13.06.2020

Lista 5

Technologie sieciowe

Dominika Szydło

1. Uruchomienie skryptu

server1.pl

```
#!/usr/bin/perl

use HTTP::Daemon;
use HTTP::Status;
use IO::File;

my $d = HTTP::Daemon->new(LocalAddr => 'localhost', LocalPort => 4321,)|| die;

print "Please contact me at: <URL:", $d->url, ">\n";

while (my $c = $d->accept) {
    while (my $r = $c->get_request) {
        if ($r->method eq 'GET') {
            $file_s= "index.html";
            $c->send_file_response($file_s);
        }
        else {
            $c->send_error(RC_FORBIDDEN)
        }
    }
    $c->close;
    undef($c);
}
```

Screenshot 1. Program po modyfikacji

Aby program zadziałał, musiałam zmienić parametr LocalAddr z „lukim” na „localhost”. Program uruchamia serwer HTTP, który po otrzymaniu GET request wysyła klientowi plik index.html. W przypadku otrzymania innego requesta, wysyła klientowi błąd 403, informujący o braku dostępu do danego pliku. Utworzyłam plik index.html w tej samej lokalizacji, gdzie zapisałam program i z której go uruchomiłam.

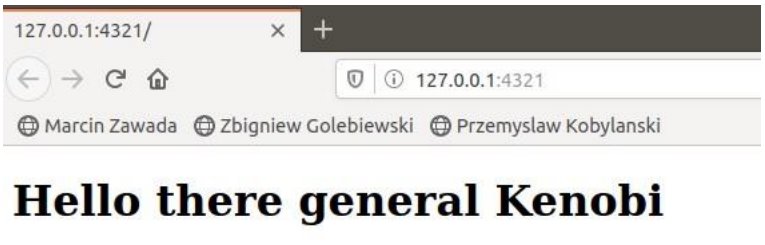
index.html > html

```
<!DOCTYPE html>
<html>
<body>
    <h1>Hello there general Kenobi</h1>
</body>
</html>
```

Screenshot 2. Plik index.html

```
domka@domka-VirtualBox:~/Desktop/TS/lista5$ perl server1.pl
Please contact me at: <URL:http://127.0.0.1:4321/>
```

Screenshot 3. Uruchomienie programu server1.pl



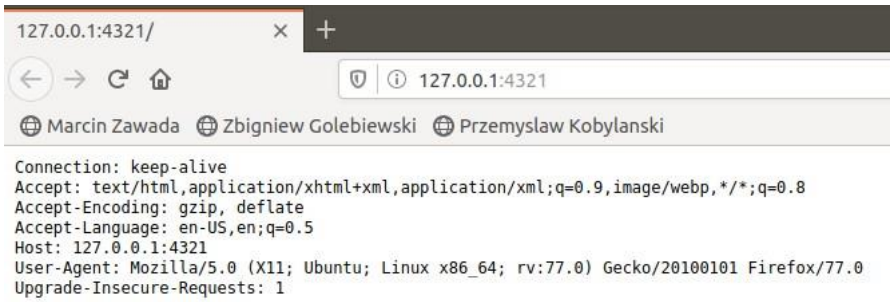
Screenshot 4. Okno przeglądarki po otwarciu linka z konsoli

2. Program odsyłający nagłówki żądań

```
11  while (my $c = $d->accept) {
12      while (my $r = $c->get_request) {
13          if ($r->method eq 'GET') {
14              my $h = $r->headers_as_string;
15              my $resp = HTTP::Response->new(200);
16              $resp->header("Content-Type" => "text/plain");
17              $resp->content($h);
18              $c->send_response($resp);
19          }
20          else {
21              $c->send_error(RC_FORBIDDEN);
22          }
```

Screenshot 5. Program server2.pl, będący modyfikacją programu server1.pl

Aby serwer wysyłał do klienta nagłówek jego żądania wystarczyła prosta modyfikacja wewnętrznej pętli programu. Najpierw tworzę zmienną przechowującą nagłówek requesta, potem tworzę nową odpowiedź o kodzie 200 (OK, znaczenie - zawartość żądanego dokumentu), ustawiam jej typ na dane tekstowe oraz nadaję jej zawartość, w postaci wartości trzymanej przez zmienną h, a na końcu odsyłam do klienta.



Screenshot 6. Okno przeglądarki po uruchomieniu programu server2.pl i otwarciu linka

3. Program obsługujący żądania do prostego tekstowego serwisu WWW

```
11  while (my $c = $d->accept) {
12      while (my $r = $c->get_request) {
13          if ($r->method eq 'GET') {
14              my $uri = $r->uri;
15              if ($uri eq "/") {
16                  $uri = "/glowna.html"
17              }
18              my $file_s= "strona".$uri;
19              $c->send_file_response($file_s);
20          }
21          else {
22              $c->send_error(RC_FORBIDDEN)
23          }
```

Screenshot 7. Program server3.pl, będący modyfikacją programu server1.pl

Na potrzeby tego zadania utworzyłam folder *strona*, w którym umieściłam trzy pliki – glowna.html, podstrona1.html oraz podstrona2.html. Aby zrealizować polecenie, zrobiłam kolejną modyfikację pętli programu. Do zmiennej uri zapisuję URI, czyli ścieżkę zasobu, którego dotyczy zapytanie GET. Jest ona jednak niepełna, ponieważ zawiera ona jedynie nazwy plików, a znajdują się one w innym folderze. Dlatego uzupełniam ścieżkę o początek „strona" i dopiero potem odsyłam klientowi żądany zasób.

```html
<!DOCTYPE html>
<html>
<body>
    <h1>Strona glowna</h1>
    <ul>
        <li><a href="glowna.html">Strona glowna</a></li>
        <li><a href="podstrona1.html">Podstrona 1</a></li>
        <li><a href="podstrona2.html">Podstrona 2</a></li>
    </ul>
</body>
</html>
```

Screenshot 8. Plik glowna.html

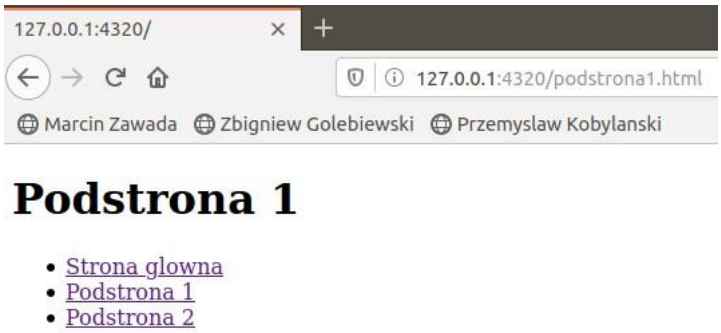
```html
<!DOCTYPE html>
<html>
<body>
    <h1>Podstrona 1</h1>
    <ul>
        <li><a href="glowna.html">Strona glowna</a></li>
        <li><a href="podstrona1.html">Podstrona 1</a></li>
        <li><a href="podstrona2.html">Podstrona 2</a></li>
    </ul>
</body>
</html>
```

Screenshot 9. Plik podstrona1.html

```html
<!DOCTYPE html>
<html>
<body>
    <h1>Podstrona 2</h1>
    <ul>
        <li><a href="glowna.html">Strona glowna</a></li>
        <li><a href="podstrona1.html">Podstrona 1</a></li>
        <li><a href="podstrona2.html">Podstrona 2</a></li>
    </ul>
</body>
</html>
```

Screenshot 10. Plik podstrona2.html

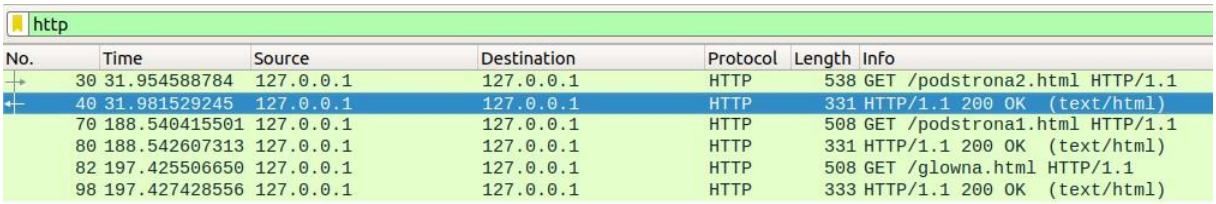
Screenshot 11. Okno przeglądarki po uruchomieniu programu server3.pl



Screenshot 12. Okno przeglądarki po kliknięciu "Podstrona1"

4. Przechwytywanie i analiza pakietów

Do przechwytywania pakietów użyłam Wiresharka. Weszłam w podstronę 2, z niej w podstronę 1, a następnie przeszłam na stronę główną. Przechwycone komunikaty występują parami – GET request od klienta i odpowiedź serwera o kodzie 200 (OK).



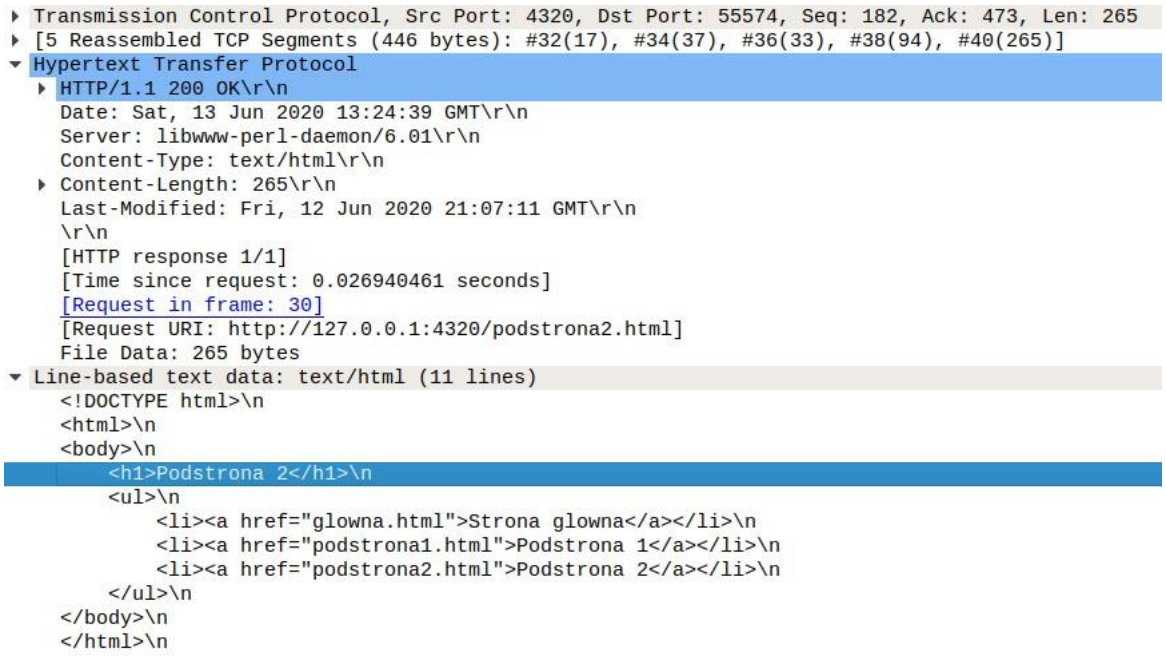
Screenshot 13. Okno Wiresharka

W przechwyconym komunikacie od klienta znajdują się informacje m.in. o adresie hosta i numerze portu, rodzaju requesta (GET), nazwie żądanego zasobu, używanej przeglądarce i systemie operacyjny czy długości komunikatu.

```
Transmission Control Protocol, Src Port: 55574, Dst Port: 4320, Seq: 1, Ack: 1, Len: 472
Hypertext Transfer Protocol
  GET /podstrona2.html HTTP/1.1\r\n
    [Expert Info (Chat/Sequence): GET /podstrona2.html HTTP/1.1\r\n]
    Request Method: GET
    Request URI: /podstrona2.html
    Request Version: HTTP/1.1
  Host: 127.0.0.1:4320\r\n
  User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:77.0) Gecko/20100101 Firefox/77.0\r\n
  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8\r\n
  Accept-Language: en-US,en;q=0.5\r\n
  Accept-Encoding: gzip, deflate\r\n
  Referer: http://127.0.0.1:4320/podstrona1.html\r\n
  Connection: keep-alive\r\n
  Upgrade-Insecure-Requests: 1\r\n
  If-Modified-Since: Fri, 12 Jun 2020 21:07:11 GMT\r\n
  Cache-Control: max-age=0\r\n
  \r\n
  [Full request URI: http://127.0.0.1:4320/podstrona2.html]
  [HTTP request 1/1]
```

Screenshot 14. Dane żądania GET o podstronę 2

W przechwyconej odpowiedzi od serwera znajdują się informacje takie jak jej kod (200 OK), data, dane serwera, typ przesyłanej zawartości (text/html) i jej długość, a także ona sama.



Screenshot 15. Odpowiedź serwera na żądanie z poprzedniego screenshota