hospedeiro que já tem um servidor Web rodando nele, então deverá usar uma porta diferente da porta 80 para o seu servidor.

Esqueleto de Código Python para o Servidor Web

Abaixo você encontrará o esqueleto do código para o servidor Web. Você deve completar o código onde indicado. Os trechos onde é necessário inserir código estão marcados com #Fill in start e #Fill in end. Cada um desses trechos pode exigir uma ou mais linhas de código.

```
# Importa o módulo socket
from socket import *
import sys  # Necessário para encerrar o programa

# Cria o socket TCP (orientado à conexão)
serverSocket = socket(AF_INET, SOCK_STREAM)

# Prepara o socket do servidor

#Fill in start
#Fill in end

while True:
    # Estabelece a conexão
    print('Ready to serve...')
    connectionSocket, addr = #Fill in start     #Fill in end

    try:
        # Recebe a mensagem do cliente (requisição HTTP)
        message = #Fill in start     #Fill in end
```

```
        filename = message.split()[1]
        f = open(filename[1:])
        outputdata = #Fill in start     #Fill in end

        # Envia a linha de status do cabeçalho HTTP
        #Fill in start
        #Fill in end

        # Envia o conteúdo do arquivo ao cliente
        for i in range(0, len(outputdata)):
            connectionSocket.send(outputdata[i].encode())
        connectionSocket.send("\r\n".encode())

        # Fecha a conexão com o cliente
        connectionSocket.close()

    except IOError:
        # Envia mensagem de erro 404 se o arquivo não for encontrado
        #Fill in start
        #Fill in end

        # Fecha o socket do cliente
        #Fill in start
        #Fill in end

serverSocket.close()
sys.exit()  # Encerra o programa
```

Executando o Servidor

- Crie um arquivo HelloWorld.html no mesmo diretório do script, com algo simples como: <html><body><h1>Hello, World!</h1></body></html>.
- Execute o servidor
- No navegador (ou em outro computador da rede), acesse: http://127.0.0.1:6789/HelloWorld.html (ou substitua 127.0.0.1 pelo IP da máquina onde o servidor está rodando.)

HelloWorld.html é o nome do arquivo que você colocou no diretório do servidor. Note também o uso do número da porta após os dois pontos. Você deve substituir esse número de porta pelo que você utilizou no código do servidor. Em seguida, tente acessar um arquivo que não existe no servidor. Você deverá receber uma mensagem “404 Not Found”.

Questão 02: Wireshark – HTTP

1. O Básico do GET do HTTP/interação de resposta

Vamos começar explorando o HTTP fazendo o download de um arquivo HTML muito simples – um arquivo pequeno e que não contém referências para objetos. Faça o seguinte:

- Inicie o navegador Web.
- Inicie o sniffer de pacotes, Wireshark, mas não inicie a captura de pacotes. Digite "http" (somente as letras sem as aspas) na janela display-filter-specification, assim somente as mensagens HTTP serão mostradas pela ferramenta.
- Espere um instante antes de iniciar a captura de pacotes.
- Entre o endereço abaixo no navegador: http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file1.html
  O navegador deve mostrar uma página bem simples com uma linha.
- Pare a captura de pacotes do Wireshark.

A sua janela do Wireshark deve ser bem parecida com a mostrada na Figura 1. Se não estiver apto a rodar o Wireshark em uma conexão de rede, você pode fazer o download do pacote que foi criado, quando os passos acima foram executados[2].

[2]Baixe o arquivo zip http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip e extraia o arquivo http-ethereal-trace-1. O arquivo zip foi coletado quando o autor do livro rodou o Wireshark enquanto executava os passos para o lab. Após baixar o arquivo, você pode abrir no Wireshark e ver as informações usando File -> Open e selecionando o arquivo http-ethereal-trace-1 trace. A tela resultante deve ser parecida com a Figura 1.

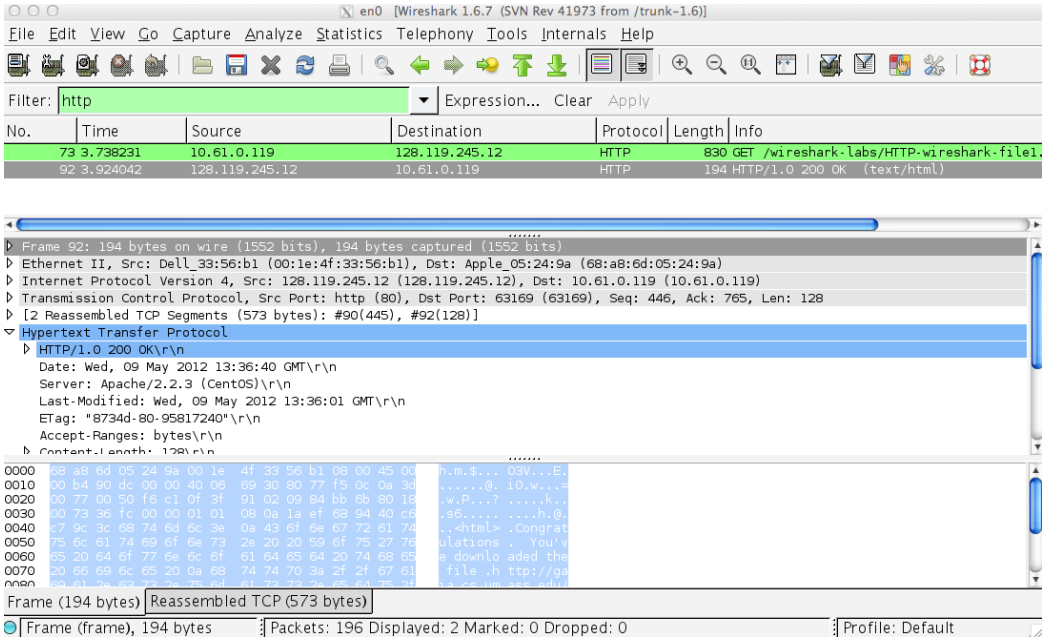

Figura 1. Tela do Wireshark depois de entrar em http://gaia.cs.umass.edu/wireshark-labs/ HTTP-wireshark-file1.html

O exemplo na Figura 1 mostra na janela de listagem duas mensagens HTTP capturadas: uma mensagem de GET (de um navegador para o servidor gaia.cs.umass.edu) e uma mensagem de resposta do servidor para o navegador. A janela com o conteúdo dos

pacotes mostra detalhes da mensagem selecionada. Lembre-se de que a mensagem HTTP é carregada num segmento TCP, o qual está num datagrama IP, o qual está num quadro Ethernet. O Wireshark mostra informações do quadro, Ethernet, IP e TCP. Queremos minimizar a quantidade de dados que não são HTTP, então mostre apenas as informações de HTTP no Wireshark e esconda o restante.

Observando a informação do GET HTTP e a mensagem de resposta, responda as questões abaixo. Quando responder, inclua uma impressão das mensagens de GET e a resposta, e indicar onde você encontrou as respostas.

a. O seu navegador está rodando a versão 1.0 ou 1.1 do HTTP? Que versão do HTTP está sendo rodada no servidor?
b. Que linguagem (se tiver) o seu navegador indica que ele pode aceitar do servidor?
c. Qual é o endereço IP do seu computador? E do servidor gaia.cs.umass.edu?
d. Qual é o código de estado retornado pelo servidor para o seu navegador?
e. Quando foi a última alteração feita no arquivo HTML?
f. Quantos bytes de conteúdo são retornados para o seu navegador?

2. GET HTTP CONDITIONAL/interação de resposta

Antes de fazer esta parte, certifique-se de que o cache do navegador esteja limpo (faça isso usando a opção para apagar o histórico do navegador). Agora, faça o seguinte:

- Inicie o navegador e certifique-se de que o cache esteja limpo.
- Inicie o Wireshark.
- Entre com a URL no navegador http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file2.html
  Seu navegador deve mostrar um arquivo bem simples HTML com cinco linhas.
- Rapidamente entre com a URL no navegador novamente (ou simplesmente selecione o botão para recarregar a página).
- Pare o Wireshark e entre com “http” na janela em que se especifica filtros para que somente as mensagens HTTP possam ser mostradas na janela de listagem de pacotes.
- (Nota: Se você não puder rodar o Wireshark em uma conexão interativamente, use o disposto no rodapé 2 para responder as perguntas abaixo)

Responda às seguintes questões:
g. Inspecione o conteúdo da primeira requisição GET do HTTP de seu navegador para o servidor. Você viu uma linha com “IF-MODIFIED-SINCE” no GET do HTTP?
h. Inspecione o conteúdo da resposta do servidor. O servidor retornou o conteúdo do arquivo explicitamente? Como isso se observou?
i. Agora investigue o conteúdo da segunda requisição GET do HTTP de seu navegador para o servidor. Você viu uma linha com “IF-MODIFIED-SINCE” no GET do HTTP? Se sim, que informação segue o cabeçalho “IF-MODIFIED-SINCE:”?
j. Qual é o código de estado do HTTP e a frase retornada do servidor em resposta ao segundo GET do HTTP? O servidor explicitamente retornou o conteúdo do arquivo? Explique