Numero: Nome: Data:

LAB08

Protocolo HTTP
Servidor Apache

A. Protocolo HTTP

Neste exercício vamos utilizar a aplicação `telnet`, ligar-nos directamente à porta TCP 80 do servidor web, e familiarizarmo-nos com os comandos do protocolo HTTP.

1. Corre a aplicação telnet e pede a página de entrada do servidor web

```
#telnet 10.10.23.13 80
GET / HTTP/1.0
(linha em branco)
(linha em branco)
```

Que é que recebeste como resposta?
Linha de status: _____
Cabeçalhos:
Connection: _____
Content-Type:_____
Dados (html, pdf, jpg, png, ...): _____

```
#telnet intranet.ualg.pt 80
GET /templates/manga/images/logo.png HTTP/1.0
(linha em branco)
(linha em branco)
```

Que é que recebeste como resposta?
Linha de status: _____
Cabeçalhos:
Connection: _____
Content-Type:_____
Dados (html, pdf, jpg, png, ...): _____

2. Neste exercício vamos enviar dados para o servidor juntamente com o URL utilizando o comando GET.
   `test2.pl` é um programa em Perl que se encontra no servidor e que devolve uma página html com todas as variáveis fornecidas pelo servidor web ao programa (ver o código em apendice). É através da variável _____ que o programa tem conhecimento dos dados que nós enviamos.

```
#telnet 10.10.23.13 80
GET /cgi-bin/test2.pl?nome=bruno&idade=23 HTTP/1.0
```

Numero: Nome: Data:

```
(linha em branco)
(linha em branco)
```

Que é que recebeste como resposta?
QUERY_STRING=_____

3. Neste exercício vamos enviar dados para o servidor web através do comando POST. `post.pl` é um programa em Perl que se encontra no servidor e que devolve uma página html com o corpo da mensagem HTTP (ver o código em apendice)

```
#telnet 10.10.23.13 80
POST /cgi-bin/post.pl HTTP/1.0
Content-Length:22
(linha em branco)
ano=2003&mes=11&dia=20
(linha em branco)
```

Que é que recebeste como resposta?
Linha de status: _____
Cabeçalhos:
Connection: _____
Content-Type:_____
Dados (html, pdf, jpg, png, ...): _____

B. Instalação do servidor Apache

4. Para instalar o servidor Apache basta executar o comando

```
#apt-get install apache2
```

5. Verifica que o serviço está a funcionar com o browser (chrome...) abrindo o URL `http://localhost/`.

Funciona?_____

- Vê o que está definido no ficheiro de configuração `/etc/apache2/apache2.conf`
- Vê quais os modulos que estão activos em /etc/apache2/mods-enabled
- Vê quais os sites que estão activos em /etc/apache2/sites-enabled

6. Altera a página de entrada do site para uma da tua preferência substituindo /var/www/index.html pela tua página html preferida. Por exemplo guarda a página de entrada do Google (www.google.pt) utilizando o menu

```
(chrome) Settings > Save Page as ... > /var/www/google.html
```

Copia a página que acabaste de guardar para /var/www/index.html:

```
#cp –a /var/www/index.html /var/www/index.html.original
#mv /var/www/google.html /var/www/index.html
```

7. Faz refresh à página `http://localhost/`

   Quais são as diferenças relativamente à página original do google? _____
   O que deves fazer para aparecer o logotipo do google? _____
   Qual o nome do serviço e o IP do servidor que tens que "assaltar" (hijack) para re- direccionares os acessos ao google para o teu servidor? _____

C. Criação do directório web pessoal de um utilizador

8. Activa o modulo "userdir" do apache

```
# cd /etc/apache2/mods-enabled
/etc/apache2/mods-enabled# ln –s ../mods-available/userdir.conf  userdir.conf
/etc/apache2/mods-enabled# ln –s ../mods-available/userdir.load  userdir.load
```

Faz restart ao serviço

```
# /etc/init.d/apache2 restart
```

9. Cria (se ainda não existir!) o utilizador cantiflas

```
# adduser cantiflas
# chmod a+x /home/cantiflas
```

10. Muda para o utilizador cantiflas e cria o directório public_html

```
# su – cantiflas
cantiflas@router_x:~$ mkdir public_html
cantiflas@router_x:~$ chmod a+rx public_html
```

11. Copia a tua página web preferida para dentro do directório public_html

```
cantiflas@router_x:~$ cp /var/www/index.html ~/public_html
```

12. Verifica com o browser que a página web pessoal do utilizador cantiflas está agora acessível em `http://localhost/~cantiflas`

Sucesso?_____

13. Qual a directiva em `userdir.conf` que terás que mudar para que a pasta web pessoal `public_html` passe a chamar-se `web_folder` ?

    _____

D. Criação de um directório com acesso restrito

14. Cria a pasta "`privado`" do utilizador cantiflas

```
$ mkdir ~/public_html/privado
$ chmod a+x ~/public_html/privado
```

15. Cria o ficheiro `~/public_html/privado/.htaccess` com o seguinte conteúdo

```
$ nano ~/public_html/privado/.htaccess

AuthType Basic
AuthName “cantiflas”
AuthUserFile “/home/cantiflas/public_html/privado/.htpasswd”
Require valid-user
```

16. Cria o ficheiro .htpasswd

```
$ cd ~/public_html/privado/
$ /usr/bin/htpasswd –c -b .htpasswd cantiflas segredo
```

17. Verifica que te é pedida uma password para aceder ao URL
`http://localhost/~cantiflas/privado`

Qual a directiva que deves incluir no ficheiro `.htaccess` para garantir que o cantiflas é um utilizador válido do servidor? _____

Quais são as directivas que estão dentro do contentor `<Files ~ "^\.ht" >` que impedem que se possa fazer download do ficheiro `.htpasswd` ?

_____

(Sugestão: a tua página web pessoal no DEEI http://www.deei.fct.ualg.pt/~axxxxx/, pode ser protegida por password utilizando os comando que aprendeste neste guião ...)

E. Criação de dois hosts virtuais

18. Altera o servidor de DNS* e introduz dois “aliases” adicionais em `/etc/bind/db.hosts`.

```
# nano /etc/bind/db.hosts

girls     IN   CNAME     server__.grs.deei.
boys      IN   CNAME     server__.grs.deei.
```

Substitui __ pelo número do teu servidor.

Faz o restart ao serviço DNS

```
# /etc/init.d/bind9 restart
```

* Nota: Se o servidor de DNS não funcionar/não existir, no mínimo é preciso que existam entradas para estes nomes no ficheiro `/etc/hosts` !

Assegura-te que o teu servidor de DNS é o *primeiro* a ser consultado:

```
# nano /etc/resolv.conf
search grs.deei
nameserver 127.0.0.1
```

Verifica!

```
# nslookup boys.grs.deei
# nslookup girls.grs.deei
```

19. Cria as pastas girls e boys em `/var/www`

```
# mkdir /var/www/girls
# mkdir /var/www/boys
# chmod a+rx /var/www/girls
# chmod a+rx /var/www/boys
```

20. Cria os ficheiros “boys.grs.deei” e “girls.grs.deei”

```
# nano /etc/apache2/sites-available/boys.grs.deei


<VirtualHost   *:80>
     ServerName     boys.grs.deei
     DocumentRoot   /var/www/boys/
</VirtualHost>

# nano /etc/apache2/sites-available/girls.grs.deei

<VirtualHost   *:80>
     ServerName      girls.grs.deei
     DocumentRoot   /var/www/girls/
</VirtualHost>
```

E activa os “virtual hosts” ...

```
# cd /etc/apache2/sites-enabled
/etc/apache2/sites-enabled# ln –s ../sites-available/boys.grs.deei boys.grs.deei
/etc/apache2/sites-enabled# ln –s ../sites-available/girls.grs.deei girls.grs.deei
```

Re-inicia o apache

```
# /etc/init.d/apache2 restart
```

21. Coloca dois ficheiros index.html **diferentes** em /var/www/girls e /var/www/boys.

22. Verifica com o browser que os URLs

```
http://boys.grs.deei/
```

`http://girls.grs.deei/`

devolvem páginas diferentes, apesar de ambos os sites estarem alojadas no mesmo servidor web...

Sucesso?_____

Faz o debugging necessario até que os dois virtual hosts estejam a funcionar correctamente ...

23. Termina aqui este laboratório.

APÊNDICE

test2.pl

```
#!/usr/bin/perl

print "Content-type:text/html\n\n";
print <<EndOfHTML;
<html><head><title>Print Environment</title></head>
<body>
EndOfHTML

foreach $key (sort(keys %ENV)) {
        print "$key = $ENV{$key}<br>\n";
}

print "</body></html>";
```

post.pl

```
#!/usr/bin/perl

print "Content-type:text/html\n\n";

read(STDIN, $buffer, $ENV{'CONTENT_LENGTH'});
@pairs = split(/&/, $buffer);
foreach $pair (@pairs) {
  ($name, $value) = split(/=/, $pair);
   $value =~ tr/+/ /;
   $value =~ s/%([a-fA-F0-9][a-fA-F0-9])/pack("C", hex($1))/eg;
   $FORM{$name} = $value;
}

print "<html><head><title>Form Output</title></head><body>";
print "<h2>Results from FORM post</h2>\n";

foreach $key (keys(%FORM)) {
             print "$key = $FORM{$key}<br>";
}

print "</body></html>";
```