

TP3 d'Informatique : Mini Serveur Web

By Lahbabi Chems Eddine et Bonne Emelyne

6 - Fonctionnement du protocole HTTP , observation du fonctionnement à l'aide de telnet.

```
(23:39:34) >> telnet www.stri.ups-tlse.fr 80
Trying 195.220.59.62...
Connected to lifou-www-nom-ups.cict.fr.
Escape character is '^]'.
GET /index.php HTTP/1.1
Host : www.stri.ups-tlse.fr
Accept : text/html
Accept : text/plain
User-Agent : Lynx/2.4 libwww/2.1.4

HTTP/1.1 200 OK
Date: Tue, 08 Nov 2016 22:44:21 GMT
Server: Apache
Transfer-Encoding: chunked
Content-Type: text/html

2096
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="en-US" prefix="og: http://ogp.me/ns#">
<head profile="http://gmpg.org/xfn/11">
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
```

7- Modification du mainClient:

```
#include <stdio.h>
#include <stdlib.h>
#include "client.h"

int main() {
    char *message;

    if(InitialisationAvecService("www.stri.ups-tlse.fr", "80") != 1) {
        printf("Erreur d'initialisation\n");
        return 1;
    }

    if(Emission("GET /index.php HTTP/1.1\nHost : www.stri.ups-tlse.fr\n\
Accept : text/html\nAccept : text/plain\nUser-Agent : Lynx/2.4 libwww/2.1.4\n\n")!=1) {
        printf("Erreur d'mission\n");
        return 1;
    }
}
```



```
while((message = Reception()) != NULL)
{
    printf("%s\n", message);
    free(message);
}

Terminaison();

return 0;
}
```

Exécution du programme :

```
(22:05:23) >> ./mainClient
Connexion avec le serveur reussie.
Emission de 128 caracteres.
HTTP/1.1 200 OK

Date: Tue, 15 Nov 2016 21:05:21 GMT
Server: Apache
Connection: close
Content-Type: text/html

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" lang="en-US" prefix="og: http://ogp.me/ns#">

<head profile="http://gmpg.org/xfn/11">

    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
```

8-a- Fonction extraitNomFichier

```
int extraitFichier(char *requete, char *tableauNomFichier, int tailleTableauNomFichier)
{
    int i;
    //Si la requete n'est pas GET la fonction retourne -1
    if(requete[0] != 'G' && requete[1] != 'E' && requete[2] != 'T')
        return -1;
    /* copier le nombre 'tailleTableauNomFichier' caractères après 'GET /'
       si dans la requete il y a plus de 5 éléments donc plus de 'GET /*
    if((strlen(requete) > 5)&&(strlen(requete)>tailleTableauNomFichier))
        memcpy(tableauNomFichier, &requete[5], tailleTableauNomFichier);
    else if((strlen(requete) > 5)&&(strlen(requete)<tailleTableauNomFichier))
        memcpy(tableauNomFichier, &requete[5], strlen(requete)-5);
    else
        return -2;
    /*Dans ces caractères copier précédement prendre le premier mot jusqu'à espace
    cette boucle compte le nombre de caractère jusqu'au prochain espace */
    for (i = 0; tableauNomFichier[i] != ' '; ++i)
    {
        if(tableauNomFichier[i] == '\n')
        {
            return -2;
        }
    }
    /*remplacer l'espace trouvé dans la chaine par '\0'
       ainsi dans le tableauNomFichier on retrouvera que le nom du fichier
       si le nom du fichier contient un espace, il y aura une erreur
    */
    tableauNomFichier[i] = '\0';

    return 0;
}
```



8-b- Fonction longueurFichier

```
size_t longueur_fichier(char *nomFichier){  
    FILE *fp;  
    int long size;  
  
    fp = fopen(nomFichier, "r");  
    if(fp == NULL)  
        return -1;  
  
    fseek(fp, 0, SEEK_END);  
    size = ftell(fp);  
    fclose(fp);  
  
    return size;  
}
```

8-c- Fonction envoyerContenuFichierTexte

```
int envoyerContenuFichierTexte(char *nomFichier){  
  
    FILE *fp;  
    char ligne[70];  
    fp = fopen( nomFichier, "r");  
    if(fp == 0)  
        return -1;  
    while( !(feof(fp)) )  
    {  
        if(fgets(ligne , 70, fp) != NULL)  
        {  
            if(Emission(ligne)!=1)  
                printf("Erreur d'emission\n");  
        }  
    }  
    fclose(fp);  
    return 0;  
}
```



8-d-Fonction `envoyerReponse200HTML`

```
int envoyerReponse200HTML(char *nomFichier)
{
    char req[500];
    char len[2];
    int lenght = 0;
    lenght = longueur_fichier(nomFichier);
    sprintf(len, "%d", lenght);
    strcpy(req, "\nHTTP:1.1 200 OK\nContent-type: text/html\nContent-length: ");
    strcat(req, len);
    strcat(req, "\n");
    if(Emission(req)!=1)
        printf("Erreur d'emission\n");
    return 0;
}
```

Teste de la fonction:

J'ai reçu: GET /Makefile HTTP/1.1	(17:29:38) >> ./mainClient
Emission de 26 caracteres.	Connexion avec le serveur reussie.
J'ai reçu: Host : 127.0.0.1	Emission de 116 caracteres.
Emission de 26 caracteres.	J'ai reçu:
J'ai reçu: Accept : text/html	HTTP/1.1 200 OK
Emission de 26 caracteres.	Content-type: text/html
J'ai reçu: Accept : text/plain	Content-length: 40
Emission de 26 caracteres.	
J'ai reçu: User-Agent : Lynx/2.4 libwww/2.1.4	
Emission de 26 caracteres.	compile:
J'ai reçu:	cd .. && make && cd serveur
Emission de 26 caracteres.	Test de message serveur.

8-e- Fonction envoyerReponse404

```
int envoyerReponse404(char *nomFichier)
{
    char req[500];
    char len[2];
    int lenght = 0;
    FILE *fp;
    fp = fopen("page404.html", "w+");
    if(fp == 0)
        return -1;
    fputs("<HTML>\n",fp);
    fputs("<HEAD><title>404 Error page</title></HEAD>\n",fp);
    fprintf(fp,"<BODY> <h1 >Error 404</h1> <p><br>Page %s Not
Found</p>\n",nomFichier);

    fputs("</BODY></HTML>\n\n",fp);
    fclose(fp);
    lenght = longueur_fichier("page404.html");
    sprintf(len, "%d", lenght);
    strcpy(req,"<HTTP/1.1 404 Not Found\nContent-type:
text/html\nContent-length: ");
    strcat(req,len);
    strcat(req,"\n");
    if(Emission(req)!=1)
        printf("Erreur d'emission\n");
    envoyerContenuFichierTexte("page404.html");
    return 0;
}
```

Test de la fonction :

```
(18:33:06) >> telnet 127.0.0.1 12321
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is '^]'.
GET /qzefqzef zqer

HTTP/1.1 404 Not Found
Content-type: text/html
Content-length: 128
<HTML>

<HEAD><title>404 Error page</title></HEAD>

<BODY> <h1 >Error 404</h1> <p><br>Page qzefqzef Not Found</p>

</BODY></HTML>

Test de message serveur.
```

8-f- Fonction envoyerReponse500

```
int envoyerReponse500(char *message)
{
    char req[500];
    char len[2];
    int lenght = 0;
    FILE *fp;
    fp = fopen("page500.html", "w+");
    if(fp == 0)
        return -1;
    fputs("<HTML>\n",fp);
    fputs("<HEAD><title>500 Error page</title></HEAD>\n",fp);
    fprintf(fp,"<BODY> <h1 >Oups erreur du serveur!</h1> <p><br> %s
</p>\n",message);

    fputs("</BODY></HTML>\n\n",fp);
    fclose(fp);

    lenght = longueur_fichier("page500.html");
```



```
sprintf(len, "%d", lenght);
strcpy(req, "\nHTTP/1.1 500 Server Error\nContent-type:
text/html\nContent-length: ");
strcat(req, len);
strcat(req, "\n");
if(Emission(req)!=1)
    printf("Erreur d'emission\n");
envoyerContenuFichierTexte("page500.html");
return 0;
}
```

Test de la fonction:

```
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is '^]'.
GET /page.html

HTTP/1.1 500 Server Error
Content-type: text/html
Content-length: 172
<HTML>

<HEAD><title>500 Error page</title></HEAD>

<BODY> <h1>Oups erreur du serveur!</h1> <p><br> Oups erreur lors de
la lecture du fichier demander. </p>

</BODY></HTML>

Test de message serveur.
```

8-g- les modifications apportée au mainServer.c

```
#include <stdio.h>
#include <stdlib.h>
#include "serveur.h"
#include <unistd.h>
int main ()
{
    char *message = NULL;
    char nomFichier[30];
    int extr_fich = 5; // retour de la fonction extraitFichier
    Initialisation();
    while(1) {
```



```
int fini = 0;
AttenteClient();
while(!fini) {
    message = Reception();
    if(message[0] == 'G' && message[1] == 'E' && message[2] == 'T')
    {
        extr_fich = extraitFichier(message, nomFichier, 30);

        if(extr_fich == 0){
            if( access( nomFichier, F_OK ) == 0 ) {
                if( access( nomFichier, R_OK ) == 0 ){
                    envoyerReponse200HTML(nomFichier);

                    envoyerContenuFichierTexte(nomFichier);
                }
                else
                    envoyerReponse500("Oups erreur lors
de la lecture du fichier demander.");
            }else {
                envoyerReponse404(nomFichier);
            }
        }
        else if(extr_fich == -2){
            envoyerReponse500("Oups erreur lors du traitement
par le serveur.");
        }
    }
    if(message != NULL) {
        printf("J'ai reçu: %s\n", message);
        free(message);

        if(Emission("Test de message serveur.\n")!=1) {
            printf("Erreur d'emission\n");
        }
    } else {
        fini = 1;
    }
}
TerminaisonClient();
}
return 0;}
```

9- Page HTML Simple: page.html

```
<HTML>
<HEAD><title>TP Info</title></HEAD>
<BODY> <h1 >Bienvenue sur TP3</h1> <p><br>La page html demandé</p>
</BODY>
</HTML>
```

10- Test du serveur

demande de la page simple : page.html



Bienvenue sur TP3

page html demandé©

Tes

demande d'une page inexistante sur le serveur:

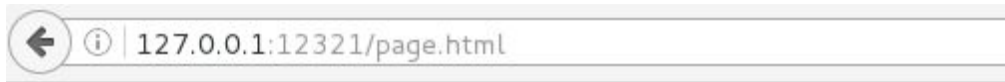


Error 404

Page zezed Not Found

Tes

demande d'une page existante "page.html" après avoir changer les droits en utilisant la commande "**sudo chmod 000 page.html**"



Oups erreur du serveur!

Oups erreur lors de la lecture du fichier demander.

Te

11- 12- La gestion des images jpg:

code source:


```
int envoyerContenuFichierBinaire(char *nomFichier)
{
    FILE *fp;
    fp = fopen(nomFichier, "rb");
    if(fp == NULL){
        fclose(fp);
        return -2;
    }
    int nb;
    char *donnees;    donnees = (char*)malloc(20000*sizeof(char));
    while((nb=fread(donnees, 1,20000,fp))>0){    //nb prend le nombre de caractère lu, ainsi
        if(EmissionBinaire(donnees, nb )== -1)    //S'il y a une erreur d'emission on arrête
            return -1;
        Emission("\n\n");
    }

    free(donnees);
    fclose(fp);
    return 0;
}
```

```
int envoyerReponse200JPG(char *nomFichier){

    char req[500];
    char len[2];
    int lenght = 0;
    lenght = longueur_fichier(nomFichier);
    sprintf(len, "%d", lenght);
    strcpy(req, "\nHTTP/1.1 200 OK\r\nContent-type: image/jpeg\r\nContent-length: ");
    strcat(req, len);
    strcat(req, "\n\n");
    if(Emission(req)!=1){
        return -1;
    }
    return 0;
}
```

Test de la gestion des images "jpg":



```

chems@debian: ~/Documents/MiniServeur/se
File Edit View Search Terminal Help

[$chems@debian] - [serveur]
(14:17:24) >> ./mainServeur
Creation du serveur reussie.
Client sur la machine d'adresse localhost connecte.
J'ai recu: GET /download.jpg HTTP/1.1

Emission de 68 caracteres.
Emission de 3 caracteres.
J'ai recu: Host: 127.0.0.1:12321

J'ai recu: Connection: keep-alive

J'ai recu: Upgrade-Insecure-Requests: 1

J'ai recu: User-Agent: Mozilla/5.0 (X11; Linux x86_64)
, like Gecko) Chrome/55.0.2883.87 Safari/537.36

J'ai recu: Accept: text/html,application/xhtml+xml,application/javascript;q=0.8

J'ai recu: Accept-Encoding: gzip, deflate, sdch, br

J'ai recu: Accept-Language: en-US,en;q=0.8
  
```

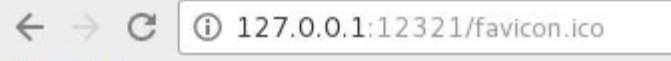
13- Gestion des images ICO


code source:

```

int envoyerReponse200ICO(char*nomFichier){
    char req[500];
    char len[2];
    int lenght = 0;
    lenght = longueur_fichier(nomFichier);
    sprintf(len, "%d", lenght); // convertir la longueur en chaine de caractère
    //formater la reponse
    strcpy(req, "\nHTTP/1.1 200 OK\r\nContent-type :image/vnd.microsoft.icon\r\nContent-length: ");
    strcat(req, len);
    strcat(req, "\n\n");
    if(Emission(req)!=1){
        return -1;
    }
    return 0;
}
  
```

Test de la gestion des images "ICO":





Name	Headers	Preview	Response	Timing
favicon.ico	<p>▼ General</p> <p>Request URL: http://127.0.0.1:12321/favicon.ico</p> <p>Request Method: GET</p> <p>Status Code: 200 OK</p> <p>Remote Address: 127.0.0.1:12321</p> <p>▼ Response Headers view parsed</p> <p>HTTP/1.1 200 OK</p> <p>Content-type: image/vnd.microsoft.icon</p> <p>Content-length: 32038</p> <p>▼ Request Headers view source</p> <p>Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8</p> <p>Accept-Encoding: gzip, deflate, sdch, br</p> <p>Accept-Language: en-US,en;q=0.8</p> <p>Connection: keep-alive</p> <p>Host: 127.0.0.1:12321</p> <p>Upgrade-Insecure-Requests: 1</p>			