Administrarea serverelor Web

Îmbunătățirea securității serverului Apache

– *Sabin Corneliu Buraga, Dragoș Acostăchioaie*

În cadrul acestui articol vom descrie o serie de metode pentru a îmbunătăți securitatea serverelor Web, în special a celui mai folosit server Web din lume – *Apache*, care potrivit statisticilor realizate de Netscraft Inc. (`http://www.netcraft.com/survey`) ocupă în prezent peste 55% din piață.

Serverele Web și securitatea

Serverele Web au ca funcționalitate de bază recepționarea de cereri anonime de la clienți și furnizarea de informații într-o manieră dorită a fi eficientă și rapidă. În fapt, un server Web este un *daemon* care acceptă conexiuni conforme protocolului HTTP, răspunzând cererilor recepționate de la clienți. Ca și alte protocoale utilizate în Internet, protocolul *HTTP* (*HyperText Transfer Protocol*) este un protocol de tip cerere-răspuns, bazat pe TCP/IP, destinat transferurilor informațiilor hipermedia.

În ordinea importanței, problemele de securitate pot fi împărțite astfel:

- Un atacator poate exploata *bug*-urile (deficiențele) existente într-un server Web sau din *script*-uri, obținând acces neautorizat la fișiere vitale din sistem (fișiere de parole, fișiere conținând surse de *script*-uri executate pe server și altele) sau pentru a deține controlul total asupra sistemului;
- Informațiile confidențiale stocate pe serverul Web (*e.g.* baze de date, aplicații etc.) pot fi distribuite unor persoane neautorizate;
- Punctele vulnerabile necunoscute ale navigatorului pot permite accesarea de informații private stocate pe mașina clientului.

Din păcate, soluțiile utilizate în mod curent nu pot rezolva toate problemele și, deseori, sunt contradictorii. De exemplu, pentru reducerea riscurilor de interceptare, multe organizații achiziționează servere Web considerate sigure, implementând o pleiadă de protocoale de criptare, dar aceste servere necesită certificate de autentificare semnate digital care trebuie actualizate periodic.

Sfaturi generale

Furnizăm în continuare o serie de sfaturi pentru creșterea nivelului de securitate a unui server Web (oricare ar fi acesta):

(1) Pe cât posibil, un server Web nu trebuie să ruleze alte servicii și să nu accepte conexiuni de la distanță (dacă deservește un Intranet); preferabil ar fi ca *daemon*-ul de poștă electronică să nu fie operațional;
(2) Serverul Web nu trebuie să ruleze sub auspicii de super-utilizator (`root` în Unix/Linux). La fel, *script*-urile CGI (Common Gateway Interface) nu trebuie executate ca `root`.
(3) Fișierele de configurare și modulele serverului nu trebuie stocate pe o partiție care poate fi exportată prin NFS către alte mașini.
(4) Permisiunile atașate fișierelor de configurare ale serverului Web trebuie setate și monitorizate cu atenție.
(5) Se va limita sau chiar inhiba execuția directivelor SSI (Server Side Includes).
(6) *Script*-urile CGI trebuie plasate într-un singur director (de obicei `cgi-bin`) și modificările lor trebuie monitorizate.
(7) În cadrul directorului `cgi-bin` nu se permite accesul nelimitat. Utilizatorii locali nu trebuie să poată instala, edita, șterge sau chiar vizualiza *script*-uri sau fișiere de configurare.
(8) Autorii scripturilor (CGI, PHP, ASP sau altele) nu trebuie să facă publice sursele programelor lor, mai ales dacă nu sunt în variante finale, deoarece ele pot conține puncte vulnerabile nebănuite sau *bug*-uri periculoase.
(9) *Script*-urile nu trebuie să poată fi preluate prin FTP anonim și nici fișierele de configurare ale serverului FTP nu trebuie accesate via WWW.
(10) Se va interzice plasarea de la distanță pe serverul Web a fișierelor de configurare (e.g. `.htaccess` în cazul Apache). Fișierele jurnal (*log*-urile) nu trebuie să poată fi accesate de utilizatorii ordinari sau de intruși. Hackerii pot altera sau distruge informațiile din aceste fișiere.
(11) Alte măsuri, mai extreme, pot fi:

- ștergerea tuturor conturilor care nu sunt necesare;
- inhibarea execuției sau ștergerea compilatoarelor;
- ștergerea programelor utilitare care nu sunt necesare pentru rularea sistemului de operare sau a serverului Web.

Configurarea serverului Apache

Pentru a asigura servicii HTTP, serverul Apache trebuie să fie instalat în sistem (în mod uzual, fiind vorba de un pachet RPM în Linux sau de un program executabil `.exe` în Windows), iar *daemon*-ul `httpd` pornit. Apache este un sistem modular, alcătuit dintr-un server de bază și mai multe module, care sunt încărcate dinamic într-un mod similar cu funcționarea modulelor din nucleul Linux.

Ne vom referi în continuare la modalitățile de configurare ale serverului Apache în Linux. Apache poate fi configurat cu ajutorul interfeței grafice `apacheconf` (vezi meniul *System::Apache Configuration Tool* din managerul de ferestre favorit). Fișierul de configurare principal este `httpd.conf` și este localizat de obicei în directorul `/etc/httpd`. Mai există două fișiere de configurare, `access.conf` și `srm.conf`, care au fost însă înlăturate începând cu versiunea 1.3.4.

Fișierul de configurare conține câte o directivă pe fiecare linie, acestea putând fi continuate pe linia următoare adăugând la sfârșitul acesteia caracterul „\". Comentariile încep cu caracterul „#" (ca în *shell*-urile Unix). Directivele din fișierul principal de configurare se referă la configurări globale ale serverului. Pentru a aplica anumite aspecte ale serverului doar unei zone din server, directivele trebuie incluse în cadrul secțiunilor `<Directory>`, `<DirectoryMatch>`, `<Files>`, `<FilesMatch>`, `<Location>` sau `<LocationMatch>`. Acest lucru poate fi realizat și prin plasarea unui fișier denumit `.htaccess` în directorul în care se dorește modificarea comportamentului serverului, conținând directivele dorite.

De asemenea, Apache oferă posibilitatea de a servi mai multe situri Web simultan, altfel spus, *găzduire virtuală* (*virtual hosting*).

Directivele pot fi specificate în cadrul secțiunii `<VirtualHost>`, caz în care se vor referi doar la un anumit sit.

Pentru alte detalii privind directivele Apache, cititorul interesat poate consulta referințele bibliografice.

Administratorul de sistem are, de asemenea, posibilitatea de a configura fișierele jurnal Apache. Serverul generează două jurnale: primul, localizat în genere în `/var/log/httpd/access_conf`, înregistrează cererile de accesare primite de către server, iar al doilea, localizat de obicei în `/var/log/httpd/error_log`, memorează erorile apărute în decursul rezolvării cererilor (pagini inexistente, erori de conexiune etc.).

Configurarea accesului la paginile Web

În anumite cazuri, este necesar să se restricționeze accesul la anumite documente, prin intermediul autentificării prin nume de utilizator și parolă sau în funcție de adresa calculatorului clientului Web.

Autentificarea. Pentru a realiza autentificarea utilizatorilor, vom parcurge doi pași:

(1) Se creează un fișier conținând numele și parolele utilizatorilor care vor avea acces la anumite date de pe serverul Web (în particular Apache);

(2) Se configurează serverul pentru a seta care resurse vor fi protejate și care sunt utilizatorii având permisiunea accesării lor, după introducerea unei parole valide.

Lista utilizatorilor și parolelor asociate va fi memorată într-o manieră similară fișierului `/etc/passwd` din Unix. Din rațiuni de securitate, fișierul conținând această listă nu va fi stocat în directoarele compuse din documente HTML, ci la o locație mai sigură (*e.g.* în directoarele `/usr/local/etc/httpd` sau `/etc/httpd`). În exemplele următoare, vom numi acest fișier `users`, iar pentru a-l manipula vom folosi comanda `htpasswd` prin care vom adăuga/șterge utilizatorii doriți.

Deși parolele sunt criptate, fișierul de autentificare este unul text, care poate fi ușor exploatat de persoane rău-voitoare. Rămâne în responsabilitatea administratorului sistemului să seteze permisiunile de rigoare asupra fișierului și directorului unde rezidă.

Crearea unui fișier de autentificare se realizează prin apelul de mai jos (putem stabili și modul de criptare, de exemplu MD5 cu opțiunea `-m` sau SHA cu opțiunea `-s`):

```
htpasswd -c /etc/httpd/users busaco
```

Autentificarea unui utilizator pentru a accesa o zona protejată

Această linie ne permite să asignăm sau să modificăm parola unui utilizator, parola fiind solicitată de la intrarea standard. Configurarea serverului se poate realiza fie prin fișierul `httpd.conf`, fie prin `.htaccess`, indicând o zonă protejată, de obicei în funcție de directoarele dorite a fi accesate pe bază de autentificare. Fișierul `.htaccess` va fi stocat în directorul asupra căruia dorim să modificăm comportamentul implicit al serverului Web (plasarea lui într-un anumit director va avea efect asupra tuturor sub-directoarelor acestuia).

Înainte de a modifica maniera de autentificare prin fișierul `.htaccess`, administratorul serverului Apache va specifica în `httpd.conf` ca autentificările să se realizeze via `.htaccess`. Pentru aceasta, vom insera directiva `AllowOverride AuthConfig`, iar fișierul `.htaccess` poate avea liniile de mai jos:

```
AuthName "zonă restrictivă"
AuthType Basic
AuthUserFile /etc/httpd/users

require valid-user
```

unde:

- `AuthName` specifică zona dorită a fi protejată și oricare resursă din această zonă va fi accesată prin intermediul unui nume de utilizator permis și o parolă (putem crea mai multe zone partajând aceleași nume și parole).
- `AuthType` desemnează metoda de autentificare folosită de protocolul HTTP (`Basic` sau metoda, mai sigură, `Digest`; serverul Apache suportă metoda `Digest` prin adăugarea unui modul special).
- `AuthUserFile` indică fișierul de autentificare generat de `htpasswd`. Autentificarea se poate institui și pe baza grupurilor de utilizatori, folosindu-se `AuthGroupFile` (vezi mai jos).

Utilizatorii permiși vor fi dați după directiva `require`. Parametrul `valid-user` indică faptul că orice nume de utilizator prezent în fișierul furnizat de `AuthUserFile` poate accesa resursele protejate, dar putem scrie o serie de nume de utilizatori permiși (*e.g.* `require user busaco dragos` care va permite accesul numai utilizatorilor `busaco` sau `dragos`, desigur după ce s-au introdus parolele corecte).

De exemplu, pentru a permite accesul în directorul `/Web` pe baza autentificării, vom plasa fișierul `.htaccess` următor (în `/home/httpd/html/Web` presupunând că directorul rădăcină al sitului Web este localizat în `/home/httpd/html/`):

```
AuthName "Numai pentru autorul cursului"
AuthType Basic
AuthUserFile /etc/httpd/users

require valid-user
```

Atunci când un utilizator va dori să viziteze o pagină stocată în acel director, va trebui să furnizeze un nume și o parolă de acces (dacă numele sau parola nu coincid cu cele din fișierul `/etc/httpd/users`, autentificarea va eșua). Vezi și figura *„Autentificarea unui utilizator pentru a accesa o zonă protejată“*.

Mecanismul general este următorul: la primul acces al unei resurse necesitând o autentificare, serverul va returna codul 401 („Unauthorized“) și va include în antetul HTTP un câmp `WWW-Authenticate` care va conține schema de autentificare ce trebuie utilizată (de exemplu, `Basic`) și numele zonei protejate (așa-numitul *realm*). Navigatorul va cere utilizatorului introducerea unui nume și a unei parole și va realiza o cerere către server, incluzând și câmpul `Authorization` care va conține numele schemei de autentificare (`Basic`), plus numele și parola furnizate.

Serverul va realiza verificarea pe baza fișierului de configurație sau a fișierului `.htaccess` și în caz de succes va expedia resursa cerută browserului. Altfel, va fi trimis din nou codul 401.

În situația în care autentificarea s-a soldat cu succes, iar utilizatorul va dori să vizualizeze o altă resursă aparținând aceleași zone protejate, atunci serverul va răspunde cu codul 401 și navigatorul va retrimite, de fiecare dată, câmpul `Authorization`.

Grupuri de autentificare. Atunci când trebuie autentificați mai mulți utilizatori, se poate folosi un fișier specificând autentificarea pe baza unui grup de utilizatori (concept similar grupurilor din Unix). Iată un exemplu:

```
require group webgroup admin
require user webadmin
```

Se va permite accesul oricărui utilizator din grupul `webgroup` sau `admin` ori utilizatorului cu numele `webadmin`.

Un fișier de grup constă din linii desemnând grupurile și membrii acestora, sintaxa generală fiind:

```
grup: utiliz1 utiliz2 ... utilizN
```

De exemplu, putem avea:

```
webgroup: busaco dragos mihaela stanasa stefan
admin: diablo socrate
```

O linie nu poate avea mai mult de 8 KB lungime.

În `.htaccess` fișierul de grupuri de utilizatori va fi dat prin `AuthGroupFile`. De exemplu, `.htaccess` ar putea conține:

```
AuthName "Numai pentru membrii WebGroup-ului"
AuthType Basic
AuthUserFile  /etc/httpd/users
AuthGroupFile /etc/httpd/webgroup

require group webgroup
require user busaco
```

Dacă se furnizează numele unui utilizator care există în cadrul unui grup, dar căruia nu i s-a asociat nici o parolă în fișierul de autentificare a utilizatorilor, atunci acel utilizator nu va avea acces în zona restrictivă.

În situația în care numărul de utilizatori crește destul de mult, se pot utiliza metode complementare, precum stocarea informațiilor de autentificare în baze de date: fișiere DBM (modulul `mod_auth_dbm`), DB (`mod_auth_db`), mSQL (prin `mod_auth_msql`) etc. Autentificările se pot realiza via Kerberos prin utilizarea unor module adiționale în cadrul serverului Apache.

Limitarea accesului. Putem restricționa folosirea metodelor de interogare HTTP prin intermediul secțiunii `<Limit>`:

```
<Limit GET POST PUT>
  require valid-user
</Limit>
```

Astfel, autentificarea se poate realiza în funcție de metodele de acces HTTP. Dacă `require` nu apare între `<Limit>` și `</Limit>`, protecția se va aplica la toate metodele.

În cazul în care dorim să limităm restricția doar la metoda POST, vom putea scrie în `.htaccess` următoarele:

```
AuthName "POST restrictiv"
AuthType Basic
AuthUserFile /etc/httpd/users

<Limit POST>
  require user dragos socrate
</Limit>
```

Numai utilizatorii `dragos` sau `socrate` vor putea realiza o cerere prin metoda POST, alți utilizatori (neautentificați) vor utiliza alte metode (*i.e.* GET). Această manieră poate fi adoptată la restricționarea accesului la un *script* CGI (se permite metoda GET, dar numai anumiți utilizatori pot folosi POST).

Restricționarea pe baza adresei clientului. Accesul la resurse (directoare sau fișiere) se poate face pe baza adresei IP sau a domeniului clientului care a formulat cererea de acces. În acest caz, se poate include între `<Limit>` și `</Limit>` (în `.htaccess` sau în fișierul de configurație a serverului Apache) intervalul de valori ale adreselor IP permise. Cererile de la alte adrese vor fi rejectate.

Un exemplu:

```
<Limit GET POST PUT>
  order deny, allow
  deny from all
  allow from 193.231.30.0/24 192.168.0.0/16
</Limit>
```

Directiva `deny` stabilește adresele sau domeniile care nu au permisiunea de acces la resurse, iar `allow` definește adresele sau domeniile de la care clienții pot accesa resursele Web. În exemplu, am specificat un set de adrese IP permise. Ordinea de verificare a restricțiilor de acces este dictată de directiva `order`.

Pentru a permite accesul clienților Web care nu fac parte din domeniul `.infoiasi.ro`, dar care pot proveni din oricare alt domeniu, vom introduce următoarele:

```
<Limit GET>
  order allow, deny
  allow from all
  deny from .infoiasi.ro
</Limit>
```

Restricționarea combinată. După cum am văzut mai sus, accesul poate fi restricționat și pe baza adresei IP sau domeniului calculatorului client. Putem utiliza, în mod combinat, ambele metode. Dacă dorim ca accesul să se realizeze fie pe baza domeniilor permise, fie folosind autentificarea prin nume și parolă, vom include directiva `Satisfy any` în fișierul `.htaccess` sau în secțiunile `<Directory>`, `<Location>` ori `<Files>` ale fișierului de configurare. Dacă apar ambele metode de restricționare (pe baza adreselor clienților și pe baza autentificării), în mod implicit serverul Web impune să fie îndeplinite amândouă.

Redirectări. În unele cazuri ar fi dorit ca, la apariția unor erori HTTP, vizitatorii să fie redirectați spre anumite documente particulare. Implicit, la o eroare, serverul Web va transmite navigatorului o pagină descriind codul și mesajul de eroare. Putem specifica să fie transmise alte documente clientului, atunci când survin erori de genul 401 („Unauthorized“), 403 („Forbidden“), 404 („Not found“) sau 500 („Server Error“). Pentru aceasta, se pot include în `.htaccess` liniile de mai jos:

```
ErrorDocument 401 /Web/not_auth.html
ErrorDocument 404 /Web/not_found.html
```

Astfel, atunci când un anumit utilizator va cere o resursă inexistentă, serverul va trimite documentul `not_found.html` din directorul `/Web`.

Putem folosi, în locul unui document local, un anumit URI (Uniform Resource Identifier) spre care să fie redirectat navigatorul. Aceasta ne permite să redirectăm automat navigatorul atunci când am mutat definitiv o resursă la alt URI (sau în alt director, aflat pe același server). Va trebui să introducem în vechiul director un fișier `.htaccess` care să cuprindă următoarele:

```
<Limit GET POST>
  deny from all
  ErrorDocument 403 /un_alt_director/index.html
</Limit>
```

De asemenea, în `httpd.conf` putem utiliza directiva `RedirectPermanent` pentru a redirecta navigatorul spre un nou URI:

```
RedirectPermanent /~catam http://www.infoiasi.ro/~catam/
```

Pentru crearea *alias*-urilor, se va folosi directiva `alias`:

```
Alias /webgroup  /home/others/webgr/html
Alias /webgroup/ /home/others/webgr/html
```

În exemplul următor putem urmări utilizarea *alias*-urilor împreună cu directivele de restricționare a accesului pe baza adreselor IP:

```
Alias /books /home/ftp/pub/books
<Directory /home/ftp/pub/books>
  Options Indexes SymLinksIfOwnerMatch
  AllowOverride none
  order deny, allow
  deny from all
  allow from 193.231.30.0/24 192.168.0.0/16
</Directory>
```

Astfel, resursele din directorul `/home/ftp/pub/books` vor putea fi accesate prin intermediul URI-ului `http://`*`nume_server`*`/books` numai de mașinile de la adresele IP specificate în exemplu.

Găzduirea virtuală

Există două metode de implementare a găzduirii virtuale: prima bazată pe nume și a doua bazată pe adrese IP. Mașinile virtuale bazate pe adresă utilizează adresa IP a conexiunii pentru a determina mașina virtuală corectă. Astfel, pentru fiecare mașină trebuie alocată o adresă IP separată. În cazul găzduirii virtuale bazate pe nume, determinarea mașinii virtuale se face pe baza numelui acesteia. Astfel, mai multe mașini pot utiliza aceeași adresă IP.

Găzduirea virtuală bazată pe nume este mai simplu de implementat, și este recomandată utilizarea acesteia. Găzduirea bazază pe adresă trebuie utilizată doar într-una din situațiile:

- trebuie suportați clienți HTTP vechi, care nu recunosc mașinile virtuale;
- sunt necesare conexiuni sigure de tip SSL (Secure Socket Layer);
- sunt utilizate sisteme de operare care nu pot diferenția mașinile decât dacă au adrese IP diferite.

Găzduirea bazată pe nume. Pentru a utiliza serviciul de găzduire virtuală, trebuie mai întâi stabilite adresa IP și portul pentru serverul care va accepta cereri pentru respectiva mașină virtuală, cu ajutorul directivei `NameVirtualHost`. În mod normal, se utilizează toate adresele IP pe care le utilizează `httpd` (vezi directivele `BindAddress` și `Listen`), precum și toate porturile pe care serverul așteaptă cereri (vezi directiva `Port`), folosind valoarea „*“. Se stabilește apoi un bloc `<VirtualHost>` … `</VirtualHost>`. Ca parametru, `<VirtualHost>` trebuie să primească aceeași valoare utilizată la directiva `NameVirtualHost`. Blocul astfel declarat trebuie să conțină cel puțin directivele `ServerName`, care să primească drept parametru numele mașinii virtuale, și `DocumentRoot`, care să specifice în ce director se află conținutul respectivei mașini. Iată un exemplu:

```
NameVirtualHost *

<VirtualHost *>
  ServerName www.dragos.ro
  DocumentRoot /usr/web/dragos
</VirtualHost>
```

De asemenea, pentru fiecare mașină virtuală pot fi stabilite mai multe nume, utilizând directiva `ServerAlias`. Astfel, dacă se introduce declarația:

```
ServerAlias dragos.ro *.dragos.ro
```

cererile pentru mașinile din domeniul `dragos.ro` vor fi rezolvate de mașina virtuală `www.dragos.ro`. Pot fi folosite caracterele *wildcard* „*“ și „?“. Evident, și înregistrările DNS (Domain Name System) pentru serviciul `named` trebuie să fie corect configurate pentru ca adresele `*.dragos.ro` să poată fi rezolvate.

Găzduirea virtuală bazată pe adresa IP. Pentru utilizarea acestui tip de găzduire, mașina trebuie configurată fie pentru a avea mai multe conexiuni fizice la rețea, fie pentru a avea mai multe interfețe virtuale, având adrese IP diferite.

Serverul *Apache* poate fi configurat în două moduri:

- să execute câte un *daemon* pentru fiecare mașină virtuală. Pentru aceasta, se utilizează directiva *Listen* pentru a stabili ce adresă IP va fi asociată daemonului;
- să execute un singur *daemon* pentru toate mașinile virtuale. Pentru aceasta, se utilizează directiva `VirtualHost` descrisă mai sus.

Sabin-Corneliu Buraga este doctorand în Computer Science la Facultatea de Informatică, Universitatea „Al.I.Cuza“ din Iași și poate fi contactat la busaco@infoiasi.ro. Dragoș Acostăchioaie este programator de sistem și aplicații Linux, administrator de rețea la firma BIOSFARM din Iași, putând fi contactat la adresa dragos@biosfarm.ro. ■ 98

Referințe bibliografice

- D. Acostăchioaie, **Administrarea și configurarea sistemelor Linux**, Editura Polirom, Iași, 2002: `http://www.biosfarm.ro/~dragos/admin`
- S. Buraga, **Tehnologii Web**, Editura Matrix Rom, București, 2001: `http://www.infoiasi.ro/~busaco/books/web.html`
- R. Fielding **et al.** (eds.), **Hypertext Transfer Protocol – HTTP/1.1**, RFC 2068, IETF, 1997: `http://www.ietf.org/rfc/rfc2068.txt`
- J. Franks **et al.**, **HTTP Authentication: Basic and Digest Access Authentication**, RFC 2617, IETF, 1999: `http://www.ietf.org/rfc/rfc2617.txt`
- G. Holden, N. Wells, M. Keller, **Apache Server Commentary**, The Coriolis Group, 1999
- G. Mourani, **Securing and Optimizing Linux: Red Hat Edition**, OpenDocs Publishing, 2000: `http://www.openna.com`
- V.V.Patriciu et al., **Securitatea informatică în UNIX și Internet**, Ed.Tehnică, București, 1998
- * * *, **Apache HTTP Server Documentation**: `http://httpd.apache.org/docs`
- * * *, **Apache Week**: `http://www.apacheweek.com`
- * * *, **The Official Red Hat Linux Reference Guide**, Red Hat Inc., Durham, 2001