Fedora 7. Księga eksperta

Autor: Andrew Hudson, Paul Hudson
Tłumaczenie: Przemysław Szeremiota
ISBN: 978-83-246-1480-6
Tytuł oryginału: Fedora 7 Unleashed
Format: 172x245, stron: 1088

Kompleksowe omówienie jednej z najpopularniejszych dystrybucji Linuksa

- Jak zainstalować i skonfigurować Fedorę?
- W jaki sposób korzystać z konsoli tekstowej i środowiska graficznego?
- Jakie aplikacje dołączono do dystrybucji Fedory?
- Jak uruchomić serwer internetowy w oparciu o Fedorę?

Fedora to rozwijana przez firmę Red Hat dostępna bezpłatnie dystrybucja systemu Linux. W odpowiedzi na ogromną popularność dystrybucji Red Hat Linux jej producent zdecydował się na opublikowanie dwóch wersji – komercyjnej i nieodpłatnej. Fedora jest więc następcą znanego od dawna Red Hat Linuksa. O możliwościach, stabilności i wszechstronności Linuksa napisano już setki artykułów i książek. Chyba nawet jego twórca nie spodziewał się, że napisany przez niego w ramach studenckich ćwiczeń system operacyjny zrewolucjonizuje współczesną informatykę, a jego wersje rozwojowe będą wykorzystywane nawet przez ogromne korporacje.

Książka „Fedora 7. Księga eksperta" to kompleksowe opracowanie poświęcone najnowszej wersji dystrybucji Fedora. Czytając ją, dowiesz się, jak zainstalować Fedorę, skonfigurować ją i uruchomić. Poznasz graficzny interfejs użytkownika i konsolę tekstową, nauczysz się korzystać z oprogramowania dołączonego do dystrybucji i łączyć z internetem. Opanujesz kwestie połączeń sieciowych i zadania administratora systemu. Przeczytasz także o uruchamianiu w Fedorze usług sieciowych, programowaniu w Linuksie i optymalizacji wydajności systemu.

- Instalacja Fedory
- Środowiska graficzne GNOME i KDE
- Praca z wierszem poleceń
- Korzystanie z poczty elektronicznej i WWW
- Pakiet OpenOffice.org
- Gry i multimedia
- Automatyzacja pracy
- Zarządzanie kontami użytkowników
- Tworzenie kopii zapasowych
- Konfiguracja sieci
- Uruchamianie serwera WWW, FTP i poczty elektronicznej
- Administrowanie bazami danych MySQL i PostgreSQL
- Programowanie w Linuksie – Perl, Python, PHP i Mono
- Ochrona systemu przed atakami z sieci
- Strojenie wydajności

Poznaj Fedorę i korzystaj z niesamowitych możliwości systemów linuksowych

Wydawnictwo Helion
ul. Kościuszki 1c
44-100 Gliwice
tel. 032 230 98 63
e-mail: helion@helion.pl

Spis treści

Rozdział 17.
Zarządzanie serwerem WWW Apache

Bieżący rozdział poświęcony jest konfiguracji i zarządzaniu serwerem WWW Apache. Omówienie obejmować będzie m.in. przegląd podstawowych elementów serwera wraz z prezentacją sposobów jego konfiguracji, również za pośrednictwem interfejsów graficznych. Dowiesz się, jak uruchomić serwer, jak go zatrzymać i przeładować, korzystając z narzędzi dostępnych w dystrybucji Fedora. Wszystko to poprzedzone będzie prezentacją jednego z najpopularniejszych na świece serwerów WWW.

Serwer WWW Apache

Apache to serwer WWW najszerzej wykorzystywany we współczesnym internecie (jeśli wierzyć statystykom publikowanym przez Netcraft, zamieszczonym częściowo w tabeli 17.1).

Tabela 17.1. *Wyniki ankiety Netcraft (dane z września 2007 roku)*[1]

Serwer WWW	Liczba witryn	Udział w rynku (procentowy)
Apache	68 228 561	50,48
Microsoft[2]	47 232 300	34,94
Google	6 616 713	4,90
SunONE	2 212 821	1,64
lighttpd	1 515 963	1,12

Trzeba zaznaczyć, że te statystyki nie obejmują serwerów Apache działających w izolowanych sieciach lokalnych, czyli w **intranecie**.

Nazwa ***Apache*** pojawiła się na wczesnych etapach rozwoju oprogramowania serwera; serwer był wtedy zbiorem łat aplikowanych do kodu źródłowego serwera NCSA (więc był to **łatany** serwer, ang. *„a patchy" server*). Przez jakiś czas po zarzuceniu projektu NCSA wielu programistów tworzyło jeszcze udoskonalające serwer łaty, nie tylko eliminujące wykryte błędy, ale również uzupełniające serwer o nowe funkcje. Tworzony w ten sposób kod programiści wymieniali między sobą w sposób kompletnie niezorganizowany.

Nie trzeba było długo czekać, aby Bob Behlendorf i Cliff Skolnick udostępnili scentralizowane repozytorium łat serwera NCSA — tak narodził się projekt Apache. Dziś rdzeń projektu stanowi stosunkowo niewielka grupa programistów, choć mile widziane są również propozycje i łaty pochodzące z zewnątrz.

[1] Ankieta objęła 135 166 473 witryn.

[2] Łącznie wszystkie wersje serwerów tej firmy.

W ciągu ostatnich kilku lat serwer Apache zyskał wielkie znaczenie komercyjne, co po części tylko wynika z popularnego modelu stosowania oprogramowania *open-source* w korporacyjnych systemach informacyjnych. Równie ważną przyczyną jest niechęć do serwera **IIS** (ang. *Internet Information Server*) firmy Microsoft, zawierającego wiele luk (wciąż odkrywane są nowe) i podatnego na szereg ataków, jak również do implementacji systemu operacyjnego Windows i jego obsługi sieci, co razem tworzy środowisko rozprzestrzeniania się robaków, takich jak Code Red, Nimda czy Blaster. Jedną z pierwszych firm, które doceniły stabilność serwera WWW, była firma IBM, aktywnie wspomagająca projekt, a w zamian korzystająca ze stabilnego i uznanego serwera WWW.

W połowie lat 90. ubiegłego wieku powstała (jako organizacja typu non profit) firma Apache Software Foundation. Zarządza nią wybierana corocznie spośród członków ASF rada dyrektorska. Firma stanowi matecznik wielu różnych otwartych projektów programowych, z serwerem Apache na czele.

Najlepszym źródłem informacji o serwerze Apache jest strona główna przedsięwzięcia Apache Software Foundation pod adresem *http://www.apache.org/* oraz witryna biuletynu Apache Week (*http://www.apacheweek.com/*); ta druga prowadzi usługę powiadamiania subskrybentów o nowych elementach projektu serwera, pozwalającą na otrzymywanie najnowszych informacji o niebezpiecznych lukach i dostępnych poprawkach.

Wskazówka

Dobrym źródłem informacji o serwerze Apache jest dokument FAQ, dostępny pod adresem *http://httpd.apache.org/docs-2.2/faq/*. Obok dokumentacji dostępnej online użytkownicy serwera mają też do dyspozycji dokumentację w formacie HTML, instalowaną w katalogu serwera Apache (z pakietu `httpd-manual`). Można się do niej odwołać za pośrednictwem samego serwera, nawiązując połączenie z serwerem lokalnym i przechodząc do strony *http://localhost/manual/index.html*. Ale lektura tak udostępnianej dokumentacji będzie możliwa dopiero po uruchomieniu serwera!

Dystrybucja Fedora zawiera wersję 2.2 serwera Apache; pakiet serwera, znajdujący się na płycie instalacyjnej, nosi nazwę `httpd`. Najnowszą wersję oprogramowania serwera w postaci pakietu RPM można pobrać z serwera FTP projektu Fedora, dostępnego za pośrednictwem programu `yum`. Można też pobrać ze strony serwera Apache najnowszą wersję kodu źródłowego i — zgodnie z tradycją linuksową — samodzielnie ją skompilować i zainstalować w systemie.

Aby określić, która z wersji serwera Apache została zainstalowana w systemie, należy uruchomić plik wykonywalny serwera z przełącznikiem `-V`:

```
# /usr/sbin/httpd -V | cat
Server version: Apache/2.2.6 (Unix)
Server built:   Sep 18 2007 09:40:44
Server's Module Magic Number: 20051115:5
Server loaded:  APR 1.2.8, APR-Util 1.2.8
Compiled using: APR 1.2.8, APR-Util 1.2.8
Architecture:   32-bit
Server MPM:     Prefork
  threaded:     no
    forked:     yes (variable process count)
Server compiled with....
...
```

Wynikiem polecenia jest wydruk obejmujący numer wersji serwera, datę i czas kompilacji oraz listę opcji wystosowanych przy kompilacji. Te same informacje, ale w wersji nieco okrojonej, można uzyskać po umieszczeniu w wierszu wywołania serwera opcji `-v`.

Instalowanie serwera Apache

Oprogramowanie serwera Apache można zainstalować z pakietu RPM bądź wykonać kompilację jego kodu źródłowego. Kod źródłowy powinien dać się skompilować w dowolnym systemie uniksowym, jak również w środowisku Win32. Jeżeli przy instalacji dystrybucji Fedora zaznaczono grupę pakietów *Serwer WWW*, w systemie automatycznie zainstalowany zostanie zarówno sam serwer Apache, jak i oprogramowanie z nim związane, w postaci kilkunastu dodatkowych pakietów (z dokumentacją włącznie).

Przystępując do instalacji nowszej wersji serwera, należy pamiętać o konieczności zatrzymania serwera działającego w tle. Co prawda, prawdopodobieństwo, że działanie serwera zakłóci proces instalacji, jest niewielkie, to nie zaszkodzi zmniejszyć je do zera. Procedura zatrzymywania serwera opisana została w podrozdziale „Uruchamianie i zatrzymywanie serwera Apache".

Instalacja serwera przy użyciu programu pirut

Co prawda, kategoria Serwer WWW jest dostępna już przy pierwotnej instalacji dystrybucji, ale często na tym etapie lepiej powstrzymać się od instalacji jakichkolwiek nadmiarowych usług, żeby po prostu ograniczyć liczbę furtek do systemu, które przed szczegółowym przygotowaniem konfiguracji trzeba uznać za potencjalnie otwarte. Jednak już po instalacji i wstępnej konfiguracji systemu Fedora można zainstalować serwer Apache z poziomu systemowej aplikacji dodawania i usuwania oprogramowania; w programie pirut (*Aplikacje*/*Dodaj/usuń oprogramowanie*) wystarczy w grupie Serwery zaznaczyć kategorię *Serwer WWW*.

Warto także zajrzeć do szczegółowej listy pakietów (zakładka *Lista*) i wybrać z niej poszczególne moduły serwera — łatwo je rozpoznać, bo wszystkie zaczynają się na „`mod_`". W dystrybucji Fedora znajduje się mnóstwo takich modułów, ale domyślnie aktywowane są tylko niektóre — resztę możesz zainstalować wybiórczo na własną rękę, również w programie pirut.

Ręczna instalacja serwera z pakietów RPM

Pakiet RPM z oprogramowaniem serwera Apache można znaleźć na jednej z płyt instalacyjnych dystrybucji Fedora lub w repozytoriach serwera FTP projektu Fedora i wielu serwerów lustrzanych. Najnowsze wersje pakietu można pobierać spod adresu *http://download.fedora.redhat.com/*
↪pub/fedora/linux/updates/7/; warto tam zaglądać jak najczęściej, aby zawsze dysponować najnowszą wersją serwera. Aktualizowane pakiety RPM zawierają zwykle ważne poprawki, usuwające znalezione błędy i eliminujące luki w zabezpieczeniach. Dlatego warto jak najszybciej instalować pojawiające się uaktualnienia — serwer będzie odporniejszy na ataki.

Uwaga

**Raportów o bezpieczeństwie serwera należy szukać w witrynie projektu Apache. Trzeba w tym celu przejść do strony *http:// httpd.*
↪apache.org/security_report.html, gdzie znajdują się odnośniki do stron podsumowujących wykryte luki w oprogramowaniu serwera, osobno dla wersji 1.3, 2.0 i 2.2. Warto też przejrzeć archiwum artykułów rozsyłanych przez listy dystrybucyjne witryny serwera (*http://httpd.apache.org/mail/*). Interfejs WWW do tego archiwum znajduje się pod adresem *http://httpd.apache.org/lists.html*.**

**Wszyscy ciekawi działania najnowszych wersji pakietów mogą z serwera Fedory pobierać pakiety w wersji eksperymentalnej (*http://*
↪download.fedora.redhat.com/pub/fedora/linux/updates/testing/7/
oraz *http://download.fedora.redhat.com/pub/fedora/linux/development/*). To dystrybucja rozwojowa, testowana jako baza dla kolejnego wydania dystrybucji Fedora. W przypadku serwera Apache instalacja takiego eksperymentalnego pakietu może być jednak uzależniona od zainstalowania sporej ilości pakietów dodatkowych, wymaganych do poprawnej instalacji serwera. W takim przypadku lepszym wyjściem może być pobranie pakietu zawierającego kod źródłowy serwera (z podkatalogu *SRPMS*) i wykonanie samodzielnej kompilacji. W ten sposób eliminuje się zależności pomiędzy pakietami.**

Ostrzeżenie

Należy unikać instalowania pakietów nieprzetestowanych (eksperymentalnych) na serwerach produkcyjnych (to jest stanowiących podstawę działalności organizacji). Pakiety takie lepiej instalować i testować na wydzielonych z części produkcyjnej węzłach systemu, najlepiej odłączonych od sieci komputerowej!

Po pobraniu pakietu RPM zawierającego oprogramowanie serwera Apache można pakiet ów zainstalować, korzystając z polecenia `rpm`:

```
# rpm -UvH httpd-nr_wersji.rpm
```

gdzie *`nr_wersji`* to numer pobranej wersji pakietu.

Do ręcznej instalacji najprościej chyba jednak wykorzystać program `yum` i jego opcję instalacji najnowszej wersji:

```
# yum update httpd
```

Pakiety RPM serwera Apache zawierają pliki instalowane w następujących katalogach:

- */etc/httpd/conf/* — katalog zawierający pliki konfiguracyjne serwera Apache, w tym plik główny, *httpd.conf*. Zawartości tego pliku poświęcony będzie punkt „Konfigurowanie serwera pod kątem wydajności szczytowej" zamieszczony w dalszej części rozdziału.
- */etc/rc.d/* — podkatalogi tego katalogu zawierają skrypty uruchamiane w ramach rozruchu systemu. Pakiet serwera Apache zawiera instalowany w podkatalogu */etc/rc.d/init.d/* skrypt startowy o nazwie *httpd*, uruchamiający usługę serwera WWW. Skrypt ten wykorzystuje się do ręcznego uruchamiania i zatrzymywania serwera z poziomu wiersza polecenia. System wykorzystuje go zaś do automatycznego uruchomienia usługi serwera WWW podczas rozruchu systemu.
- */var/www/* — tutaj instalowane są domyślnie ikony i rysunki, programy *CGI* (ang. *Common Gateway Interface*) oraz pliki HTML udostępniane przez serwer. Jeżeli użytkownik zechce przechowywać udostępniane treści w innym katalogu, powinien wprowadzić stosowne zmiany w konfiguracji serwera.
- */var/www/manual/* — w katalogu tym znajduje się kopia dokumentacji serwera WWW w formacie HTML (instalowana z pakietu `httpd-manual`). Dokumentację tę można przeglądać za pośrednictwem samego serwera, przechodząc na stronę *http://localhost/manual/*.
- */usr/share/man/* — rozprowadzany w dystrybucji Fedora pakiet RPM serwera Apache zawiera również dokumentację w formacie podręcznika systemowego `man`. Strona podręcznika dla polecenia `httpd` znajduje się np. w sekcji `8` podręcznika.
- */usr/sbin/* — w tym katalogu umieszczane są pliki wykonywalne serwera. Chodzi zarówno o plik wykonywalny programu serwera, jak i pliki programów pomocniczych.
- */usr/bin/* — tu instalowane są niektóre z narzędzi wchodzących w skład pakietu serwera; jednym z takich narzędzi jest program `htpasswd`, służący do generowania plików haseł serwera.
- */var/log/httpd/* — katalog plików dziennika serwera Apache. Domyślnie właśnie w tym katalogu tworzone są dwa najważniejsze dla administratora serwera pliki: *access_log* oraz *error_log*. Zestaw plików dziennika można uzupełnić o pliki, w których umieszczane są dodatkowe informacje o działaniu serwera. Dodatkowe informacje znajdują się w podrozdziale „Rejestrowanie".

- */usr/src/redhat/SOURCES/* — katalog, w którym może znajdować się archiwum programu `tar` zawierające kod źródłowy serwera Apache wraz z ewentualnymi łatami. Katalog tworzony jest po zainstalowaniu pakietu kodu źródłowego (SRPM) serwera Apache.

Uruchomiony serwer Apache tworzy też w podkatalogu */var/run/* plik *httpd.pid*, zawierający identyfikator procesu serwera Apache.

Uwaga

Przy aktualizacji serwera polegającej na instalacji nowszej wersji pakietu RPM nie należy obawiać się nadpisania plików konfiguracyjnych bieżącej wersji. Pliki te są przez program instalacyjny zachowywane z nazwami rozszerzonymi przyrostkiem *.rpmnew* (np. plik *httpd.conf* otrzyma nazwę *httpd.conf.rpmnew*).

Samodzielna kompilacja kodu źródłowego serwera

Kod źródłowy serwera Apache można uzyskać na kilka sposobów. W ramach dystrybucji Fedora dostępne są pakiety SRPM zawierające kod źródłowy Apache wraz z pakietami dostosowującymi ten kod do specyfiki dystrybucji. Najnowsze, stabilne wersje kodu źródłowego serwera Apache dla dystrybucji Fedora można więc pobrać z repozytorium dostępnego w witrynie *http://fedora.redhat.com* po kliknięciu odnośnika *Download*. Po zainstalowaniu jednego z takich pakietów w podkatalogu */usr/src/redhat/SOURCES/* pojawi się archiwum programu `tar` zawierające kod źródłowy serwera oraz zestaw łat.

Kod źródłowy można też pobrać bezpośrednio ze strony *http://www.apache.org/*. W czasie przygotowywania tego wydania najnowsza dostępna w ten sposób wersja oprogramowania serwera nosiła numer 2.2.6 i zajmowała ok. 6 MB (spakowana jako *.tar.gz*; w archiwum *.tar.bz2* zajmuje około 4,5 MB). W gałęzi 1.3 najnowszą wersją była wersja 1.3.39 (2,4 MB), a gałąź 2.0 doczekała się wydania 2.0.61 (5,8 MB). Co prawda, wiele witryn, głównie ze względu na zgodność oprogramowania, wciąż używa starszych wersji serwera, jednak wiele stron obsługiwanych jest przez wersje najnowsze.

Po zdobyciu pliku archiwum programu `tar` należy rozpakować go do dowolnie wybranego katalogu tymczasowego, np. */tmp*. Po rozpakowaniu archiwum w katalogu tymczasowym pojawi się katalog o nazwie *httpd-nr_wersji* (bądź *apache_nr_wersji* w przypadku wersji 1.3), np. *httpd-2.2.6*.

Wskazówka

Po rozpakowaniu archiwum kodu źródłowego powstaje katalog zawierający kod źródłowy, a w nim zwyczajowo już znajdują się pliki *README* oraz *INSTALL*. Warto zapoznać się z treścią tych plików przed próbą kompilacji i instalacji oprogramowania.

Konfiguracja kodu źródłowego za pomocą skryptu configure

Aby ułatwić sobie proces kompilacji kodu źródłowego serwera Apache, można skorzystać ze znajdującego się w katalogu głównym kodu źródłowego skryptu `configure`. Po określeniu w wierszu wywołania skryptu opcji `--prefix` można określić katalog docelowy instalacji; domyślne jest to katalog */usr/local/apache/*:

```
# ./configure --prefix=/sciezka_dostepu_do_katalogu
```

Wynikiem działania skryptu będzie plik *Makefile*, który można wykorzystać do kompilacji kodu serwera.

Kolejnym krokiem jest wpisanie polecenia **`make`**, inicjującego kompilację. Po zakończeniu kompilacji należy — dysponując uprawnieniami użytkownika *root* — wpisać polecenie **`make install`**. Spowoduje to zainstalowanie serwera w systemie. Po zakończeniu instalacji można już przystąpić do edycji plików konfiguracyjnych, patrz podrozdział „Ustawienia konfiguracyjne serwera".

Wskazówka

Bezpieczniejszą metodą instalacji nowej wersji oprogramowania serwera Apache jest skorzystanie z polecenia `ln` i utworzenie za jego pomocą dowiązań symbolicznych (wymienionych w punkcie „Ręczna instalacja serwera z pakietów RPM") do nowo skompilowanych plików. Chodzi o to, że domyślne katalogi instalacyjne przy samodzielnej kompilacji różnią się od katalogów zdefiniowanych w pakiecie RPM. Rozbieżność w położeniu katalogów instalacji może zaś uniemożliwić automatyczne uruchomienie serwera za pośrednictwem skryptu startowego.

Warto też przed zainstalowaniem nowej wersji oprogramowania serwera wykonać kopie zapasowe wszystkich ważniejszych plików konfiguracyjnych (np. całego katalogu */etc/httpd*), a następnie skorzystać z polecenia `rpm` w celu usunięcia z systemu oprogramowania serwera instalowanego z pakietu RPM. Wtedy można bezpiecznie zainstalować i przetestować nową wersję, a w razie konieczności w prosty sposób przywrócić poprzedni stan systemu i poprzednie ustawienia konfiguracji serwera.

Przede wszystkim zaś warto korzystać z pakietów (czy to plików skompilowanych, czy też pakietów kodu źródłowego) RPM Warto pamiętać, że przy samodzielnej kompilacji i instalacji kodu źródłowego nie istnieje opcja usunięcia instalacji serwera z poziomu narzędzi zarządzających pakietami!

Jak szybko uruchomić serwer Apache?

W dystrybucji Fedora można skonfigurować i uruchomić serwer Apache, a następnie przetestować jego działanie, wykonując tylko dwie czynności. Najpierw jednak trzeba upewnić się, czy serwer jest zainstalowany. Można zaznaczyć serwer do instalacji podczas instalowania systemu albo później zainstalować serwer z odpowiedniego pakietu RPM (w tym ostatnim przypadku warto zajrzeć do rozdziału 7., w którym opisaliśmy m.in. sposób korzystania z polecenia `rpm`).

Następnie należy z uprawnieniami użytkownika *root* utworzyć własną stronę domową, umieszczając plik *index.html* w podkatalogu */var/www/html/*. Wcześniej warto wykonać kopię zapasową pierwotnej strony (najlepiej całego katalogu *www*), aby można było przywrócić poprzedni stan systemu, kiedy okaże się to konieczne.

Dalej wystarczy uruchomić serwer (z uprawnieniami użytkownika uprzywilejowanego), korzystając z następującego polecenia:

```
# service httpd start
```

Można też skorzystać ze skryptu startowego serwera, umieszczonego w katalogu */etc/rc.d/init.d*:

```
# /etc/rc.d/init.d/httpd start
```

Następnie można już uruchomić dowolną przeglądarkę WWW i sprawdzić, czy pod adresem *localhost* (albo innym, zgodnym z nazwą węzła danego komputera) albo adresem IP danego komputera dostępna jest utworzona wcześniej strona HTML. Dla przykładu, zakładając wykorzystanie przeglądarki `links`, użytkownik powinien wykonać następujące polecenie:

```
# links http://localhost/
```

Ze względów bezpieczeństwa nie należy uruchamiać serwera w imieniu użytkownika *root*, jeśli komputer jest podłączony do internetu albo choćby do sieci wewnętrznej (intranet) danej organizacji. Na szczęście, Apache działa, niezależnie od sposobu uruchomienia, w imieniu użytkownika i grupy `apache` (definiuje to ustawienie parametru `User` oraz parametru `Group` pliku konfiguracyjnego serwera). Mimo tych środków ostrożności, zarządzanie i uruchamianie serwera powinno odbywać się w imieniu użytkownika `apache`, zdefiniowanego w pliku */etc/passwd* następującym wpisem:

```
apache:x:48:48:Apache:/var/www:/sbin/nologin
```

Jeśli w oknie przeglądarki pojawiła się odpowiednia strona, można uaktywnić na stałe usługę Apache, korzystając z narzędzia `setup` bądź programu `ntsysv`. W obu tych programach usługa serwera Apache występuje jako `httpd`.

Położenie plików serwera Apache po skompilowaniu i zainstalowaniu

Pliki serwera, kiedy jest instalowany z pakietu kodu źródłowego, umieszczane są w rozmaitych podkatalogach domyślnego katalogu instalacyjnego (*/usr/local/apache*) bądź dowolnego katalogu wskazanego przez opcję `--prefix`. W wersjach poprzedzających 1.3.4 pliki były domyślnie rozmieszczane w katalogu */usr/local/etc/httpd*.

Oto lista podkatalogów (wraz z krótkim opisem zawartości) katalogu instalacyjnego serwera Apache kompilowanego samodzielnie z kodu źródłowego.

- */usr/local/apache/conf/* — katalog zawierający różne pliki, a wśród nich plik konfiguracyjny serwera Apache, *httpd.conf*. O plikach konfiguracyjnych powiemy więcej w podrozdziale „Edycja pliku httpd.conf".
- */usr/local/apache/* — zawiera m.in. podkatalogi *cgi-bin*, *icons* oraz *htdocs* zawierające z kolei odpowiednio: programy CGI, standardowe ikony oraz domyślnie instalowane dokumenty HTML.
- */usr/local/apache/bin/* — katalog plików wykonywalnych serwera.
- */usr/local/apache/logs/* — katalog plików dziennika. Domyślnie serwer tworzy dwa pliki dziennika: `access_log` oraz `error_log`, ale administrator może zdefiniować dowolną liczbę własnych plików dziennika, zawierających rozmaite informacje (patrz podrozdział dotyczący rejestrowania działania serwera). Domyślna lokalizacja plików dziennika serwera Apache w dystrybucji Fedora to */var/log/httpd*.

Uruchamianie i zatrzymywanie serwera Apache

Zakładamy w tym momencie, że w systemie zainstalowany jest serwer Apache, dysponujący domyślną konfiguracją. Serwer Apache powinien być uruchamiany w ramach rozruchu systemu, jako jedna z usług systemowych, uruchamiana po zakończeniu konfigurowania sieci i ewentualnych zapór sieciowych. Sposób konfigurowania rozruchu systemu opisany został w rozdziale 11., „Automatyzacja pracy".

Czas na pierwsze uruchomienie serwera. W poniższych punktach pokazany jest zarówno sposób ręcznego uruchomienia i zatrzymania serwera Apache, jak i sposób skonfigurowania systemu Fedora do automatycznego uruchamiania usługi serwera w fazie rozruchu systemu.

Ręczne uruchamianie serwera Apache

Serwer Apache może zostać uruchomiony ręcznie, poleceniem wykonanym z poziomu konsoli lub okna terminala i wymagającym dysponowania uprawnieniami użytkownika *root*. Program serwera, o nazwie `httpd`, rozpoznaje szereg opcji wywołania, za których pośrednictwem można

ustawiać wartości niektórych parametrów, np. określić lokalizację pliku konfiguracji. Plik wykonywalny serwera Apache rozpoznaje również opcje innego rodzaju, umożliwiające selektywne interpretowanie pliku konfiguracyjnego, np. przez wskazanie innego niż zdefiniowany w konfiguracji pliku dziennika. Opcja `-v` inicjuje wydruk wersji programu. Podobna opcja, `-V`, pokazuje wydruk rozszerzony, uwzględniający opcje, z jakimi skompilowany został plik wykonywalny serwera.

Sposób uruchamiania serwera i lista dostępnych opcji wyświetlana jest po uruchomieniu serwera z opcją `-h` (przy założeniu, że polecenie uruchamiane jest przez użytkownika *root*):

```
# httpd -h
Usage: httpd [-D name] [-d directory] [-f file]
             [-C "directive"] [-c "directive"]
             [-k start|restart|graceful|graceful-stop|stop]
             [-v] [-V] [-h] [-l] [-L] [-t] [-S]
  Options:
  -D name            : define a name for use in <IfDefine name> directives
  -d directory       : specify an alternate initial ServerRoot
  -f file            : specify an alternate ServerConfigFile
  -C "directive"     : process directive before reading config files
  -c "directive"     : process directive after reading config files
  -e level           : show startup errors of level (see LogLevel)
  -E file            : log startup errors to file
  -v                 : show version number
  -V                 : show compile settings
  -h                 : list available command line options (this page)
  -l                 : list compiled in modules
  -L                 : list available configuration directives
  -t -D DUMP_VHOSTS  : show parsed settings (currently only vhost settings)
  -S                 : a synonym for -t -D DUMP_VHOSTS
  -t -D DUMP_MODULES : show all loaded modules
  -M                 : a synonym for -t -D DUMP_MODULES
  -t                 : run syntax check for config files
```

Wśród pozostałych opcji znajduje się opcja wyświetlania **modułów**, czyli wydzielonych elementów serwera. Opcje te zwane są **dyrektywami konfiguracyjnymi** i sterują działaniem poszczególnych modułów. Serwer Apache dysponuje około pięćdziesięcioma **modułami dynamicznymi**, czyli obiektami programowymi ładowanymi przez serwer wtedy, kiedy potrzebny jest zawarty w nich kod obsługujący pewną funkcję serwera.

Niezwykle przydatna opcja `-t` pozwala na sprawdzenie poprawności składniowej pliku konfiguracyjnego serwera. Zawsze po wprowadzeniu zmian w konfiguracji warto sprawdzić, czy nie naruszyły one poprawności pliku. Kontrola taka jest tym ważniejsza, że błąd pliku konfiguracyjnego może spowodować załamanie serwera lub jego błędne działanie.

Uwaga

Przy kompilacji i instalacji Apache z kodu źródłowego (a nie z rozprowadzanych z dystrybucją Fedora pakietów RPM) należy uruchamiać serwer ręcznie, z poziomu wiersza poleceń, korzystając z uprawnień użytkownika *root* (jak przy testowaniu serwera). Ma to dwie przyczyny:

- **Serwer korzysta domyślnie z portu o numerze 80 (domyślny port usługi HTTP), a tylko użytkownik uprzywilejowany może uruchamiać aplikacje okupujące porty o numerach mniejszych od 1024.**
- **Jedynie procesy będące własnością użytkownika *root* mogą zmieniać własne identyfikatory UID i GID (zaś serwer Apache po uruchomieniu i zajęciu portu nasłuchu zmienia te identyfikatory na określone w pliku konfiguracyjnym serwera). Kiedy Apache zostanie uruchomiony przez innego użytkownika, serwer nie będzie mógł odrzucić uprawnień tego użytkownika przez zmianę identyfikatora UID.**

Należy też zauważyć, że choć większość prezentowanych dalej przykładów pokazuje uruchamianie serwera Apache w imieniu użytkownika *root*, to taki tryb uruchamiania należy stosować wyłącznie w fazie testów. Właściwym trybem uruchamiania serwera jest uruchamianie go w ramach rozruchu systemu.

Skrypt startowy /etc/rc.d/init.d/httpd

Dystrybucja Fedora steruje uruchamianiem i zatrzymywaniem usług, w tym usługi serwera Apache za pośrednictwem skryptów znajdujących się w katalogu */etc/rc.d/init.d*. Główny skrypt sterujący działaniem serwera Apache to */etc/rc.d/init.d/httpd*; skrypt ten realizuje swoje zadania za pośrednictwem programu `apachectl`, rozprowadzanego wraz z programem serwera Apache.

Skrypt */etc/rc.d/init.d/httpd* wykorzystuje do sterowania działaniem serwera następujące (m.in.) opcje wywołania:

- `start` — opcja inicjująca uruchomienie serwera w czasie rozruchu systemu. Skrypt z tą opcją może też wykorzystywać użytkownik *root* w celu ręcznego uruchomienia serwera.
- `stop` — opcja powodująca zatrzymanie serwera. Należy korzystać z niej zamiast polecenia `kill`.

Uwaga

Plik *`/etc/rc.d/init.d/httpd`* to skrypt powłoki wykorzystywany do uruchamiania i zatrzymywania serwera w ramach rozruchu i zatrzymywania systemu; nie należy mylić go z plikiem wykonywalnym serwera, instalowanym w katalogu *`/usr/sbin`*. Plik *`/usr/sbin/httpd`* to plik wykonywalny programu serwera (czyli serwer jako taki), a skrypt *`/etc/rc.d/init.d/httpd`* to jedynie skrypt powłoki odwołujący się do programu sterującego `apachectl`. Opis niektórych innych usług uruchamianych za pośrednictwem skryptów katalogu *`/etc/rc.d/init.d`* znajduje się w rozdziale 11.

- `reload` — opcja inicjująca wysłanie do procesu serwera sygnału HUP, co zmusza serwer do ponownego odczytania zawartości plików konfiguracyjnych po ich modyfikacji.
- `restart` — opcja ta jest wygodną metodą zatrzymywania i następnie bezzwłocznego uruchamiania procesu serwera WWW. Jeżeli serwer nie działa, w wyniku tego polecenia zostanie uruchomiony.
- `condrestart` — opcja działająca podobnie jak opcja `restart`, tyle że w przypadku, kiedy serwer akurat nie działa, nie zostanie uruchomiony.
- `status` — skrypt po określeniu tej opcji zwraca ciąg zawierający wykaz identyfikatorów PID działających w systemie procesów serwera.

Aby np. sprawdzić bieżący stan serwera, należy wykonać polecenie:

```
# /etc/rc.d/init.d/httpd status
```

co spowoduje wyświetlenie komunikatu podobnego do poniższego:

```
httpd (pid 16631 16630 16629 16628 16627 16626 16625 16624 16623) jest
↪uruchomiony...
```

Ciąg ten sygnalizuje, że serwer działa; w rzeczy samej, w pamięci znajduje się 9 egzemplarzy procesu serwera.

Poza wymienionymi wcześniej opcjami, skrypt `httpd` rozpoznaje jeszcze cztery:

- `help` — wyświetla listę opcji uruchomienia programu `httpd` (skrypt przekazuje wywołanie do programu `/usr/sbin/httpd`).
- `configtest` — prosty test konfiguracji serwera, ograniczający się do zwrócenia komunikatu `Syntax OK` w przypadku poprawności konfiguracji; identyczny test można wykonać, uruchamiając program serwera z opcją `-t`:

```
# httpd -t
```

- `fullstatus` — wyświetla raport o stanie serwera.
- `graceful` — opcja podobna do parametru `restart`, ale zatrzymanie serwera z tą opcją nie powoduje zerwania bieżąco obsługiwanych połączeń.

Wskazówka

Po wprowadzeniu dowolnych zmian w konfiguracji serwera należy skorzystać z opcji `reload` skryptu. Pozwala to na zaoszczędzenie czasu potrzebnego na zatrzymanie i ponowne uruchomienie serwera.

Sterowanie serwerem Apache — polecenie service

Zamiast wywołania skryptu */etc/rc.d/init.d/httpd* można zastosować polecenie `service` i za jego pośrednictwem uruchamiać, zatrzymywać i przeładowywać usługę Apache. Polecenie `service` wykorzystuje się w połączeniu z nazwą usługi (według nazw skryptów uruchomieniowych umieszczonych w katalogu */etc/rc.d/init.d*) i ewentualnej opcji:

```
# service nazwa_usługi opcja
```

Aby np. przeładować program serwera Apache za pośrednictwem polecenia `service`, należy skonstruować następujące polecenie:

```
# service httpd restart
```

Spowoduje to zatrzymanie i ponowne uruchomienie serwera, jeśli ten był już uruchomiony, bądź samo jego uruchomienie, jeśli wcześniej nie działał.

Sterowanie serwerem Apache — polecenie chkconfig

Polecenie `chkconfig` stanowi dostępny z poziomu wiersza powłoki interfejs skryptów uruchomieniowych usług dystrybucji Fedora. Polecenie może zostać wykorzystane do określania, które z usług będą uruchamiane, przeładowywane i zatrzymywane w określonych stanach systemu (przy rozruchu systemu, jego przeładowywaniu bądź zatrzymywaniu) i poszczególnych poziomach uruchomieniowych, np. w trybie pojedynczego użytkownika (poziom 0), trybie sieciowym (poziom 3.) i trybie graficznym (poziom 5.).

Zatem w celu przejrzenia bieżących ustawień systemu Fedora należy zajrzeć do pliku */etc/inittab* i odszukać w nim określenie domyślnego poziomu uruchomieniowego:

```
# grep id: /etc/inittab
id:3:initdefault:
```

Taki wpis umieszczony w pliku */etc/inittab* informuje, że system ma uruchamiać się w trybie konsoli (bez uruchamiania serwera X), z pełnym dostępem do sieci (poziom uruchomieniowy 3.).

Polecenie `chkconfig` można wykorzystać do podejrzenia zdefiniowanego działania serwera Apache na poszczególnych poziomach uruchomieniowych systemu:

```
# chkconfig --list | grep httpd | sed -e 's/ączone/./g'
httpd           0:wył.   1:wył.   2:wył.   3:wył.   4:wył.   5:wył.   6:wył.
```

Z powyższego wynika, że serwer Apache jest wyłączony na wszystkich poziomach uruchomieniowych, w tym na poziomach 3. i 5. (to główne poziomy uruchomieniowe — innych raczej się nie wykorzystuje; można jeszcze korzystać z poziomu 4., dostosowując go do własnych potrzeb). W celu uaktywnienia rozruchu usługi serwera Apache przy uruchamianiu systemu na 3. poziomie uruchomieniowym należy wykonać następujące polecenie:

```
# chkconfig --level 3 httpd on
```

Do zweryfikowania nowych ustawień można ponownie wykorzystać polecenie `chkconfig`:

```
# chkconfig --list | grep httpd | sed -e 's/ączone/./g'
httpd           0:wył.   1:wył.   2:wył.   3:wł.   4:wył.   5:wył.   6:wył.
```

Aby uaktywnić uruchamianie serwera Apache również wtedy, kiedy system wchodzi na 5. poziom uruchomieniowy, czyli użytkownik loguje się do systemu za pośrednictwem interfejsu graficznego, należy znów skorzystać z polecenia `chkconfig`, opcji `level` i `on`, ale tym razem trzeba określić poziom numer 5:

```
# chkconfig --level 5 httpd on
```

Ponowna weryfikacja stanu usług systemowych da następujące rezultaty:

```
# chkconfig --list | grep httpd | sed -e 's/ączone/./g'
httpd           0:wył.   1:wył.   2:wył.   3:wł.   4:wył.   5:wł.   6:wył.
```

W celu wycofania usługi serwera Apache z danego poziomu uruchomieniowego należy skorzystać z opcji `off` polecenia `chkconfig`.

Graficzny interfejs konfiguracji serwera Apache

Pewne podstawowe aspekty działania serwera Apache można konfigurować za pośrednictwem narzędzia działającego w środowisku graficznym, a noszącego nazwę `system-config-httpd` (instalowanego z pakietu RPM o tej samej nazwie). Narzędzie to stanowi wygodny środek konfiguracji elementów, takich jak nazwa użytkownika i grupy serwera Apache, położenie pliku zawierającego identyfikator procesu serwera czy ustawienia wpływające na wydajność obsługi żądań (np. maksymalna liczba współbieżnych połączeń) — wszystko bez potrzeby ręcznego wprowadzania dyrektyw do pliku konfiguracyjnego.

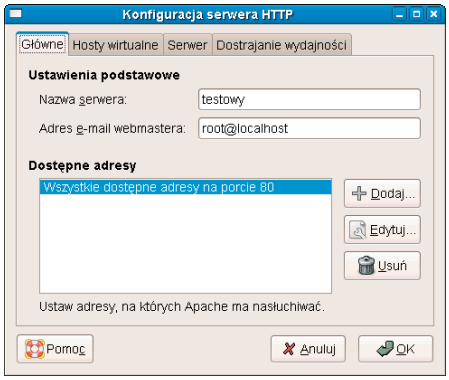
Ostrzeżenie

Korzystając z narzędzia `system-config-httpd`, nie należy podejmować prób ręcznego ingerowania w zawartość pliku konfiguracyjnego *httpd.conf*. Wszelkie zmiany wprowadzone ręcznie zostaną nadpisane przy następnym uruchomieniu graficznego programu konfiguracji.

Program ten uruchamia się, wybierając z menu GNOME pozycje *System/Administracja/Ustawienia serwera/HTTP*; można też uruchomić program ręcznie, wpisując w oknie terminala polecenie:

```
$ system-config-httpd &
```

Po naciśnięciu klawisza *Enter* (bądź wybraniu stosownej pozycji menu) wyświetlony zostanie monit o podanie hasła. Po wprowadzeniu hasła na ekranie wyświetlone zostanie okno główne programu, widoczne na rysunku 17.1.

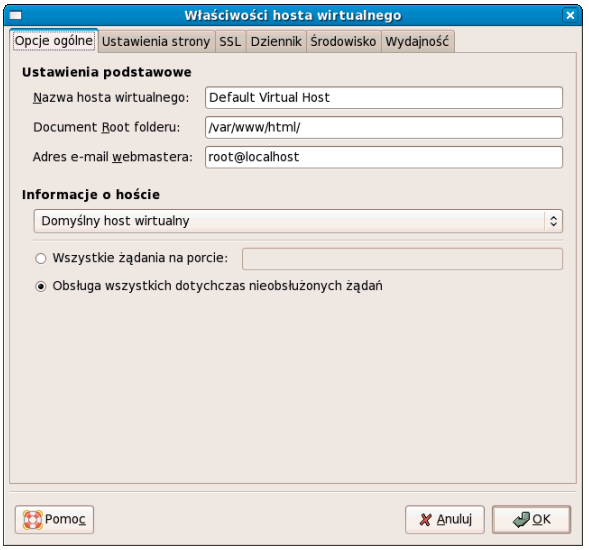


Rysunek 17.1. Główne okno programu system-config-httpd, dającego możliwość wygodnej konfiguracji podstawowych elementów serwera WWW

W ramach zakładki *Główne* można ustawić nazwę serwera, podać adres administratora witryny oraz zdefiniować numer portu, pod którym Apache będzie prowadził nasłuch w oczekiwaniu na napływające żądania. Tutaj można też ustawić nasłuch na dodatkowych portach, dla potrzeb serwerów wirtualnych.

Konfigurowanie węzłów wirtualnych

W ramach zakładki *Hosty wirtualne* można definiować cechy właściwe dla każdego z serwerów wirtualnych. Lista nazw owych serwerów wyświetlana jest na liście w kolumnie *Nazwa*. Po wskazaniu węzła wirtualnego i naciśnięciu przycisku *Edytuj* wywołuje się okno dialogowe ustawień węzła wirtualnego, prezentowane na rysunku 17.2. Najbardziej podstawowe ustawienia węzła wirtualnego zebrane są na zakładce *Opcje ogólne*.



Rysunek 17.2. Okno dialogowe Właściwości hosta wirtualnego programu system-config-httpd daje dostęp do szeregu opcji konfigurujących węzły wirtualne serwera Apache

Po wywołaniu zakładki *Ustawienia strony* w panelu opcji wyświetlane są charakterystyczne dla danego węzła wirtualnego parametry, takie jak lista plików indeksu katalogu zwracanych do klienta w obliczu braku w katalogu domyślnego pliku indeksowego, *index.html*.

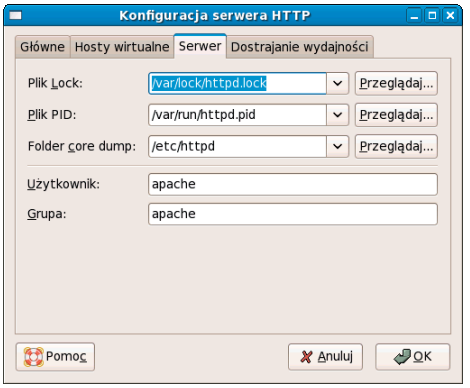
Zakładka *SSL* daje dostęp do ustawień włączających i wyłączających komunikację za pośrednictwem protokołu SSL, określających ustawienia certyfikatów i definiujących nazwy i położenie plików dzienników modułu zawiadującego komunikacją SSL. Zakładka *Dziennik* grupuje opcje konfigurujące położenie pliku dziennika błędów oraz dziennika odwołań i pozwala na zdefiniowanie poziomu rejestrowania, czyli ilości i rodzaju informacji trafiających do plików dziennika.

Opcje kategorii *Środowisko* służą do konfigurowania modułu `mod_env`, wykorzystywanego do przekazywania zmiennych środowiskowych do programów CGI.

Konfigurowanie serwera

Zakładka *Serwer* w oknie głównym programu `system-config-httpd`, której zawartość widoczna jest na rysunku 17.3, pozwala na konfigurowanie takich aspektów działania serwera jak nazwa i położenie pliku blokady oraz pliku identyfikatora procesu serwera. W obu przypadkach znakomicie sprawdzają się ustawienia domyślne. Zakładka pozwala poza tym na wskazanie katalogu, w którym ewentualnie zostanie umieszczony zrzut pamięci procesu serwera w przypadku jego błędnego zakończenia.

Na tej zakładce można wreszcie wskazać użytkownika i grupę, w których imieniu uruchamiany będzie serwer Apache. Jak już wspominaliśmy, ze względów bezpieczeństwa najlepiej uruchamiać serwer Apache w imieniu użytkownika `apache` i członka grupy `apache` (tak, jak to jest ustawione domyślnie).



Rysunek 17.3. Zakładka Server programu system-config-httpd

Konfigurowanie serwera pod kątem wydajności szczytowej

Opcje zakładki *Dostrajanie wydajności* służą do takiego dostosowywania konfiguracji serwera, które zapewni maksymalną wydajność. Opcje tej zakładki pozwalają na ustawienie maksymalnej liczby połączeń, limitu czasu podtrzymania połączenia oraz liczby żądań obsługiwanych w ramach jednego połączenia. Ustawiając ten ostatni parametr, należy pamiętać, że każde odebrane przez serwer połączenie obsługiwane jest za pośrednictwem oddzielnego egzemplarza procesu serwera. Każdy z takich egzemplarzy zajmuje pewną ilość zasobów systemu, w tym zasobów najcenniejszych, czyli czasu procesora i pamięci operacyjnej. Kolejne podpowiedzi dotyczące zwiększania wydajności serwera WWW zostały umieszczone w rozdziale 31., „Strojenie wydajności".

Ustawienia konfiguracyjne serwera

W tym momencie, jeśli, śledząc tekst, wykonujesz prezentowane w nim czynności, powinieneś dysponować działającym serwerem WWW; serwer ten nie musi jednak odpowiadać w pełni Twoim wymaganiom. Być może nieodpowiednia jest np. lokalizacja plików strony WWW udostępnianej przez serwer. Stąd w tym podrozdziale pokazane zostaną podstawy konfiguracji serwera.

Konfiguracja sterująca działaniem serwera Apache zapisywana jest w pojedynczym pliku o nazwie *httpd.conf*, umieszczonym w podkatalogu */etc/httpd/conf/*. Parametry zdefiniowane w tym pliku sterują niemal wszystkimi aspektami działania serwera, pozwalając na wskazanie katalogu dokumentów serwera (domyślnie */var/www*), określenie nazwy pliku przechowującego identyfikator procesu serwera (domyślnie */etc/httpd/run/httpd.pid*) czy nawet limitu czasu odpowiedzi serwera (domyślnie 300 sekund). Serwer odczytuje owe parametry przy uruchamianiu (albo przeładowywaniu) usługi. Można też wymusić ręcznie ponowny odczyt pliku konfiguracyjnego,

korzystając z polecenia `/etc/rc.d/init.d/httpd reload`. Możliwość ta przydaje się po wprowadzeniu zmian w pliku konfiguracyjnym (sposób przeładowywania serwera omawiany był w podrozdziale „Uruchamianie i zatrzymywanie serwera Apache").

Dyrektywy konfiguracyjne

Konfigurację działania serwera definiuje się za pośrednictwem dyrektyw konfiguracyjnych, będących swego rodzaju poleceniami i opcjami dla procesu serwera. Dyrektywy instruują serwer o konieczności uruchomienia określonych funkcji i o parametrach działania serwera (np. wskazują położenie plików ważnych dla poprawnego działania serwera). Obecnie Apache rozpoznaje blisko 300 dyrektyw konfiguracyjnych; są one definiowane w pliku konfiguracji przy użyciu następującej składni:

```
dyrektywa opcja opcja ...
```

Każda dyrektywa definiowana jest w osobnym wierszu pliku konfiguracyjnego. Niektóre z dyrektyw jedynie definiują wartości pewnych parametrów, np. przypisują nazwę pliku, podczas gdy inne pozwalają na sterowanie działaniem funkcji serwera. Niektóre z dyrektyw specjalnych, zwanych blokowymi, zapisuje się podobnie jak znaczniki języka HTML. Dyrektywy blokowe ujmowane są w znaki nawiasów ostrych, np.: *<Dyrektywa>*. Dyrektywa blokowa ogranicza zwykle całą grupę dyrektyw, stosowanych np. wyłącznie do wybranego, określonego w nagłówku dyrektywy katalogu dokumentów serwera:

```
<Directory katalog/w/drzewie/dokumentów/serwera>
  dyrektywa opcja opcja
  dyrektywa opcja opcja
  ...
</Directory>
```

Dyrektywy blokowe ograniczane są znacznikami podobnymi do tych, które znamy z języka HTML, więc kończą się wierszem *</Directory>*.

Wskazówka

Po zainstalowaniu i uruchomieniu serwera Apache (koniecznie wraz z pakietem dokumentacji, `httpd-manual`) indeks dyrektyw można znaleźć, odwołując się do strony *http://localhost//manual/↪mod/directives.html*.

Edycja pliku httpd.conf

Większość ustawień umieszczanych w domyślnej wersji pliku konfiguracyjnego to ustawienia działające poprawnie zwłaszcza wtedy, jeżeli serwer został zainstalowany w domyślnej lokalizacji i nie ma służyć do obsługi zadań niestandardowych. Można wręcz przyjąć zasadę, że jeżeli dana dyrektywa nie ma dla nas jasnego znaczenia, najlepiej zostawić ją w postaci domyślnej.

W kolejnych podpunktach opisane zostaną te dyrektywy pliku konfiguracyjnego, które niekiedy warto dostosować do własnych potrzeb.

ServerRoot

Dyrektywa `ServerRoot` definiuje bezwzględną ścieżkę do głównego katalogu serwera. Dyrektywa ta informuje serwer o położeniu wszystkich plików konfiguracyjnych i plików zasobów. Wiele z tych zasobów wskazywanych jest w plikach konfiguracyjnych znajdujących się w podkatalogach katalogu `ServerRoot`.

W przypadku instalacji serwera z pakietu RPM dyrektywa `ServerRoot` powinna wskazywać katalog */etc/httpd*. W przypadku samodzielnej kompilacji serwera wartością dyrektywy powinna być ścieżka */usr/local/apache/* bądź dowolna inna ścieżka zdefiniowana przy konfiguracji kodu źródłowego serwera.

Listen

Dyrektywa `Listen` określa numer portu, pod którym serwer będzie prowadził nasłuch w oczekiwaniu na żądania nawiązania połączenia. Domyślnie portem tym jest port o numerze 80, standardowym numerze portu usługi HTTP. Można jednak przypisać serwer do dowolnego innego numeru portu — można np. uruchomić serwer testowy, który nie powinien być dostępny dla osób łączących się przypadkowo. Oczywiście, zmiana numeru portu nie ma nic wspólnego z zabezpieczeniem serwera przed dostępem z zewnątrz! O rzeczywistych metodach zabezpieczania serwera WWW można poczytać w podrozdziale „Uwierzytelnianie i kontrola dostępu".

User oraz Group

Dyrektywy `User` oraz `Group` powinny być ustawione na numer identyfikatora użytkownika (`UID`) oraz grupy (`GID`), na których rzecz uruchamiany jest serwer. W dystrybucji Fedora należy ową dyrektywę ustawić tak, aby serwer Apache nie dysponował zbyt szerokimi uprawnieniami w sensie dostępu do systemu plików. Domyślnie dyrektywy te wskazują użytkownika `apache`, specjalnego użytkownika serwera Apache. Jeżeli zachodzi potrzeba ustawienia innych numerów `UID` i `GID`, należy pamiętać, że serwer działa z uprawnieniami wskazanego użytkownika, co oznacza, że w przypadku naruszenia bezpieczeństwa czy to serwera, czy tworzonych samodzielnie skryptów CGI, ewentualne szkody będą proporcjonalne do uprawnień procesu serwera WWW. Jeśli serwer działa w imieniu użytkownika *root* albo innego użytkownika uprzywilejowanego, intruz, który, wykorzystując lukę w zabezpieczeniach, przejmie kontrolę nad serwerem, będzie miał naprawdę szerokie pole do popisu. Analizując potencjalne ryzyko, należy zawsze zakładać złośliwość użytkownika i szkody, jakie powstaną w wyniku wykonania przez niego polecenia np. `rm -rf /`. Jeśli użytkownik będzie dysponował uprawnieniami użytkownika *root*, polecenie to usunie wszystkie pliki systemu. Taka wizja powinna wystarczająco przekonać do ograniczania uprawnień serwera.

Zamiast określać wartości dyrektyw `User` i `Group` za pośrednictwem nazw użytkownika i grupy, można przypisać do nich wartości liczbowe `UID` i `GID`. W takim przypadku należy się

upewnić, czy podane numery odpowiadają dokładnie wybranemu użytkownikowi i grupie; należy też pamiętać o poprzedzeniu numeru znakiem kratki (#).

Oto przykład określania użytkownika i grupy serwera za pośrednictwem nazwy:

```
User apache
Group apache
```

Identyczny efekt można osiągnąć, określając wprost numery `UID` i `GID`:

```
User #48
Group #48
```

Wskazówka

Jeżeli w systemie znajduje się wpis użytkownika innego niż *root*, o numerach `UID` i `GID` równych `0`, oznacza to, że system jest nadużywany przez złośliwego, a może wręcz groźnego użytkownika.

ServerAdmin

Dyrektywa `ServerAdmin` powinna wskazywać adres poczty elektronicznej administratora serwera WWW; warto, aby był to prawdziwy, poprawny adres poczty elektronicznej (np. *webmaster@gnulix.org*), ponieważ będzie udostępniany odwiedzającym stronę w przypadku błędu działania serwera.

ServerName

Dyrektywa `ServerName` określa nazwę węzła zwracaną przez serwer. Należy ustawić ją na pełną, kwalifikowaną nazwę domenową, np. *www.moja.domena*, a nie po prostu *www*. Ma to szczególne znaczenie, kiedy serwer dostępny jest spoza sieci lokalnej.

Dyrektywy tej nie trzeba ustawiać, jeśli serwer ma zwracać nazwę zgodną z nazwą domenową węzła, na którym działa. Kiedy brakuje wartości tej dyrektywy, serwer określi nazwę samodzielnie, realizując zapytanie odwrotnego DNS. Często jednak lepiej prezentować odwiedzającym przyjazną wersję nazwy domenowej, np. *www.moja.domena*. Tak czy inaczej wartość `ServerName` powinna być prawidłową nazwą z punktu widzenia DNS danej sieci. Jeżeli więc zarządzanie serwerem DNS realizowane jest przez osoby trzecie, należy je poprosić o uzupełnienie bazy o nazwę określoną w dyrektywie `ServerName`.

DocumentRoot

Jest to dyrektywa wskazująca ścieżkę (bezwzględną) do drzewa dokumentów serwera, czyli do głównego katalogu, z którego pochodzą udostępniane przez serwer pliki. Domyślnie katalogiem tym jest */var/www/html*. W przypadku samodzielnej kompilacji i instalacji kodu źródłowego serwera dyrektywa ta ma wartość */usr/local/apache/htdocs* (chyba że przy konfigurowaniu

kodu źródłowego podano inny katalog docelowy instalacji). Przed wersją 1.3.4 dyrektywa ta była definiowana w pliku *srm.conf*.

UserDir

Dyrektywa `UserDir` włącza lub wyłącza i definiuje katalog (określany względem katalogu domowego użytkownika), w którym serwer Apache ma szukać dokumentów HTML publikowanych przez użytkowników. Katalog ten jest określany względem katalogu domowego, ponieważ każdy użytkownik systemu może mieć możliwość publikowania własnych stron. Domyślnie dyrektywa ta jest nieaktywna (oznaczona symbolem komentarza #).

Domyślną wartością dla tej dyrektywy jest `public_html`. Po uaktywnieniu dyrektywy z taką wartością każdy użytkownik może utworzyć w swoim katalogu domowym podkatalog `public_html` i umieścić w nim dokumenty HTML tworzące jego prywatną stronę WWW. Strona ta będzie dostępna pod adresem *http://nazwa_serwera/~nazwa_użytkownika*, gdzie *nazwa_użytkownika* to systemowy identyfikator danego użytkownika. W wersjach wcześniejszych od 1.3.4 dyrektywa ta była zapisywana w pliku *srm.conf*.

DirectoryIndex

Dyrektywa `DirectoryIndex` wskazuje plik, który powinien zostać wykorzystany w roli indeksu katalogu podczas obsługi odwołań pod adresy zadane w postaci: *http://nazwa_serwera/nazwa_*
↪katalogu/.

Czasami, kiedy w katalogu brakuje pliku *index.html*, warto wskazać tu inny plik (lub listę plików), zawierający np. odpowiedni komunikat. Najciekawszym zastosowaniem tej dyrektywy jest wskazanie w niej programu CGI, który będzie uruchamiany w momencie odwołania do katalogu. W wartości dyrektywy można też uwzględnić tych użytkowników, którzy swoje strony tworzą w systemie Windows, i dodać do listy plików pozycję *index.htm*. Dyrektywa `DirectoryIndex` może więc mieć np. postać `DirectoryIndex index.html index.cgi index.htm`. W wersjach wcześniejszych od 1.3.4 dyrektywa ta była zapisywana w pliku *srm.conf*.

Moduły MPM

Serwer Apache od wersji 2.0 korzysta z nowej wewnętrznej architektury, obsługującej moduły wieloprzetwarzania *MPM* (ang. *multiprocessing modules*). Moduły te są przez serwer wykorzystywane do implementacji modelu działania wielowątkowego; moduły są skompilowane do pliku wykonywalnego serwera. Moduły MPM pozwalają na lepszą pracę serwera na większej liczbie platform, zwiększając jego stabilność i skalowalność.

Apache może korzystać z co najwyżej jednego modułu MPM na raz. Moduły MPM różnią się od zestawu modułów rozprowadzanych z serwerem Apache (patrz podrozdział „Moduły serwera Apache"); służą one do implementowania ustawień, limitów i innych czynności serwera. Każdy z modułów obsługuje własny zestaw ustawień, również zwanych **dyrektywami**, których wartości sterują szczegółowymi aspektami działania procesów serwera.

Dla Linuksa dostępne są następujące moduły MPM:

- `mpm_common` — zestaw dwudziestu dyrektyw wspólnych dla wszystkich modułów MPM;
- `prefork` — moduł implementujący tryb pracy serwera, w którym naśladuje on zachowanie serwerów z serii 1.3 (obsługa żądań przez osobne procesy potomne serwera);
- `worker` — właściwy tryb pracy serwerów serii 2.0, implementujący serwer hybrydowy, wieloprocesowy i wielowątkowy.

Odpowiedni MPM pozwala na ograniczenie zasobów wykorzystywanych przez serwer, przy zachowaniu pełnej zdolności do obsługi natłoku żądań i pełnej stabilności. Moduł `worker` udostępnia dyrektywy sterujące liczbą obsługiwanych jednocześnie połączeń.

Uwaga

Dla innych platform dostępne są inne moduły implementacji wieloprzetwarzania, np. `mpm_netware` dla węzłów pracujących w systemie Netware czy też `mpm_winnt` dla wersji przeznaczonej do systemu Windows NT, moduł o nazwie `perchild`, który pozwala na przypisywanie osobnych identyfikatorów UID dla poszczególnych procesów serwera. Więcej informacji na temat modułów MPM dla serwera Apache można znaleźć: *http://www.apache.org/*.

Pliki konfiguracyjne .htaccess

Apache wykorzystuje specjalne pliki konfiguracyjne, określane mianem plików *.htaccess*. W plikach tych mogą występować niemal wszystkie te dyrektywy, które umieszczane są w głównym pliku konfiguracji, *httpd.conf*. Plik *.htaccess*, określony dyrektywą `AccessFileName` w pliku *httpd.conf* (albo *srm.conf* w wersjach poprzedzających 1.3.4), definiuje zwykle konfigurację dla określonego katalogu. Administrator systemu może w konfiguracji serwera WWW określić zarówno nazwę pliku konfiguracji dodatkowej, jak i wskazać, które z fragmentów konfiguracji serwera będą nadpisywane zawartością plików konfiguracyjnych *.htaccess*.

W celu ograniczenia elementów konfiguracji, które mogą być przesłonięte zawartością plików *.htaccess*, wykorzystuje się dyrektywę `AllowOverride`. Dyrektywę tę można ustawić globalnie bądź powiązać z poszczególnymi katalogami. Dla przykładu, w pliku *httpd.conf* można umieścić następującą konstrukcję:

```
# Każdy katalog, do którego dostęp ma serwer Apache, może
# dysponować osobną konfiguracją uzupełniającą bądź eliminującą
# funkcje realizowane w danym katalogu (i jego podkatalogach).
#
# Na początek należy określić bardzo restrykcyjną konfigurację domyślną.
#
```

```
<Directory />
  Options FollowSymLinks
  AllowOverride None
</Directory>
```

Dyrektywa Options

Dyrektywa `Options` określa domyślne opcje konfiguracyjne serwera Apache. Dyrektywa może przyjmować wartość `None` bądź `All` albo stanowić połączenie dowolnych spośród wartości `Indexes`, `Includes`, `FollowSymLinks`, `ExecCGI` i `Multiviews`. Znaczenie poszczególnych wartości wyjaśniamy w tabeli 17.2.

Tabela 17.2. *Wartości dyrektywy Options*

Wartość	Znaczenie
`None`	Dla katalogu nie będzie dostępna żadna z poniższych opcji.
`All`	Dla katalogu dostępne będą wszystkie poniższe opcje.
`Indexes`	W przypadku braku pliku index.html albo innego określonego wartością dyrektywy `DirectoryIndex` serwer wygeneruje dla katalogu stronę z listą plików katalogu.
`Includes`	W katalogu mogą być wykorzystywane wstawki **SSI** (ang. *Server Side Includes*). Opcję tę można też zapisać jako `IncludesNoExec`, jeśli w katalogu mają być dostępne wstawki serwera z wyjątkiem wstawki `exec`. Ze względów bezpieczeństwa zawsze warto wyłączać tę możliwość w tych katalogach, nad którymi administrator nie ma pełnej kontroli, np. w katalogu definiowanym dyrektywą `UserDir`.
`FollowSymLinks`	Pozwala na dostęp do katalogów, które są powiązane z katalogiem dokumentu za pośrednictwem dowiązań symbolicznych. Ustawienie tej możliwości dla poszczególnych katalogów nie jest zalecane (jest dopuszczalne w rzadkich przypadkach), w żadnym zaś razie nie należy uaktywniać tej opcji dla całego serwera. Opcja stanowi potencjalną lukę w zabezpieczeniach, ponieważ pozwala na wyjście poza katalog dokumentów, co potencjalnie daje dostęp do tych fragmentów systemu plików, które nigdy nie powinny być dostępne spoza systemu.
`ExecCGI`	W danym katalogu można uruchamiać programy CGI nawet wtedy, jeśli katalog ten nie jest zdefiniowany w dyrektywie `ScriptAlias`.
`Multiviews`	Element modułu `mod_negotiation`; kiedy klient inicjuje żądanie udostępnienia dokumentu, którego nie można odnaleźć, serwer próbuje wyszukać dokument najbliższy żądanemu, patrz *http://localhost/manual/mod/*.

Uwaga

Dyrektywy te obejmują skutkami swego działania również wszystkie podkatalogi danego katalogu.

Dyrektywa AllowOverride

Dyrektywa `AllowOverride` określa, które z opcji konfiguracyjnych mogą zostać przesłonięte ustawieniami zawartymi w pliku *.htaccess* dla danego katalogu. Można np. osobno zdefiniować możliwość przesłaniania ustawień dla głównego katalogu dokumentów oraz dla katalogów użytkowników, definiowanych dyrektywą `UserDir`.

Możliwość ta jest szczególnie przydatna w katalogach publikacji użytkowników, ponieważ użytkownicy ci zwykle nie mają dostępu do głównego pliku konfiguracji serwera.

Dyrektywa `AllowOverride` może mieć wartość `All`, `None` bądź stanowić kombinację wartości `Options`, `FileInfo`, `AuthConfig` oraz `Limit`. Ich znaczenie wyjaśniamy w tabeli 17.3.

Tabela 17.3. *Znaczenie argumentów dyrektywy AllowOverride*

Wartość	Znaczenie
`Options`	Plik *.htaccess* może uzupełniać opcje nieuwzględnione w dyrektywie `Options` dla danego katalogu.
`FileInfo`	Plik *.htaccess* może zawierać dyrektywy modyfikujące informacje o typie przechowywanych w katalogu dokumentów.
`AuthConfig`	Plik *.htaccess* może zawierać dyrektywy uwierzytelniania i autoryzacji dostępu.
`Limit`	Plik *.htaccess* może zawierać dyrektywy `allow`, `deny` i `order`.

Uwierzytelnianie i kontrola dostępu

Serwery WWW często służą do udostępniania danych, które niekoniecznie powinny być dostępne szerokiej publiczności. Stąd konieczność blokowania dostępu do pewnych materiałów bądź jego ograniczania dla określonych użytkowników. Serwer Apache umożliwia dwie metody realizacji tego rodzaju ograniczeń dostępu: uwierzytelnianie i kontrolę dostępu. Dostęp do poszczególnych elementów witryny WWW można ograniczać na podstawie rozmaitych kryteriów, np. na podstawie adresu IP bądź nazwy węzła klienta albo podania właściwego identyfikatora użytkownika i hasła.

Ostrzeżenie

Zezwalanie użytkownikom na publikowanie własnych materiałów za pośrednictwem danego serwera powoduje szereg problemów związanych z bezpieczeństwem. Każdy administrator serwera udostępnianego w internecie powinien zapoznać się z dokumentem *The World Wide Web Security FAQ*, dostępnym pod adresem *http://www.w3.org/Security/Faq/www-security-faq.html*.

Ograniczanie dostępu dyrektywami Allow oraz Deny

Jedną z prostszych metod ograniczania dostępu do materiałów udostępnianych przez serwer WWW jest ograniczanie dostępu do określonej grupy użytkowników, identyfikowanych adresami IP lub nazwami węzłów komputerów, z których inicjowane są żądania dostępu do dokumentów. Służą do tego dyrektywy `Allow` i `Deny`.

Obie dyrektywy przyjmują argument w postaci tzw. wyrażenia adresowego. W wyrażeniu tym można wykorzystywać następujące wartości:

- `all` — obejmuje wszystkie węzły, bez względu na adres IP;
- *<nazwa_węzła>* bądź *<nazwa_domeny>*, czyli zarówno kwalifikowana, jak i częściowa nazwa domenowa (np. `test.gnulix.org` albo `gnulix.org`);
- adres IP, również zarówno pełny, jak i określony cząstkowo (np. `212.85.67` albo `212.85.67.66`);
- para adres sieci i maska sieci, np. `212.85.67.0/255.255.255.0`;
- adres sieciowy określony w bezklasowym formacie wewnątrzdomenowym (CIDR), np. `212.85.67.0/24`. Zaprezentowana notacja CIDR określa sieć i maskę podsieci te same jak w przykładzie wyżej.

Jeśli jest to możliwe, to najlepiej oprzeć kontrolę dostępu na adresach IP, a nie na nazwach węzłów. Daje to większą wydajność dostępu, ponieważ eliminuje konieczność realizowania wyszukiwania nazw — adres IP klienta jest zaś zawsze nieodłączną częścią żądania.

Dyrektywy `Allow` i `Deny` mogą też być wykorzystywane do udostępniania bądź blokowania dostępu do materiałów witryny, w zależności od obecności bądź braku określonych zmiennych środowiskowych. Poniższy zapis np. blokuje realizację żądań inicjowanych z kontekstu, w którym zdefiniowana jest zmienna środowiskowa o nazwie `NOACCESS`:

```
Deny from env=NOACCESS
```

Domyślnie serwer Apache stosuje najpierw wszystkie określone dyrektywy `Deny`, a dopiero potem rozszerza ewentualny zbiór klientów dopuszczonych, analizując dyrektywy `Allow`. Aby zmienić kolejność przetwarzania tych dyrektyw, można wykorzystać dyrektywę `Order`. Apache interpretuje ją na trzy sposoby:

- `Order Deny,Allow` — dyrektywy `Deny` będą przetwarzane przed dyrektywami `Allow`. Jeżeli węzeł klienta nie jest wymieniowy w dyrektywach blokujących, zostanie dopuszczony do zasobu. To domyślny porządek interpretowania dyrektyw `allow` i `deny`.
- `Order Allow,Deny` — dyrektywy `Allow` przetwarzane są przed dyrektywami `Deny`; jeśli węzeł nie ma jawnie przyznanego dostępu, dostęp do zasobu zostanie odmówiony.
- `Order Mutual-failure` — do zasobów dopuszczane są wyłącznie te węzły, które obejmowane są dyrektywami `Allow` i równocześnie nie występują w dyrektywach `Deny`. Węzeł, który nie jest wymieniony ani w dyrektywach `Allow`, ani w dyrektywach `Deny`, nie zostanie obsłużony.

Pora na przykład. Przypuśćmy, że do zasobu `server-status` dostęp mają mieć wyłącznie użytkownicy łączący się z domeny serwera. Jeśli nazwą domeny jest *gnulix.org*, to w pliku konfiguracyjnym powinny się znaleźć następujące wiersze:

```
<Location /server-status>
  SetHandler server-status
  Order Deny,Allow
  Deny from all
  Allow from gnulix.org
</Location>
```

Uwierzytelnianie

Uwierzytelnianie to proces sprawdzania, czy odwiedzający jest tym, za kogo się podaje. Serwer Apache można skonfigurować tak, aby umożliwiał dostęp do określonych zasobów witryny WWW wyłącznie tym klientom, którzy są w stanie potwierdzić swoją tożsamość. Apache wykorzystuje kilka metod uwierzytelniania; najprostsza jest metoda określana mianem *Basic Authentication*.

Metoda uwierzytelniania prostego polega na podaniu przez klienta nazwy użytkownika oraz hasła, mającego gwarantować mu dostęp do chronionych zasobów. Serwer Apache określa możliwość dostępu do danego zasobu, weryfikując poprawność hasła. Najpierw jednak sprawdza podany identyfikator użytkownika — dopiero wtedy, kiedy taki użytkownik jest zdefiniowany, serwer przystępuje do kontroli zgodności hasła. Jeśli hasło również się zgadza, serwer udostępnia klientowi chronione zasoby.

Protokół HTTP jest protokołem bezstanowym, co oznacza, że każde żądanie kierowane do serwera WWW jest obsługiwane niezależnie od żądań poprzednich; stąd konieczność dołączania danych uwierzytelniających do każdego kolejnego żądania. Z tego powodu każde żądanie odwołujące się do zasobów chronionych będzie nieco obszerniejsze i realizowane dłużej niż żądania kierowane do obszarów niezabezpieczonych. Aby ograniczyć zużycie zasobów i zwiększyć wydajność obsługi żądań, należy chronić wyłącznie te obszary witryny, które tego bezwzględnie wymagają.

Ostrzeżenie

Nie należy wykorzystywać w roli pliku haseł serwera Apache systemowego pliku */etc/passwd*. Przy stosowaniu prostej metody uwierzytelniania nazwy użytkowników i hasła są przesyłane pomiędzy klientem a serwerem w postaci tekstu zakodowanego algorytmem base64, który można odczytać niemal równie łatwo jak otwarty tekst. Nazwy użytkowników i hasła są umieszczane w każdym żądaniu przesyłanym do serwera. Stąd każdy, kto zainstaluje program węszyciela sieciowego, może te informacje łatwo przechwycić.

Żeby skorzystać z prostej metody uwierzytelniania, należy utworzyć plik z listą użytkowników, którzy mają prawo odwoływać się do chronionych zasobów. Plik ten zawiera pary: identyfikator użytkownika i hasło. Plik taki jest więc podobny do systemowego pliku haseł, */etc/passwd*.

Do tworzenia pliku użytkowników dla serwera Apache służy program `htpasswd`. Program ten stanowi element pakietu oprogramowania serwera Apache. W przypadku instalacji serwera z pakietu RPM jest instalowany w katalogu */usr/bin*. Uruchomienie programu `htpasswd` bez podania jakichkolwiek opcji spowoduje wyprowadzenie następującego komunikatu:

```
Usage:
        htpasswd [-cmdpsD] passwordfile username
        htpasswd -b[cmdpsD] passwordfile username password

        htpasswd -n[mdps] username
        htpasswd -nb[mdps] username password
 -c  Create a new file.
 -n  Don't update file; display results on stdout.
 -m  Force MD5 encryption of the password.
 -d  Force CRYPT encryption of the password (default).
 -p  Do not encrypt the password (plaintext).
 -s  Force SHA encryption of the password.
 -b  Use the password from the command line rather than prompting for it.
 -D  Delete the specified user.
On Windows, NetWare and TPF systems the '-m' flag is used by default.
On all other systems, the '-p' flag will probably not work.
```

Jak widać, program `htpasswd` nie jest narzędziem specjalnie skomplikowanym. Aby np. utworzyć plik użytkowników serwera o nazwie *klientela* zawierający pojedynczy wpis użytkownika o nazwie `janek`, należy wykonać następujące polecenie:

```
# htpasswd -c klientela janek
```

Program zapyta o hasło dla użytkownika. W celu uzupełnienia pliku o kolejne wpisy użytkowników należy powyższą procedurę powtórzyć, usuwając z wiersza wywołania programu przełącznik `-c`.

Można też tworzyć pliki grup użytkowników. Pliki takie mają format zbliżony do formatu systemowego pliku */etc/groups*. W każdym wierszu pliku należy umieścić nazwę grupy, a za nią dwukropek i listę identyfikatorów użytkowników, oddzielonych spacjami. Wpis taki mógłby wyglądać np. tak:

```
klientela: janek hifly wojt andrew
```

Gdy dysponujemy umiejętnością tworzenia pliku użytkowników, można przejść do omówienia sposobu, w jaki za pośrednictwem konfiguracji serwera można chronić dostęp do niektórych zasobów witryny WWW.

Aby wskazać serwerowi plik użytkowników, należy odpowiednio ustawić dyrektywę `AuthUserFile`. Dyrektywa ta przyjmuje jako argument ścieżkę dostępu do pliku użytkowników. Jeśli ścieżka nie jest ścieżką bezwzględną (tzn. nie rozpoczyna się od znaku ukośnika), to zakłada się,

że została określona względem katalogu `ServerRoot`. Podobnie wskazuje się serwerowi plik grup użytkowników — służy do tego dyrektywa `AuthGroupFile`.

Następnie należy za pośrednictwem dyrektywy `AuthType` określić metodę uwierzytelniania. W naszym omówieniu przyjęliśmy metodę prostą, czyli dyrektywa otrzymuje wartość `Basic`.

Teraz należy zdecydować, do której ze sfer należeć będzie chroniony zasób. Sfery (ang. *realms*) grupują różne zasoby, do których mogą się odwoływać ci sami użytkownicy. Sfera definiowana jest dowolnym ciągiem znakowym. Powinna być widoczna w okienku uwierzytelniania wyświetlanym w oknie przeglądarki użytkownika odwołującego się do zasobów chronionych. Z tego względu ciąg określający sferę powinien być dla użytkowników znaczący. Sfera uwierzytelniania definiowana jest dyrektywą `AuthName`.

Na koniec należy określić grupę użytkowników autoryzowanych do korzystania z zasobu. Dokonuje się tego za pośrednictwem dyrektywy `Require`. Dyrektywę tę można definiować z trzema kategoriami wartości.

- Z opcją `valid-user` — każdy z użytkowników wymienionych w pliku użytkowników będzie dopuszczany do zasobu chronionego (zasób będzie więc udostępniany wszystkim, którzy wprowadzą poprawny identyfikator i hasło).
- Z listą użytkowników uprawnionych, wymienionych po słowie `users`.
- Z listą grup użytkowników uprawnionych, wymienionych po słowie `group`. Listy grup i użytkowników są oddzielane spacjami.

Wracając do prezentowanego wcześniej przykładu z chronionym zasobem `server-status`, możemy teraz w miejsce kontroli dostępu bazującej na nazwie (albo adresie) węzła klienta kontrolować dostęp i wymagać uwierzytelnienia od użytkowników chcących odwołać się do zasobów. Taką konfigurację definiują następujące wpisy w pliku konfiguracji serwera:

```
<Location /server-status>
  SetHandler server-status
  AuthType Basic
  AuthName "Server status"
  AuthUserFIle "klientela"
  Require valid-user
</Location>
```

Kontrola dostępu raz jeszcze

Jeżeli serwer zostanie skonfigurowany z uwzględnieniem zarówno uwierzytelniania użytkowników, jak i z kontrolą dostępu na bazie źródła pochodzenia żądania, będzie wymagał od użytkowników spełnienia obu warunków dostępu do zasobu chronionego. A co w przypadku, kiedy pożądane jest, aby dostęp był przyznawany po spełnieniu dowolnego z kryteriów, czyli albo po zakwalifikowaniu się do grupy adresów węzłów dopuszczonych, albo po uwierzytelnieniu użytkownika (również spoza takiej grupy). Taką możliwość można uwzględnić za pośrednictwem

dyrektywy `Satisfy`. Dyrektywa ta może przyjąć wartość `All` (wartość domyślną) bądź `Any`. Ustawienie jej na wartość `All` oznacza, że użytkownik odwołujący się do zasobu chronionego będzie musiał zaspokoić wymagania zarówno metod kontroli dostępu, jak i metod uwierzytelniania. Wartość `Any` pozwala na skorzystanie z zasobu już po spełnieniu jednego z dwóch warunków dopuszczenia do zasobu.

Poniżej prezentowany jest jeszcze jeden przykład konfiguracji z kontrolą dostępu, znów chroniący zasób `server-status`. Tym razem ochrona zostanie ustawiona tak, aby użytkownicy domeny `gnulix` byli dopuszczani do zasobu bez dodatkowych ceregieli, a wszyscy klienci spoza domeny musieli przedstawić poprawne informacje uwierzytelniające:

```
<Location /server-status>
  SetHandler server-status
  Order Deny,Allow
  Deny from all
  Allow from gnulix.org
  AuthType Basic
  AuthName "Server status"
  AuthUserFIle "klientela"
  Require valid-user
  Satisfy Any
</Location>
```

Materiały publikowane za pośrednictwem serwera WWW można też chronić kilkoma innymi metodami, ale ich omówienie spowodowałoby rozrost tego rozdziału do postaci osobnej książki, a zaprezentowane metody uwierzytelniania prostego i kontroli dostępu sprawdzają się w wielu typowych konfiguracjach. Więcej przykładów zabezpieczania można znaleźć w dokumentacji serwera Apache.

Moduły serwera Apache

Rdzeń serwera Apache jest stosunkowo niewielki; większość zaawansowanych funkcji serwera Apache implementowanych jest bowiem w postaci modułów. Każdy z nich odpowiedzialny jest za obsługę innego aspektu działania serwera. Dodając i usuwając moduły, można więc wybiórczo uruchamiać i blokować funkcje serwera WWW wedle konkretnych potrzeb.

Wraz z serwerem Apache rozprowadzanych jest przeszło 60 różnych modułów. Jeszcze więcej modułów publikowanych jest przez programistów niezależnych. Pod adresem *http://modules.apache.org/* znajduje się ogólnodostępne repozytorium modułów dla serwera Apache, znane jako *Apache Module Registry*. Moduły zainstalowane wraz z oprogramowaniem serwera są zebrane w katalogu */etc/httpd/modules*, jednak katalog ten jest jedynie dowiązaniem symbolicznym do właściwego katalogu modułów, czyli */usr/lib/httpd/modules*. Oto ich lista (jej zawartość może się różnić, w zależności od sposobu instalacji i numeru wersji serwera):

mod_actions.so
mod_alias.so
mod_asis.so
mod_auth_basic.so
mod_auth_digest.so
mod_authn_alias.so
mod_authn_anon.so
mod_authn_dbd.so
mod_authn_dbm.so
mod_authn_default.so
mod_authn_file.so
mod_authnz_ldap.so
mod_authz_dbm.so
mod_authz_default.so
mod_authz_groupfile.so
mod_authz_host.so
mod_authz_owner.so
mod_authz_user.so
mod_autoindex.so
mod_cache.so
mod_cern_meta.so
mod_cgid.so
mod_cgi.so
mod_dav_fs.so
mod_dav.so
mod_dbd.so
mod_deflate.so
mod_dir.so
mod_disk_cache.so
mod_dumpio.so
mod_env.so
mod_expires.so
mod_ext_filter.so
mod_file_cache.so
mod_filter.so
mod_headers.so
mod_ident.so
mod_imagemap.so
mod_include.so
mod_info.so
mod_ldap.so
mod_log_config.so
mod_log_forensic.so
mod_logio.so
mod_mem_cache.so
mod_mime_magic.so
mod_mime.so
mod_negotiation.so
mod_proxy_ajp.so
mod_proxy_balancer.so
mod_proxy_connect.so
mod_proxy_ftp.so
mod_proxy_http.so
mod_proxy.so
mod_rewrite.so
mod_setenvif.so
mod_speling.so
mod_status.so
mod_suexec.so
mod_unique_id.so
mod_userdir.so
mod_usertrack.so
mod_version.so
mod_vhost_alias.so

Każdy z modułów uzupełnia konfigurację serwera o nowe dyrektywy. Jak można się domyślić, dodatkowych dyrektyw, opcji i przełączników jest o wiele za dużo, aby opisać je w jednym rozdziale. W kolejnych punktach zostaną jedynie — z konieczności — opisane funkcje wybranych modułów.

mod_authz_host

Moduł `mod_authz_host` pozwala na sterowanie (za pośrednictwem dyrektyw `Allow`) funkcją kontroli dostępu do serwera WWW opartej na adresach IP, nazwach węzłów i obecności

zmiennych środowiskowych. Dzięki temu modułowi można np. zezwolić na pełny dostęp do materiałów udostępnianych przez serwer z poziomu sieci lokalnej, natomiast szerokiej publiczności nawiązującej połączenie z serwerem spoza sieci lokalnej udostępniać jedynie podzbiór tych materiałów, patrz podrozdział „Uwierzytelnianie i kontrola dostępu".

mod_alias

Moduł `mod_alias` manipuluje ciągami URL umieszczonymi w napływających żądaniach HTTP, przekierowując np. klienta pod właściwy adres URL. Za pomocą tego modułu można też odwzorowywać części systemu plików na poszczególne elementy hierarchii witryny WWW. Dla przykładu dyrektywa:

```
Alias /images/ /home/janek/graphics/
```

spowoduje, w odpowiedzi na żądanie zawierające URL zaczynający się od ciągu */images/*, pobranie zawartości katalogu */home/janek/graphics/*. Przy tym klient inicjujący żądanie nie będzie nawet wiedział o dokonaniu podmiany. W przypadku przekierowania natomiast klient otrzyma w odpowiedzi URL wskazujący właściwe położenie żądanych zasobów. Więcej zaawansowanych funkcji manipulowania ciągami URL oferuje moduł `mod_rewrite`.

mod_asis

Moduł `mod_asis` pozwala na szczegółowe określenie wszystkich informacji, które mają zostać odesłane w odpowiedzi na żądanie. Odpowiedzi generowane z użyciem tego modułu tworzone są z pominięciem wszelkich nagłówków, które w innym przypadku mogłyby być dołączone do odpowiedzi. Wszystkie pliki z rozszerzeniem *.asis* przesyłane są do klienta bez jakichkolwiek zmian i uzupełnień.

W ramach prostego przykładu można założyć, że z danej strony przeniesiono materiały pod inny adres. Teraz wypada poinformować odwiedzających o fakcie przeprowadzki, a najlepiej automatycznie skierować ich pod właściwy adres. W celu zaprezentowania odwiedzającym odpowiedniego komunikatu i skierowania ich do właściwej strony należy umieścić w pustym katalogu plik z rozszerzeniem *.asis* i następującą treścią:

```
Status: 301 Stary adres!
Location: http://gnulix.org/nowy/strona.html
Content-type: text/html

<html>
  <head>
    <title>Strona przeniesiona</title>
  </head>
  <body>
    <h1>Strona została przeniesiona. Obecnie dostępna jest pod adresem:
      <a href="http://gnulix.org/nowy/strona.html">Nowy Adres</a>.
    </h1>
  </body>
</html>
```

mod_auth_basic i mod_authn_file

Ta para modułów pełni rolę zarezerwowaną w wersji 2.0 dla pojedynczego modułu `mod_auth`, a więc implementuje prostą metodę uwierzytelniania użytkowników bazującą na przechowywanych w pliku użytkowników identyfikatorach i zaszyfrowanych hasłach. Plik użytkowników jest podobny do systemowego pliku */etc/passwd*; tworzy się go i wypełnia poleceniem `htpasswd`. Uwierzytelnianiu poświęcony był podrozdział „Uwierzytelnianie i kontrola dostępu".

mod_authn_anon

Moduł `mod_authn_anon` implementuje uwierzytelnianie anonimowe, podobne do znanego z serwerów FTP. Moduł pozwala na określenie identyfikatorów użytkowników tych klientów, którzy łączą się z serwerem jako goście. Kiedy taki użytkownik próbuje się zalogować, zostanie zapytany o podanie hasła w postaci adresu poczty elektronicznej. Serwer może rejestrować podawane adresy.

mod_authn_dbm

Moduł `mod_authn_dbm` pozwala na wykorzystywanie przy uwierzytelnianiu pliku bazy danych Berkeley DB w miejsce pliku użytkowników.

mod_auth_digest

Rozszerzenie modułu uwierzytelniania prostego; w trybie uwierzytelniania Digest użytkownik nie przesyła informacji uwierzytelniających otwartym tekstem, ale szyfruje je algorytmem MD5. Taki schemat uwierzytelniania został zdefiniowany w dokumencie RFC 2617. W porównaniu z metodą uwierzytelniania prostego uwierzytelnianie to jest dużo bezpieczniejsze. Niestety, nie wszystkie przeglądarki WWW obsługują tę metodę uwierzytelniania.

Do tworzenia pliku haseł dla tej metody uwierzytelniania służy program `htdigest`. Jego opcje pokrywają się mniej więcej z opcjami analogicznego programu dla metody Basic, `htpasswd`. Więcej informacji można znaleźć na stronie podręcznika systemowego pod hasłem `htdigest`.

mod_autoindex

Moduł `mod_autoindex` pozwala na dynamiczne tworzenie listy plików indeksu katalogowego. Indeks jest uzupełniany informacjami dodatkowymi tak, aby przypominał listingi katalogów uzyskiwane podczas sesji FTP za pośrednictwem polecenia `ls`.

mod_cgi

Moduł ten pozwala na uruchamianie przez serwer programów CGI. Programy CGI to wykonywalne pliki umieszczone w podkatalogu */var/www/cgi-bin*; służą do dynamicznego generowania danych (zwykle dokumentów HTML).

mod_dir oraz mod_env

Moduł `mod_dir` służy do określania, które z plików powinny zostać zwrócone automatycznie, kiedy użytkownik odwołuje się do katalogu. Domyślnie plikiem takim jest *index.html*. Jeżeli serwer ma obsługiwać również użytkowników tworzących strony WWW w systemie Windows, listę plików indeksowych warto rozszerzyć o plik *index.htm*, tak jak poniżej:

```
DirectoryIndex index.html index.htm
```

Moduł `mod_env` zawiaduje przekazywaniem zmiennych środowiskowych do skryptów CGI i SSI.

mod_expires

Moduł `mod_expires` służy do uzupełniania materiałów publikowanych w witrynie o informacje o dacie ważności. Moduł uzupełnia generowane odpowiedzi HTTP o nagłówki `Expires` i `Cache-Control`. Określanie czasu ważności strony pozwala na buforowanie stron WWW przez przeglądarki bądź serwery buforujące WWW.

mod_headers

Moduł ten służy do manipulowania nagłówkami odpowiedzi HTTP generowanych przez serwer. Można dzięki niemu usuwać nagłówki, dodawać je i zastępować własnymi. Moduł udostępnia w tym celu dyrektywę `Header`. Ważna jest kolejność definiowania dyrektyw `Header`. Otóż włączenie danego nagłówka wartością `set` dyrektywy, a następnie wyłączenie go wartością `unset` spowoduje usunięcie nagłówka. Dyrektywy `Header` mogą być rozmieszczane w niemal dowolnych miejscach w pliku konfiguracyjnym. Dyrektywy te są przetwarzane w następującej kolejności:

1. Dyrektywy dla całego serwera,
2. Dyrektywy dla węzła wirtualnego,
3. Dyrektywy w blokach `<Directory>` i plikach *.htaccess*,
4. Dyrektywy w blokach `<Location>`,
5. Dyrektywy w blokach `<Files>`.

mod_include

Moduł `mod_include` pozwala serwerowi na obsługę tzw. wstawek *SSI* (ang. *Server Side Includes*).

mod_info oraz mod_log_config

Moduł `mod_info` udostępnia klientom obszerne informacje o konfiguracji serwera. Za jego pomocą można np. wyświetlić w oknie przeglądarki listę wszystkich zainstalowanych modułów oraz dyrektyw zdefiniowanych w plikach konfiguracyjnych.

Moduł `mod_log_config` definiuje format i zawartość plików dziennika, patrz podrozdział „Rejestrowanie".

mod_mime oraz mod_mime_magic

Moduł `mod_mime` pozwala na podejmowanie przez serwer próby określania typu MIME pliku na podstawie jego rozszerzenia.

Moduł `mod_mime_magic` pozwala z kolei na określanie typu MIME pliku na podstawie zawartości pliku.

mod_negotiation

Za pośrednictwem modułu `mod_negotiation` można przy generowaniu odpowiedzi wybrać jedną z kilku wersji dokumentu, najlepiej dostosowaną do możliwości klienta. Wybór może opierać się na kilku różnych kryteriach, określanych w procesie negocjacji. Można np. wybierać pomiędzy różnymi wersjami językowymi dokumentów, formatami plików oraz metodami kompresji.

mod_proxy

Moduł `mod_proxy` implementuje dla serwera Apache mechanizm buforowania i pośredniczenia. Dzięki temu serwer może pośredniczyć w przekazywaniu FTP, AJP13, CONNECT, HTTP/0.9, HTTP/1.0 oraz HTTP/1.1. Uaktywnienie modułu `mod_proxy` nie jest najlepszym pomysłem, jeżeli serwer ma obsługiwać dużą liczbę użytkowników.

mod_rewrite

Moduł `mod_rewrite` to istny szwajcarski scyzoryk w dziedzinie manipulowania ciągami URL. Pozwala na wykonywanie niemal dowolnych, dających się wymyślić manipulacji, również z wykorzystaniem złożonych wyrażeń regularnych. W sumie mało jest operacji na URL, których nie można by przeprowadzić za pośrednictwem tego modułu.

Wskazówka

Możliwości modułu `mod_rewrite` można lepiej poznać, zaglądając pod adres *http://localhost/manual/misc/rewriteguide.html*.

mod_setenvif

Moduł `mod_setenvif` pozwala na manipulowanie zmiennymi środowiskowymi. Otóż można tu za pomocą ciągów symboli dopasowywanych do tekstu, znanych jako **wyrażenia regularne**, warunkowo zmieniać wartości zmiennych środowiskowych. Istotna jest kolejność występowania w pliku konfiguracji wprowadzanych przez moduł dyrektyw `SetEnvIf`. Każda kolejna dyrektywa `SetEnvIf` może bowiem znieść działanie poprzedniej dyrektywy, jeśli obie odnoszą się do tej samej zmiennej środowiskowej. Warto o tym pamiętać, aby uniknąć niespodzianek i trudnych do wykrycia błędów.

mod_speling

Moduł `mod_speling` implementuje funkcję kontroli poprawności i poprawiania nieprawidłowo wpisanych adresów URL. Jeśli np. żądany URL nie zostanie znaleziony, serwer może przystąpić do utworzenia listy plików leżących w tym samym katalogu i dopasowania jednego z tych plików jako najbliższego żądaniu. Dopasowanie polega na próbie poprawienia tylko jednej literówki.

mod_status

Moduł ten można wykorzystywać do generowania strony WWW zawierającej informacje o bieżącym stanie serwera Apache. Strona ta zawiera informacje o wewnętrznym stanie procesów serwera i statystyki ich działania. Informacje te są bezcenne w procesie dostosowywania konfiguracji tak, aby osiągnąć maksymalną wydajność serwera. Równie przydatne są przy próbie określania źródła ewentualnych problemów.

mod_ssl

Moduł `mod_ssl` implementuje protokół SSL (w wersji 2. oraz 3.) oraz TLS (wersja 1.). Moduł wprowadza do pliku konfiguracyjnego przynajmniej 30 dyrektyw, definiujących sposób oraz tryb szyfrowania i uwierzytelniania klientów.

mod_unique_id

Moduł ten dla każdego napływającego żądania generuje unikalny identyfikator żądania. Identyfikator ten umieszczany jest w zmiennej środowiskowej `UNIQUE_ID`.

mod_userdir

Moduł `mod_userdir` pozwala na odwzorowywanie w drzewie dokumentów serwera podkatalogów znajdujących się w katalogach domowych użytkowników.

mod_usertrack

Moduł `mod_usertrack` służy do generowania dla kolejnych sesji użytkownika specjalnych cookies sesji. Można dzięki nim śledzić poczynania użytkownika odwiedzającego kolejne strony witryny. Aby po takich odwiedzinach został jakiś ślad, należy uaktywnić rejestrowanie cookies w pliku dziennika.

mod_vhost_alias

Moduł `mod_vhost_alias` obsługuje dynamiczną konfigurację węzłów wirtualnych, przydatną, jeśli serwer służy jako dostawca usług internetowych i ma obsługiwać większą liczbę węzłów wirtualnych. Jednak dla przeciętnego użytkownika standardowa obsługa węzłów wirtualnych serwera Apache powinna być jak najbardziej wystarczająca.

Serwer Apache potrafi obsługiwać węzły wirtualne na dwa sposoby. Można do jednego adresu IP przypisać wiele nazw węzłów albo nazwy te przypisywać do osobnych adresów IP. W obu tych przypadkach przypisaniem i działaniem serwerów wirtualnych sterują różne zestawy dyrektyw (więcej o serwerach wirtualnych w następnym podrozdziale).

Jak widać, liczba dostępnych modułów i opcji tych modułów powoduje, że jakikolwiek wyczerpujący ich opis musiałby zająć znacznie więcej niż jeden rozdział. Kompletną dokumentację modułów serwera Apache można znaleźć w udostępnianej przez sam serwer dokumentacji rozprowadzanej w dystrybucji Fedora albo publikowanej na stronach Apache Software Foundation.

Serwery wirtualne

Jedną z najpopularniejszych usług udostępnianych przez serwer WWW jest możliwość obsługi domeny wirtualnej. Domena taka, znana również jako **węzeł wirtualny** bądź **serwer wirtualny**, jest kompletną witryną WWW, dysponującą własną nazwą, zupełnie tak, jakby była obsługiwana przez dedykowany jej serwer WWW zainstalowany na dedykowanym witrynie węźle sieci. To jednak pozór, ponieważ węzeł wirtualny wraz z jego „domeną" jest obsługiwany przez pojedynczy serwer, zdolny do równoczesnej obsługi wielu takich węzłów. Apache implementuje tę możliwość za pośrednictwem dyrektyw pliku konfiguracyjnego.

Teraz serwer Apache potrafi również dynamicznie tworzyć witryny wirtualne, korzystając z pomocy modułu `mod_vhost_alias`. Moduł ten jest przeznaczony głównie dla dostawców usług internetowych, którzy obsługują znaczną liczbę użytkowników i ich stron WWW. Jako

że jest przeznaczony do zastosowań profesjonalnych, jego omówienie wykracza poza tematykę tego — mającego przecież charakter wprowadzający — rozdziału. Dlatego w niniejszym podrozdziale opisany zostanie jedynie tradycyjny sposób definiowania węzłów wirtualnych, adekwatny do potrzeb przeciętnego użytkownika.

Węzły wirtualne rozróżniane adresami IP

Jeżeli komputer z systemem Linux zostanie już tak skonfigurowany, że będzie odpowiadał na połączenia dla kilku różnych adresów IP, to uwzględniająca ten stan konfiguracja serwera Apache jest już stosunkowo prosta. Wystarczy jedynie umieścić w pliku konfiguracyjnym *httpd.conf* jedną dyrektywę blokową `VirtualHost` dla każdego z adresów, pod którymi mają działać niezależne od siebie witryny WWW:

```
<VirtualHost 212.85.67.67>
 ServerName gnulix.org
 DocumentRoot /home/virtual/gnulix/public_html
 TransferLog /home/virtual/gnulix/logs/access_log
 ErrorLog /home/virtual/gnulix/logs/error_log
</VirtualHost>
```

W nagłówku dyrektywy `VirtualHost` należy umieszczać nie nazwy węzłów, ale ich adresy IP.

Wewnątrz dyrektywy blokowej `VirtualHost` można określać dowolne dyrektywy konfiguracyjne. Można np. w ramach tej dyrektywy umieścić dyrektywy `AllowOverride` o działaniu odmiennym, niż to zdefiniowano dla całego serwera. Wszystkie zaś dyrektywy węzła wirtualnego, które nie zostaną określone wprost, przejmują wartości z dyrektyw określonych dla całego serwera.

Węzły wirtualne rozróżniane nazwami

Węzły wirtualne rozróżniane nazwami pozwalają na uruchomienie więcej niż jednej witryny na węźle dysponującym pojedynczym adresem IP. Utworzone tak „domeny" należy jednak umieścić w systemie DNS jako rekordy CNAME dla danego komputera. Kiedy klient protokołu HTTP (czyli przeglądarka WWW) nadeśle żądanie udostępnienia dokumentu przechowywanego na serwerze, w żądaniu znajduje się zmienna wskazująca nazwę serwera, który powinien zawierać dokument. I właśnie na podstawie tej zmiennej serwer może wytypować węzeł wirtualny do obsługi żądania.

Uwaga

Niektóre ze starszych przeglądarek WWW nie potrafią rozróżniać węzłów za pomocą nazw, ponieważ możliwość ta została wprowadzona wraz z wersją 1.1 protokołu HTTP, a starsze przeglądarki są zwykle zgodne jedynie z wersją HTTP 1.0. Jeżeli jednak przeglądarka jest choć częściowo zgodna z protokołem HTTP 1.1, to z pewnością daje tę właśnie możliwość.

Węzły wirtualne rozróżniane nazwami wymagają umieszczenia w pliku konfiguracyjnym jednego tylko dodatkowego wpisu. Otóż należy poinformować serwer WWW, który z adresów IP ma w systemie DNS więcej niż jedną nazwę. Służy do tego dyrektywa `NameVirtualHost`:

```
NameVirtualHost 212.85.67.67
```

W pliku konfiguracyjnym powinien znaleźć się blok dyrektyw dla każdej nazwy przypisanej do adresu. Podobnie jak w przypadku węzłów wirtualnych rozróżnianych adresami IP, w ramach dyrektywy blokowej `VirtualHost` trzeba definiować tylko te dyrektywy, które powinny mieć wartości inne od dyrektyw definiowanych globalnie, dla całego serwera. Koniecznie natomiast trzeba w bloku zdefiniować dyrektywę `ServerName`, ponieważ tylko ona rozróżnia poszczególne serwery wirtualne:

```
<VirtualHost 212.85.67.67>
 ServerName bugserver.gnulix.org
 ServerAlias bugserver
 DocumentRoot /home/bugserver/htdocs
 ScriptAlias /home/bugserver/cgi-bin
 TransferLog /home/bugserver/logs/access_log
</VirtualHost>

<VirtualHost 212.85.67.67>
 ServerName pts.gnulix.org
 ServerAlias pts
 DocumentRoot /home/pts/htdocs
 ScriptAlias /home/pts/cgi-bin
 TransferLog /home/pts/logs/access_log
 ErrorLog /home/pts/logs/error_log
</VirtualHost>
```

Wskazówka

Obsługując węzły wirtualne w sieci lokalnej bądź intranecie, można udostępnić użytkownikom skrócone nazwy węzłów, aby nie musieli za każdym razem wpisywać w pasku adresu pełnej kwalifikowanej nazwy domeny. Dla przykładu, przy użyciu nazwy skróconej można odwołać się do adresu *http://bugserver/index.html* zamiast do *http://bugserver.gnulix.org/index.html*. W takim przypadku serwer Apache nie może wiedzieć, że oba te adresy odnoszą się do tego samego węzła wirtualnego. Można to obejść, definiując dwie dyrektywy blokowe `VirutalHost`, jedną dla nazwy *bugserver*, drugą dla pełnej nazwy *bugserver.gnulix.org*, ale łatwiej po prostu umieścić wewnątrz dyrektywy blokowej węzła wirtualnego dyrektywę definiującą alias węzła:

```
ServerAlias bugserver
```

Więcej o działaniu dyrektywy blokowej `VirtualHost` napisano w dokumentacji serwera, dostępnej pod adresem *http://localhost/manual*.

Rejestrowanie

Serwer Apache umożliwia rejestrowanie w plikach dziennika informacji o niemal dowolnych aspektach działania serwera. Rejestrowanie tych informacji może być pomocne do:

- lepszego zarządzania zasobami systemowymi (w przypadku informacji dotyczących wykorzystania procesów serwera);
- wykrywania włamań (w przypadku dokumentowania nieprawidłowych żądań HTTP);
- diagnostyki błędów (w przypadku rejestrowania błędów przetwarzania żądań).

Po uruchomieniu serwera Apache tworzone są domyślnie dwa pliki dzienników: *access_log* (plik dziennika napływających żądań) oraz *error_log* (plik dziennika błędów). Znajdują się one zwykle w katalogu */var/log/httpd* (w zależności od konfiguracji serwera, w katalogu tym mogą się też pojawić pliki dziennika modułu implementującego protokół SSL: *ssl_access_log*, *ssl_error_log* oraz *ssl_request_log*). Wszystkie pliki dziennika, z wyjątkiem *error_log* (czyli standardowo jedynie plik *access_log*), są generowane w formacie określonym parą dyrektyw `CustomLog` oraz `LogFormat`. Dyrektywy te umieszczane są, rzecz jasna, w pliku konfiguracyjnym *httpd.conf*.

Nowy format pliku dziennika można zdefiniować następująco:

```
LogFormat "%h %l %u %t \"%r\" %>s %b" common
```

Znacznik formatu `common` to definicja formatu bazowego, domyślnego dla wszystkich dzienników. Warto pamiętać, że większość dostępnych narzędzi automatycznie kontrolujących i analizujących zawartość plików dziennika działa w oparciu o założenie, że analizowany plik generowany jest właśnie zgodnie z formatem `common`, ewentualnie formatem `combined`; oba formaty definiowane są w domyślnych plikach konfiguracyjnych.

W definicji formatu `LogFormat` można stosować następujące zmienne.

`%a`	Zdalny adres IP.
`%A`	Lokalny adres IP.
`%b`	Liczba przesłanych bajtów, z wyłączeniem nagłówków HTTP. Zmienna rejestrowana w formacie *CLF* (ang. *Combined Log Format*), co oznacza m.in., że dla żądań pozbawionych danych w dzienniku zapisana zostanie nie wartość `0`, ale znak `-`.
`%B`	Liczba przesłanych bajtów, z wyłączeniem nagłówków HTTP.
`%{`*`ZMIENNA`*`}e`	Wartość zmiennej środowiskowej *`ZMIENNA`*.

`%f`	Nazwa pliku dziennika.
`%h`	Nazwa węzła zdalnego.
`%H`	Protokół żądania.
`%{`*`NAGŁÓWEK`*`}i`	Zawartość nagłówka *`NAGŁÓWEK`*`:` umieszczonego w żądaniu.
`%l`	Nazwa zdalnego pliku dziennika (określona przez `identd`).
`%m`	Metoda żądania.
`%{`*`NOTA`*`}n`	Wartość noty *`NOTA`* innego modułu.
`%{`*`NAGŁÓWEK`*`}o`	Zawartość nagłówka *`NAGŁÓWEK`*`:` umieszczonego w odpowiedzi.
`%p`	Port serwera obsługującego żądanie.
`%P`	Identyfikator procesu obsługującego żądanie.
`%q`	Zawartość ciągu zmiennych formularza, poprzedzona znakiem `?`. W przypadku braku ciągu odwołania zmienna przyjmuje wartość pustą.
`%r`	Pierwszy wiersz żądania.
`%s`	Status; dla żądań przekierowanych wewnętrznie będzie to status żądania oryginalnego. Ostatni status żądania uzyskuje się, zapisując zmienną jako `%>s`.
`%t`	Czas, w formacie przyjętym dla standardowego formatu dziennika (w standardowym zapisie angielskim).
`%{`*`format`*`}t`	Czas, w formacie definiowanym ciągiem *`format`*, który powinien być zgodny z formatem ciągu przyjętym dla funkcji `strftime(3)`, patrz podrozdział „Podstawowe dyrektywy SSI”, w którym znajduje się pełna lista opcji formatowania.
`%T`	Liczba sekund, jakie upłynęły w czasie realizacji żądania.
`%u`	Identyfikator użytkownika zdalnego; w przypadku kiedy status (`%s`) to `401`, identyfikator może być nieprawdziwy.
`%U`	Umieszczony w żądaniu ciąg URL.
`%V`	Nazwa serwera obsługującego żądanie, zgodnie z dyrektywą `UseCanonicalName`.
`%v`	Wartość (w postaci kanonicznej) dyrektywy `ServerName` serwera obsługującego żądanie.

Każdą zmienną można poprzedzić wyrażeniem, które będzie warunkowało umieszczenie wartości zmiennej w wierszu dziennika. W przypadku niespełnienia warunku w miejsce wartości zmiennej pojawi się znak `-`. Warunki definiowane są w postaci liczb odpowiadających wartościom

zwracanym przez serwer. Dla przykładu, zapis `%!401u` spowoduje wyświetlenie wartości zmiennej `REMOTE_USER`, chyba że zwracany przez serwer kod obsługi żądania to `401`.

Położenie i format pliku dziennika określa się za pośrednictwem dyrektywy `CustomLog`:

```
CustomLog logs/access_log common
```

Jeżeli podana w dyrektywie ścieżka nie jest ścieżką absolutną (jak to jest w przykładzie powyżej), to jest ona interpretowana względem katalogu określonego dyrektywą `ServerRoot`.

Warto zajrzeć

W sieci dostępna jest niezwykle bogata dokumentacja serwera Apache. Informacji dodatkowych o zagadnieniach opisywanych w rozdziale warto poszukać również pod następującymi adresami:

- *http://news.netcraft.com/archives/web_server_survey.html* — strona z podsumowaniem wrześniowej ankiety firmy Netcraft, obejmującej 135 166 473 respondentów (witryn) (dane z września 2007 roku). Ankiety wykazują od lat, że to właśnie Apache jest najpopularniejszym serwerem WWW w publicznym internecie.
- *http://www.apache.org/* — witryna grupy Apache Group, zawierająca m.in. wyczerpującą dokumentację serwera Apache.
- *http://www.apachetoday.com/* — strona publikująca najnowsze doniesienia o postępach w pracach nad rozwojem i poprawianiem kodu serwera. Tutaj można też przeczytać znakomite artykuły techniczne.
- *http://modules.apache.org/* — repozytorium modułów dla serwera Apache.

Serwerowi Apache poświęcono też kilka znakomitych książek. Jedną z nich jest *Apache Server Unleashed* (Sams Publishing).

Przydatne polecenia systemu Linux

Przy zarządzaniu serwerem Apache przydadzą się następujące polecenia:

- **`apachectl` — narzędzie kontroli działania serwera, rozprowadzane wraz z serwerem Apache;**
- **`system-config-httpd` — graficzne narzędzie konfiguracji serwera Apache;**
- **`httpd` — program serwera Apache;**
- **`konqueror` — przeglądarka WWW dla środowiska KDE;**
- **`elinks` — tekstowa przeglądarka stron WWW;**
- **`firefox` — znakomita, wolna i otwarta przeglądarka WWW.**