ATENŢIONARE!

Acest material a fost elaborat de Cristian Oftez, în cadrul S.C. Centrul de Pregătire în Informatică S.A., partener de implementare a proiectului POSDRU /3/1.3/S/5.

Versiunea materialului de instruire: V2.0

11. Securizarea unui server Debian

Securitatea într-un sistem Debian

Sistemele Linux sunt cunoscute în întreaga lume pentru stabilitatea lor şi pentru arhitectura sa construită în jurul unor principii solide ce asigură un nivel de securitate şi integritate a datelor foarte greu de egalat. Motivul pentru care securizarea sistemelor informatice a devenit un subiect atât de important este în primul rând expansiunea Internet-ului, care odată cu nenumăratele facilități pe care le-a oferit, a adus şi o serie de dezavantaje. Principalul punct de interes în ziua de azi îl constituie informațiile electronice, cu diferite caractere – chiar şi cele private – pe care încercăm să le protejăm prin diferitele mijloace disponibile.

Deşi prin natura lor sistemele Linux sunt foarte sigure, acestea nu sunt infailibile. Nu putem vorbi despre un sistem 100% sigur nici măcar la nivel de concept, ci doar de sisteme cu grad ridicat de securitate, care vor necesita foarte mult efort – în principal sub formă de cunoştințe şi mai ales timp – pentru a putea fi compromise. De asemenea, este important de ştiut că măsurile extreme de securitate pot fi foarte intruzive, mergând de la afectarea până la pierderea unor funcționalități. De aceea este nevoie să se stabilească clar un nivel acceptabil de securitate în funcție de rolul sistemului ce trebuie protejat – luându-se în calcul şi consecințele pierderii funcționalității în cazul unor politici de securitate prea restrictive. Un exemplu de astfel de politică ar fi restricționarea fizică a unui server exclusiv la rețeaua locala - ar oferi un nivel foart bun de securitate, eliminând riscul compromiterii din exterior, însă funcționalitățile oferite ar fi restrânse exclusiv la mediul local, total nepotrivit pentru un serviciu de **email** sau server **HTTP**.

Există multe referințe foarte bune în ceea ce priveşte securizarea unui sistem informatic, însă nu le vom enumera pe toate, limitându-ne la aspectele cele mai importante. În acest scop vom defini următorii termeni:

- **Riscul** reprezintă posibilitatea ca un sistem să fie compromis într-un fel sau altul. Riscul poate fi cuantificat pe mai multe nivele, în funcție severitate sau alți factori. Nivelul acceptabil de risc este reprezentat că raportul considerat optim dintre accesabilitate şi funcții pe de-o parte şi gradul politicii de securitate pe altă parte. Nivelul acceptabil variază de la organizație la organizație şi este strict legat de natura acestora cât şi de scopul final.
- **Vulnerabilitatea** reprezintă gradul în care sistemul este protejat.
- **Amenințarea** este de multe ori sursa care generează nivelul de risc, şi poate fi descrisă atât prin acțiuni malițioase cât şi prin evenimente aleatoare cum ar fi defecțiuni sau dezastre naturale. În cazul evenimentelor ce au ca rezultat pierderea de date, studiile recente arată că în marea parte a cazurilor eroarea umană reprezintă principalul factor de risc. Evidențiam astfel faptul că riscurile de securitate sunt reprezentate şi de pierderea datelor, subliniand importanța unei politici de backup bine pusă la punct.

Securizarea generală a unui sistem Linux

Deşi acesta problemă este una mult prea generală pentru a putea fi tratată punctual, există un set de practici recomandate în cadrul procesului de administrare a serverelor Linux care poate reduce semnificativ riscul compromiterii sistemului.

Alegerea de parole complexe este unul dintre cele mai importante aspecte într-un sistem. Date fiind drepturile privilegiate ale contului **root**, este clar că acesta nu trebuie să aibă asociată o parolă „slabă”. Multe companii private sau de stat folosesc aşa numitele **politici de parole,** pentru reglementarea parolelor folosite în sistemele informatice pe care acestea le deţin. Aceste politici descriu diverse aspecte precum:

- **Compoziția unei parole** – în mod optim aceasta trebuie să aibă o dimensiune minimă, să conțină atât caractere speciale, numerice cât şi litere, să conțină elemente de capitalizare (combinații de litere mici şi mari) şi să nu aibă în componență cuvinte sau bucăți de cuvinte ce pot fi regăsite în dicționare. Un exemplu de parolă bună ar fi „**W9]alKi8gj”** – aceasta conține atât cifre, cât şi caractere speciale dar şi combinații între litere mari şi litere mici. La polul celălalt, parole slabe pot fi cele ce reprezintă nume precum „**Maria”** sau cuvinte ce pot fi regăsite în dicționare precum „**pantofi”**.
- **Viața unei parole** este folosită pentru a marca perioada de valabilitate a unei parole. Când termenul unei parole expiră, aceasta va trebui schimbată neapărat. Duratele de viață a parolelor variază în general ca multiplu de 30 de zile, însă această abordare nu este obligatorie. O politică de parole balansată poate prevedea durate de 90 până la 120 de zile. Principalul motiv pentru această regulă este diminuarea riscului de compromitere al parolelor prin atacuri de tip **forța brută** – în care se încearcă în mod iterativ toate combinațiile posibile de parole. Acest tip de atac poate dura mult timp, prin urmare o politică de schimbare periodică a parolelor va face imposibilă aflarea acestora în timp util. O altă problemă asociată cu această abordare o constituie comoditatea utilizatorilor, aceştia încercând de multe ori schimbarea minimală a parolei – de exemplu prin adăugarea unui singur caracter la sfârşitul vechii parole. O politică corectă de parole va interzice alegerea unei noi parole care seamănă prea mult cu vechea parolă.
- **Practici referitoare la parole** specifică regulile ce trebuie urmate în legătură cu parolele alese. Acestea pot să varieze de la interzicerea divulgării parolei (chiar şi colegilor), interzicerea notării parolei pe suport fizic (precum hârtia), până la menționarea sancțiunilor ce vor fi aplicate utilizatorilor care nu respectă politicile alese.

Restricționarea serviciilor sistemului reprezintă o altă abordare foarte eficientă în vederea securizării unui sistem Linux. Primul lucru pe care un administrator trebuie să-l facă este să determine scopul final al server-ului pe care îl întreține. Dacă scopul final este întreținerea unui simplu server **HTTP**, atunci servicii precum **DNS** nu-şi vor avea rostul. Ideea din spatele acestei

abordări este limitarea numărului de servicii pornite la minimul posibil, fără a afecta însă nivelul de funcționalitate dorit. Principalul motiv pentru care limitatarea numărului de servicii pornite creşte nivelul de securitate este că se reduce numărul de vulnerabilități expuse la un moment dat – orice serviciu deschis poate să prezinte diferite vulnerabilități ce pot fi exploatate de către utilizatori rău intenționați. După ce stabilirea exactă a scopului final al server-ului, ar trebui făcută o listă cu toate servicile esențiale astfel încât scopul ales să poată fi atins. Se recomandă dezinstalarea serviciilor care nu vor fi folosite - din moment ce nu fac altceva decât să ocupe spațiu de stocare; acestea pot fi reinstalate foarte uşor în funcție de necesitățile administratorului.

Dezinstalarea programelor inutile este o altă metodă de securizare a serverelor. În cazul distribuției Debian, sunt incluse o mulțime de programe care nu vor avea nici un rol într-un server aflat în producție. Toate programele neesențiale vor constitui un risc, putând fi folosite prin intermediul unor conturi compromise pentru preluarea controlului sistemului. Printre cele mai periculoase programe se numără compilatoare precum **g++** care pot fi folosite pentru construirea unor utilitare care pot fi folosite pentru atacarea altor sisteme. Acestea din urmă pot fi atacuri de tip **denial of service, spam, etc**, deci trebuie vizate utilitarele care ar putea fi folosite în acest scop. Deşi înlăturarea utilitarelor va îmbunătăți nivelul de securitate, acest procedeu nu oferă nici un fel de garanție, ci mai degrabă o tentativă de întârziere - un atacator mai experimentat putând să-şi descarce singur utilitarele de care are nevoie prin intermediul Internet-ului.

Actualizarea sistematică a software-ului folosit este în cele mai multe cazuri soluția optimă pentru prevenirea atacurilor care au la bază exploatarea vulnerabilităților. De regulă actualizările programelor conțin atât imbunatiri legate de funcționalitățile şi performanțele acestora, cât şi soluții la problemele de securitate cunoscute. Se poate realiza şi actualizarea preferențială în cazul în care acest lucru se doreşte, însă administratorul va trebui să fie la curent cu vulnerabilitățile existente pentru a putea lua decizii informate. În acest scop este recomandată abonarea la o listă de email specifică distribuției Debian, strict legată de securitate. Pentru abonarea la această listă, vizitați pagina **http://lists.debian.org/debian-security-announce/,** unde administratorul va primi periodic atenționări.

Restricționarea conturilor de utilizator se poate realiza în scopul limitării pagubelor ce pot rezulta din cauza unor conturi compromise. În acest scop se poate limita abilitatea utilizatorilor de a se loga în sistem la una din console sau folosind protocolul **ssh.** Restricționarea procesului de login se face prin modificarea shell-ului implicit pentru fiecare utilizator dorit în parte; shell-ul implicit este **bash**, şi îl vom schimba într-un shell care nu va permite utilizatorului să execute nimic. Un astfel de shell este **/bin/false** pentru distribuția Debian, şi poate fi setat pentru fiecare utilizator, folosind utilitarul **usemod**: **usermod –s /bin/false [nume_utilizator]**

Comanda anterioară va stabili că shell implicit **/bin/false** eliminând astfel posibilitatea contului respectiv de a se mai loga în consolă - contul utilizatorului nu a fost eliminat însă, ci a rămas activ. Este recomandată restricționarea accesului la consola pentru utilizatorii care nu au nevoie neapărată de facilitățile oferite de aceasta.

Rootkit-uri

Rootkit-urile sunt aplicații al căror scop este mascarea compromiterii unui sistem. Atacatorii pot folosi **rootkit-uri** pentru înlocuirea unor fişiere importante ale sistemului, pentru ascunderea proceselor şi a fişierelor proprii cât şi modificarea fişierelor de jurnalizare. **Rootkit-urile** se pot deghiza sub forma unor module **LKM (Loadeable Kernel Module)** sau a unor programe obişnuite.

Rootkit-uri LKM – aşa cum am menționat în secțiunea corespunzătoare acestora, **LKM-urile** sunt module ce pot fi inserate în mod dinamic în kernel pentru extinderea funcționalității acestuia. Dezavantajul major este că **rootkit-urile** se pot da drept un modul **LKM**, fiind mult mai greu de detectat. Sub această formă acestea pot ascunde procese, directoare sau chiar conexiuni active fără nici un fel de modificare a programelor sistemului, astfel încât administratorul să nu-şi dea seama că un sistem a fost compromis.

Detectarea rootkit-urilor se poate realiza folosind două abordări diferite:

- **Abordarea pro-activa** presupune folosirea unui **LKM** special conceput pentru protejarea sistemului impotriva **LKM-urilor** nocive, sau dezactivarea încărcării modulelor în întregime. Avantajul acestei abordări este faptul că intervenția are loc înainte ca sistemul să fie compromis, prevenind în loc să trateze.
- **Abordarea reactivă** este că nu încarcă sistemul prea mult şi nici nu elimină din funcționalitățile acestuia. Principiul de funcționare este simplu: se compară o tabelă de apelare a sistemului cu o copie a acesteia recunoscută drept „curată”. Dezavantajul major al acestei abordări este faptul că administratorul nu va fi notificat decât după compromiterea sistemului. Detectarea rootkit-urilor folosind această abordare poate fi realizată cu un utilitar numit **chkrootkit**, disponibil în pachetul cu acelaşi nume. Acesta poate fi instalat prin executarea comenzii **apt-get install chkrootkit**, şi va căuta semne ale existenței unor rootkit-uri cunoscute în sistem. După instalare, se recomandă scanări periodice ale sistemului cu acest utilitar şi actualizarea permanentă a acestuia la ultima versiune existentă – astfel încât acesta să fie la curent cu ultimele rootkit-uri şi pentru asigurarea facilităților de detecție corespunzătoare.

Instalarea securizată a pachetelor

O altă amenințare puternică sunt pachetele mascate – aceleaşi pachete specifice distribuției Debian **(.deb)** folosite pentru gestionarea programelor din sistem. Pachetele pot fi modificate astfel încât să se instaleze programe ce pot compromite sistemul – fie versiuni modificate ale unor programe legitime, fie programe care nu au nici o legătură cu numele pachetului instalat, dar care sunt conținute în acesta. Din moment ce marea majoritate a programelor disponibile sunt open-source, acestea pot fi modificate cu uşurință de aproape oricine astfel încât să pretindă că sunt programe legitime şi, de aceea, o politică strictă de gestiune a pachetelor este foarte importantă.

Având în vedere faptul că sistemul **apt** îşi poate procura pachetele din mai multe tipuri de surse, de la unități optice până la servere **ftp** în Internet, este recomandat ca administratorul să nu instaleze niciodată pachete ce provin din surse nesigure. Probabil cea mai periculoase surse sunt cele descărcate de pe serverele **ftp**, în special cele de tipul „third-party” – mai exact arhivele mai puțin cunoscute, care pot fi alterate uşor.

Există însă mecanisme pentru validarea pachetelor, astfel încât să se poată identifica dacă originea pachetelor este legitimă. Metoda cea mai folosită se numeşte **secure apt** şi se bazează pe un sistem de criptare bazat pe chei publice şi private, cu generarea unor amprente digitale unice pentru fiecare pachet. Se poate alcătui o listă de chei globală pentru un sistem, prin care se marchează sursele legitime, eliminând în mare măsură riscul instalării unor pachete ce pot compromite sistemul.

CHROOT este un concept des folosit în lumea sistemelor Linux, şi presupune modificarea directorului root (atenție: este vorba de directorul / al sistemului de fişiere, şi nu de contul de utilizator **root**) în mod individual pentru anumite procese (şi implicit şi pentru procesele fiu ale acestora). Se elimină în acest mod posibilitatea de accesare a oricărui alt director în afara celui stabilit în acest scop, crescând considerabil gradul de securitate al unui sistem – în cazul în care un serviciu va fi compromis de către un atacator, acesta nu va avea acces la părți esențiale ale sistemului de operare.

Procedeul de **chrooting** are mai multe întrebuințări, precum crearea unor medii de testare oarecum separate de restul sistemului, controlul dependințelor sau pur şi simplu pentru îmbunătățirea securității.

Deşi constituie un mecanism defensiv foarte bun, acesta are totuşi o serie de limitări:

- Invalidarea de către userul **root** – mecanismul nu este conceput pentru a funcționa şi pentru contul **root**. Acesta este singurul cont care are dreptul de a implementa acest mecanism.
- Greutate în configurare – procedeul de **chroot** variază de la proces la proces şi nu există multe aspecte comune. Din acest motiv, aplicarea mecanismului pentru mai multe servicii diferite poate fi relativ dificilă,

datorită funcționalităților diferite ale acestora – fiecare serviciu va necesita un set diferit de fişiere, fişiere de tip dispozitiv şi librării pentru a funcționa. Aceste resurse vor trebui să fie disponibile şi accesibile în noul mediu pentru că procesele să funcționeze.

- Mecanismul **chroot** nu a fost conceput pentru restricționarea resurselor precum dispozitivele de intrare/ieşire, mediile de stocare sau modul de utilizare al procesorului. Din acest motiv, un serviciu compromis – chiar şi sub influența mecanismului **chroot** – ar putea fi folosit pentru atacuri de alte tipuri asupra sistemului.

Chroot reprezintă o metodă de securizare foarte puternică, deşi există o serie de premize în care poate fi ocolită – orice utilizator cu drept de **root** va putea dezactiva acest mecanism. Un alt dezavantaj al acestei proceduri este că de regulă, o mare parte din fişierele funcționale necesare diferitelor procese trebuie duplicate în noile locații. Pentru a preveni scenariul în care mediul **chroot** este dezactivat de către **root**, serviciile configurate să funcționeze în acest regim vor trebui configurate astfel încât acestea să nu ruleze cu drept de **root**. Există utilitare dedicate pentru facilitarea configurării de medii **chroot** precum **makejail**. Acesta poate fi instalat prin executarea comenzii următoare: **apt-get install makejail**. Acest program are rolul de a încerca să intuiască toți parametrii necesari pentru construirea unui **chroot jail**. Pentru mai multe detalii despre pachetul menționat anterior, apelați comanda **man makejail** după instalarea pachetului.

Securizarea rețelei

O altă vulnerabilitate importantă o reprezintă mediul de rețea, în cadrul căreia mesajele vor fi transmise în clar în mod implicit. Oricine cu acces fizic la o rețea locală va putea intercepta tot traficul ce se desfăşoară în această.
Acest fenomen reprezintă un risc deosebit de mare în prezent, în mare parte datorită transmiterii în clar a parolelor şi a conturilor de utilizator – acest fenomen nu este neapărat datorat neglijenței utilizatorilor, ci mai degrabă felului în care anumite protocoale funcționează. Din acest motiv se recomandă evitarea instalării şi folosirii serviciilor ce transmit datele şi mesajele utilizatorilor în clar, precum telnet, NIS sau FTP.

Configurarea unui firewall

Programul **iptables** este foarte des folosit în mediul Linux. Acesta oferă funcții de filtrare a pachetelor, NAT (Network Address Translation) şi „mangling”. Cel mai des însă, **iptables** este folosit ca firewall şi pentru NAT. Configurarea poate fi greoaie pentru începători, de aceea se recomandă folosirea unor utilitare destinate configurării grafice a firewall-ului, precum **firestarter**. Acest program poate fi instalat cu ajutorul comenzii **apt-get install firestarter** şi rulat cu **firestarter**.

Principiul de funcționare **iptables** este simplu: acesta oferă administratorului posibilitatea creării unor şiruri de reguli legate la traficul de pachete TCP/IP. Numele utilitarului provine de la modul în care acesta funcționează: împarte şirurile de reguli în nişte aşa numite tabele (tables), fiecare asociată cu un tip diferit de prelucrare a pachetelor TCP. Există cinci şiruri (chains) predefinite, care au asociată o politică (policy). Politica specifica ce se va întâmpla cu pachetul în momentul în care acesta va ajunge la sfârşitul unui şir – de exemplu DROP, când acesta va fi refuzat. Cele cinci şiruri sunt:

- **PREROUTING** – Orice pachet va ajunge aici înaintea unei operații de redirectare
- **INPUT** – Pachetele vor fi distribuite local
- **FORWARD** – Toată pachetele care au fost redirecționate şi nu corespund pentru distribuirea locală vor ajunge în acest şir.
- **OUTPUT** – Pachetele transmise de sistem vor ajunge în acest şir.
- **POSTROUTING** – Reprezintă momentul imediat următor deciziei de redirecționare al unui pachet, oarecum complementar pentru şirul **PREROUTING**.

Fiecare regulă dintr-un şir va descrie un mod specific de funcționare, şi poate conține şi parametri precum **destinație** sau **sursă**. Pe măsură ce un pachet înaintează printr-un şir, fiecare regulă din acel şir va fi interpretată sistematic pentru pachetul în cauză, cu două finalizări posibile: fie pachetul examinat în acel moment nu corespunde regulei curente (caz în care se va aplica regula imediat următoare), fie pachetul va corespunde regulei actualea şi se va lua o decizie în privința acestuia - pachetul va fi blocat, sau va merge mai departe. Acest sistem oferă o serie foarte largă de posibilități de configurare, administratorul având astfel control deplin asupra pachetelor ce tranzitează sistemul.

Configurarea prin intermediul consolei, configurarea **firewall-ului** se poate realiza cu ajutorul utilitarului **iptables** şi o serie specifică de parametrii.

Printre cei mai importanți parametrii amintim:

- **-A** va specifica şirul în care va fi adăugată noua regulă
- **-L** va afişa pe ecran toate regulile actuale
- **-p** specifică protocolul la care se referă regulă
- **--dport** specifică portul destinației
- **-j** specifică acțiuni posibile: acestea pot fi

    - **ACCEPT –** va accepta pachetul şi va înceta să prelucreze regulile din acest şir.
    - **REJECT –** va refuza pachetul şi va notifica expeditorul asupra acestui aspect
    - **DROP –** va refuza pachetul fără nici un fel de atenționare.
    - **LOG –** va jurnaliza acest pachet şi va continua să prelucreze regulile din şirul actual.
- **-I** va însera o regulă pe o poziție specifică într-un şir. Folosit în modul următor, acesta va însera regulă pe poziția a 10-a din şirul **INPUT**: **-I INPUT 10**.
- **-s** specifică adresa (opțional cu mască) sursă
- **-d** specifică adresa (opțional cu mască) destinație

În continuare vom examina câteva exemple cu descrierea funcțională a acestora:

- **#iptables –A INPUT –p tcp –dport 22 –j ACCEPT** va avea ca efect acceptarea tuturor pachetelor pe portul 22, folosind protocolul TCP care vor intra în sistem.
- **#iptables –A INPUT –s 199.200.201.202 –j DROP** va avea ca efect respingerea silențioasă a tuturor pachetelor ce intră în sistem, pe orice port de la adresa 199.200.201.202.
- **#iptables –A INPUT –p ICMP –j DROP** va refuza orice pachet folosind protocolul **ICMP** (ping) care intră în sistem.
- **#iptables –L** va afişa regulile existente

Configurarea sistemului astfel încât acesta să încarce un set de reguli definit în mod automat la pornire, se realizează în doi paşi. Primul pas presupune configurarea regulilor şi salvarea lor într-un fişier prin intermediul comenzii **iptables-save > /etc/nume_fişier**. Această comandă va avea ca efect salvarea regulilor definite în fişierul menționat. Pasul doi presupune modificarea fişierului **/etc/network/interfaces** astfel încât fişierul creat la pasul anterior să fie încărcat în mod automat pentru o interfață anume. În acest scop trebuie editat fişierul **/etc/network/interfaces** şi adăugată următoarea linie pentru interfața dorită: **pre-up iptables-restore < /etc/nume_fişier**. În mod alternativ se poate folosi comanda **iptables-restore < /etc/nume_fişier** în cadrul unuia din scripturile de pornire ce pot fi găsite în **/etc/init.d**.

Securizarea serviciilor

O altă secțiune importantă în ceea ce priveşte securitatea, este cea dedicată securizării serviciilor ce rulează pe un sistem Linux. Procesul de securizare poate fi împărțit în două secțiuni: limitarea zonelor de influență a serviciilor şi configurarea acestora în aşa fel încât să nu poată fi utilizate în alte scopuri decât cele pentru care au fost concepute.

Securizarea serviciului SSH

Serviciul SSH, menționat în capitolul de administrare generală reprezintă alternativa sigură prin care utilizatorii unui sistem Linux se pot conecta la acesta la distanță, prin protocolul TCP. Un prim pas în vederea securizării sistemelor, păstrând facilitățile de **remote login** este înlocuirea serviciului **telnet** cu **ssh**, asigurându-se astfel confidențialitatea conexiunilor. În continuare vom analiza o serie de schimbări în configurația server-ului **ssh** care pot duce la o îmbunătățire a serviciului - fişierul de configurare implicit al server-ului descris este **/etc/ssh/sshd_config** şi în marea parte a cazurilor trebuie creat.

- **PermitRootLogin no** – nu va permite autentificarea utilizatorilor drept **root** (nici măcar cu parola corectă), limitând astfel posibilitatea compromiterii parolei de root prin atacuri de tip forța brută.
- **PermitEmptyPasswords no** – va interzice folosirea parolelor goale.
- **AllowUsers user1 user2** … **-** descrie o serie de utilizatori care vor avea acces la acest serviciu. Această opțiune este una exclusivă.
- **Port [număr] –** va schimba port-ul pe care serviciul va răspunde la conexiuni. Deşi acest tip de securitate este unul foarte bun (nepermițând atacatorilor să-şi dea seama cu uşurință ce servicii rulează pe un anumit sistem), necesită cunoştința utilizatorilor ce vor folosi acest serviciu, şi ca urmare nu va putea fi aplicat cu succes în scenarii extinse.
- **ListenAddress [adresa_ip]** – permite restricționarea serviciului la o interfață ethernet anume. Ca urmare directă, serviciul **ssh** nu va fi disponibil decât prin cadrul interfeței menționate.
- **Banner [nume_fişier]** – va afişa fişierul menționat drept **MOTD** (Message of the day), putând să fie folosit ca un avertisment.

Securizarea serviciului HTTP

Server-ul Apache descris în secțiunile anterioare reprezintă ca orice alt serviciu pornit, o posibilă poartă de intrare în sistem pentru atacatori. Creşterea nivelului de securitate pentru serviciul **HTTP** se poate realiza în mai multe feluri, precum restricționarea accesului la fişierele ce alcătuiesc site-urile găzduite, restricționarea porturilor şi adreselor IP la care server-ul va putea fi accesat, sau chiar configurarea unui mediu **chroot** pentru serviciul httpd.

Fişierele ce alcătuiesc site-urile găzduite prin serviciul respectiv ar trebui să fie accesibile de orice user din sistem, dar în niciun caz modificabile de userul **www-data** sau de oricine în grupul cu acelaşi nume. Userul menționat anterior este contul cu drepturile căruia procesul server-ului Apache rulează, şi cel mai probabil va reprezenta şi drepturile de care va dispune un eventual atacator care a reuşit să compromită serviciul **HTTP.** Din acest motiv, se recomandă ca fişierele site-urilor să nu poată fi modificate de către userul **www-data,** dar şi modificarea site-ului implicit care ar putea furniza informații nedorite eventualilor atacatori – precum versiunea de apache folosită sau distribuția folosită.

Securizarea serverului DNS bind9

Din motive de securitate, se recomandă că serviciul să urmeze procesul **chrooting**. Acesta este o tehnică cunoscută de securizare şi presupune schimbarea directorului root (**/**) pentru instanța procesului responsabil şi copiii săi. În urmă acestui procedeu procesul în cauză va considera directorul setat ca fiind rădăcina sistemului de fişiere şi prin urmare nu va avea acces în afara acesteia. Pentru a efectua această schimbare va fi necesarea mutarea efectivă a fişierelor de configurare ale serviciul într-o locație care este sigură, schimbarea fişierului de configurație al daemon-ului, simularea unor fişiere de tip dispozitiv speciale şi crearea unor legături simbolice astfel încât instanța server-ului să poată găsi fişierele de configurație. Configurarea unui mediu **chroot** pentru server-ul **bind9** este printre cele mai simple proceduri de acest tip, şi de aceea reprezintă exemplul ideal (şi datorită faptului că este unul dintre cele mai vulnerabile servicii):

Pasul 1. Crearea noilor directoare se poate face într-o locație precum **/var/named** şi în scopul acestui exercițiu, vom alege această locație:

```
mkdir –p /var/named/etc
mkdir –p /var/named/dev
mkdir –p /var/named/var/run/bind/run
mkdir –p /var/named/var/cache/bind
```

Pasul 2. Editarea fişierului de configurație presupune modificarea fişierului **/etc/default/bind9** şi schimbarea liniei **OPTIONS** cu nouă linie

```
OPTIONS=”-u bind –t /var/named”
```

Pasul 3. Mutarea directorului de configurare bind în noua locație, şi crearea unei legături simbolice către noua locație astfel încât daemon-ul să poată găsi fişierele de configurare şi să eliminăm orice fel de probleme în eventualitatea unei actualizări a programului:

```
mv /etc/bind /var/named/etc
ln –s /var/named/etc/bind /etc/bind
```

Pasul 4. Simularea dispozitivelor speciale null şi random şi schimbarea setului de permisiuni se poate realiza executând următoarele comenzi. Programul **mknod** se foloseşte pentru crearea fişierelor de tip dispozitiv.

```
cd /var/named/dev
mknod null c 1 3
mknod random c 1 8
chown bind:bind –R ../etc/bind
chown bind:bind –R ../var/*
chmod a+rw *
```

Pasul 5. Adăugarea facilităților de jurnalizare se realizează prin crearea fişierului **/etc/rsyslog.d/bind-chroot.conf** şi adăugarea următoarei linii în acesta:

```
$AddUnixListenSocket /var/named/dev/log
```

Pasul 6. Repornirea serviciilor DNS şi de jurnalizare se va face conform comenzilor de mai jos:

```
/etc/init.d/rsyslog restart
/etc/init.d/bind9 start
```

În acest moment, avem un server **bind9** configurat astfel încât să ruleze într-un mediu **chroot**. Şi alte servicii pot fi modificate în acest scop, însă pentru a menține dimensiunea acestui manual cât mai mică, vom trata doar acest serviciu, acesta fiind unul dintre cele mai vulnerabile servicii existente pe sistemele Linux – în mare parte datorită complexității sale.