Universidade Estadual de Campinas

Faculdade de Engenharia Elétrica e de Computação

Laboratório de Software Básico (EA872)

Relatório 4
Protocolo HTTP

Aluna: Gabriela de Castro Surita
gabsurita@gmail.com

Professor: Christian Rodolfo Esteve Rothenberg

Exercícios

1. (1 pt) Faça um telnet para uma máquina onde exista um servidor HTTP real instalado, passando o port do servidor (80).

(a) (0,2 pts) Envie o comando, documente o resultado e explique a diferença entre o resultado de OPTIONS e o de HEAD feito anteriormente no mesmo servidor.

O resultado do comando OPTIONS difere de HEAD pois as informações retornadas são diferentes. Em especial, o comando OPTIONS retorna os métodos aceitos pelo servidor (campo "Allow").

```
OPTIONS / HTTP/1.1
Host: www.dca.fee.unicamp.br

HTTP/1.1 200 OK
Date: Sat, 17 Sep 2016 02:51:08 GMT
Server: Apache/2.2.22 (Ubuntu)
Allow: OPTIONS,GET,HEAD,POST
Vary: Accept-Encoding
Content-Length: 0
Content-Type: text/html
```

(b) (0,2 pts) Envie o comando, documente o resultado e explique se houve (e qual foi a) diferença entre o resultado esperado e o obtido.

Como indica a informação obtida pelo campo OPTIONS, o método TRACE não foi autorizado pelo servidor e uma mensagem de erro (405 - "Method not allowed") for retornada.

```
TRACE / HTTP/1.1
Host: www.dca.fee.unicamp.br

HTTP/1.1 405 Method Not Allowed
Date: Sat, 17 Sep 2016 02:59:31 GMT
Server: Apache/2.2.22 (Ubuntu)
Allow:
Vary: Accept-Encoding
Content-Length: 311
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD␣HTML␣2.0//EN">
```

```
<html><head>
<title>405 Method Not Allowed</title>
</head><body>
<h1>Method Not Allowed</h1>
<p>The requested method TRACE is not allowed for the URL /.</p>
<hr>
<address>Apache/2.2.22 (Ubuntu) Server at www.dca.fee.unicamp.br
Port 80</address>
</body></html>
```

(c) (0,2 pts) Envie o comando, documente o resultado e explique se houve (e qual foi a) diferença entre o resultado esperado e o obtido.

Conforme esperado, o resultado do POST é um acesso bem sucedido, similar a resposta recebida pelo GET neste caso específico (Em geral, POST é utilizado para submissão de conteúdo, como formulários, pelo cliente HTTP).

```
POST / HTTP/1.1
Host: www.dca.fee.unicamp.br

HTTP/1.1 200 OK
Date: Sat, 17 Sep 2016 03:04:12 GMT
Server: Apache/2.2.22 (Ubuntu)
Last-Modified: Tue, 13 May 2014 19:54:12 GMT
ETag: "622a7-d6-4f94d6d90fb93"
Accept-Ranges: bytes
Content-Length: 214
Vary: Accept-Encoding
Content-Type: text/html
X-Pad: avoid browser bug

<html>
<head>
<meta http-equiv="Refresh" content="0;
url=http://www.fee.unicamp.br/?q=node/145"/>
</head>
<body>
<p>Please follow
<a href="http://www.fee.unicamp.br/?q=node/145/">this link</a>.</p>
</body>
</html>
```

(d) (0,2 pts) Envie o comando, documente o resultado e explique se houve (e qual foi a) diferença entre o resultado esperado e o obtido.

A resposta novamente é não autorizada, igual ao método TRACE, conforme esperado.

```
DELETE / HTTP/1.1
Host: www.dca.fee.unicamp.br

HTTP/1.1 405 Method Not Allowed
Date: Sat, 17 Sep 2016 03:07:54 GMT
Server: Apache/2.2.22 (Ubuntu)
Allow: OPTIONS,GET,HEAD,POST
Vary: Accept-Encoding
Content-Length: 322
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD␣HTML␣2.0//EN">
<html><head>
<title>405 Method Not Allowed</title>
</head><body>
<h1>Method Not Allowed</h1>
<p>The requested method DELETE is not allowed for the URL /index.html.</p>
<hr>
<address>Apache/2.2.22 (Ubuntu) Server at www.dca.fee.unicamp.br
Port 80</address>
</body></html>
Connection closed by foreign host.
```

(e) (0,2 pts) Envie o comando, documente o resultado e explique se houve (e qual foi a) diferença entre o resultado esperado e o obtido.

O método AGORA_VAI não é um método padrão em servidores HTTP, e não está implementado no servidor em questão. Sendo assim, a resposta obtida é de método não implementado (501). Vale notar que a resposta incluí uma chamada semelhante à OPTIONS seguida de uma mensagem de erro.

```
AGORA_VAI / HTTP/1.1
Host: www.dca.fee.unicamp.br

HTTP/1.1 501 Method Not Implemented
Date: Sat, 17 Sep 2016 03:09:55 GMT
```

```
Server: Apache/2.2.22 (Ubuntu)
Allow: OPTIONS,GET,HEAD,POST
Vary: Accept-Encoding
Content-Length: 309
Connection: close
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD␣HTML␣2.0//EN">
<html><head>
<title>501 Method Not Implemented</title>
</head><body>
<h1>Method Not Implemented</h1>
<p>AGORA_VAI to /index.html not supported.<br />
</p>
<hr>
<address>Apache/2.2.22 (Ubuntu) Server at www.dca.fee.unicamp.br
Port 80</address>
</body></html>
```

2. (0,5 pts) Crie e apresente testes de troca de mensagens com um servidor web usando telnet, testes estes que resultem em pelo menos 5 códigos de erro HTTP distintos.

1. Exemplo de método não implementado (501). Quando o método especificado no cabeçalho não é descrito no servidor.

```
$ telnet www.fee.unicamp.br 80
AAA
<!DOCTYPE HTML PUBLIC "-//IETF//DTD␣HTML␣2.0//EN">
<html><head>
<title>501 Method Not Implemented</title>
</head><body>
<h1>Method Not Implemented</h1>
<p>AAA to /index.html not supported.<br />
</p>
<hr>
<address>Apache/2.2.22 (Ubuntu) Server at vm1.fee.unicamp.br Port 80</addres
</body></html>
```

2. Exemplo de requisição errada (400). Quando os campos do cabeçalho não correspondem aos valores esperados.

```
$ telnet www.fee.unicamp.br 80
A A A

HTTP/1.1 400 Bad Request
Date: Sat, 17 Sep 2016 14:07:52 GMT
Server: Apache/2.2.22 (Ubuntu)
Vary: Accept-Encoding
Content-Length: 310
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>
<hr>
<address>Apache/2.2.22 (Ubuntu) Server at vm1.fee.unicamp.br Port 80</addres
</body></html>
```

3. Exemplo de página movida pemanentemente (301). Utilizado quando um endereço é obsoleto ou, no caso, é apenas um atalho para outro.

```
$ telnet www.fb.com.br 80
GET / HTTP/1.1
Host: www.fb.com.br

HTTP/1.1 301 Moved Permanently
Server: nginx/1.8.0
Date: Sat, 17 Sep 2016 14:09:13 GMT
Content-Type: text/html
Content-Length: 0
Connection: keep-alive
X-Powered-By: PHP/5.4.43
Cache-Control: public, max-age=15
X-Abuse: URL redirection provided by freedns.afraid.org - please report
any misuse of this service
Location: http://www.viajandoparaorlando.com.br
```

4. Exemplo de redireção com página encontrada (302). Utilizado quando um endereço refere-se à outro devido à algum motivo próprio. No caso, por questões de região e língua, a versão

brasileira do site é indicada. Esta requisição difere da anterior (301) pois não é incondicional, ou seja, podem existir casos onde a redireção não é exatamente esta.

```
$ telnet www.google.com 80
Trying 190.98.170.148...
Connected to www.google.com.
Escape character is '^]'.
GET / HTTP/1.1
Host: www.google.com

HTTP/1.1 302 Found
Cache-Control: private
Content-Type: text/html; charset=UTF-8
Location: http://www.google.com.br/?gfe_rd=cr&ei=Z5HxVezQKMiB8QeY747ABg
Content-Length: 262
Date: Sat, 17 Sep 2016 14:19:19 GMT
Server: GFE/2.0

<HTML><HEAD><meta http-equiv="content-type"
content="text/html;charset=utf-8">
<TITLE>302 Moved</TITLE></HEAD><BODY>
<H1>302 Moved</H1>
The document has moved
<A HREF="http://www.google.com.br/?gfe_rd=cr&amp;ei=Z5HxVezQKMiB8Qe
Y747ABg">here</A>.
</BODY></HTML>
```

5. Exemplo de acesso não autorizado (401). No caso, a página em questão exige autenticação.

```
$ telnet wiki.lsc.ic.unicamp.br 80
GET / HTTP/1.1
Host: wiki.lsc.ic.unicamp.br

HTTP/1.1 401 Unauthorized
Server: nginx/1.8.0
Date: Sat, 17 Sep 2016 14:23:22 GMT
Content-Type: text/html
Content-Length: 194
Connection: keep-alive
WWW-Authenticate: Basic realm="Restricted"

<html>
<head><title>401 Authorization Required</title></head>
<body bgcolor="white">
```

```
<center><h1>401 Authorization Required</h1></center>
<hr><center>nginx/1.8.0</center>
</body>
</html>
```

3. (4 pts) Redirecione a saída de http-dump para um arquivo de saída qualquer e acesse este "servidor" com dois navegadores de fornecedores diferentes (Firefox, Internet Explorer, Chrome, Opera, Safari etc). Em seguida adapte o programa criado no exercício 5 da Atividade 3 para guardar os comandos, parâmetros e valores de requisições HTTP nas listas como especificado naquele exercício. Observe que a linha inicial que traz um dos poucos comandos HTTP não tem o símbolo de dois pontos (:) e precisará de tratamento especial para separar e guardar seus parâmetros. É preciso atentar também para linhas que tenham este símbolo mais de uma vez (por causa da especificação de número de porta). Além de documentar seu código no relatório, coloque exemplos de funcionamento que mostrem o programa varrendo as listas e imprimindo na tela todos os dados obtidos das mesmas.

(a) Exemplos de entrada e saída

```
$ cat request.http
GET / HTTP/1.1
Host: localhost:8080
Connection: keep-alive
Cache-Control: max-age=0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (X11; Fedora; Linux x86_64) AppleWebKit/537.36 (KHT
Accept-Encoding: gzip, deflate, sdch
Accept-Language: en-US,en;q=0.8,pt-BR;q=0.6,pt;q=0.4
Cookie: _ga=GA1.1.590075715.1465236650

$ ./http < request.http
Host: localhost
Requisicao: GET
Local: /
Protocolo: HTTP/1.1
Host: localhost
Porta: 8080
Connection :  keep-alive,
Cache-Control :  max-age=0,
```

```
Accept :  text/html, application/xhtml+xml, application/xml;q=0.9, image/we
Upgrade-Insecure-Requests :  1,
User-Agent :  Mozilla/5.0 (X11; Fedora; Linux x86_64) AppleWebKit/537.36 (K
like Gecko) Chrome/50.0.2661.86 Safari/537.36,
Accept-Encoding :  gzip,  deflate,  sdch,
Accept-Language :  en-US, en;q=0.8, pt-BR;q=0.6, pt;q=0.4,
Cookie :  _ga=GA1.1.590075715.1465236650,


$ cat request.http
GET / HTTP/1.1
Host: 127.0.0.1:8080
User-Agent: Mozilla/5.0 (X11; Fedora; Linux x86_64; rv:38.0) Gecko/20100101
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive

$ ./http < request.http
Host: 127.0.0.1
Requisicao: GET
Local: /
Protocolo: HTTP/1.1
Host: 127.0.0.1
Porta: 8080
User-Agent : Mozilla/5.0(X11;Fedora;Linuxx86_64;rv : 38.0) Gecko/20100101 F
Accept :  text/html, application/xhtml+xml, application/xml;q=0.9, */*;q=0.8
Accept-Language :  en-US, en;q=0.5,
Accept-Encoding :  gzip,  deflate,
Connection :  keep-alive,
```

(b) Analisador léxico modificado

```
%{
  #include "http.tab.h"
%}

%%
#.*                     {}
,                       {return VIRGULA;}
Host[ ]*:               {return HOST;}
[^,:\n\t\r]+:           {strcpy(yylval.text, yytext); return COMANDO;}
[^,:\n\t\r]+            {strcpy(yylval.text, yytext); return PARAMETRO;}
[\n]                    {return NOVALINHA;}
.                       {}
%%
```

(c) Analisador sintático modificado

```
%{
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "queue.h"

/* Tipo comando */
typedef struct comando{
  char* texto;
  Queue* filaParam;
} comando;

char* req  = NULL;
char* path = NULL;
char* protocolo = NULL;
char* host = NULL;
int   porta = 80;

/* Fila de comandos */
Queue* filaComandos;

/* Comando atual para insercao de parametros */
comando* comandoAtual;

/* Contador de Linha */
int nlinhas = 1;

/* Remove ocorrencias de caracter em string */
void clearString(char* input, char rem);

%}

%union {
  char text[1024];
}

%token
COMANDO
PARAMETRO
VIRGULA
DOISPONTOS
NOVALINHA
HOST
```

```
%%
prog:
requisicao linhas | linha;

linhas:
linha linhas | linha ;

linha:
comando | parametros | host | fimLinha;

parametros:
parametro VIRGULA parametros | parametro;

comando:
COMANDO {

  /* Limpa a string de ":" e " " */
  clearString(yylval.text, ' ');
  clearString(yylval.text, ':');

  //printf("Comando: %s\n", yylval.text);

  /* Cria novo comando */
  comandoAtual = (comando*)malloc(sizeof(comando));
  char* texto = (char*)malloc(sizeof(char)*strlen(yylval.text));
  strcpy(texto, yylval.text);
  comandoAtual->texto = texto;
  comandoAtual->filaParam = newQueue();

  /* Insere comando na fila */
  inQueue(&filaComandos, (char*)comandoAtual);

};

parametro:
PARAMETRO {

  //printf("Parametro: %s\n", yylval.text);

  /* Verifica se o comando atual e valido */
  if(comandoAtual){
    /* Cria novo parametro */
    char* texto = (char*)malloc(sizeof(char)*strlen(yylval.text));
    strcpy(texto, yylval.text);
```

```
    /* Insere parametro na fila */
    Queue* parametros = comandoAtual->filaParam;
    inQueue(&parametros, texto);
  }
  /* Caso: parametro em comando invalido */
  else{
    printf("Comando␣invalido␣na␣linha␣%d\n", nlinhas);
  }
};

requisicao:
PARAMETRO{

  char* texto = yylval.text;

  /* Divide string por espacos */
  for(int i=0; i<strlen(texto); i++){
    if(texto[i]=='␣'){
      texto[i] = '\0';
    }
  }

  /* Monta requisicao */
  req =  (char*)malloc(sizeof(char)*strlen(texto));
  strcpy(req, texto);

  texto+=strlen(req);

  path = (char*)malloc(sizeof(char)*strlen(texto));
  strcpy(path, texto);

  texto+=strlen(path)+1;

  protocolo = (char*)malloc(sizeof(char)*strlen(texto));
  strcpy(protocolo, texto);
}

fimLinha

;

host:

HOST
```

```
COMANDO{
  /* Limpa a string de ":" e " " */
  clearString(yylval.text, '␣');
  clearString(yylval.text, ':');

  printf("Host:␣%s\n", yylval.text);

  host = (char*)malloc(sizeof(char)*strlen(yylval.text));
  strcpy(host, yylval.text);
}


PARAMETRO{
  clearString(yylval.text, '␣');
  clearString(yylval.text, ':');

  porta = atoi(yylval.text);
}

;


fimLinha:
NOVALINHA{
  nlinhas++;
  comandoAtual = NULL;
};

%%


int main(){

  filaComandos = newQueue();
  yyparse();

  printf("Requisicao:␣%s\n", req);
  printf("Local:␣%s\n", path);
  printf("Protocolo:␣%s\n", protocolo);
  printf("Host:␣%s\n", host);
  printf("Porta:␣%d\n", porta);


  /* Imprime conteudo da lista */
  while(notNullQueue(&filaComandos)){
```

```
    comando* iter = (comando*)outQueue(&filaComandos);

    char* texto = iter->texto;
    printf("%s : ", texto);
    free(texto);

    Queue* params = iter->filaParam;

    while(notNullQueue(&params)){
      char* texto = outQueue(&params);
      printf("%s, ", texto);
      free(texto);
    }

    destroyQueue(&params);
    free(iter);
    printf("\n");
  }

  destroyQueue(&filaComandos);

  free(req);
  free(path);
  free(protocolo);
  free(host);
}


/* Remove ocorrencias de caracter em string
    Adaptado de http://stackoverflow.com/questions/4161822/ */
void clearString(char* input, char rem){

  char *src, *dest;
  src = dest = input;

  while(*src != '\0'){
      if(*src != rem){
          *dest = *src;
          dest++;
      }
      src++;
  }
  *dest = '\0';
}
```