

Instalacja MySQL oraz Apache2 z obsługą: SSL, CGI, PHP4, PHP5, mod_rewrite

Autor : Artur Kwiatkowski alias Fo
e-mail : artur.kwiatkowski@gazeta.pl

Słowo wstępu

Instalacja oprogramowania przeprowadzona została wg. poniższych wskazówek pomyślnie na dystrybucjach: debian sarge (3.1) oraz slackware 10.0

Znak # oznacza, iż polecenie wykonywane było jako użytkownik root.

Użytkownik [user] : oznacza zwykłego użytkownika - nie mającego praw root'a. Użytkownika takiego możemy stworzyć poprzez wykonanie jako root następującego polecenia:

```
# adduser
```

i odpowiedzi na wyświetlane pod konsolą pytania takie jak nazwa użytkownika, hasło.

Instalacja bazy danych MySQL

- źródła : <http://dev.mysql.com/downloads/mysql/4.0.html>
- instalowana wersja podczas tworzenia artykułu :
<http://dev.mysql.com/get/Downloads/MySQL-4.0/mysql-4.0.24.tar.gz/from/pick#mirrors>
- dokumentacja : <http://dev.mysql.com/doc/>

Pierwszym krokiem będzie pobranie źródeł oraz dodanie grupy i użytkownika systemowego pod którym uruchamiany jest serwer baz danych MySQL

```
# su - c [user]
# wget http://sunsite.icm.edu.pl/mysql/Downloads/MySQL-4.0/mysql-4.0.24.tar.gz
# exit
# groupadd mysql
# useradd -g mysql mysql
```

Następnie rozpakowujemy pobrane źródła:

```
# tar -zxvf mysql-4.0.24.tar.gz
```

Przechodzimy do katalogu do którego ów źródła zostały rozpakowane i przystępujemy do procesu kompilacji:

```
# cd mysql-4.0.24
# ./configure --prefix=/usr/local/mysql --with-charset=latin2
# make
# make install
```

Po pomyślnej instalacji bazy danych, musimy zainstalować jeszcze bazę danych mysql - bez której nasz serwer nie będzie poprawnie funkcjonować :

```
# ./scripts/mysql_install_db
```

Po zainstalowaniu się w katalog /usr/local/mysql/var bazy mysql zmieniamy właściciela i grupę katalogu w którym egzystuje MySQL :

```
# chown -R mysql /usr/local/mysql
# chgrp -R mysql /usr/local/mysql
```

Dzięki temu zabiegowi nasz serwer baz danych będzie uruchamiany z prawami normalnego użytkownika (mysql). Następnie do katalogu, który określiliśmy przy kompilacji parametrem --sysconfdir kopiujemy plik my.cnf

```
# cp ./support-files/my-medium.cnf /etc/my.cnf
```

Uruchamiamy serwer MySQL :

```
# /usr/local/mysql/bin/mysqld_safe &
```

Sprawdzamy czy rzeczywiście zaczął on funkcjonować :

```
# /usr/local/mysql/bin/mysqladmin ping
```

Powinna zostać wyświetlona następująca informacja :

```
# mysqld is alive
```

Zdefiniujemy teraz hasło do naszego serwera mysql tak aby nie pozostał on otwarty :

```
# /usr/local/mysql/bin/mysqladmin -u root password nasze_haslo
```

Aby uprościć sobie dostęp do najważniejszych narzędzi, utworzymy do nich symboliczne dowiązania :

```
# ln -s /usr/local/mysql/bin/mysqladmin /usr/local/bin/mysql.admin
# ln -s /usr/local/mysql/bin/mysql /usr/local/bin/mysql.console
```

Zobaczmy jeszcze czy aby napewno do naszego serwera nie można się dostać bez hasła :

```
# mysql.console
ERROR 1045: Access denied for user: 'root@localhost' (Using password: NO)
```

Tak więc bez hasła nikt na root'a mysql się nie dostanie, sprawdźmy jeszcze tylko czy my możemy się do niego dostać :

```
# mysql.console -u root -p
Enter password : *****
```

Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 13 to server version: 4.0.23a-log

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

```
# exit
```

Aby nasz serwer mysql po restarcie maszyny został uruchomiony dokładamy odpowiedni wpis :

➤ dla slackware :

```
# echo '/usr/local/mysql/bin/mysqld_safe &' >> /etc/rc.d/rc.local
```

➤ dla debian'a :

```
# touch /etc/init.d/initialization
# echo '/usr/local/mysql/bin/mysqld_safe &' >> /etc/init.d/initialization
```

Jeżeli nie chcemy aby dostęp do mysql był również z zewnątrz (poza localhost) w pliku konfiguracyjnym MySQL : /etc/my.cnf dokonujemy małych modyfikacji - dopisujemy w sekcji [mysqld] :

➤ /etc/my.cnf:

```
[mysqld]
bind-address = 127.0.0.1
```

Instalacja serwera www - Apache2

- źródła : <http://httpd.apache.org/download.cgi>
- instalowana wersja podczas tworzenia artykułu :
<http://www.apache.net.pl/httpd/httpd-2.0.54.tar.gz>
- dokumentacja : <http://httpd.apache.org/docs-2.0/>

Apache2 instalować będziemy z obsługą SSL'a - tak więc przed dotknięciem się do źródeł apache2, instalujemy najnowszą wersję OpenSSL

- źródła : <http://www.openssl.org/source/>
- instalowana wersja podczas tworzenia artykułu :
<http://www.openssl.org/source/openssl-0.9.7g.tar.gz>
- dokumentacja : <http://www.openssl.org/docs/>

```
# su - c [user]
# wget http://www.openssl.org/source/openssl-0.9.7g.tar.gz
# exit
# tar -zxvf openssl-0.9.7g.tar.gz
# cd openssl-0.9.7
# ./config --prefix=/usr/local/ssl
# make
# make install
```

Po zainstalowaniu SSL'a zabieramy się za naszego apache2 :

```
# tar -zxvf httpd-2.0.54.tar.gz
# cd httpd-2.0.54
# ./configure --prefix=/usr/local/apache2 --enable-so --enable-modules=rewrite --enable-shared=rewrite --enable-ssl --enable-rewrite
# make
# make install
```

Po zainstalowaniu serwera apache2, zaczynamy zabawę z naszym httpd.conf, w tym celu zaprzęgamy do roboty nasz ulubiony edytor tekstu, u mnie był to debianowy nano, ale modyfikacji można równie dobrze dokonywać z poziomu edycji w MC (midnight commander). Poniżej załączam opis - jak powinny wyglądać odpowiednie linijki w httpd.conf

➤ /usr/local/apache2/conf/httpd.conf

```

"ServerRoot /usr/local/apache2"
Listen nasze_ip:80
Listen nasze_ip:443

ServerName nasze_ip

DocumentRoot "/home/www" #- powiedzmy, że takie ;)

<Directory "/home/www">
Options Indexes FollowSymLinks
....
</Directory>

DirectoryIndex index.html index.htm

AddDefaultCharset ISO-8859-2 #(tej linijki w httpd.conf nie znajdziemy, trzeba ją dopisać)

NameVirtualHost nasze_ip:80
NameVirtualHost nasze_ip:443

#(sekcja virtualhost - przykładowy nowy Vhost bez obsługi SSL)

<VirtualHost nasze_ip:80>
ServerAdmin admin@space
DocumentRoot /home/www/localhost
ServerName nasze_ip
ErrorLog logs/localhost-error_log
CustomLog logs/localhost-access_log common
</VirtualHost>

#(przykładowy nowy Vhost z obsługą SSL)

<VirtualHost nasze_ip:443>
DocumentRoot /home/www/localhost
ServerName nasze_ip:443
ServerAdmin root@space
ErrorLog logs/localhost-error_log
CustomLog logs/localhost-access_log common
SSLEngine on
SSLCipherSuite ALL:!ADH:!EXPORT56:RC4+RSA:+HIGH:+MEDIUM:+LOW:+SSLv2:+EXP:+eNULL
SSLCertificateFile /usr/local/apache2/conf/ssl.crt/server.crt
SSLCertificateKeyFile /usr/local/apache2/conf/ssl.key/server.key
SetEnvIf User-Agent ".MSIE.*" \
nokeepalive ssl-unclean-shutdown \
downgrade-1.0 force-response-1.0
</VirtualHost>

```

Teraz należy jeszcze co niego zmodyfikować plik : /usr/local/apache2/conf/ssl.conf i zahashować co poniektóre linijki, ponieważ obsługa virtualhosta z SSL ustawiać będziemy w httpd.conf smile.gif

➤ /usr/local/apache2/conf/ssl.conf

```
#Listen 443
#<VirtualHost _default_:443>
# General setup for the virtual host
#DocumentRoot "/home/www/"
#ServerName nasze_ip:443
#ServerAdmin root@space
#ErrorLog /usr/local/apache2/logs/error_log
#TransferLog /usr/local/apache2/logs/access_log
#SSLEngine on
#SSLCipherSuite (to jest jedna linia )
ALL:!ADH:!EXPORT56:RC4+RSA:+HIGH:+MEDIUM:+LOW:+SSLv2:+EXP:+eNULL
#SSLCertificateFile /usr/local/apache2/conf/ssl.crt/server.crt
#SSLCertificateKeyFile /usr/local/apache2/conf/ssl.key/server.key
#SSLOptions +FakeBasicAuth +ExportCertData +CompatEnvVars +StrictRequire
#<Files ~ "\.(cgi|shtml|phtml|php3?)$" >
#  SSLOptions +StdEnvVars
#</Files>
#<Directory "/usr/local/apache2/cgi-bin">
#  SSLOptions +StdEnvVars
#</Directory>
#SetEnvIf User-Agent ".*MSIE.*" \
#  nokeepalive ssl-unclean-shutdown \
#  downgrade-1.0 force-response-1.0
#</VirtualHost>
```

Ok modyfikacji dość, teraz zrobimy sobie dla wygody linka do apachectl

```
# ln -s /usr/local/apache2/bin/apachectl /usr/local/bin/apache2
```

Jak również dołożymy odpowiednią linijkę do rc.local lub naszego debianowego skryptu initialization aby po restarcie maszyny serwer został zainicjowany :

- dla slackware :

```
# echo '/usr/local/apache2/bin/apachectl startssl' >> /etc/rc.d/rc.local
```

- dla debian'a :

```
# echo '/usr/local/apache2/bin/apachectl startssl' >> /etc/init.d/initialization
```

oraz co się tyczy stricto Debiana : o ile nie zrobiliśmy już tego wcześniej aby nasz skrypt startowy wogóle był skrytem startowym systemu, wywołać musimy następującą komendę :

```
# update-rc.d /etc/init.d/initialization defaults
```

Zainstalowaliśmy wcześniej openssl'a tak więc teraz kiedy mamy również zainstalowany serwer www, damy mu możliwość współpracy z SSL.

Generujemy główny certyfikat dla naszego serwera :

```
# openssl genrsa -des3 -out server.key 1024
```

(pamiętaj jakie ustawiasz hasło dla certyfikatu)

Usuujemy hasło z klucza aby umożliwić automatyczny start apache2

```
# openssl rsa -in server.key -out server.pem
```

(podajemy to samo hasło które wpisaliśmy podczas generowania certyfikatu)

Generujemy podpis certyfikatu

```
# openssl req -new -key server.pem -out server.csr
```

Wypełniamy pola : country, state, city w momencie kiedy zostaniemy zapytani o 'organization unit' zostawiamy to pole puste [enter] pole common name - w nim powinniśmy podać albo adres ip naszego serwera albo jego domene, zalecam domene. pola 'a challenge password' nie musimy wypełniać.

Generujemy tzw. self-signed certificate - certyfikat podpisany 'przez siebie' (podajemy takie samo hasło jakie podawaliśmy podczas generowania server.pem)

```
# openssl x509 -req -days 365 -in server.csr -signkey server.pem -out server.crt
```

Ok, kiedy wszystko już mamy przygotowane, zabieramy się do instalacji naszego prywatnego klucza oraz certyfikatu do apache2 :

```
# mkdir /usr/local/apache2/conf/ssl.crt
# cp server.crt /usr/local/apache2/conf/ssl.crt/server.crt
# mkdir /usr/local/apache2/conf/ssl.key
# cp server.pem /usr/local/apache2/conf/ssl.key/server.key
```

Kiedy wszystko już praktycznie jest gotowe, sprawdzamy czy nie mamy jakichś błędów w httpd.conf :

```
# apache2 -t
```

Syntax OK.

Jeżeli wszystko jest ok, włączamy naszego apache2 z obsługą SSL

```
# apache2 startssl
```

Spojrzymy dla pewności w procesy i wywołajmy url w przeglądarce aby mieć pewność że serwer pracuje prawidłowo.

W procesach widnieć powinno kilka pozycji podobnych do poniższej :

```
nobody 14682 0.0 1.0 15296 5168 ? S 09:05 0:00 /usr/local/apache2/bin/httpd -k start -DSSL
```

Obsługa mod_rewrite

Aby nasz mod_rewrite z którym skompilowaliśmy apache2 funkcjonował poprawnie należy w httpd.conf dokonać następujących modyfikacji :

Wyedytujemy httpd.conf w okolicach linijki 335 zaraz pod DocumentRoot "/home/www", struktura musi być następująca jeżeli chcemy mieć możliwość korzystania z mod_rewrite z poziomu .htaccess

➤ /usr/local/apache2/conf/httpd.conf

```
<Directory />
Options FollowSymLinks
AllowOverride FileInfo
</Directory>
```

Kawałek dalej :

```
<Directory "/home/www">
Options Indexes FollowSymLinks
AllowOverride FileInfo
Order allow,deny
Allow from all
</Directory>
```

W dowolnym miejscu w httpd.conf dorzucamy jeszcze następujące linijeczki :

```
# mod_rewrite support
<IfModule mod_rewrite.c>
RewriteEngine On
RewriteLog "/usr/local/apache2/logs/rewrite.log"
RewriteLogLevel 9
</IfModule>
```

Jako iż logów mod_rewrite'a potrafi przybyć całkiem sporo i to w niedługim czasie, zalecam zadbanie o odpowiednią ich rotację, lub po prostu nie dopisywanie dwóch przed ostatnich linii do httpd.conf.

Aby przetestować czy mod_rewrite działa poprawnie, należy do katalogu dla VirtualHosta, którego sobie utworzymy wprowadzić plik .htaccess o następującej zawartości :

```
RewriteEngine On
RewriteCond %{HTTP_HOST} ^.*$ [NC]
RewriteRule ^(.*)$ http://www.domain.com/ [R]
```

Dzięki dobrodziejstwu .htaccess i mod_rewrite po wejściu na url, dla którego DocumentRoot wskazuje na katalog zawierający .htaccess z powyższą zawartością zostaniemy przekierowani na adres www.domain.com

Obsługa CGI

Aby mieć możliwość uruchamiania na naszej maszynie skryptów perl'a (.cgi .pl) należy w pliku konfiguracyjnym httpd.conf dokonać paru małych modyfikacji :

➤ /usr/local/apache2/conf/httpd.conf

Modyfikujemy linijkę DirectoryIndex

```
DirectoryIndex index.html index.htm index.cgi index.pl
```

Dodajemy linijeczkę :

```
AddHandler cgi-script .cgi .pl
```

Oraz określamy w jakich katalogach mają być parsowane pliki .cgi oraz .pl - jako skrypty CGI, w naszym przypadku wszystkie katalogi cgi-bin we wszystkich podkatalogach * w katalogu /home/www

```
<Directory "/home/www/*/cgi-bin">
Options +ExecCGI
</Directory>
```

Teraz sprawdzimy czy nasze skrypty CGI aby napewno działają poprawnie, otwieramy nasz ulubiony edytor i wpisujemy :

```
#!/usr/bin/perl
print "Content-type: text/html\n\n";
print "Hello, Im a CGI script and Im gonna print You some environment variables to make You belive that Im
working fine smile.gif \n";
print "
";

foreach $key (keys %ENV)
{
    print "$key --> $ENV{$key}
";
}
}
```

Nasz kod zapisujemy jako plik .cgi lub .pl w miejscu w którym nadaliśmy możliwość parsowania plików jako skrypty CGI. Następnie nadajemy odpowiednie prawa dla pliku :

```
# chmod 755 /home/www/localhost/cgi-bin/skrypt.cgi
# chmod +x /home/www/localhost/cgi-bin/skrypt.cgi
```

Teraz sprawdzamy już tylko w przeglądarce czy wszystko działa poprawnie podając w url'u ścieżkę do naszego skryptu cgi np. : http://nasze_ip/cgi-bin/skrypt.cgi. Jeżeli wszystko działa poprawnie w przeglądarce powinno pokazać się co nieco zmiennych.

Obsługa php4

- źródła : <http://www.php.net/downloads.php>
- instalowana wersja podczas tworzenia artykułu :
<http://pl.php.net/distributions/php-4.3.11.tar.gz>
- dokumentacja : <http://pl.php.net/manual/pl/>

Zatrzymujemy serwer apache :

```
# apache2 stop
```

Następnie pobieramy źródła, rozpakowujemy je i przystępujemy do kompilacji. php4 zainstalujemy jako moduł dla apache2.

Moja instalacja php4 była non-standard - tak więc doinstalowałem wcześniej rzeczy takie jak :

- ❖ Zlib 1.2.2 - <http://prdownloads.sourceforge.net/libpng/zlib-1.2.2.tar.gz?download>
- ❖ curl 7.13.1 - <http://curl.haxx.se/download.html>
- ❖ gd 2.0.33 - <http://www.boutell.com/gd/>
- ❖ libmcrypt 2.5.7 –
http://sourceforge.net/project/showfiles.php?group_id=87941&package_id=91774&release_id=178782
- ❖ mcrypt 2.6.4 –
http://sourceforge.net/project/showfiles.php?group_id=87941&package_id=91948&release_id=178780
- ❖ mhash 0.9.2 - http://sourceforge.net/project/showfiles.php?group_id=4286
- ❖ libiconv 1.9.2 - <ftp://ftp.gnu.org/gnu/libiconv/>
- ❖ libjpeg-6b - <http://site.n.ml.org/info/libjpeg/>
- ❖ libtiff 3.7.1 - <http://dl.maptools.org/dl/libtiff/>

Instalacja tych elementów to prawie zawsze samo :

```
./configure
make
make install
```

Mimo to przed wykonaniem powyższego należy się upewnić wydając polecenie

```
./configure --help
```

Pamiętać również należy o tym aby nasz plik /etc/ld.so.conf zawierał w sobie między innymi :

```
/lib
/usr/lib
/usr/local/lib
/usr/local/ssl/lib
```

Co sprawi iż aplikacje, które będziemy instalować (nie tylko rzeczy zawarte w tym artykule ale również cokolwiek innego) będą posiadać informacje na temat środowiska bibliotek naszego systemu - krócej - plik ld.so.conf zawiera informacje dla środowiska bibliotek systemu w jakich katalogach leżą biblioteki które należy uwzględnić - po dopisaniu powyższych linii do ld.so.conf wydajemy polecenie # ldconfig

Jeżeli instalację przeprowadzasz na systemie debian – odnajdź nazwę pakietu na stronie www.packages.debian.org i użyj narzędzia apt-get do zainstalowania wyżej wymienionych bibliotek. Pamiętaj, że w niektórych przypadkach warto również doinstalować biblioteki z gałęzi development (dev).

```
# su -c [user]
# wget http://pl.php.net/distributions/php-4.3.11.tar.gz
# exit
# tar -zxvf php-4.3.11.tar.gz
# cd php-4.3.11.tar.gz
```

Jeżeli na serwerze masz zainstalowanego jakiegoś MTA (mail transport agent) z obsługą imap'u to do poniższego configure dołączyć możesz --with-imap --with-imap-ssl

Poniższe argumenty dla ./configure znajdować się muszą w jednej ciągłej linii.

```
# ./configure --prefix=/usr/local/php4 --with-config-file-path=/usr/local/php4 --sysconfdir=/usr/local/php4 --with-apxs2=/usr/local/apache2/bin/apxs --with-mysql=/usr/local/mysql --with-openssl=/usr/local/ssl --enable-discard-path --enable-debug --enable-track-vars --enable-versioning --with-openssl=/usr/local/ssl --with-xml --enable-bcmath --with-bz2 --enable-calendar --with-jpeg-dir=/usr/local --with-png-dir=/usr/lib --with-tiff-dir=/usr/local --with-ttf=/usr/lib --with-curl --with-db --with-dbase --with-pear --enable-exif --enable-ftp --with-gettext --with-iconv --with-iconv-dir=/usr/local --enable-mbstring --with-mcrypt --with-mhash --with-zlib --with-gd --with-gd-native-ttf --with-xslt-sablot=/usr/lib --enable-xslt --enable-wddx --with-kerberos --with-ncurses
# make
# make install
```

Teraz zapewne jako iż instalowaliśmy php4 jako moduł do apache2 w pliku :
/usr/local/apache2/conf/httpd.conf - powinna znajdować się linijka odpowiedzialna za
załadowanie modułu : LoadModule php4_module modules/libphp4.so

Do httpd.conf dorzucamy jeszcze samą obsługę parsowania php - ponieważ samo
załadowanie modułu nie wystarcza, tak więc dopisujemy np. pod LoadModule php4_module
modules/libphp4.so linijki :

➤ /usr/local/apache2/conf/httpd.conf

```
AddType application/x-httpd-php .php  
AddType application/x-httpd-php .php4
```

Oraz naszą linijkę DirectoryIndex wzbogacamy o dwa kolejne rozszerzenia :

```
DirectoryIndex: index.php index.php4
```

Teraz skoro mamy już obsługę php4, sprawdźmy czy działa ona poprawnie :

```
# apache2 startssl  
# touch /home/www/localhost/phpinfo.php  
# echo '<?php phpinfo(); ?>' >> /home/www/localhost/phpinfo.php
```

Wywołujemy w przeglądarce plik phpinfo.php i naszym oczom powinna się ukazać ładna
informacja na temat php4.

Obsługa php5

Obsługę php5 w tym przypadku zamontujemy jako CGI.

Kwestia dwóch wersji php jako modułów na jednym porcie bez żadnych ProxyPass'ów itp itd. jest sprawą w chwili tworzenia tego artykułu na tyle skomplikowaną iż w tej wersji artykułu nie będzie ona uwzględniona.

- źródła : <http://www.php.net/downloads.php>
- instalowana wersja podczas tworzenia artykułu :
<http://pl.php.net/distributions/php-5.0.4.tar.gz>
- dokumentacja : <http://pl.php.net/manual/pl/>

Ściągamy źródła, rozpakowujemy i instalujemy na pokładzie php5 jako CGI.

```
# apache2 stop
# su - c [user]
# wget http://pl.php.net/distributions/php-5.0.4.tar.gz
# tar -zxvf php-5.0.4.tar.gz
# cd php-5.0.4
```

Podobnie jak ostatnim razem - jeżeli na naszym systemie posiadamy oprogramowanie pozwalające na dołączenie funkcji --with-imap --with-imap-ssl, dołączamy je.

Poniższe argumenty użyte przy configure to tylko przykładowe rozwiązanie.

Wszystko w jednej linii.

```
# ./configure --prefix=/usr/local/php5 --with-config-file-path=/usr/local/php5 --sysconfdir=/usr/local/php5 --with-mysql=/usr/local/mysql --with-zlib --with-bzip --with-gd --enable-force-cgi-redirect --enable-bcmath --enable-calendar --enable-ctype --enable-dbase --enable-discard-path --enable-exif --enable-filepro --enable-ftp --enable-gd-imgstrttf --enable-gd-native-ttf --enable-inline-optimization --enable-mbstr-enc-trans --enable-mbstring --enable-mbregex --enable-track-vars --enable-versioning --enable-wddx --enable-bz2 --with-dom=/usr/include/libxml2 --with-ftp --with-gettext --with-gmp --with-jpeg-dir=/usr/local --with-mcal=/usr/include --with-mcrypt --with-mhash --with-png-dir=/usr/lib --with-iconv --with-ncurses --with-xml --with-xslt-sablot=/usr/lib --enable-xslt --with-kerberos --enable-fastcgi --with-openssl=/usr/local/ssl --with-tiff-dir=/usr/local --with-curl
# make
# make install
```

Teraz musimy wyedytować ponownie `/usr/local/apache2/conf/httpd.conf` i dorzucamy do niego taki wpis :

➤ `/usr/local/apache2/conf/httpd.conf`

```
AddType application/x-httpd-php5 .php5
ScriptAlias /php/ "/usr/local/php5/bin/"
Action application/x-httpd-php5 "/php/php"
```

Oraz oczywiście do linii `DirectoryIndex` dorzucamy `index.php5`

Następnie pozostaje już tylko start `apache2` (`# apache2 startssl`) i utworzenie pliku w `/home/www/localhost/phpinfo.php5` z zawartością `<? phpinfo(); ?>`

Warto mieć również na względzie jeżeli chodzi o kwestie bezpieczeństwa pliki `php.ini` i dyrektywe : `disable_functions` która u mnie przy obu wersjach `php` zawiera następującą zawartość :

```
disable_functions = shell_exec, exec, system, passthru, escapeshellcmd, escapeshellarg
```

Prawa Autorskie

AUTOR NIE PONOSI ODPOWIEDZIALNOŚCI ZA SZKODY POWSTAŁE W WYNIKU UŻYWANIA TEGO DOKUMENTU.

Za żadne szkody autor tego dokumentu nie jest odpowiedzialny, bez względu na to, co się może przydażyć (włączając w to bez żadnych ograniczeń celowe, przypadkowe, powstałe w wyniku jakichkolwiek przyczyn straty finansowe, utratę danych, upadek firmy i inne możliwe straty) w wyniku używania tego dokumentu.