Serwer Apache w połączeniu z OpenSSL

Autorzy: Radosław Bednarski, Tomasz Kaleta IV FDS

STRESZCZENIE

Dokument zawiera sposób integracji Apache'a z OpenSSL i modułu modSSL.
Poniższy opis przeprowadzi nas przez kompilację, instalację oraz konfigurację poszczególnych pakietów.

SPIS TREŚCI

1. WSTĘP

Korzystamy ze sklepów i banków internetowych, dokonujemy transakcji poprzez Internet nie zdając sobie sprawy z niebezpieczeństwa na jakie się narażamy w czasie wykonywania tych czynności. Większość danych wysyłanych z naszego komputera jest niezabezpieczonych i może zostać wykorzystane do włamania się do naszego komputera, do przechwycenia haseł do naszego konta pocztowego, a nawet do przechwycenia numeru naszej karty kredytowej.

Aby zadbać o swoje bezpieczeństwo w Internecie powinniśmy zadbać o jedną z najważniejszych rzeczy jaką jest. szyfrowanie przesyłanych danych.

We wszystkich bankach internetowych realizowana jest transmisja szyfrowana, która zabezpiecza przesyłane hasła przed przechwyceniem.

1.1 Cel projektu

W tym projekcie skupimy się na przedstawieniu jednego ze sposobów integracji serwera protokołu HTTP – Apache z protokołem szyfrowania SSL - OpenSSL. Dzięki takiemu połączeniu osiągniemy bezpieczną wymianę danych pomiędzy użytkownikiem a serwerem.

1.2 Wykorzystany sprzęt i oprogramowanie

Do realizacji projektu wykorzystaliśmy komputer klasy PC o następujących parametrach: procesor Pentium 90MHz, pamięć operacyjna 32 MB RAM-u, dysk twardy 3,2 GB, dwie karty sieciowe 3com509.

Jako system operacyjny wybraliśmy Debian GNU/Linux Potato rev5 opartym na jądrze 2.2.19. Oprogramowanie wykorzystane do stworzenia serwera protokołu HTTPS (HTTP + SSL) wykorzystaliśmy następujące pakiety: apache_1.3.27 [9], openssl-0.9.7 [11] oraz mod_ssl-2.8.12-1.3.27 [10]

1.3 Prawa autorskie

W tym podrozdziale zamieszczone są odnośniki do licencji każdego z użytych pakietów:

a) Licencja dla pakietu Apache – Licencja pakietu Apache
b) Licencja dla pakietu OpenSSL – Licencja pakietu OpenSSL
c) Licencja dla pakietu modssl – Licencja pakietu Modssl

Debian GNU/Linux wspiera oprogramowanie wolnodostępne. Ponieważ wiele programów opieraja się na wielu różnych licencjach, stworzono Debian Free Software Guidelines (DFSG) (Wytyczne Debiana dotyczące Oprogramowania Wolnodostępnego), które określają jednoznacznie czym jest oprogramowanie wolnodostępne w rozumieniu Debiana. Tylko oprogramowanie zgodne z tymi wytycznymi może znaleźć się w głównej dystrybucji Debiana. Debian posiada większość pakietów opartych na licencji GNU (Oryginał GNU). [4][6]

2. SERWER APACHE

2.1 Trochę historii

Apache jest najpopularniejszym i najchętniej wykorzystywanym serwerem HTTP w systemach operacyjnych bazujących na Unix'ie, w obecnej chwili używa go 63% użytkowników. Serwer ten dostępny też jest dla innych systemów operacyjnych min. BeOS, OS/2, NetWare, MacOS, Windows. Projekt Apache powstał w 1996 roku, został utworzony przez grupę wybranych programistów, którzy chcieli stworzyć lepszy serwer protokołu http od tych, które były aktualnie dostępne. Projekt ten został oparty na kodzie źródłowym programu httpd 1.3 stworzonego przez organizację NCSA (National Center for Supercomputing Applications) na początku roku 1995. Rozwojem serwera Apache zajmuje się Apache Group - zespół programistów ochotników którzy rozwijają program jako Apache Projekt. Główna baza informacji na temat programu znajduje się pod adresem http://www.apache.org/ gdzie znajdziemy więcej informacji na ten temat.

2.2 Możliwości

Apache posiada niemal nieograniczone możliwości, współpracuj on z php, cgi, asp, perl, ssl, sql, itp. Możliwości serwera rozszerzane są poprzez dodawanie modułów wspomagających obsługę dodatkowych funkcji dla serwera. Przykładowa lista modułów:

Tabela 1.

Wybrane moduły umieszczone w dystrybucji serwera Apache wersja 1.3.x	
Core	Podstawowe funkcje zawsze dostępne w dystrybucji
mod_access	Określa kontrolę dostępu w oparciu o hosta lub domenę
mod_actions	Odpowiada za wykonywanie skryptów CGI w zależności od typu danych lub sposobu pobrania. Odzwzorowuje skrypty CGI na plik typu MIME
mod_auth	Włącza uwierzytelnianie użytkownika
mod_cgi	Włącza obsługę - wykonywanie skryptów CGI
mod_include	Pozwala włączać zawartości plików lub wyniki działania skryptu do zwykłych plików HTML i zwracać ich zawartość klientowi.
mod_info	Włącza użycie programu odpowiedzialnego za informację o ustawieniach serwera Apache.
mod_log_agent	Zapisywanie w logach nazw i wersji przeglądarek internetowych klientów.
mod_negotiation	Odpowiedzialny za uzgadnianie najlepszej reprezentacji danych w przeglądarce klienta. Wprowadzony ze względu na zgodność z HTTP/1.1, umożliwia negocjację zawartości MIME
mod_proxy	Apache staje się serwerem proxy dla stron WWW, przyspiesza dostęp do często używanych danych, gdy serwer WWW (komputer) jest wykorzystywany do zapamiętywania danych.
mod_userdir	Ustawienia dotyczące katalogów domowych użytkowników. Definiuje, gdzie użytkownicy mogą tworzyć publiczne strony www
mod_usertrack	Śledzenie zachowania użytkowników za pomocą Cookies (tzw. ciasteczka), szczególnie przydatne dla np. stałych klientów w sklepach internetowych lub do określania preferencji użytkownika.

Całą listę dostępnych modułów w wersji 1.3.x serwera Apache jest dostępna pod tym adresem:

a) Posegregowane zgodnie z typami modułów
http://httpd.apache.org/docs/mod/index-bytype.html

b) Posegregowane alfabetycznie
http://httpd.apache.org/docs/mod/index.html

Każdy moduł możemy załadować na dwa sposoby:

a) Podczas kompilacji źródeł można wkompilować obsługę wybranych modułów do Apache'a

b) Po zainstalowaniu i uruchomieniu Apache'a można dynamicznie ładować do niego dowolne moduły

Wkompilowanie modułów bezpośrednio w kod serwera może mieć wpływ na jego szybkość działania, ma to szczególnie wpływ w systemach wykorzystywanych u provider'ów, gdzie ilość zapytań użytkowników potrzebuje szybko i sprawnie działającego serwera.
W naszym przypadku nie ma to większego znaczenia.

2.3 Instalacja

Instalację serwera Apache'a możemy przeprowadzić na kilka sposobów:

a) Kompilując źródła wcześniej ściągnięte z Internetu

b) Za pomocą gotowej „paczki" deb (w przypadku Debian'a) lub rpm (dla Red Hat'a)

c) Przy pomocy narzędzia Dselect korzystającego z polecenia dpkg dostępnego w dystrybucji wykorzystanej przez nas do realizacji projektu

d) Bespośrednio wykorzystując polecenie dpkg --install pakiet.tar.gz

Możliwości instalacji programu w Linux'ie nie kończą się na tych 4 metodach, jeżeli nie napisaliśmy o jakiejś metodzie to nie znaczy, że ona nie istnieje, lecz że my o niej nic nie wiemy.

My opiszemy instalację ze źródeł, która pozwala na całkowitą kontrolę nad instalowanym pakietem. Daje to większe bezpieczeństwo i stabilniejszą pracę na instalowanym serwerze.

2.4 Instalacja ze źródeł

Instalacja Apache'a ze źródeł sprowadza się do rozpakowania, skonfigurowania i skompilowania ściągniętego wcześniej pakietu.

Rozpakowujemy ściągnięty pakiet:

```
tar -zxf apache_1.3.27.tar.gz
```

Zostaje utworzony katalog `apache_1.3.27`, w którym zostają umieszczone rozpakowane pliki Apache'a

Przystępujemy do konfiguracji źródeł:

Plik konfiguracyjny `Configuration` znajduje się w katalogu `apache_1.3.27/src`. Konfiguracja polega na „odhaszowaniu" pewnych linijek, lub sporządzeniu odpowiednich wpisów do tego pliku.

Pakiet Apache możemy skonfigurować poprzez dołączony skrypt configure (w katalogu `apache_1.3.27`), który umożliwia automatyczne skonfigurowanie pakietów poprzez przekazywanie parametrów do skryptu np.:

```
./configure --prefix=/usr/local/apache \
--enable-module=expires \
--enable-module=headers \
--enable-module=log_agent \
--enable-module=log_referer \
--enable-module=usertrack
```

Dla naszych potrzeb wystarczy standardowa konfiguracja, więc wpisujemy polecenie:

```
./configure
```

```
Configuring for Apache, Version 1.3.27
 + Warning: Configuring Apache with default settings.
 + This is probably not what you really want.
 + Please read the README.configure and INSTALL files
 + first or at least run './configure --help' for
 + a compact summary of available options.
 + using installation path layout: Apache (config.layout)
Creating Makefile
Creating Configuration.apaci in src
Creating Makefile in src
 + configured for Linux platform
 + setting C compiler to gcc
 + setting C pre-processor to gcc -E
 + checking for system header files
 + adding selected modules
 + using builtin Expat
 + checking sizeof various data types
 + doing sanity check on compiler and options
```

```
  Creating Makefile in src/support
  Creating Makefile in src/regex
  Creating Makefile in src/os/unix
  Creating Makefile in src/ap
  Creating Makefile in src/main
  Creating Makefile in src/lib/expat-lite
Creating Makefile in src/modules/standard
```

Następnie wydajemy polecenia:

make

make install

wynikiem poprawnego zakonczenia opercaji instalowania serwera Apache'a powinna być następująca plansza:

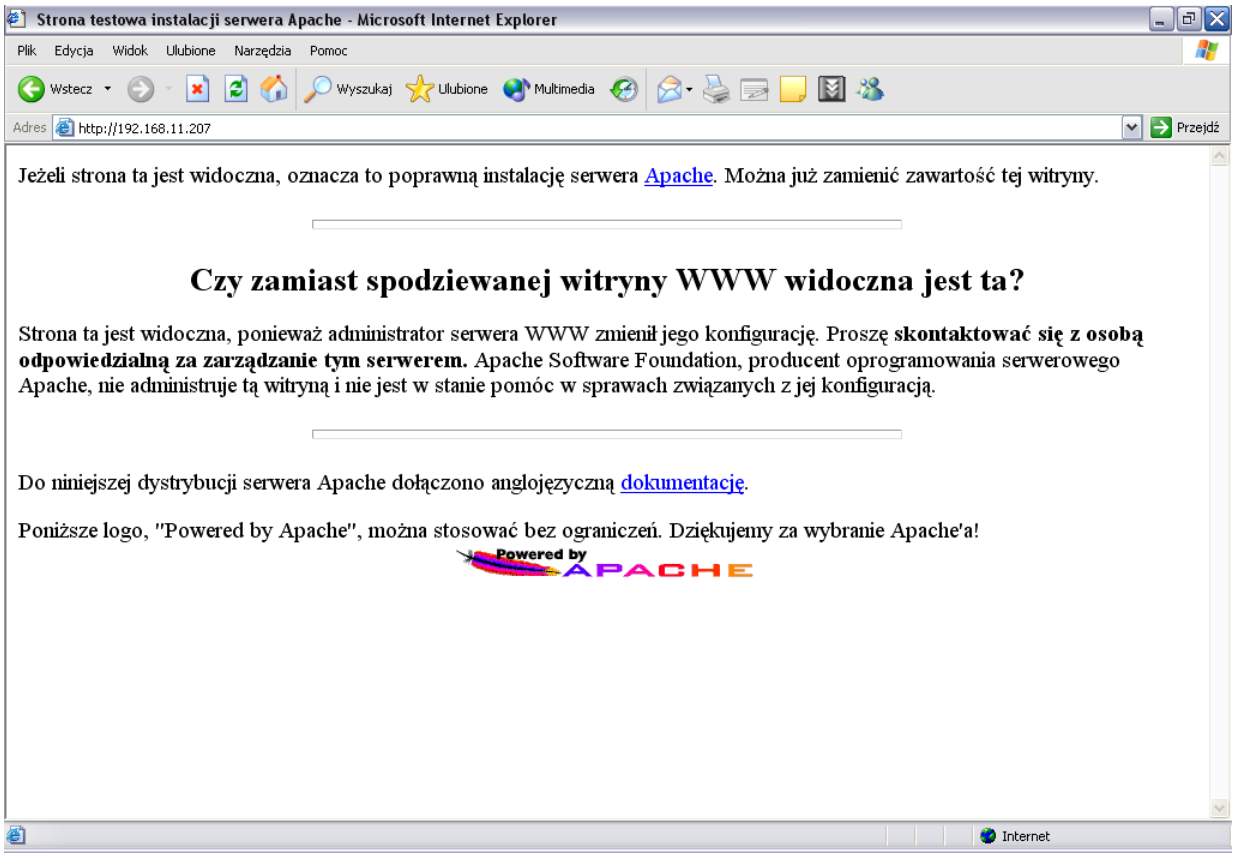
```
  +--------------------------------------------------------+
  | You now have successfully built and installed the      |
  | Apache 1.3 HTTP server. To verify that Apache actually |
  | works correctly you now should first check the         |
  | (initially created or preserved) configuration files   |
  |                                                        |
  |   /usr/local/apache/conf/httpd.conf
  |                                                        |
  | and then you should be able to immediately fire up     |
  | Apache the first time by running:                      |
  |                                                        |
  |   /usr/local/apache/bin/apachectl start
  |                                                        |
  | Thanks for using Apache.       The Apache Group        |
  |                                http://www.apache.org/  |
  +--------------------------------------------------------+
```

2.5 Uruchamianie

W tym momencie możemy uruchomić Apache'a poleceniem

```
/usr/local/apache/bin/apachectl start
```

Aby sprawdzić czy nasz serwer httpd działa poprawnie wpisujemy w Internet Explorerze następującą linię: http://adres_ip_naszego_serwera
Po wykonaniu tego polecenia powininna pojawić się następująca strona:



Apache jest zwykle uruchamiany jako demon `httpd`, który kontroluje wszystkie prośby połączeń do serwera na port 80. Jest to możliwe dzięki wywoływaniu Apache'a poprzez demaona internetowego inetd, za każdym razem gdy przychodzi żądanie połączenia do serwera Apache.

Parametry linii komend

Listę parametrów, które możemy podać podczas uruchamiania httpd uzyskujemy wpisując polecenie ./usr/local/apache/bin/httpd -help lub man httpd

2.6 Konfiguracja

Konfiguracja Apache'a opiera się o 3 pliki:

a) http.conf - główne ustawienia Apache'a
b) access.conf – definiuje ograniczenia dostępu na stronach www
c) srm.conf – określa miejsce przechowywania dokumentów na serwerze i definiuje w jaki sposób przedstawić je użytkownikowi

Pliki te znajdują się w katalogu /usr/local/apache/conf

Dokładny opis dyrektyw używanych w plikach konfiguracyjnych możemy znaleźć na stronie projektu, lub http://www.jtz.org.pl/Inne/Apache/Mod/core.html

2.7 Pliki logów

Przy plikach logów należy zwrócić szczególną uwagę na to, aby katalog w którym je Apache przechowuje nie był dostępny dla każdego. Z tych plików można uzyskać uid przypisany demonowi httpd, jest to o tyle groźne, że demon ten wykorzystuje uid root'a
Pliki przechowywane są w katalogu /usr/local/apache/logs, w katalogu tym oprócz plików logów znajduje się również plik httpd.pid. Plik ten zawiera numer identyfikatora procesu httpd (pid).

Po więcej informacji na temat konfigurowania i użytkowania serwera Apache odsyłam do strony projektu Apache. [9]

3. PAKIET OPENSSL

Pakiet OpenSSL jest niekomercyjną implementacją protokołu SSL (Secure Socket Layer v 2.0 i v 3.0) oraz protokołu TLS (Transport Layer Security). Pakiet ten umożliwia stworzenie agencji uwierzytelniającej (Certificate Authority) i zarządzanie wydawanymi przez nią certyfikatami. Pakiet OpenSSL dostępny jest na rożne platformy systemowe min.: Linux, Unix, Windows.
Protokół SSL, który został zaimplementowany w OpenSSL został stworzony przez firmę Netscape. Służy on do bezpiecznej komunikacji poprzez szyfrowane połączenie. [1]

3.1 Instalacja

Instalowanie pakietu OpenSSL przeprowadzimy ze źródeł, jak wspomnieliśmy przy okazji kompilowania źródeł Apache'a jest to lepsza metoda, dająca większe możliwości, lecz dla początkującego użytkownika może być ona nieco skomplikowana.

Następującymi poleceniami skonfigurujemy i zainstalujemy pakiet OpenSSL

Rozpakowujemy pakiet:

```
tar -zxf openssl-0.9.7.tar.gz
```

Następnie:

```
./config --prefix=/usr/local/openssl
make
make install
```

3.2 Certyfikaty

Posiadanie certyfikatów umożliwia w czasie komunikacji potwierdzenie tożsamości obiektu, z którym chcemy wymieniać dane. Certyfikat taki zawiera dane pozwalające zidentyfikować obiekt, jak i pozwalające stwierdzić autentyczność danych identyfikacyjnych. Każdy certyfikat musi być podpisany przez CA - Certificate Authority, który to przyjmujemy jako jednostkę wiarygodną, podającą prawdziwe dane. [1]

Wystawiony certyfikaty posiada następujące pola:

a) Nazwę certyfikowanego obiektu

b) Identyfikator obiektu

c) Klucz publiczny obiektu

d) Czas ważności

e) Nazwę wystawcy certyfikatu

f) Identyfikator wystawcy

g) Podpis wystawcy - zawiera skrót całego certyfikatu zaszyfrowanego za pomocą klucza prywatnego wystawcy

3.3 Tworzenie CA - Certificate Authority

Tworzone certyfikaty powinniśmy gdzieś umieszczać, w związku z czym stworzymy sobie strukture katalogów zalecaną przez openssl. Tworzymy katalog CA, w którym umieszczamy poszczególne katalogi: [8]

`certs/` - zawiera certyfikat CA
`crl/` - nie wykorzystywany,
`newcerts/` - przechowuje wszystkie wygenerowane certyfikaty,
`private/` - przechowuje klucz prywatny CA

Aby stać się pełnoprawnym CA potrzebujemy klucza prywatnego i certyfikatu. Generujemy je w następujący sposób:

Klucz prywatny:
```
OpenSSL> genrsa -des3 -out ca.key 1024
```

```
Certyfikat:
OpenSSL> req -new -x509 -key ca.key -out ca.crt -days 1024
```

Po wydaniu tego polecenia system spyta o hasło dla klucza prywatnego, klucz ten zostanie zaszyfrowany. Następnie uzupełniamy dane identyfikacyjne tworzonego certyfikatu.

Zostaną stworzone dwa pliki:

a) `ca.crt` - zawierający certyfikat CA

b) `ca.key` - zawierający zaszyfrowany klucz prywatny CA

3.4 Tworzenie certyfikatów dla obiektów

Po wygenerowaniu certyfikatu dla CA możemy przystąpić do wydawania certyfikatów dla innych obiektów.

Certyfikat tworzymy rozpoczynając od stworzenia pliku w określonym formacie, który zawiera dane komputera jak i dane klucza prywatnego dla niego. Pamiętajmy o zapamiętaniu haseł, które podajemy przy generowaniu kluczy. [5]

Stworzymy dla naszego Apache'a, klucz prywatny oraz certyfikat:

```
OpenSSL> genrsa -des3 -out server.key 1024
```

Gdy mamy już klucz prywatny, pozostało wygenerować .csr („prośba" o wydanie certyfikatu)

```
OpenSSL> req -new -key server.key -out server.csr
```

Na zapytania dotyczące danych komputera trzeba rzetelnie odpowiadać, ponieważ dane te będę dostępne dla każdego kto będzie chciał sprawdzić autentyczność tego komputera.
Pole Common Name powinno składać się z nazwy serwera i nazwy domeny nazwa_serwera.nazwa_domeny, np.: komputer.prz-rzeszow.pl. [12]

Po stworzeniu pliku o rozszerzeniu .csr dla naszego komputera musimy podpisać tę prośbę za pomocą wcześniej wygenerowanego certyfikatu naszgo CA. Najpewniejszą metodą podpisywania certyfikatu jest skorzystanie ze skryptu sign.sh dołączonego do modssl'a. Skrypt ten umieszczony jest w katalogu /mod_ssl-2.8.12-1.3.27/pkg.contrib/. Kopiujemy potrzebne do podpisania plik (ca.crt, ca.key, server.csr, server.key) do katalogu ze skryptem, gdzie następnie wykonujemy skrypt: [12]

```
./sign.sh server.csr
```

Po tych czynnościach otrzymujemy plik server.crt, który wraz z server.key kopiujemy do jakiegoś osobnego katalogu np.: etc/ssl/apache.

W ten oto sposób mamy stworzony certyfikat dla naszego obiektu.

4. PAKIET MODSSL

Pakiet modssl jest modułem dołączanym do Apache'a, dla obsługi SSL.

4.1 Instalacja

Rozpakowujemy moduł modssl:
tar -zxf mod_ssl-2.8.12-1.3.27.tar.gz

Następnie wykonujemy kolejno polecenia:

```
./configure \
      --with-apache=../apache_1.3.27 \
      --with-ssl=../openssl-0.9.7 \
      --with-crt=/CA/newcerts/newcert.pem \
      --with-key=/CA/newcerts/sitekey.pem \
      --prefix=/usr/local/apache \
      --enable-module=ssl
```

5. INTEGRACJA APACHE'A Z SSL

Na koniec jeszcze przekompilowanie Apache'a, aby współpracował z modułem ssl:

```
cd ../apache_1.3.27
SSL_BASE = ../openssl-0.9.6b \
    ./configure \
    --enable-module=ssl \
    --prefix=/usr/local/apache
    --[różne inne opcje]
make
make install
```

Jeszcze parę zmian w pliku httpd.conf, aby SSL działał dla virtualnych hostów: [12]

```
SSLEngine on
SSLCertificateFile sciezka_dostepu/server.crt
SSLCertificateKeyFile sciezka_dostepu/server.key
```

Na koniec musimy uruchomić Apache'a:

```
usr/local/apache/bin/apachectl startssl
```

Sprawdzamy czy SSL działa jakimkolwiek programem skanującym porty np.: nmap'em: `nmap adres_serwera`, powinien być widoczny otwarty port 443 https

Po tej operacji mamy działającego Apache'a z obsługą SSL.

LITERATURA

[1] Daniel Rychcik „SSL – Teoria i zasada działania” 2000

[2] Henryk Liniowski „Apache i SSL” Poznań 2001

[3] Craig Hunt „Serwery Sieciowe Linuxa” Mikom Warszawa 2000

[4] Mario Camou, John Goerzen, Aaron Van Couweberghe „Debian Linux” Helion Gliwice 2001

[5] Strona internetowa projektu JTZ (Jak To Zrobić) http://www.jtz.org.pl

[6] Strona internetowa projektu Debian GNU/Linux http://www.debian.org

[7] Nick DeClario „Building a secure web server using Apache and OpenSSL” 9/19/2000 Tłumaczenie: Robert Sobieski „Budowanie zabezpieczonego web serwera przy użyciu Apache i OpenSSL”

[8] Michał Żurak Praca Magisterska „Bezpieczne kanały komunikacyjne w oparciu o szyfrowanie pakietów IP - modyfikacja jądra systemu Linux” Wrocław 2000

[9] Strona internetowa projektu Apache http://www.apache.org

[10] Strona internetowa projektu Modssl http://www.modssl.org

[11] Strona internetowa projektu OpenSSL http://www.openssl.org

[12] Zasoby strony internetowej http://www.linuxindex.pl