WWW - Apache + PHP

Dawid Bachoń & Jakub Fladziński

Wprowadzenie

Apache to przeznaczony dla wielu systemów operacyjnych wszechstronny otwarty serwer HTTP umożliwiający hostowanie wielu stron jednocześnie.

Został uruchomiony 1995 roku i był najbardziej popularnym serwerem HTTP od kwietnia 1996. Apache był najpopularniejszym serwerem aż do czerwca 2014, po którym został na krótko zdetronizowany przez serwer Microsoftu. Udział w rynku serwerów HTTP zaczął dzielić się na coraz więcej graczy, co doprowadziło ze Apache od Marca 2016 nie był już w stanie zdobyć pozycji lidera. W serwerach internetowych Apache zajmuje 28% rynku co czyni go 2 najpopularniejszym serwerem www, miano lidera przypada nginx z udziałem aż 34%.

W Polsce według danych serwisu Amudom z 2017 roku aż 53% serwerów używało Apache. Prawdopodobnie związane jest to z tym, że w polskich technikach wykorzystywany jest zestaw oprogramowania open source LAMP.

PHP jest to interpretowany, skryptowy język programowania, nastawiony na programowanie aplikacji webowych, generowania i budowania stron www.

Rasmus Lerdorf w 1994 roku stworzył PHP jako zestaw skryptów perla, do monitorowania stron www. W 1997 roku cały kod PHP został napisany od nowa w języku C co znacznie zwiększyło wydajność oraz stabilność. Najważniejszą jednak zmianą było wprowadzenie modułowości. Na przestrzeni lat wprowadzono do języka PHP wiele usprawnień i nowych funkcjonalności. W PHP 5 wprowadzono obsługę XML, ZIP, JSON oraz domyślnie wkompilowany silnik baz danych SQLite. Początkowo prosty język został wzbogacony o wyrażenia lambda i domknięcia znane z innych zaawansowanych języków jak Java. Obecnie wykorzystywaną jest wersja 7 języka, nad którą prace rozpoczęły się 2014 roku, mająca jako główny cel przyspieszenie pracy stron internetowych. Jest to też pierwsza wersja która pozwala na określenie typów argumentów funkcji.

Zalety

- Setki tysięcy osób w Polsce posiadających chociaż podstawową wiedzę na temat programowania w języku PHP oraz obsługi Apache.
- Prosta konfiguracja podstawowych parametrów serwera
- Zarówno PHP jak i Apache są open source, co czynie je darmowe.
- Kompatybilność z wieloma systemami operacyjnymi
- Możliwość rozbudowywania podstawowych funkcjonalności serwera poprzez instalowanie modułów np. do obsługi bazy danych
- PHP wykonuje się po stronie serwera, co daje przewagę bezpieczeństwa porównując ze skryptami wykonywanymi po stronie klienta

Wady

- Apache jest gorszy przy dużym obciążeniu niż nginx.
- Zaawansowana konfiguracja serwera jest trochę skomplikowana co może odstraszać początkujących użytkowników.
- PHP jest językiem wysokiego poziomu co czyni go niezbyt wydajnym

Konkurencyjne rozwiązania

Apache nie jest jedynym popularnym serwerem http, innymi przykładowymi serwerami są:

- Nginx filmy Nginx, Inc.
- Internet Information Services (IIS) od firmy Microsoft
- LiteSpeed Web Server od LiteSpeed Technologies
- Google Web Server (GWS) od Google

Instalacja

Aby zainstalować PHP oraz Apache potrzebne są uprawnienia root'a.

Serwer Apache instalujemy używając komendy (Fedora 32 serwer posiada już zainstalowany):

```
dnf install -y httpd
```

Po pomyślnej instalacji:

- zezwalamy na startowanie usługi wraz ze startem maszyny

  systemctl enable httpd.service

- uruchamiamy usługę

  systemctl start httpd

- sprawdzamy stan usługi

  systemctl status httpd

Następnie konfigurujemy zaporę sieciową aby umożliwiła nam dostęp do serwera web:

- zezwalamy na http

  firewall-cmd --permanent --add-service=http

- zezwalamy na https

  firewall-cmd --permanent --add-service=https

- odświeżamy ustawienia

  systemctl reload firewalld

Zainstalujemy przydatne moduły

- moduł ssl służący m.in do https

  dnf install -y mod_ssl

Sprawdzenie czy serwer działa możemy dokonać poprzez:

- znalezienie naszego adresu IP komenda: **ip a**
- wpisanie adresu do przeglądarki ***http://adres-ip***

Instalacje rozpoczynamy od zainstalowania PHP komendą:

dnf install -y php

Następnie instalujemy dodatkowe moduły do obsługi baz danych oraz skryptów:

- moduł mysql

  dnf install -y php-mysqli

- moduł postgress

  dnf install -y php-pgsql

- dodatkowo zainstalujemy perl'a

  dnf install -y perl perl-CGI

Po instalacji modułów należy zrestartować Apache by mógł używać PHP

systemctl restart httpd

Aby SELinux nie blokował ładowania PHP musimy go zarejestrować.

chcon -t texrel_shlib_t /etc/httpd/modules/libphp7.so

Pliki konfiguracyjne

Pliki konfiguracyjne z Apache znajdują się w **/etc/httpd/conf.** Główny plik konfiguracyjny nosi nazwę **httpd.conf** jeżeli nie mamy dostępu do tego pliku lub chcemy dokonać konfiguracji tylko w obrębie jednego katalogu, możemy użyć pliku **.htaccess**. Plik ten pozwala na konfiguracje Apache w obrębie katalogu i jego podkatalogach. Jednakże pozwala dokonać ustawień jedynie takich na jakie zezwala główny plik konfiguracyjny httpd.conf.

Przykładowy plik httpd.conf można znaleźć pod adresem: https://github.com/dbachon/Apache/blob/main/httpd.conf

Plik httpd.conf powinien zawierać konfiguracje:

- nadrzędny katalog z konfiguracja serwera
  - **ServerRoot "/etc/httpd"**
- adres pod którym będzie nasłuchiwał
  - **Listen 127.0.0.1:80** lub **Listen 80** gdy konfigurujemy resztę w virtual host
- nazwę serwera z portem jak w Listen
  - **ServerName localhost:80**
- nazwę użytkownika oraz grupy
  - **User apache**
  - **Group apache**
- nazwę administratora
  - **ServerAdmin root@localhost**
- nadrzędny katalog z stron www serwera
  - **DocumentRoot "/var/www/html"**

Omówienie wybranych dyrektyw

- **DirectoryIndeks** - określa jaka strona ma być przekazana użytkownikowi odwiedzającego nas serwis - domyślnie index.html
- **Directory>/patch/name</Directory>** - służy do ograniczania dostępu do katalogu, nadawania określonych dyrektyw na katalog. można także zamiast nazwy użyć regex
- **IncludeOptional** - określa położenie katalogu dodatkowych plików konfiguracyjnych.
  - np **IncludeOptional conf.d/*.conf**
- **ScriptAlias** - określa położenie katalogu skryptów CGI
- **AllowOverride** - określa jakie typy dyrektyw mogę być określane w AccessFileName
- **AccessFileName** - plik konfiguracyjny dla poszczególnych katalogów, domyślnie **.htaccess**

Konstrukcje służące do konfiguracji kontroli dostępu:

- **Order** - określa kolejność wykonywania dyrektyw Allow i Deny, domyślnie Deny, Allow
  - wykonywane są zawsze obie próby dopasowania
  - jeżeli zostaną dopasowane obydwie dyrektywy, kolejność order ma znaczenie na ostateczny wynik:
    - **Allow, Deny** - ostatecznie żądanie zostanie odrzucone
    - **Deny, Allow** - ostatecznie żądanie zostanie zaakceptowane
- **Allow** - dyrektywa ta określa jakie hosty mają dostęp do serwera
  - **Allow from all|host|env=[!]env-variable [host|env=[!]env-variable] …**
  - możemy podać pełen adres ip, nazwę domeny, częściowy adres ip, a także parę: adres sieci + maska
  - przykładowe użycie:
    - **Allow from 10.1.2.3 example.org**
    - **Allow from 192.168.1.104 192.168.1**
    - **Allow from 10.1.0.0/255.255.0.0**
- **Deny** - dyrektywa ta określa jakie hosty mają ograniczony dostęp do serwera
  - **Deny from all|host|env=[!]env-variable [host|env=[!]env-variable] …**
  - możemy podać pełen adres ip, nazwę domeny, częściowy adres ip, a także parę: adres sieci + maska
  - przykładowe użycie:
    - **Deny from 10.1.2.3 example.org**
    - **Deny from 192.168.1.104 192.168.1**
    - **Deny from 10.1.0.0/255.255.0.0**

Skrypty CGI

Common **G**ateway **I**nterface definiuje sposoby interakcji serwera internetowego z zewnętrznym programem lub skryptem generującymi zawartość. CGI to prosty sposób do umieszczania dynamicznych zawartości na swoich stronach, przy wykorzystaniu dowolnego języka programowania.

Aby CGI działało poprawnie należy dodać w pliku konfiguracyjnym Apache httpd.conf dyrektywy :

- ładującą moduł CGI do Apache

  LoadModule cgid_module modules/mod_cgid.so

- ScriptAlias informuje Apache, że określony katalog jest przeznaczony dla programów CGI. Apache przyjmie, że każdy plik w tym katalogu jest programem CGI i podejmie próbę jego wykonania, gdy ten konkretny zasób jest wymagany przez klienta. Dla domyślnych lokalizacji instalacji Apache dyrektywa wygląda następująco:

  ScriptAlias "/cgi-bin/" "/usr/local/apache2/cgi-bin/"

  Krótko mówiąc oznacza to że każde zapytanie pod /cgi-/bin/ powinno być kierowane do katalogu podanego jako drugi parametr **ScriptAlias** i powinno być realizowane jako skrypt CGI.

- Programy CGI są domyślnie ograniczone do katalogów ze względów bezpieczeństwa. Jednakże administrator może pozwolić użytkownikom mających strony internetowe w swoich domowych katalogach na użycie cgi. Aby wykonywanie skryptów cgi było dozwolone w określonym katalogu należy dodać dyrektywę:

  <Directory "/usr/local/apache2/htdocs/somedir">
  Options +ExecCGI
  </Directory>

  oraz wskazać jakie typy plików uznawane są za skrypty cgi - służy do tego dyrektywa:

  AddHandler cgi-script .cgi

  Mamy też możliwość dodania, że wszystkie pliki uznawane będą za skrypty. przykładowa dyrektywa:

  <Directory "/home/*/public_html/cgi-bin">
  Options ExecCGI
  SetHandler cgi-script
  </Directory>

  Oznacza ona, że wszystko co jest w katalogu **/home/*/public_html/cgi-bin** jest uznawane za skrypt CGI.

Cechy programów CGI

- Wszystkie dane wyjściowe muszą być poprzedzone nagłówkiem typu MIME, jest nim nagłówek HTML wyglądający zazwyczaj:

  Content-type: text/html

- Pozostałe dane wyjściowe muszą być w formacie obsługiwanym przez przeglądarkę.

Prosty skrypt test.cgi napisany w bash znajdujący się w **/var/www/cgi-bin**

```
#!/usr/bin/bash
echo "Content-type: text/html\n\n"
echo "<b>Hello, user.</b>"
```

Wywołanie programu polega na wpisaniu w przeglądarkę adresu strony:

localhost/cgi-bin/test.cgi

W przeglądarce pojawi się napis: **Hello, user**

Serwery wirtualne i przekierowania

Aby móc posiadać więcej niż jedną stronę internetową na serwerze Apache potrzebujemy użyć funkcjonalności serwerów wirtualnych.

Serwery wirtualne mogą być bazowane:

- na nazwach
- na IP

Bazowanie na nazwach serwerów wirtualnych jest zazwyczaj prostsze gdyż wymaga jedynie skonfigurowania serwera DNS tak aby mapował każdą nazwę hosta na poprawny adres IP, a następnie skonfigurowanie Apache tak by rozpoznawał różne nazwy hostów. Hosting oparty na nazwach jest zalecany ponieważ zmniejsza zapotrzebowanie na “rzadkie” adresy IP.

Przebieg wyboru hosta przez serwer oparty o nazwy:

- rozpoznawanie oparte o protokół IP
- po otrzymaniu żądania serwer dopasowuje najbardziej konkretny argument dopasowania na podstawie adresu i portu
- jeżeli jest więcej niż jeden host wirtualny zawierający powyższą konfigurację Apache będzie dalej porównywać dyrektywy serwera znajdujące się w żądaniu

  <VirtualHost>ServerNameServerAlias

  jeżeli nie uda dopasować się **ServerName** oraz **ServerAlias** zostanie uruchomiony pierwszy pasujący **VirtualHost**

Przykładowe ustawienie wirtualnego serwera(httpd.conf) aby do domeny ***www.example.com*** dodać jeszcze jedną dostępną pod adresem **other.example.com**

```
<VirtualHost *:80>
    ServerName www.example.com
    ServerAlias example.com
    DocumentRoot "/www/domain"
</VirtualHost>
<VirtualHost *:80>
    ServerName other.example.com
    DocumentRoot "/www/otherdomain"
</VirtualHost>
```

Aby serwer był dostępny pod wieloma nazwami konfigurujemy

```
SerwerAlias example.com *.example.com
```

W powyższym przypadków serwer rozpozna stronę perfiks.example.com niezależnie od perfiksu.

Przekierowania służą do odesłania żądania pod inny adres. Często jest to wykorzystywane gdy trwają prace modernizacyjne, koncern posiada kilka domen o podobnych nazwach jak na przykład facebook, wpisując w przeglądarce facebook.pl lub facebook.com wylądujemy na tej samej stronie innternetowej.

Najłatwiejszym sposobem na przekierowanie jest użycie dyrektywy **Redirect** na przykład:

```
<VirtualHost *:80>
        ServerName www.domain1.com
        Redirect / http://www.domain2.com
</VirtualHost>
<VirtualHost *:80>
        ServerName www.domain2.com
        . . .
</VirtualHost>
```

Musimy zwrócić uwagę że to przekierowanie działa tylko dla pojedynczej strony a nie całej witryny.

Dyrektywa Redirect ma format:

```
Redirect [typ] [adres strony]
```

Gdzie typ może przyjmować:

HTTP Code	Status	Description
301	permanent	The resource has permanently moved
302	temp	The resource has temporarily moved
303	seeother	The resource has been replaced and refer to new resource
305	UseProxy	Use proxy to access site
307	Temp	The resource has temporarily moved
410	Tegone	The resource has permanently removed

Strony użytkowników

W systemie z wieloma użytkownikami każdy użytkownik może mieć swoją własną stronę internetową w katalogu domowym. Aby umożliwić posiadanie stron przez użytkowników wykorzystamy dyrektywę **UserDir.** Opcję tą należy włączyć w pliku **httpd-userdir.conf,** znajdującego się w katalogu /**etc/httpd/conf.d/.**

Proces umożliwienia użytkownikom posiadania własnych stron internetowych

- UserDir ustawiamy na enabled

  UserDir enabled

- Dodajmy ścieżkę do katalogu gdzie będzie znajdować się katalog ze stronami

  UserDir public_html

  podanie ścieżki jako nazwy katalogu oznacza że katalog będzie znajdował się bezpośrednio w katalogu domowym użytkownika.

- Istnieje też możliwość zablokowania dostępu do stron z wyjątkiem do kilku użytkowników wpisując

  UserDir disabled

  UserDir enabled testuser1 testuser2

- Aby nadać każdemu użytkownikowi własny katalog cgi-bin, można użyć dyrektywy wprowadzającej określony podkatalog do katalogu macierzystego użytkownika w pliku **/etc/httpd/conf.d/cgi-enabled.conf**

  ```
  <Directory "/home/*/public_html/cgi-bin/">
      Options ExecCGI
      SetHandler cgi-script
  </Directory>
  ```

- Należy pamiętać aby zezwolić na wykonywanie skryptów CGI komenda:

  setsebool -P httpd_unified 1

Przykładowa konfiguracja pliku **httpd-userdir.conf**

```
<IfModule mod_userdir.c>
    UserDir enabled
    UserDir public_html
</IfModule>
<Directory "/home/*/public_html">
    AllowOverride FileInfo AuthConfig Limit Indexes
    Options MultiViews Indexes SymLinksIfOwnerMatch IncludesNoExec
    Require method GET POST OPTIONS
</Directory>
```

Uchwyty server-status i server-info

Plik odpowiedzialny za konfigurację: **httpd.conf**. można też dokonać konfiguracji w plikach **/etc/httpd/conf.d/server-status.conf** oraz **/etc/httpd/conf.d/server-status.conf**

server-status - zawiera informacje o aktywności i wydajności serwera
Przykładowa konfiguracja uchwytu **server-status:**

```
<Location "/server-status">
   SetHandler server-status
   Require host example.com # dostęp jedynie dla określonych hostów
</Location>
```

Informacje dostępne pod adresem: `http://your.server.name/server-status`

server-info - zawiera informacje o konfiguracji serwera
Przykładowa konfiguracja uchwytu **server-info:**

```
<Location "/server-info">
   SetHandler server-info
   Require host example.com # dostęp jedynie dla określonych hostów
</Location>
```

Informacje dostępne pod adresem: `http://your.host.example.com/server-info`

Modyfikacja uprawnień użytkowników

Aby umożliwić zabezpieczenie hasłem katalogu(strony) użytkownika potrzebne jest ustawienie dyrektywy **AllowOverride AuthConfig** w pliku httpd.conf w sekcji <Directory> albo w pliku **.htaccess**.

Kolejnym krokiem jest:

- utworzenie pliku z hasłami
  - **htpasswd -c /etc/httpd/passwd/passwords username**
- dodawanie dodatkowych użytkowników do pliku z hasłami
  - **htpasswd /etc/httpd/passwd/passwords username2**

Dodanie pliku dozwolonych grup

- utworzenie pliku o nazwie grupy z lista urzytkownikow w katalogu **/etc/httpd/passwd/groups/:**
    - **echo "GroupName: user1 user2 user323" > /etc/httpd/passwd/groups/GrupName**

Przykładowa konfiguracja ograniczenia dostępu do pliku konfiguracja w klauzulach <Directory>:

```
AuthType Basic          #typ uwierzytelniania
AuthName "By Invitation Only"
# Optional line:
AuthBasicProvider file
AuthUserFile "/etc/httpd/passwd/passwords"  #lokalizacja pliku .htpasswd
Require user username1 username2  #uprawnieni użytkownicy (valid-user wszyscy autoryzowani)
AuthGroupFile " /etc/httpd/passwd/groups"  #lokalizacja pliku z danymi grupami
Require group GroupName1 GroupName 2 #uprawnione grupy użytkowników
```

Moduł mod_rewrite

Moduł ten służy np. do zamiany adresów URL na łatwe do zapamiętania (tzw. pretty links), tworzenie przekierowań.

Wykorzystanie tego modułu realizujemy za pomocą pliku **.htaccess** w folderze strony internetowej.

- Przykład 1

W zależności czy użytkownik włącza stronę z przeglądarki mobilnej czy zwykłej - włącza mu się inna wersja strony

```
RewriteCond  "%{HTTP_USER_AGENT}"  "(iPhone|Blackberry|Android)"
RewriteRule  "^/$"          "/homepage.mobile.html"

RewriteRule  "^/$"          "/homepage.std.html"
```

- Przykład 2

Zamiana “brzydkiego” linku na “ładny”:

http://example.com/results.php?category=books

na

http://example.com/category/books

RewriteRule ^category/([A-Za-z0-9-]+)/?$ results.php?category=$1

Sposoby uruchamiania i zatrzymywania serwera Apache

Uruchomić, zatrzymać bądź zrestartować serwer możemy na kilka sposobów.

- polecenie service httpd
  service httpd start|stop|restart
- polecenie apachectl z parametrem k
  apachectl -k start|stop|restart
- polecenie systemctl
  systemctl start|stop|restart httpd

Sposoby monitorowania serwera Apache

Podstawowymi mechanizmami monitorowania i diagnozowania serwera Apache

- polecenie systemctl
  - **systemctl status httpd**
- wykorzystanie mechanizmu server-status
- wykorzystanie mechanizmu server-info
- komunikaty o błędach oraz działaniu serwera zawarte w logach
  - katalog **/etc/httpd/logs/** zawiera on także plik z informacjami o ruchu sieciowym na naszych stronach
- wykorzystanie phpinfo aby dowiedzieć się o statusie PHP, zainstalowanych i aktywnych modułach.
- skrypty CGI służące do np rejestrowania aktywności itp.

Zasada działania PHP oraz alternatywne rozwiązania

PHP jest językiem skryptowym więc jest wykonywany w czasie działania programu przez program zwany interpreterem. Co przyczynia się do tego że jest wolniejszy niż języki kompilowane. Skrypt PHP wykonywany jest po stronie serwera co zapewnia większy poziom bezpieczeństwa od skryptów wykonywanych po stronie serwera jak np: javascript.

PHP możemy używać jako m.in:

- skryptów CGI
- modułów serwera
- do tworzenia samodzielnej strony internetowej lub używając HTML z PHP

Alternatywne rozwiązania to np:

- perl
- javascript

Konfiguracja komputera

Jednym z najważniejszych konfiguracji komputera potrzebnych do poprawnego działania serwera Apache jest:

- skonfigurowanie nazwy maszyny
    - pełną nazwę maszyny sprawdzamy komendą
        - **hostname --fqdn**
    - zmianę nazwy dokonujemy w pliku
        - **/etc/hostname**
    - następnie wszystkie poprzednie nazwy zmieniamy na nowe w pliku
        - **/etc/hosts**
- ustawienie zapory sieciowej
- zezwolenie na określone mechanizmy w SELinux

Wykonywanie skryptów i programów przez serwer WWW

Skrypty są przetwarzane na serwerze WWW w momencie gdy użytkownik żąda informacji. Tego rodzaju skrypty mogą być uruchamiane przed załadowaniem się strony internetowej. Potrzebne są do wszystkiego co wymaga danych dynamicznych.

żądanie klienta → przetwarzanie serwera → przesłanie informacji zwrotnej do klienta

Cechy:

- kod źródłowy nie jest widoczny dla użytkownika
- dowolna technologia po stronie serwera(brak zależności od przeglądarki)
- wykonuje się na serwerze
- możliwość spersonalizowanej odpowiedzi na podstawie praw użytkownika, pełnionej funkcji, etc.
- większe bezpieczeństwo danych

Zadanie WWW na laboratorium krok po kroku

1. Instalujemy PHP
    a. **dnf install -y php**
    b. **dnf install -y pg-pgsql**
    c. **dnf install -y php-mysql**
    d. **chcon -t texrel_shlib_t /etc/httpd/modules/libphp7.so**

2. Instalujemy perla oraz perl-CGI
    a. **dnf install -y perl**
    b. **dnf install -y perl-CGI**

3. Zezwalamy na strony domowe użytkowników
    a. w pliku **/etc/httpd/conf.d/userdir.conf** dopisujemy:

```
<IfModule mod_userdir.c>
        UserDir enabled
        UserDir public_html
</IfModule>
<Directory "/home/*/public_html">
        AllowOverride FileInfo AuthConfig Limit Indexes
        Options MultiViews Indexes SymLinksIfOwnerMatch
IncludesNoExec
        Require method GET POST OPTIONS
</Directory>
```

    b. **setsebool -P httpd_enable_homedirs true**

4. Konfigurujemy server-info
    a. w pliku **/etc/httpd/conf.d/server-info.conf** dopisujemy:

```
<Location "/server-info">
        SetHandler server-info
        Require ip 192.168.YYY.0/24
</Location>
```

    b. aby mieć możliwość sprawdzenia we własnej przeglądarce dodajemy adres naszego komputera np:

```
Require ip 192.168.YYY.0/24 172.18.0.14
```

5. Konfigurujemy server-status
    a. w pliku **/etc/httpd/conf.d/server-status.conf** dopisujemy:

```
<Location "/server-status">
        SetHandler server-status
        Require ip 192.168.YYY.0/24;
</Location>>
```

    b. aby mieć możliwość sprawdzenia we własnej przeglądarce dodajemy adres naszego komputera np:

```
Require ip 192.168.YYY.0/24 172.18.0.14
```

6. Tworzymy i konfigurujemy użytkownika testuser
    a. dodajemy użytkownika
        i. **adduser testuser**
    b. ustalamy hasło
        i. **passwd testuser**
    c. tworzymy mu katalog do stron internetowych
        i. **mkdir /home/testuser/public_html**
    d. nadajemy potrzebne uprawnienia oraz własność
        i. **chown testuser /home/testuser/public_html**
        ii. **chmod 755 /home/testuser/public_html**
        iii. **chmod 755 /home/testuser**

7. Tworzymy użytkownikowi testuser stronę index.php
    a. **nano /home/testuser/public_html/index.php**
        i. Wpisujemy w treści

```
<?PHP phpinfo(); ?>
```

8. Tworzymy dodatkowy katalog i dodajemy przykładowe strony
    a. **mkdir /var/www/html2**
    b. **nano /var/www/html/index.html**
        i. Wpisujemy w treści np:

```
<h1> Strona numer 1 </h1>"
```

    c. **nano /var/www/html2/index.html**
        i. Wpisujemy w treści np:

```
<h1> Strona numer 2 </h1>"
```

9. Konfigurujemy serwery wirtualne
    a. w pliku **/etc/httpd/conf.d/vhost.conf** dopisujemy:

```
<VirtualHost *:80 *:443>
        DocumentRoot "/var/www/html"
        ServerName labXXX.domlabXXX.studmat.uni.torun.pl
</VirtualHost>
<VirtualHost *:80 *:443>
        DocumentRoot "/var/www/html2"
        ServerName www.domlabXXX.studmat.uni.torun.pl
</VirtualHost>
```

10. Zezwalamy na wykonywanie skryptów **CGI**
    a. w pliku **/etc/httpd/conf.d/cgi-enabled.conf** dopisujemy:

```
<Directory "/var/www/cgi-bin">
        Options +ExecCGI
        AddHandler cgi-script .cgi .pl
</Directory>
```

11. Tworzymy przykładowy skrypt **CGI** w perlu
    a. **nano /var/www/cgi-bin/index.cgi**
        i. dodajemy przykładową zawartość

```
#!/usr/bin/perl
print "Content-type: text/html\n\n";
print "<b>CGI Page</b>";
```

    b. **chmod 755 /var/www/cgi-bin/index.cgi**

12. Restartujemy serwer Apache oraz zezwalamy na połączenia internetowe
    a. **systemctl restart httpd;**
    b. **setsebool -P httpd_can_network_connect=1**
    c. **setsebool -P httpd_unified 1**

13. Konfigurujemy zaporę sieciową
    a. zezwalamy na http
        i. **firewall-cmd --permanent --add-service=http**
    b. zezwalamy na https
        i. **firewall-cmd --permanent --add-service=https**
    c. restartujemy zaporę
        i. **systemctl restart firewalld**

14. Logujemy się na użytkownika testuser
    a. przechodzimy do katalogu public_html
        i. **cd public_html**
    b. tworzymy plik index.html i uzupełniamy
        i. **nano index.html**
            1. wpisujemy np:
               Strona użytkownika testuser!
    c. strona będzie dostępna pod adresem
        i. **192.168.YYY.XXX/~testuser/**

15. Sprawdzamy działanie server-info i server-status
    a. aby sprawdzić działanie **server-info** wpisujemy w przeglądarce
        i. **adres_ip/server-info**
    b. aby sprawdzić działanie **server-status** wpisujemy w przeglądarce
        i. **adres_ip/server-status**
    c. należy pamiętać by nasz adres był dodany do dozwolonych adresów server-info oraz server-status (punkt 4 oraz 5)

16. Dodajemy do katalogu **/var/www/html** plik **index.php**
    a. **echo "<?PHP phpinfo(); ?>" > /var/www/html/index.php**
    b. strona będzie dostępna pod adresem:
        i. **`https://addres_ip/index.php`**
    c. aby działało szyfrowanie należy skonfigurować skonfigurować SSL konfiguracja znajduję sie w innym referacje(WWW - OpenSSL).

17. Aby pod adresami *labXXX.domlabXXX.studmat.uni.torun.pl* oraz *www.domlabXXX.studmat.uni.torun.pl* byly widoczne różne strony internetowe należy skonfigurować virtualhost (punkt 9).
    a. aby wyjść z przeglądarki komputera należy posiadać odpowiednio skonfigurowany plik hosts.

18. Sprawdzenie z jakich adresów IP łączyli się użytkownicy odwiedzający stronę *www.domlabXXX.studmat.uni.torun.pl* dokonujemy w logach serwera znajdują się one domyślnie w **/etc/httpd/logs/access_log**
    a. **cat /etc/httpd/logs/access_log | grep www.domlabXXX.studmat.uni.torun.pl**

Bibliografia:

https://news.netcraft.com/archives/category/web-server-survey/
https://pl.wikipedia.org/wiki/Apache_HTTP_Server
https://httpd.apache.org/
https://pl.wikipedia.org/wiki/PHP
https://www.tecmint.com/install-lamp-apache-mariadb-and-php-on-fedora-23/
https://httpd.apache.org/docs/2.4/howto/cgi.html
https://httpd.apache.org/docs/current/vhosts/name-based.html
https://www.w3docs.com/snippets/apache/how-to-redirect-a-web-page-with-apache.html
http://www.yolinux.com/TUTORIALS/ApacheRedirect.html
https://httpd.apache.org/docs/2.4/howto/public_html.html
https://en.wikipedia.org/wiki/Web_server
http://www.kik.pcz.pl/so-add/KSL/lekcje/l_25.html
https://httpd.apache.org/docs/2.4/mod/mod_status.html
https://httpd.apache.org/docs/2.4/mod/mod_info.html
http://httpd.apache.org/docs/2.0/mod/mod_rewrite.html
https://www.digitalocean.com/community/tutorials/how-to-set-up-mod_rewrite
http://miro.borodziuk.eu/index.php/2017/07/28/serwer-www-apache/
http://httpd.apache.org/docs/current/mod/mod_access_compat.html#allow
http://httpd.apache.org/docs/current/mod/mod_access_compat.html#order
http://httpd.apache.org/docs/current/mod/mod_access_compat.html#deny
https://pl.wikibooks.org/wiki/PHP/Odpowiedzi/Podstawy_j%C4%99zyka
https://pl.wikipedia.org/wiki/Perl
https://scand.com/company/blog/php-vs-javascript-difference-between/
https://www.bbc.co.uk/bitesize/guides/znkqn39/revision/3
https://www.geeksforgeeks.org/difference-between-sever-side-scripting-and-client-side-scripting/
http://kb.rootbox.com/zmiana-hostname-w-systemie-linux/