

- Uruchom ten skrypt, przetestuj, zastanów się jak działa.
- Nawiąż połączenie za pomocą przeglądarki internetowej.

Skrypt podlegał drobnej modyfikacji przez problem z uruchomieniem:

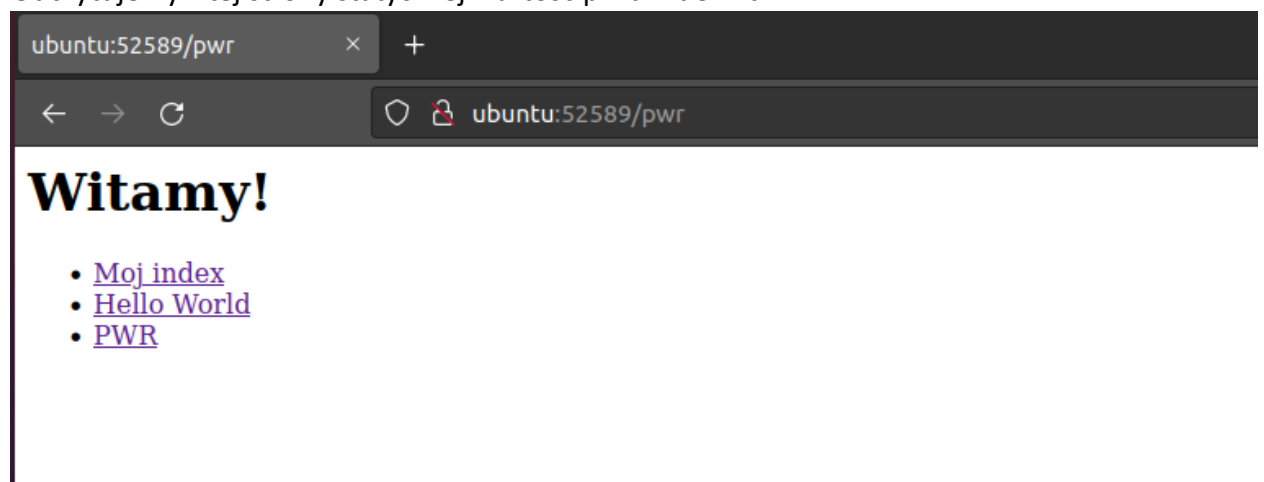
```
1 use HTTP::Daemon;  
2 use HTTP::Status;  
3 #use IO::File;  
4  
5 my $d = HTTP::Daemon->new || die;  
6  
7 print "Please contact me at: <URL:", $d->url, ">\n";  
8  
9  
10 while (my $c = $d->accept) {  
11     while (my $r = $c->get_request) {  
12         if ($r->method eq 'GET') {  
13  
14             $file_s = "/index.html"; # index.html - jakiś istniejący plik  
15             $c->send_file_response($file_s);  
16  
17         }  
18         else {  
19             $c->send_error(RC_FORBIDDEN)  
20         }  
21     }  
22     $c->close;  
23     undef($c);  
24 }  
25 ]
```

Został usunięty localhost and localport

Po uruchomieniu:

```
nscnbs@ubuntu:~$ cd Lista5_TS/  
nscnbs@ubuntu:~/Lista5_TS$ perl server3.pl  
Please contact me at: <URL:http://ubuntu:52589/>
```

Odczytujemy z tej strony statycznej wartość pliku index.html

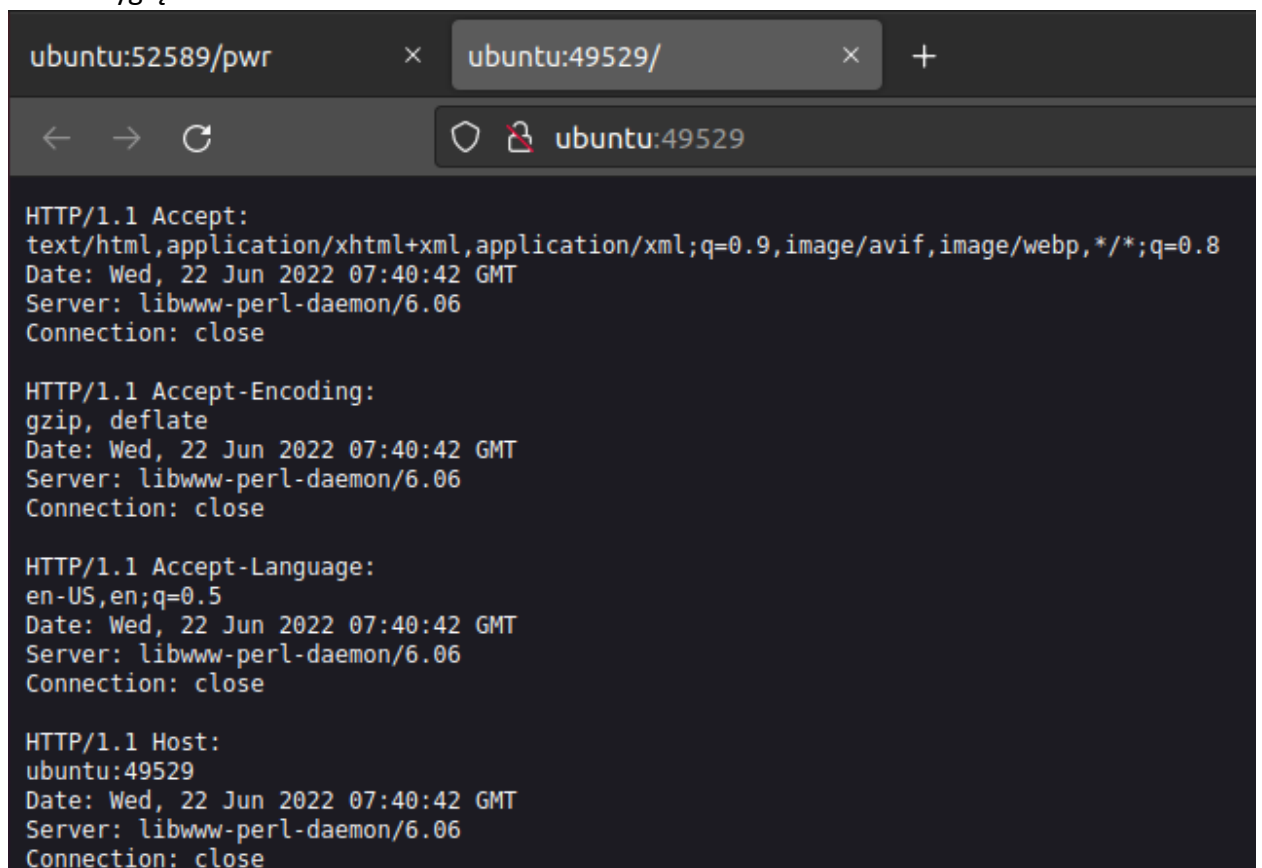


- Zmień skrypt (lub napisz własny serwer w dowolnym języku programowania) tak aby wysyłał do klienta nagłówki jego żądania.

Biblioteka HTTP::Request w Perl'u ma w sobie parametr \$r, który posiada metodę header_field_names. Za pomocą drobniej modyfikacji server3.pl zrobmy wysyłanie nagłówka żądania do klienta:

```
1 use HTTP::Daemon;
2 use HTTP::Status;
3
4 my $d = HTTP::Daemon->new || die;
5 print "Please contact me at: ", $d->url, "\n";
6
7
8 while (my $c = $d->accept) {
9     while (my $r = $c->get_request) {
10         if ($r->method eq 'GET') {
11             foreach ($r->header_field_names) {
12                 $c->send_response($_ . ":\n" . $r->header($_));
13             }
14         }
15         else {
16             $c->send_error(RC_FORBIDDEN);
17         }
18     }
19     $c->close;
20     undef($c);
21 }
```

Jak to wygląda:



```
ubuntu:52589/pwr × ubuntu:49529/ × +
← → ↻ 🔒 ubuntu:49529

HTTP/1.1 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Date: Wed, 22 Jun 2022 07:40:42 GMT
Server: libwww-perl-daemon/6.06
Connection: close

HTTP/1.1 Accept-Encoding:
gzip, deflate
Date: Wed, 22 Jun 2022 07:40:42 GMT
Server: libwww-perl-daemon/6.06
Connection: close

HTTP/1.1 Accept-Language:
en-US,en;q=0.5
Date: Wed, 22 Jun 2022 07:40:42 GMT
Server: libwww-perl-daemon/6.06
Connection: close

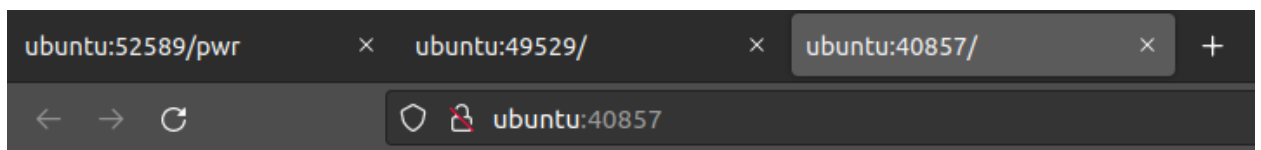
HTTP/1.1 Host:
ubuntu:49529
Date: Wed, 22 Jun 2022 07:40:42 GMT
Server: libwww-perl-daemon/6.06
Connection: close
```

- Zmień skrypt (lub napisz własny serwer w dowolnym języku programowania) tak aby obsługiwał żądania klienta do prostego tekstowego serwisu WWW (kilka statycznych

ston z wzajemnymi odwołaniami) zapisanego w pewnym katalogu dysku lokalnego komputera na którym uruchomiony jest skrypt serwera.

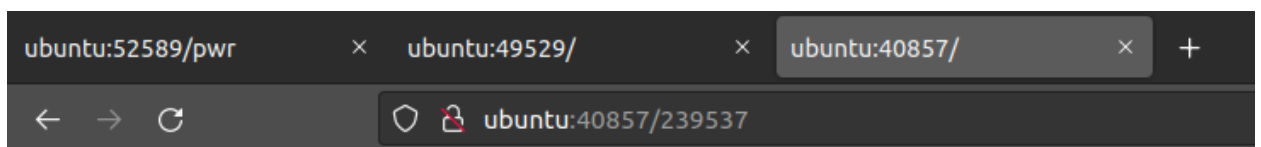
Struktura strony index.html:

- Mój index
 1. 239537
- Hello
 2. Hello World
- PWR
 3. PWR



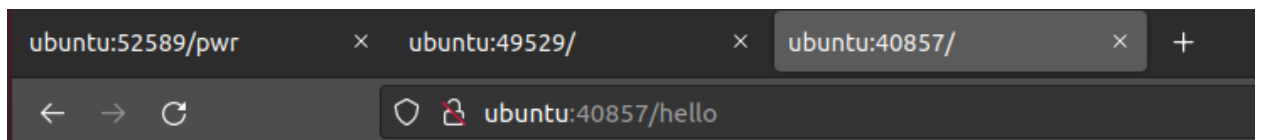
Witamy!

- [Moj index](#)
- [Hello World](#)
- [PWR](#)



239537

My index



Success!

Hello World

Kod:

```

4 my $d = HTTP::Daemon->new || die;
5 print "Please contact me at: ", $d->url, "\n";
6
7
8 while (my $c = $d->accept) {
9     while (my $r = $c->get_request) {
10         if ($r->method eq 'GET') {
11             $path = $r->uri->path;
12
13             if (index($path, "/239537") == 0 && length($path) == 7) {
14                 $c->send_file_response("./239537/hello.html");
15             }
16             elsif (index($path, "/hello") == 0 && length($path) == 6) {
17                 $c->send_file_response("./helloworld/hello2.html");
18             }
19             elsif (index($path, "/pwr") == 0 && length($path) == 4) {
20                 $c->send_file_response("./pwr/pwr.html");
21             }
22             elsif (length($path) == 1) {
23                 $c->send_file_response("./index.html");
24             }
25             else {
26                 $c->send_error(RC_FORBIDDEN);
27             }
28         }
29     }
30 }
31

```

Przechwyć komunikaty do/od serwera za pomocą analizatora sieciowego - przeanalizuj ich konstrukcję.

The screenshot shows the Wireshark interface with a packet capture of an HTTP transaction. The left pane shows the packet list with packet 5133 selected. The middle pane shows the packet details for the selected packet, highlighting the Hypertext Transfer Protocol section. The right pane shows the raw packet data in hexadecimal and ASCII.

Packet List:

No.	Time	Source
5125	41.905852718	127.0.1.1
5126	41.905855958	127.0.1.1
5127	41.905866068	127.0.1.1
5128	41.905869564	127.0.1.1
5129	41.905891446	127.0.1.1
5130	41.905895314	127.0.1.1
5131	41.905904526	127.0.1.1
5132	41.905907681	127.0.1.1
5133	41.906320174	127.0.1.1

Packet Details:

- Frame 5133: 143 bytes on wire (1144 bytes captured) on interface eth0
- Ethernet II, Src: Linux cooked capture, Dst: 08:00:00:00:00:00
- Internet Protocol Version 4, Src: 127.0.1.1, Dst: 127.0.1.1
- Transmission Control Protocol, Src Port: 80, Dst Port: 80, Seq: 3373, Len: 10
- Hypertext Transfer Protocol
 - GET / HTTP/1.1
 - Host: ubuntu:41653
 - User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:101.0) Gecko/20100101 Firefox/101.0
 - Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
 - Accept-Language: en-US,en;q=0.5
 - Accept-Encoding: gzip, deflate
 - Connection: keep-alive
 - Upgrade-Insecure-Requests: 1

Raw Data:

```

0000  00 00 03 04 00 06 00 00 00 00 00 00 00 00 00 00
0010  45 00 00 7f c3 da 40 00 40 00 00 00 00 00 00 00
0020  7f 00 00 01 a2 b5 88 5a 1b ed 00 00 00 00 00 00
0030  80 18 02 00 ff 73 00 00 01 01 00 00 00 00 00 00
0040  1d 3b b3 ae 3c 68 74 6d 6c 3e 00 00 00 00 00 00
0050  79 3e 0a 0a 20 20 20 20 3c 6e 00 00 00 00 00 00
0060  33 37 3c 2f 68 31 3e 0a 0a 20 00 00 00 00 00 00
0070  4d 79 20 69 6e 64 65 78 3c 2f 00 00 00 00 00 00
0080  62 6f 64 79 3e 0a 0a 3c 2f 6e 00 00 00 00 00 00

```

Wireshark interface showing a packet capture of HTTP traffic. The packet list pane displays a series of HTTP requests and responses between 127.0.0.1 and 127.0.1.1. The packet details pane shows the structure of the selected packet (No. 4646), including the Ethernet II header and the Hypertext Transfer Protocol (HTTP) header.

No.	Time	Source	Destination	Protocol	Length	Info
4646	25.842540245	127.0.0.1	127.0.1.1	HTTP	412	GET / HTTP/1.1
4662	25.860790638	127.0.1.1	127.0.0.1	HTTP	279	HTTP/1.1 200 OK (text/html)
4746	27.131977302	127.0.0.1	127.0.1.1	HTTP	364	GET /favicon.ico HTTP/1.1
4760	27.132754605	127.0.1.1	127.0.0.1	HTTP	121	HTTP/1.1 403 Forbidden (text/html)
5117	41.903311538	127.0.0.1	127.0.1.1	HTTP	449	GET /239537 HTTP/1.1
5133	41.906320171	127.0.1.1	127.0.0.1	HTTP	143	HTTP/1.1 200 OK (text/html)
5139	43.745502095	127.0.0.1	127.0.1.1	HTTP	448	GET /hello HTTP/1.1
5154	43.747404693	127.0.1.1	127.0.0.1	HTTP	148	HTTP/1.1 200 OK (text/html)
5182	45.659303031	127.0.0.1	127.0.1.1	HTTP	446	GET /pwr HTTP/1.1
5197	45.661058319	127.0.1.1	127.0.0.1	HTTP	3377	HTTP/1.1 200 OK (text/html)
5199	45.810654683	127.0.0.1	127.0.1.1	HTTP	358	GET /stylh.css HTTP/1.1
5212	45.811340935	127.0.1.1	127.0.0.1	HTTP	121	HTTP/1.1 403 Forbidden (text/html)
5303	46.125763148	127.0.0.1	127.0.1.1	HTTP	367	GET /logo_ki.png HTTP/1.1
5316	46.126495842	127.0.1.1	127.0.0.1	HTTP	121	HTTP/1.1 403 Forbidden (text/html)
5321	46.197598582	127.0.0.1	127.0.1.1	HTTP	401	GET /naglowek2.html HTTP/1.1
5326	46.199140737	127.0.1.1	127.0.0.1	HTTP	201	HTTP/1.1 403 Forbidden (text/html)
5328	46.211922056	127.0.0.1	127.0.1.1	HTTP	403	GET /top_nav_pl0.html HTTP/1.1
5331	46.214227409	127.0.1.1	127.0.0.1	HTTP	238	HTTP/1.1 403 Forbidden (text/html)
5333	46.216443175	127.0.0.1	127.0.1.1	HTTP	400	GET /bot_nav0.html HTTP/1.1
5336	46.217169254	127.0.1.1	127.0.0.1	HTTP	238	HTTP/1.1 403 Forbidden (text/html)
5945	170.950887527	192.168.59.130	34.122.121.32	HTTP	143	GET / HTTP/1.1
5947	171.135991055	34.122.121.32	192.168.59.130	HTTP	204	HTTP/1.1 204 No Content

Frame 4646: 412 bytes on wire (3296 bits), 412 bytes captured (3296 bits) on interface any, id 0

0000 00 00 03 04 00 06 00 00 00 00 00 00 c9 d1 08 00
0010 45 00 01 8c 9d 6f 40 00 40 06 9c fa 7f 00 00 01 E.....@.....
0020 7f 00 01 01 88 5a a2 b5 af 76 a5 0e 1b ed 6b f7Z.....v.....k.....
0030 00 18 02 00 00 81 00 00 01 01 08 0a 1d 3b 74 ef:.....t.....