Sprawozdanie z listy 5. - Technologie sieciowe

Jakub Bachanek

9 czerwca 2022

1 Przykładowy program serwera protokołu HTTP

1.1 Opis działania programu

Za pomocą *HTTP::Daemon* zostaje utworzony serwer HTTP, który nasłuchuje na gnieździe o ustalonym adresie IP oraz numerze portu. Przy połączeniu zwraca do klienta odpowiedź jako plik index.html.

1.2 Połączenie za pomocą przeglądarki internetowej

Nawiązujemy połączenie za pomocą przeglądarki internetowej poprzez wpisanie adresu ip oraz numeru portu w pasek adresu. Przy udanej próbie zobaczymu zawartość strony index.html, natomiast w razie niepowodzenia wyświetlony zostanie stosowny komunikat.

1.3 Nagłówek żądania

Po lekkiej modyfikacji programu zwraca on do klienta nagłówek jego żądania.

Jest to zrealizowane w taki sposób:

```
my $response = HTTP::Response->new();
$response->content($r->header_as_string);
$c->send_response($response);
```

Przykładowy nagłówek:

```
Cache-Control: max-age=0
Connection: keep-alive
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,i
mage/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate, br
Accept-Language: pl-PL,pl;q=0.9,en-US;q=0.8,en;q=0.7
Host: 127.0.0.1:4321
If-Modified-Since: Wed, 08 Jun 2022 17:50:13 GMT
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/101.0.4951.64 Safari/537.36
Sec-Ch-Ua: " Not A;Brand";v="99", "Chromium";v="101"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Linux"
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: none
Sec-Fetch-User: ?1
Upgrade-Insecure-Requests: 1
```

1.4 Prosty tekstowy serwis WWW

Serwis składa się z trzech stron ze wzajemnymi odwołaniami do siebie.

Sposób działania:

```
my $my_path = $r->uri->path;

if ($my_path eq "/") {
	$my_path = "/index_1.html"
}

$c->send_file_response(".".$my_path);
```

1.5 Przechwytywanie komunikatów

Do przechwycenia komunikatów w obie strony możemy użyć programu *Wireshark*. Ustawiamy nasłuchiwanie na Loopback, a następnie generujemy ruch sieciowy. Podczas analizy można zauważyć, że informacje są przesyłane w czystej formie tekstowej. Jest to spowodowane brakiem zabezpieczenia w postaci szyfrowania, które występuje w *HTTPS*.

Przykładowe przechwycenia:

```
    25 7.650533242   127.0.0.1          127.0.0.1          HTTP       814 GET /page_1.html HTTP/1.1
Frame 25: 814 bytes on wire (6512 bits), 814 bytes captured (6512 bits) on interface lo, id 0
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
Transmission Control Protocol, Src Port: 41182, Dst Port: 4321, Seq: 724, Ack: 389, Len: 748
Hypertext Transfer Protocol
▸ GET /page_1.html HTTP/1.1\r\n
  Host: 127.0.0.1:4321\r\n
  Connection: keep-alive\r\n
  sec-ch-ua: " Not A;Brand";v="99", "Chromium";v="101"\r\n
  sec-ch-ua-mobile: ?0\r\n
  sec-ch-ua-platform: "Linux"\r\n
  Upgrade-Insecure-Requests: 1\r\n
  User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/101.0.4951.64
  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8
  Sec-Fetch-Site: same-origin\r\n
  Sec-Fetch-Mode: navigate\r\n
  Sec-Fetch-User: ?1\r\n
  Sec-Fetch-Dest: document\r\n
  Referer: http://127.0.0.1:4321/\r\n
  Accept-Encoding: gzip, deflate, br\r\n
  Accept-Language: pl-PL,pl;q=0.9,en-US;q=0.8,en;q=0.7\r\n
  If-Modified-Since: Wed, 08 Jun 2022 17:50:40 GMT\r\n
```

```
    41 7.652060443   127.0.0.1          127.0.0.1          HTTP       272 HTTP/1.1 200 OK  (text/html)
Frame 41: 272 bytes on wire (2176 bits), 272 bytes captured (2176 bits) on interface lo, id 0
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
Transmission Control Protocol, Src Port: 4321, Dst Port: 41182, Seq: 570, Ack: 1472, Len: 206
[8 Reassembled TCP Segments (387 bytes): #27(17), #29(37), #31(33), #33(25), #35(21), #37(46), #39(2), #41(206)]
Hypertext Transfer Protocol
▸ HTTP/1.1 200 OK\r\n
  Date: Thu, 09 Jun 2022 14:58:53 GMT\r\n
  Server: libwww-perl-daemon/6.06\r\n
  Content-Type: text/html\r\n
▸ Content-Length: 206\r\n
  Last-Modified: Wed, 08 Jun 2022 17:50:40 GMT\r\n
  \r\n
  [HTTP response 2/4]
  [Time since request: 0.001527201 seconds]
  [Prev request in frame: 4]
  [Prev response in frame: 23]
  [Request in frame: 25]
  [Next request in frame: 43]
  [Next response in frame: 59]
  [Request URI: http://127.0.0.1:4321/page_1.html]
  File Data: 206 bytes
```

Kod "200 OK" oznacza, że zasób działa poprawnie i serwer zwrócił oczekiwaną odpowiedź.

1.6 Wnioski

Poprzez analizę przykładowego programu możemy zobaczyć jak działa prosty serwer HTTP. Znacznie bardziej rozbudowane serwery WWW stoją za obsługą połączeń w sieci, z której na co dzień korzystamy.