Computer Networks - HW 2

Report : Simple Web Server (C)

Stan Andrei (2E3)

andrei.stan@info.uaic.ro

Decembrie 2020

Universitatea Alexandru Ioan Cuza
Facultatea de Informatica Iasi

1 Introducere

Acest document consta in prezentarea raportului, aceasta avand ca si baza proiectul "Simple Web server".

Pentru acest proiect trebuie construit un server concurent ce deschide / afiseaza clientiilor conectati o pagina html principala, aceasta pagina avand acces spre fisierele .txt si .html din directorului curent.

Conectarea la web server se face dintr-un browser cu o adressa anume, iar la accesarea unui fisier se afiseaza adresa HTTP respectiva.

2 Tehnologiile utilizate

- Am folosit prptocolul de tip TCP.

- Implementarea TCP a fost realizata prin socket-uri.

- Transmisia de date se realizeaza printr-un fork.

- In codul C a fost implementa pagina HTML impreuna cu elemente CSS.

3 Arhitectura aplicatiei

Cum serverul trebuie sa fie concurent (multicast / broadcast), am folosit protocolul de tip TCP. Nu pot folosi UDP, acesta fiind unicast.

Pagina html (+ css) cu content-ul necesar este implementat in codul c printr-un string fara limita, astfel permitand o oarecare simplificare a codului, pastrand totul la un loc, evitand folosirea altor fisiere.

Chiar daca in pagina (structura) html sunt link-uri spre fisierele necesare, acestea nu pot fi accesate decat daca serverul trimite datele acestora spre client (prin fork, cu write, fd, sendfile, etc).

4 Detalii de implementare

- Implementarea protocolului va consta in socket-uri, file, client, server descriptors.

```c
struct sockaddr_in server, from;
socklen_t sin_len = sizeof(from);
int sd, cd;

int on1 = 1;
int on2 = 1;

char buf[2048];
int fdimg;
int fdtxt;
int fdhtml;
int sent;
int len;

if((sd = socket(AF_INET, SOCK_STREAM, 0)) == -1)
  {
    perror("[Error]: Socket");
    exit(1);
  }

bzero(&server,sizeof(server));
```

```c
server.sin_family = AF_INET;
server.sin_addr.s_addr = INADDR_ANY;
server.sin_port = htons(2077);

setsockopt(sd, SOL_SOCKET, SO_REUSEADDR, (void *) &on1, sizeof(int));


if(bind(sd, (struct sockaddr *) &server, sizeof(server)) == -1)
  {
    perror("[Error]: bind");
    close(sd);
    exit(1);
  }

if(listen(sd, 5) == -1)
  {
    perror("[Error]: listen");
    close(sd);
    exit(1);
  }
```

Figure 2:

- Se va face o bucla continua pentru primirea clientiilor, bufferul aratandu-ne in timp real cand se conecteaza un client (pe langa alte detalii precum os, si browserul clientului)

```
while(1)
  {
    len = sizeof(from);

  cd = accept(sd, (struct sockaddr *) &from, &sin_len);

  if(cd < 0)
    {
      perror("[Error]: The connection to the client has failed...\n");
      continue;
    }

  printf("[Info]: A client has connected!\n");
  fflush(stdout);

  sleep(0);

  if(!fork())
```

- Deasemenea si intr-un fork (sau mai multe) se va realiza transmiterea datelor (fisierelor) spre client, fisierele fiind hard-codate in program (exista posibilitate de automatizare).

```
if(!fork())
    {
      close(sd);
      memset(buf, 0, 2048);
      read(cd, buf, 2047);

      printf("%s", buf);

      if (!strncmp(buf, "GET /Background.jpg", 13))
          {
            printf("[Info]: Request to send file... getting file... sending bytes...\n");
            write(cd, imageheader, sizeof(imageheader) - 1);
            fdimg = open("Background.jpg", O_RDONLY);
            sent = sendfile(cd, fdimg, NULL, 1000000);
            printf("[Info]: File sent... number of bytes sent : %d \n\n", sent);
            close(fdimg);
          }
      else if (!strncmp(buf, "GET /test.txt", 13))
```

- Pagina html va fi implementata in program deasupra la main, primele 2 linii si '/r/n' fiind necesare la o astfel de implementare, implementarea codului html fiind unul comun.

```
char webpage[] =
"HTTP/1.1 200 Ok\r\n"
"Content-Type: text/html; charset=UTF-8\r\n\r\n"
"<!DOCTYPE html>\r\n"
"<html><head><title>Test Title</title>\r\n"
"<style> body { background-image: url('http://localhost:2077/Background.jpg'); background-repeat: repeat;} </style>\r\n"
"<style>"
"a:link {color: red;}"
"a:visited {color: red;}"
"a:hover {color: green;}"
"a:active {color: blue;}"
"</style>\r\n"
"<body>\r\n"
"<p><b><a href='http://localhost:2077/test.txt' target='_blank'>This is a link</a></b></p>"
"<a href='http://localhost:2077/test.txt'>Click here to open test.txt!</a><br>\r\n"
"<a href='test.txt' download>Click here to download test.txt!</a></body><br><br>\r\n"
"<a href='http://localhost:2077/test.html'>Click here to open test.html!</a><br>\r\n"
"<a href='test.html' download>Click here to download test.html!</a></body><br>\r\n"
"</html>\r\n";
```

- (Bonus) Pe langa acestea mi-am configurat router-ul (port fowarding protocols, dmz mapping, etc) si firewall-ul de la OS, pentru ca server-ul sa fie posibil de accesat de pe orice browser, de pe orice calculator, telefon, tableta, etc.

Conectarea la server se poate face fie local (localhost:2077), fie public (92.86.74.50:2077).

- (Bonus) Fisierele au butoane de descarcare.

5 Concluzii

Va trebui sa inbunatatesc caliatatea paginii HTML, sa ii dau mai multe elemente de aspect / functionabilitate.

Deasemenea va trebui sa incerc sa optimizez lucrul cu fisiere (acces, transfer, etc).

Alta concluzie : imi place sa lucrez cu web server :)

6 Bibliografie

- Cursurile si Seminarele profesorilor Paun Andrei si Alboaie Lenuta : Click me!.

- Tipurile de MIME : Click me!

- Informatii despre implementarea html in c : Click me!

- Tips and error fixes : friends and the internet