EA872 Laboratório de Programação de Software Básico
Atividade 4

Vinícius Esperança Mantovani **RA 247395**

Entrega (limite): 23/08/2023, em sala de aula

Exercício 9.1.1-1)

O programa “http-dump.c” funciona do seguinte modo:

1) De início, há a definição de algumas variáveis e, em seguida, a verificação do número de argumentos, para garantir que tenha-se um argumento único, que é número da porta em que o servidor será aberto.
2) Em seguida, é aberto um soquete para aceitar os pedidos de conexão recebidos pelo servidor por meio da porta determinada.
3) Depois de criado o soquete, ele é ligado ao endereço do servidor, ou seja, passa a ser vinculado ao servidor aberto pelo programa, para permitir que o servidor aceite eventuais conexões com ele.
4) Em seguida, o código implementa uma fila de até cinco tentativas de conexão, ou seja, permite que cinco pedidos de conexão fiquem pendentes enquanto um deles é executado.

Até o ponto acima, as mensagens impressas no terminal são apenas mensagens correspondentes a erros, como na criação de soquete, na associação do soquete ao servidor…

5) Após todo esse processo de inicialização e configuração do servidor e de sua porta, é amostrada uma mensagem comunicando que o servidor já está aceitando a comunicação (Figura 1) e, inicia-se um laço infinito.

```
[pininsu@fedora ativ4]$ ./http-dump 5544

./http-dump já está aceitando conexões de clientes HTTP.
```

Figura 1: mensagem de aceitação do servidor

6) Neste laço, de início, é aceito um pedido de comunicação e, em caso de erro é mostrada uma mensagem referente a isso. Essa aceitação é feita por meio de uma função chamada “accept”.
7) Em sequência, o soquete é posto para receber uma mensagem por meio da função “recv” e, em caso de erro, uma mensagem é mostrada.

8) Por fim, o programa imprime o que recebeu de mensagem no soquete loopando entre as mensagens armazenadas em “area” para imprimí-las.
9) Por fim, são fechados os soquetes e a comunicação.

O resultado da execução do programa em um exemplo de output para a comunicação feita do browser “Firefox” por mensagens passadas ao servidor está na Figura 2.

```
Mensagem recebida:
GET / HTTP/1.1
Host: localhost:5544
Connection: keep-alive
Cache-Control: max-age=0
sec-ch-ua: "Chromium";v="116", "Not)A;Brand";v="24", "Brave";v="116"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "Linux"
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/a
Sec-GPC: 1
Accept-Language: en-US,en
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Accept-Encoding: gzip, deflate, br
```

Figura 2: Output para uma tentativa de conexão do “Firefox”

Exercício 9.1.1-2)

a) GET: Solicita dados determinados do servidor para serem enviados ao cliente.
b) ACCEPT = text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8. Apresenta ao servidor os tipos de conteúdo que podem ser entendidos pelo cliente.
c) ACCEPT-ENCODING = gzip, deflate, br. Apresenta ao servidor os tipos de codificações que o cliente é capaz de decodificar e entender.
d) ACCEPT-LANGUAGE = en-US,en. Específica que a linguagem preferida pelo cliente é inglês estadunidense.
e) CACHE-CONTROL = max-age=0. Significa que o cache o cache é mantido por 0 segundos. Ou seja, a resposta é memorizada pelo cache por 0 segundos (não é armazenada).
f) CONNECTION = keep-alive. Significa que o modo de conexão é “keep-alive”, ou seja, a conexão se mantém enquanto o navegador está se comunicando com o servidor.
g) HOST = localhost:5544, significa que o servidor está hospedado localmente e, a conexão com ele é feita pela porta 5544;
h) USER_AGENT = Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36.Significa que a tentativa de conexão e as mensagens foram feitas e passadas pelo navegador da “Mozilla”, “Firefox”.

Todos os valores estão na Figura 2.

Exercício 9.1.2-1)

a) É necessário especificar o port por conta de o padrão telnet ser 23, logo, ao tentar conectar em um servidor que opera com protocolo http, no caso de não especificarmos o port 80 incorremos em uma tentativa infindável de conexão com o servidor. Ou seja, se não especificamos a porta como 80, o servidor não pode ser acessado, pois telnet busca na porta 23.

b) No caso do teste feito, foi estabelecida a conexão da mesma forma que foi com a port 443, ao menos, aparentemente (o output no terminal foi o mesmo, apresentado na Figura 3 abaixo). No entanto, pesquisando a respeito, é possível ver que o port 443 é utilizado para serviços com protocolo HTTPS, ou seja, essa porta é usada para transações seguras entre cliente e servidor.

```
[pininsu@fedora ativ4]$ telnet www.dca.fee.unicamp.br 80
Trying 143.106.148.94...
Connected to www.dca.fee.unicamp.br.
Escape character is '^]'.
Connection closed by foreign host.
```

Figura 3: Resposta do terminal para a conexão por meio do port 80 ao servidor pedido

Exercício 9.1.2-2)

a)

```
HTTP/1.1 302 Found
Date: Wed, 23 Aug 2023 19:16:13 GMT
Server: Apache
Location: https://www.dca.fee.unicamp.br/
Content-Length: 215
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>302 Found</title>
</head><body>
<h1>Found</h1>
<p>The document has moved <a href="https://www.dca.fee.unicamp.br/">here</a>.</p>
</body></html>
```

b) A diferença entre o resultado da figura abaixo, usando “head” e o da figura acima “GET”, está no fato de que o HEAD solicita ao servidor o envio de informações sobre o recurso em questão (HTTP/1.1), enquanto que o GET, além disso, solicita o recurso em sí.

```
HTTP/1.1 302 Found
Date: Wed, 23 Aug 2023 19:18:57 GMT
Server: Apache
Location: https://www.dca.fee.unicamp.br/
Content-Type: text/html; charset=iso-8859-1
```

c) Não há diferenças entre o que se expõe na imagem abaixo (GET /index.html HTTP/1.1) e o que se expôs anteriormente (GET HTTP/1.1), mas, cabe ressaltar que o uso de /index.html significa que o que se quer receber é o recurso HTTP/1.1 da página /index.html e, essa ausência de diferenças ocorre por conta de /index.html ser o arquivo padrão do site, o que faz com que GET procure nele por padrão.

```
HTTP/1.1 302 Found
Date: Wed, 23 Aug 2023 19:20:17 GMT
Server: Apache
Location: https://www.dca.fee.unicamp.br/index.html
Content-Length: 225
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>302 Found</title>
</head><body>
<h1>Found</h1>
<p>The document has moved <a href="https://www.dca.fee.unicamp.br/index.html">here</a>.</p>
</body></html>
```

d) Ocorre neste caso que a mensagem enviada ao servidor está escrita de maneira incorreta, portanto, o servidor retorna uma mensagem “Bad Request” nos informando sobre isso e, apesar de executar o comando, a conexão com o servidor é encerrada por seu host.

```
GET HTTP/1.1
Host: www.fee.unicamp.brHTTP/1.1 400 Bad Request
Date: Wed, 23 Aug 2023 19:22:15 GMT
Server: Apache
Content-Length: 226
Connection: close
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>400 Bad Request</title>
</head><body>
<h1>Bad Request</h1>
<p>Your browser sent a request that this server could not understand.<br />
</p>
</body></html>
Connection closed by foreign host.
```