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

10. Servicii

Motivul pentru care tehnologia calculatoarelor a evoluat la stadiul în care se află în ziua de azi se datorează în mare parte facilităților pe care aceasta a putut să le ofere şi datorită capacității de distribuire a informațiilor la cât mai mulți utilizatori, într-un timp cât mai mic cu putință.

La baza acestor facilități au stat serviciile, sau mai exact programele ce rulează pe sistemele informatice, ce fac posibilă oferirea unei game largi de utilități unor utilizatori. Datorită naturii serviciilor oferite, unde activitatea principală este reprezentată de schimbul de informații (de diverse tipuri), marea parte a acestora se raportează – dar nu se rezumă exclusiv – la mediul de comunicare, sau mai exact la rețele de calculatoare (printre care şi cea mai extinsă rețea de calculatoare din lume, cunoscută sub numele de Internet). Un bun exemplu de serviciu oferit deasupra infrastructurii de comunicare existente îl reprezintă paginile web, care folosesc rețeaua internet pentru schimbul de informații.

În orice serviciu oferit există două entități principale: cea care oferă serviciul (server) şi cea care beneficiază de serviciul oferit (client). În cazul exemplului anterior, paginile web (cu care aproape orice utilizator de calculatoare este familiar) sunt oferite de un calculator (numit **server**) printr-un protocol bine stabilit (numit **HTTP** sau **Hypertext Transfer Protocol**), către un alt calculator care a apelat serviciul respectiv (numit **client**).

Mai exact, atât **server-ul** cât şi **client-ul** nu sunt altceva decât nişte programe cu funcții bine stabilite, menite să comunice unul cu celălalt (de regulă printr-un protocol) pentru a oferi un serviciu – în cazul precedent, serverul este reprezentat de un **server HTTP** care are rolul de a transfera pagina către client, care nu este altceva decât un **web browser** precum Internet Explorer sau Mozilla Firefox.

Serviciile informatice nu se rezumă la un singur sistem de operare, şi de regulă clientul şi server-ul sunt independente din punct de vedere al platformei pe care rulează – concret, deşi marea majoritate a serverelor HTTP rulează sub Linux, paginile web pot fi accesate şi de către utilizatorii de Windows sau MacOS; acest lucru este posibil datorită protocolului de comunicare folosit, care este independent de sistemul de operare.

Serverele care oferă un serviciu (cunoscute şi sub numele de **daemons**) reprezintă programe care rulează de regulă în fundal, oferind un nivel de interacțiune locală foarte scăzută, spre deosebire de programele convenționale. Acestea sunt menite să lucreze în mod autonom pe fundal, în baza unor configurații prestabilite, şi să ofere un serviciu unui client în momentul în care apare o cerere pentru acel serviciu. Serverul HTTP de exemplu, aşteaptă cereri de la browsere web (de regulă pe portul 80) şi le transferă acestora paginile web cerute.

Administrarea generala a serviciilor Linux

În cadrul sistemelor de tip Unix, programele de tip **daemon** au nume specifice, astfel încât să facă identificarea lor uşoară. De regulă, acestea au nume care se termină cu litera „d” (prescurtarea de la **daemon**): httpd – pentru serverul web, sau sshd pentru serviciul ce oferă posibilitatea conectării remote prin intermediul unui „**secure shell**”.

În ceea ce priveşte marea majoritate a serviciilor în medii de tip Unix, părintele proceselor ce reprezintă **daemon**-urile este (de regulă) procesul Init (cu PID = 1). Procesele devin servicii în mod normal după ce urmează procedeul **fork** descris în capitolul aferent acestora, şi după ce părintele lor îşi termină execuția (fapt ce are ca efect „adoptarea” procesului de către Init), principalul motiv pentru acest mod de funcționare fiind disocierea daemon-ului de o consolă.

Sistemele Linux pornesc serviciile instalate odată cu procesul de boot, dat fiind faptul că multe servicii au ca scop îndeplinirea unor funcții exclusiv locale, precum programarea unor sarcini la anumite ore sau monitorizarea dispozitivelor ataşate.

În mod uzual, serviciile reduc interacțiunea locală cu utilizatorii sistemului la minim (sau chiar deloc), astfel încât să se izole de factori externi, dar permit schimbarea configurației acestora prin intermediul unor fişiere bine stabilite de configurare. De regulă, orice schimbare în fiserele de configurare ale unui serviciu va intra în vigoare după repornirea serviciului respectiv.

Listarea serviciilor şi pornirea automată

Orice serviciu în Linux este de fapt un program care aşteaptă să primească cereri (de regulă prin intermediul protocolului TCP/IP), şi cărora le răspunde în funcție de tipul de serviciu de care este responsabil.

Pe departe, cea mai simplă metodă de gestiune a serviciilor sistemului poate fi utilizată prin intermediul interfeței grafice, mai exact, prin intermediul utilitarelor incluse în GNOME. Această abordare este în principiu utilă indiferent de distribuția utilizată, fiind strict legată doar de utilizarea mediului desktop GNOME.

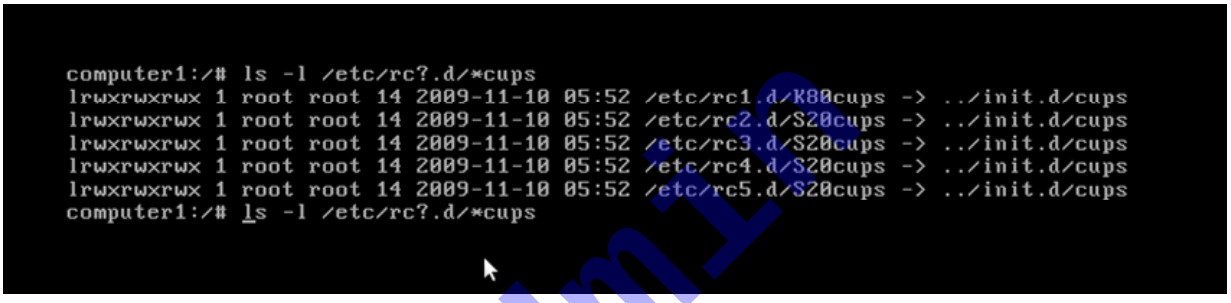
În secțiunea dedicată proceselor, am aflat mai multe despre directoarele ce conțin script-urile de inițializare pentru fiecare run-level în parte (**/etc/rc*.d/**). Serviciile pot fi pornite automat, prin intermediul script-urilor de inițializare menționate, sau pot fi pornite manual, prin executarea comenzii specifice acelui serviciu.

Pornirea sau oprirea manuală (sau orice altă modificare) a unui serviciu implică de regulă, folosirea unui script specific serviciului dorit, urmat de un parametru ce reprezintă starea în care se doreşte acel serviciu. Marea majoritate a script-urilor legate de servicii pot fi găsite în directorul **/etc/init.d/**. Un exemplu bun este cazul în care se doreşte repornirea (oprirea urmată de pornirea imediată – pentru reîncărcarea fişierului de configuratie de exemplu) serviciului responsabil de funcționarea în rețea, prin executarea

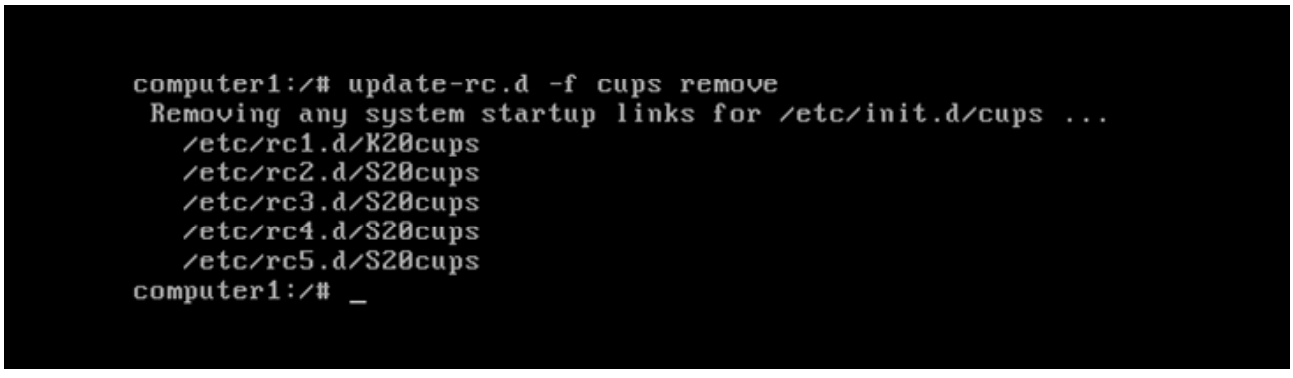
comenzii **/etc/init.d/networking restart**. Similar, parametrul restart poate fi înlocuit cu **start** sau **stop**.

Modificarea listei de servicii ce pornesc automat, odată cu sistemul, poate fi realizată în mai multe feluri, însă, tradițional, cea mai comună implică folosirea utilitarului **update-rc.d**. Acesta nu face altceva decât să elimine sau să adauge legături către serviciile dorite în directoarele rc*.d (realizabil şi manual). Ca alternativă în mod text, la utilitarul update-rc.d, utilizatorul poate să instaleze şi să utilizeze un program numit **sysvconfig** ce oferă o interfață ceva mai uşor de folosit.

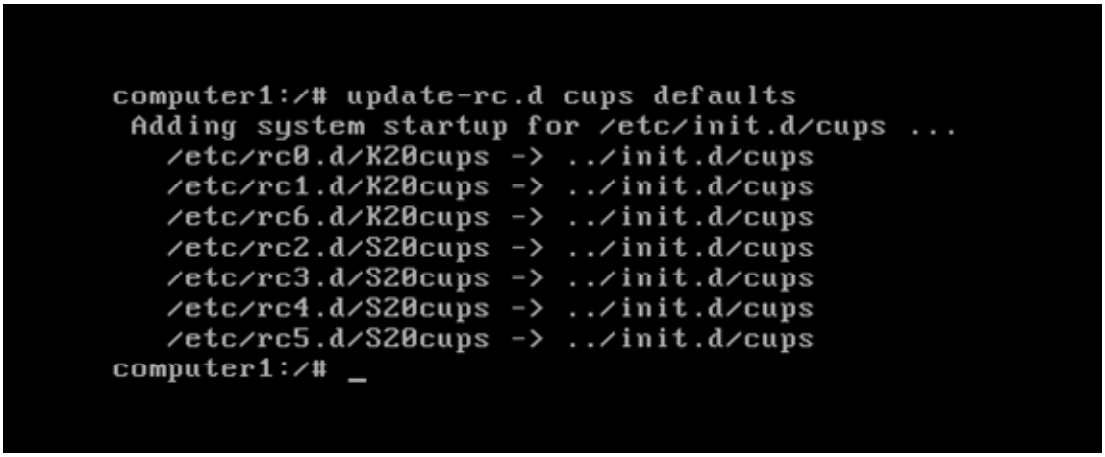
Pentru a detalia mai bine modul în care trebuie folosit utilitarul **update-rc.d**, putem lua exemplul serviciului **cups** (prescurtarea de la Common Unix Printing System), care este responsabil cu tipărirea documentelor, fiind instalat implicit şi configurat să pornească automat pe majoritatea distribuțiilor Linux. Dacă ar fi să examinăm toate script-urile responsabile cu serviciul **cups**, am observa că acesta este prezent în mai multe locuri:



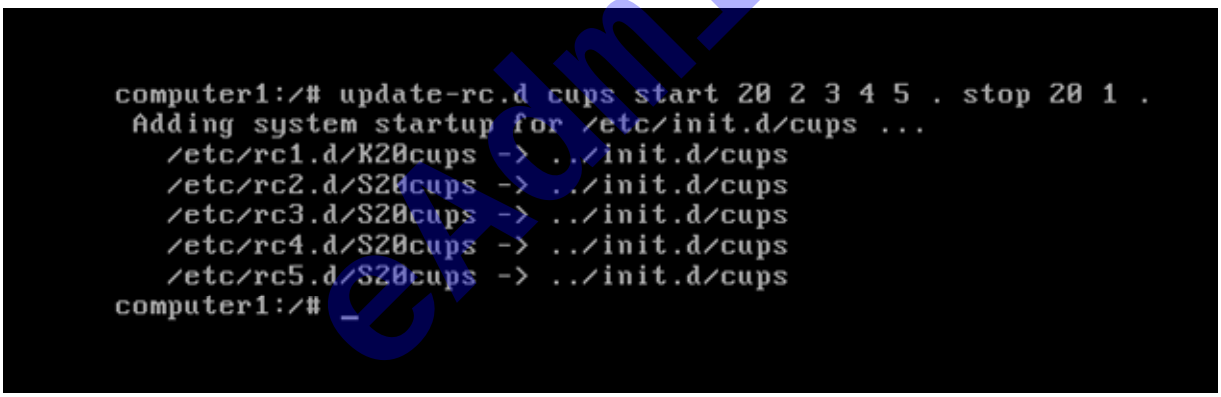
Legături către script-ul responsabil cu serviciul **cups** se pot găsi în toate cele 5 run-levels, cu mențiunea că cele al căror nume începe cu **K** (kill) se referă terminarea serviciului, iar al căror nume începe cu **S** sunt responsabile pentru pornirea serviciului respectiv. Dacă dorim să excludem serviciul de la inițializarea automată, odată cu pornirea sistemului, ar trebui să apelăm comanda **update-rc.d -f cups remove** (parametrul -f forțează eliminarea, cups reprezintă numele serviciului țintă, iar remove specifică eliminarea serviciului din procesul de inițializare).



Pentru a reface schimbarea şi a configura serviciul să pornească automat, odată cu sistemul, trebuie apelată comanda **update-rc.d cups defaults**.

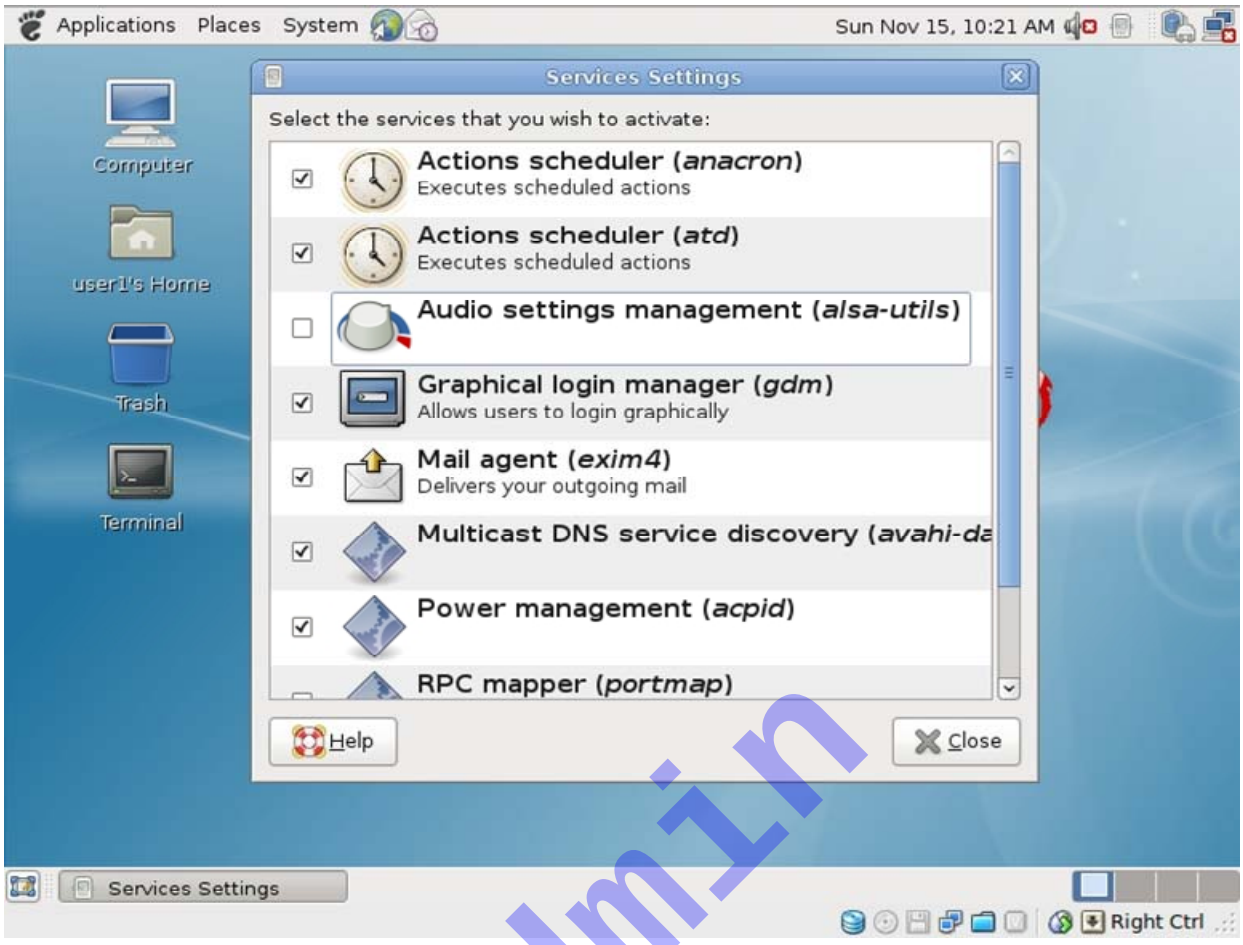


Diferența este că script-ul a fost reconfigurat în mai multe locații decât în mod inițial. Aşa cum am menționat anterior, numărul prezent în denumirea script-ului de pornire sau oprire reprezintă prioritatea sa de execuție (fişierele fiind ordonate după nume), iar prima literă (K sau S) simbolizează dacă scriptul este de oprire sau de pornire. Pentru a aduce configurația la nivelul la care era inital, trebuie analizată prima poză, în care observăm ce tipuri de script-uri erau şi ce prioritate aveau. Comanda necesară este: **update-rc.d cups start 20 2 3 4 5 . stop 20 1 .**



Comanda de mai sus poate fi explicată foarte simplu în modul următor: cups reprezintă numele serviciului, „start 20 2 3 4 5 .” simbolizează necesitatea configurării scriptului pentru pornire cu prioritatea 20 pe nivelele 2,3,4 şi 5, iar punctul simbolizează sfârşitul secvenței. Secvența următoare de parametru este similară, dar se referă la script-uri de oprire, cu prioritatea 20 numai pentru nivelul 1.

Dacă utilitarul **update-rc.d** permite o configurare în detaliu al fiecărui serviciu în parte, cel din modul grafic, inclus în suita GNOME este ceva mai simplu. Acesta poate fi găsit în mediul grafic în meniul **System -> Administration -> Services**.



Indiferent de metoda folosită, rezultatul însă trebuie să fie acelaşi mereu, iar Linux se diferențiază de alte sisteme de operare prin gradul foarte mare de libertate pe care îl oferă. Specific multor utilitare de configurare din suita de aplicații GNOME, utilizatorul ar putea fi rugat să introducă parola de root pentru a face schimbări importante în configurația sistemului.

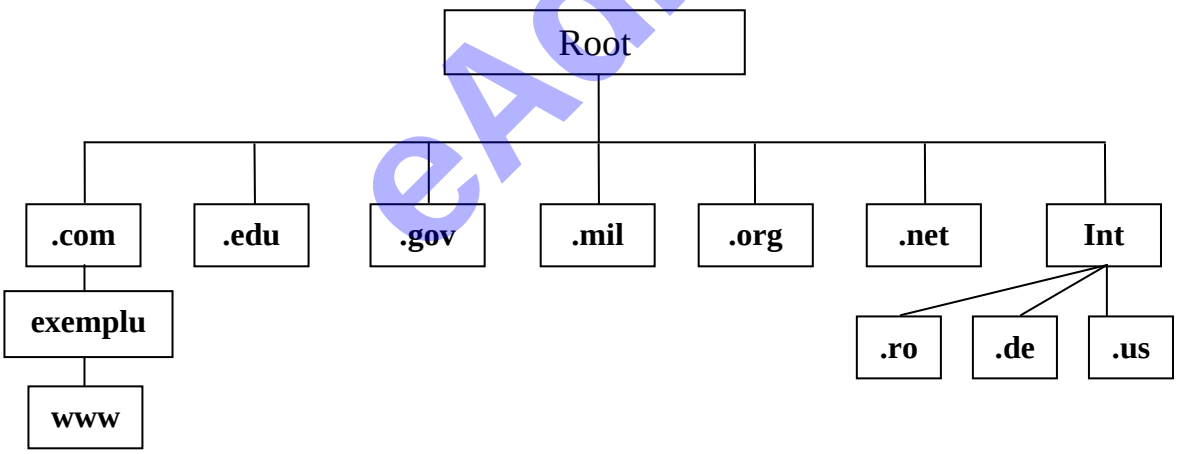
OBSERVAȚIE: Pe lângă directoarele corespunzătoare pentru fiecare run-level, mai există şi script-ul **/etc/rc.local**, care este executat înaintea oricăror altor script-uri, care poate fi folosit pentru executarea unor comenzi dorite, indiferent de run-level. În continuare vom prezenta metodele de instalare şi configurare a serviciilor importante, inclusiv o serie de exemple. În cazul exemplelor, se presupune că toate comenzile vor fi lansate cu privilegile contului **root**.

Instalarea şi configurarea serviciului DNS

DNS (Domain Name Server) este serviciul responsabil pentru rezolvarea numelor de internet, uşor de reținut de către utilizatori, în adrese IP, făcând astfel posibilă comunicarea între toate dispozitivele conectate la o rețea. Sistemul funcționează folosind o structură ierarhică de nume, şi asociază numele de domenii (de exemplu **google.com**) cu adrese binare de 32 de biți (în cazul adreselor **IPv4**) sau de 128 de biți (dacă este vorba de adrese **IPv6**).

Sistemul DNS funcționează prin distribuirea responsabilității asupra anumitor domenii prin intermediul serverelor numite **authoritative name servers (ANS)**. Aceste servere sunt responsabile de nişte domenii proprii şi pot transfera responsabilitatea asupra subdomeniilor către alte servere ANS. Acest mecanism de funcționare a permis astfel descentralizarea serviciului de DNS, un aspect foarte important în rețelele foarte mari (cum este Internet-ul) şi a consolidat calitatea serviciului printr-un sistem tolerant la defecte.

Structura spațiului de nume a fost organizată de la început ca un arbore. Orice nume de domeniu este compus din mai multe părți numite etichete, separate prin puncte. Eticheta cea mai din dreapta se numeşte top-level domain – în cazul domeniului **www.google.com**, domeniul de top este .**com**. Continuând în stânga, fiecare subdomeniu reprezintă o subdiviziune a domeniului precedent, teoretic acest proces de subdiviziune putând să suporte până la 127 de nivele independente.



În desenul de mai sus se poate observa domeniul **exemplu**. Acesta aparține domeniului **.com** şi are la rândul său unul sau mai multe subdomenii – **www** în acest caz.

Datorită faptului că toate numele apartiand unui domeniu trebuie să fie unice, întregul sistem trebuie centralizat într-un singur punct, care să asigure unicitatea tuturor subdomeniilor sale. Deşi este foarte importantă existența unei autorități în privința asigurării unicității numelor dintr-un domeniu, centralizarea întregii baze de date conținând informații despre **toate** domeniile şi subdomeniile existente (sau doar cele de top-level) nu este posibilă din mai multe motive - ar fi foarte greu de gestionat o bază de date

atât de mare, ar elimina caracterul tolerant la defecte al sistemului, iar întregul sistem ar trebui să aibă o capacitate foarte de mare de lucru, astfel încât să poată oferi acest serviciu tuturor utilizatorilor existenți.

Ca urmare a acestor necesități, fiecare domeniu are un server DNS asociat, care va răspunde la cererile pentru domeniul respectiv şi în general va avea şi un administrator desemnat să gestioneze înregistrările DNS din acel server. Unul din avantajele acestei distribuții este că deşi serviciul este oferit prin intermediul foarte multor servere individuale, întregul sistem are un comportament unitar.

Comportamentul normal al unui server DNS poate fi simplificat la două funcții majore: tratarea cererilor de rezolvare a numelor în adrese IP de la programe sau alte servere DNS. În momentul în care server-ul va primi o cerere, acesta poate să răspundă în 3 feluri diferite:

- Va răspunde direct cu adresa IP căutată în cazul în care server-ul în cauză are înregistrarea respectivă
- Va demara un proces repetitiv prin care va încerca să contacteze alte servere DNS în tentativa de aflare a adresei IP, până când va reuşi să răspundă cu adresa IP căutată.
- Va eşua căutarea fie printr-un răspuns cu adresa unui alt server, care este mai probabil să conțină mai multe informații, fie pentru că numele căutat nu există.

Probabil cea mai importantă caracteristică a servelor DNS este fiabilitatea. Datorită faptului că există multe servere pentru fiecare nivel ar ierarhiei menționate anterior, există alternative în cazul în care unul dintre ele nu funcționează.

O altă caracteristică importantă a unui server DNS este abilitatea de **caching**, mai exact procesul prin care memorează temporar toate adresele IP prelucrate în urma unei cereri. Timpul de memorare este stabilit de atributul **TTL** (Time to Live), şi este ataşat în mesajul în care a primit IP-ul respectiv.

În cadrul sistemelor Linux, cel mai răspândit server DNS se numeşte **bind9** ce se poate instala în cadrul distribuției Debian rulând comanda: **apt-get install bind9**

Odată instalat, server-ul va fi pornit automat. Din moment ce urmează să configurăm acest daemon, acesta va trebui oprit prin rularea comenzii: **/etc/init.d/bind9 stop**

Observăm că în urma instalării daemon-ului, a fost creat script-ul de inițializare în directorul **/etc/init.d**. Vom analiza în continuare atât configurația server-ului **bind** cât şi câteva exemple de configurare particularizată.

Server-ul **bind** îşi desfăşoară activitatea în baza fişierelor de configurare şi a fişierelor de zonă. Fişierele de configurare conțin parametrii generali care descriu modul în care daemon-ul respectiv va funcționa, pe când fişierele de zonă sunt folosite pentru descrierea domeniilor despre spațiile de nume.

Fiecare fişier de zonă conține informații folosite pentru atribuirea unor identități pentru sisteme individuale, şi opțional poate conține şi directive.

Să analizăm unul dintre fişierele de zonă implicite, folosite pentru identificarea **localhost-ului** ce poate fi găsit în directorul de configurare (în mod implicit acesta este **/etc/bind/db.local**):

```
$TTL    604800
@       IN      SOA     localhost. root.localhost. (
                              2         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@       IN      NS      localhost.
@       IN      A       127.0.0.1
@       IN      AAAA    ::1
```

După prima examinare, putem observa că fişierele de zonă folosesc un format bine stabilit pentru introducerea intrărilor. Se poate observa de asemenea că există mai multe tipuri de înregistrări cu diferite roluri:
Intrarea de tip „SOA” (Start of Authority) va fi prima înregistrare dintr-un fişier de zonă, şi declarară informații esențiale despre un spațiu de nume. Formatul înregistrărilor de tipul Start of Authority este:

```
@    IN    SOA <server_nume> <email_hostmaster> (
                <număr_serial>
                <time-to-refresh>
                <time-to-retry>
                <time-to-expire>
                <minimum-TTL> )
```

Fiecare dintre atributele specificate are un rol special: **<număr_serial>** acționează ca un indicator al versiunii fişierului, astfel încât server-ul DNS să ştie că această zonă trebuie actualizată în cazul în care a fost modificată; **<time-to-refresh>** transmite tuturor serverelor „slave” timpul pe care trebuie să îl aştepte înainte să întrebe serverul „master” dacă au intervenit schimbări asupra zonei respective; **<time-to-retry>** transmite serverelor „slave” intervalul pe care trebuie să-l aştepte înaintea unei noi cereri de refresh (câmpul **număr_serial** este folosit de către serverele slave pentru a determina momentul în care să ceară informații despre o zonă); **<minimum-TTL>** specifică timpul minim pe care alte servere de nume trebuie să memoreze zona respectivă.(intervalul este măsurat în secunde).

Rolul unei înregistrări SOA este de a indica faptul că server-ul DNS pe care îl reprezintă este într-adevăr cea mai bună sursă de informații despre domeniul în cauză. De asemenea, numele server-ului menționat în cadrul înregistrării SOA **trebuie** să figureze neapărat şi cu o înregistrare de tip **NS**.

Intrarea de tip „NS” (sau Name Server) are rolul de a anunța serverele cu autoritate pentru o anumită zonă. Orice zonă trebuie să aibă neapărat cel puțin o înregistrare de acest tip, deoarece scopul său principal este identificarea numelor serverelor DNS şi prin urmare, disponibilitatea unor nume de domenii. Formatul înregistrărilor de acest tip este:

```
(nume)    IN    NS    (server_de_nume)
```

Intrarea de tip „A” este folosită pentru asocierea directă între o adresă IP şi un nume. Formatul standard pentru acest tip de înregistrare este:

```
(nume)    IN    A     (adresa_ip)
```

În exemplul anterior, se poate vedea o asociere între adresa IP **127.0.0.1** şi numele **localhost**. În acest caz, caracterul „@” folosit pentru definirea conceptului de **origine**, în cazul de față, numele zonei menționate în fişierul de configurare – în acest exemplu, numele de zonă fiind **localhost**, caracterul „@” ar fi putut fi înlocuit cu numele „localhost” fără să existe nicio diferență funcțională, însă nu se recomandă folosirea fără o înțelegere bună a directivei $ORIGIN.

Intrarea de tip „CNAME” (sau Canonical name) este folosită pentru asocierea între două nume – cunoscute şi drept alias. Un exemplu bun este înregistrarea următoare

```
www       IN    CNAME    server
```

care descrie crearea unui alias pentru numele **server,** mai exact alias-ul **www**.

Intrarea de tip „MX” – (sau Mail Exchange) este folosită pentru identificarea locației unde ar trebui redirectate toate mail-urile către zona respectivă. Formatul acestei intrări este:

```
(nume)    MX    (prioritate)    (nume_server_mail)
```

Un exemplu pentru acest tip de intrare ar fi:

```
@    MX    10    mail1.exemplu.ro
@    MX    20    mail2.exemplu.ro
```

În exemplul anterior, putem identifica două intrări pentru două servere de email diferite, cu priorități diferite. Valoarea specifică prioritatea respectivului server, cu alte cuvinte, intarile cu o valoare mai mică vor avea o prioritate mai mare.

Fişierele ce conțin informații despre zone (precum **/etc/bind/db.local**) se găsesc în mod normal în directorul de configurare **bind9**.Tot în această locație se recomandă adăugarea oricăror fişiere de zonă ulterioare, pentru păstrarea unui şablon.

Deşi fişierele de zonă conțin toate informațiile necesare pentru instantierea unui nume de domeniu, server-ul **bind9** aşteaptă includerea acestor fişiere de zonă (prin intermediul format clar) în fişierul destinat definirii zonelor **/etc/bind/named.conf.local** (locație implicită). Acest fişier este folosit pentru adăugarea oricăror zone ulterioare. Pentru adăugarea unui fişier de zonă în

lista activă a server-ului, trebuie definită o **nouă zonă** în cadrul fişierului menționat anterior, urmând următoarea sintaxă pentru fiecare zonă definită:

```
zone „nume_zonă” {
type (tip);
file „fişier_de_zonă”;
};
```

Fiecare zonă definită în acest fel va trebui să aleagă numele de zonă identic cu domeniul pe care îl reprezintă, şi se recomandă alegerea unui nume de fişier de genul **db.(nume_domeniu)** pentru o evidență clară. Se poate alege modelul de funcționare al server-ului **bind9** (**master** sau **slave**) prin specificarea atributului **type.**

Configurația server-ului **bind9** este foarte flexibilă, acesta bazându-se pe un sistem în care se pot adăuga oricâte fişiere adiționale de configurare folosind directiva **include** în cadrul fişierului principal de configurare **/etc/bind/named.conf**. Acesta va include de obicei fişierele speciale **named.conf.options** (ce descrie parametrii funcționali ai server-ului) şi **named.conf.local** (folosit pentru a descrie zonele adiționale), dar va defini şi o serie de zone speciale folosite pentru procesele de **forwarding** şi **reverse look-up**. Pentru informații referitoare la toți parametri **options** consultați pagina de manual: **man named.conf**.

Chiar şi fără nicio zonă definită, server-ul instalat va funcționa ca un server DNS normal; având lista de servere root acesta poate rezolva orice fel de cerere va primi.

În continuarea celor prezentate anterior, vom parcurge un exemplu de configurare al unui server DNS prin instantierea unui domeniu fictiv, numit **exemplu.ro**.

Pasul 1. Crearea fişierului de zonă se poate face în directorul recomandat **/etc/bind**, lângă celelalte fişiere de zonă. În contextul exemplului curent, vom realiza un fişier de zonă relativ simplu, pentru o mai bună înțelegere a acestuia. Acesta poate fi creat de la zero, însă pentru a salva timp, se poate crea o copie a unei zone deja existente, urmând să o modificăm pe aceasta. În acest sens, vom modifica o copie a fişierului de zonă **db.local** (prezentat mai devreme): **cp /etc/bind/db.local /etc/bind/db.exemplu.ro**

Pasul 2. Modificarea fişierului de zonă presupune actualizarea înregistrărilor existente astfel încât zona nou createa să reflecte domeniul **exemplu.ro**: **mcedit /etc/bind/db.exemplu.ro**
După modificare, fişierul **db.exemplu.ro** ar trebui să arate similar cu fişierul prezentat în poza de mai jos:

```
$TTL    604800
@       IN      SOA     ns.exemplu.ro. root.exemplu.ro. (
                              2         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@       IN      NS      ns.exemplu.ro.
@       IN      A       192.168.2.100
www     IN      A       192.168.2.100
```

În cadrul primei înregistrări (de tip **SOA**) putem observa că server-ul **ns.exemplu.ro** a fost configurat ca fiind serverul de nume principal pentru domeniul respectiv. De asemenea, în cadrul celei de-a două înregistrări (de tip **NS**) specificăm că serverul DNS **ns.exemplu.ro** va fi principala autoritate pentru @, mai exact pentru domeniul **exemplu.ro**. Următoarele două înregistrări de tip **A** specifica numele asociate domeniului **exemplu.ro**: primul este @, şi se referă chiar la domeniul **exemplu.ro** – şi ca urmare este obligatoriu –, iar al doilea descrie subdomeniul **www** al domeniului **exemplu.ro**, asociand ambele nume cu adresa fictivă 192.168.2.100.

OBSERVAȚII: Fişierul trebuie neapărat să se încheie cu o linie nouă, goală, pentru a fi valid. De asemenea trebuie observate punctele adiționale după numele de domeniu (**exemplu.ro.**) – acestea sunt obligatorii, fiind folosite pentru descrierea domeniului top-level root şi nu reprezintă o greşeală.

Pasul 3. Crearea zonei „exemplu.ro” în cadrul fişierului **/etc/bind/named.conf.local**. Pentru că noul fişier de zonă creat să fie considerat de către server-ul DNS, trebuie definită o nouă zonă caracteristică pentru acesta, prin adăugarea următoarelor linii în fişierul **named.conf.local**:

```
zone „exemplu.ro” {
type master;
file „/etc/bind/db.exemplu.ro”;
};
```

Sintaxa acestor linii trebuie respectată la caracter, altfel server-ul **bind9** nu va ştii să le interpreteze, generând un mesaj de eroare. Numele zonei (în acest caz **exemplu.ro**) trebuie să fie identic cu numele domeniului pe care îl reprezintă.

Pasul 4. Verificarea zonelor se poate realiza după creare cu ajutorul utilitarului inclus **named-checkzone.** Sintaxa acestuia este: **named-checkzone domeniu fişier_zonă** sau, în cazul exemplului de față:

```
named-checkzone exemplu.ro db.exemplu.ro
```

Acesta va afişa pe ecran orice erori va întâlni în cadrul fişierului de zonă. În contextul exemplului curent, am menționat în fişierul de zonă, că server-ul **ns.exemplu.ro** va reprezenta autoritatea pentru acest domeniu, însă în urma executării comenzii de mai sus, vom fi atenționați că acesta nu are nici un fel de intrare de tip **A**. Corectarea acestei probleme se poate realiza prin adăugarea liniei

```
ns IN A 192.168.2.100
```
prin care se va adăuga şi subdomeniul **ns** pentru domeniul **exemplu.ro**.

Pasul 5. Verificarea fişierelor de configurare după modificarea lor, se poate realiza cu ajutorul utilitarului **named-checkconf (fişier_de_configurare)**. Acest pas poate fi sărit, însă poate furniza indicii importante în cazul în care server-ul **bind9** nu mai poate să pornească.

Pasul 6. Restartarea server-ului bind9 se poate face prin comanda:
```
/etc/init.d/bind9 restart
```
Această comandă va forța recitirea fişierelor de configurație, server-ul încărcând astfel informațiile despre noul domeniu **exemplu.ro**.

Pasul 7. Testarea server-ului DNS se poate face în mai multe feluri: fie prin includerea IP-ului stației ce rulează daemon-ul în lista de servere DNS (adăugarea unei linii în fişierului **/etc/resolv.conf** de tipul **nameserver ip_server**), fie prin utilizarea unor utilitare DNS lookup, precum **dig** sau **nslookup**. În cazul de față, vom testa server-ul cu ajutorul utilitarului **dig**:
```
dig @adresă_server_DNS nume_domeniu [+short]
```
Comanda anterioară va întreba serverul menționat prin parametrul **adresă_server_DNS** de toate înregistrările referitoare la domeniul **nume_domeniu**, pe care le va afişa pe ecran, impartitie în secțiuni de tipul **answer, question,** etc. Parametrul opțional **+short** va forța programul să afişeze doar adresa IP a numelui de domeniu căutat în cazul în care aceasta poate fi determinată. Comanda necesară pentru exemplul curent – în cazul în care este rulată de pe acelaşi sistem ca şi server-ul:
```
dig @localhost exemplu.ro +short
```
În acest moment, domeniul **exemplu.ro** va fi configurat şi disponibil tuturor utilizatorilor care au configurat ca server DNS, sistemul pe care server-ul rulează.

În cazul în care se doreşte configurarea domeniului **exemplu.ro** pentru a suporta şi **reverse lookup** (traducerea din adresă IP în nume de domeniu), mai trebuie urmați câțiva paşi adiționali:

Pasul 8. Crearea zonei reverse se realizează prin adăugarea următoarelor linii în fişierului **/etc/bind/named.conf.local**:
```
zone „2.168.192.in-addr.arpa” {
type master;
notify no;
file „/etc/bind/db.192”;
};
```
Numele zonei trebuie să respecte formatul de mai sus, mai exact să conțină primele 3 părți ale adresei IP asociate cu domeniul (în ordine inversă), urmat de „**in-addr.arpa**”. Se poate observa opțiunea **notify (**cu valoarea **no**) care semnalizează serverului DNS să nu înştiințeze serverele **slave** în momentul în care acesta primeşte o actualizare a fişierelor sale de zonă.

Pasul 9. Crearea fişierului de zonă trebuie să respecte anumite condiții, precum stabilirea numele fişierului de zonă. Acesta trebuie să înceapă (conform pasului anterior) cu **db.**, urmat de prima parte a adresei IP, în cazul de față fiind vorba de **db.192**.

```
$TTL    604800
@       IN      SOA     ns.exemplu.ro. root.exemplu.ro. (
                              2         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@       IN      NS      ns.
10      IN      PTR     ns.exemplu.ro.
10      IN      PTR     www.exemplu.ro
10      IN      PTR     exemplu.ro
```

Fişierul de zonă este oarecum similar cu cel pentru domeniul **exemplu.ro**, cu deosebirea înregistrărilor PTR, folosite pentru procesul de **reverse lookup**. Pentru fiecare înregistrare de tip A din fişierul **db.exemplu.ro** va trebui creată o înregistrare de tip PTR.

Pasul 10. Verificarea reverse lookup se face tot cu utilitarul dig:

dig @localhost 2.168.192.in-addr.arpa. AXFR

Pentru mai multe informații referitoare la configurarea şi securizarea server-ului **bind9**, consultați partea dedicată acestuia în secțiunea de securitate.

Instalarea şi configurarea serviciului HTTP

Serviciul HTTP este unul din cele mai răspândite servicii oferite în prezent prin sistemele informatice. Un server web nu este altceva decât un serviciu (daemon) care oferă conținut informatic sub forma paginilor web, prin intermediul protocolului HTTP (Hypertext Transfer Protocol).

Rolul principal al oricărui server HTTP este acela de a pune la dispoziția clienților pagini web şi conținutul ataşat acestora – fişiere adiționale sau imagini. Clienții cei mai comuni ai acestui serviciu sunt programele de **web browsing** care, în baza unei cereri specifice către server, va primi resursa dorită.

Deşi rolul inițial al serverelor web a fost acela de trimitere a informațiilor, acestea au evoluat pentru o experiență mai bogată a utilizatorilor, putând să şi primească informații de la aceştia. Au apărut astfel două tipuri de conținut web: **static** şi **dinamic**, acesta din urmă putând fi generat în timp real, atât local (javascript) cât şi direct de către server, prin conceptul de server-side scripting. Rolul serverelor web nu a rămas doar la nivelul transferului de pagini web, ci a ajuns chiar să ofere interfețe de configurare în dispozitive dedicate precum imprimante şi routere.

Printre cele mai folosite servere HTTP se numără şi server-ul **Apache**, o inițiativă Open-Source şi unul dintre pionierii internet-ului, şi este conceput să ruleze atât pe sisteme de tip Unix cât şi pe Windows NT. Conform datelor oficiale, peste 50% din toate paginile transferate din 1996 până azi au fost servite de către un server **Apache** (http://www.apache.org).

Apache a fost construit în jurul unei structuri modulare, prin care se pot extinde funcții de bază precum facilități de server-side scripting, securitate şi modele de autentificare. Putem menționa o serie de module mai comune grupate pe categorii:

- Module de autentificare: **mod_access, mod_auth, mod_digest, mod_auth_digest**
- Module de limbaje de programare: **Perl, Python, PHP**
- Module de comunicare**: mod_ssl, mod_proxy.**

Apache oferă şi posibilitatea implementării conceptului **Virtual Hosting**, care permite deservirea mai multor pagini web diferite simultan, de către o singură instanță a unui server.

Instalarea se poate face în mai multe feluri, însă se recomandă (ca şi în cazul tuturor celorlalte programe) folosirea sistemului **apt**, pentru a rezolva problema dependințelor şi actualizărilor manuale. Instalarea prin intermediul sistemului **apt** se poate face apelând comanda:

```
apt-get install apache2 apache2-doc apache2-suexec
```

Pe lângă pachetul **apache2** (care va include automat toate pachetele necesare), se recomandă şi instalarea pachetelor adiționale **apache2-doc** (ce conține documentația aferentă, foarte utilă de altfel) şi **apache2-suexec** care oferă utilizatorilor de **Apache** posibilitatea rulării programelor **CGI** şi **SSI**

drept alți utilizatori, în scopul reducerii riscurilor de securitate. În urma instalării, utilizatorul va putea vizualiza mesajele de inițializare ale tuturor modulelor incluse. În cadrul acestui ghid de instalare vom trata şi câteva module opționale extrem de utile (PHP, Ruby şi Python) ce oferă suport pentru o serie de facilități precum paginile php. Deşi acestea sunt opționale, instalarea acestora este recomandată în baza necesităților funcționale ale utilizatorului.

Pentru instalarea modulului PHP trebuie executată următoarea comandă:

```
apt-get install libapache2-mod-php5 php5 php5-common php5-curl php5-dev php5-gd php5-idn php-pear php5-imagick php5-imap php5-mcrypt php5-memcache php5-mhash php5-ming php5-mysql php5-pspell php5-recode php5-snmp php5-sqlite php5-suhosin php5-tidy php5-xcache php5-xmlrpc php5-xsl
```

Această listă de pachete (deşi destul de lungă) constituie o selecție din cele mai importante module php5 disponibile, şi unele dintre acestea asigură funcții importante pentru script-urile ce vor rula pe server.

Pentru instalarea modulului Ruby vom executa comanda:

```
apt-get install libapache2-mod-ruby
```

Pentru instalarea modulului Python vom executa comanda:

```
apt-get install libapache2-mod-python
```

Odată instalate toate modulele adiționale, putem începe să analizăm structura configurației server-ului **Apache**. Urmând o cale prin care se încearcă păstrarea compatibilității cu versiuni ale server-ului ceva mai vechi, se pot schimba parametri prin fişierul **/etc/apache2/httpd.conf** însă fişierul principal de configurație este **apache2.conf**, localizat în acelaşi director. Folosind acelaşi sistem de includere a mai multor fişiere de configurație într-unul singur, se poate centraliza totul, făcând astfel configurația întregului ansamblu mai flexibilă. Un alt fişier important de configurație este **/etc/apache2/ports.conf** în care se specifică port-urile TCP pe care server-ul va aştepta cereri (în versiunile mai vechi de apache, se folosea un singur fişier httpd.conf care include toate directivele, inclusiv port-uri, servere virtuale şi alți parametri funcționali). În mod normal, se recomandă stocarea oricăror configurații particularizate în directorul **/etc/apache2/conf.d**, acest director fiind automat inclus în configurația globală (prin directiva **Include /etc/apache2/conf.d** ce poate fi localizată la finalul fişierului **apache.conf**), asigurându-se în acest fel un sistem prin care fişierele principale de configurare rămân în mare parte nealterate.

Folosind directiva include pusă la dispoziția utilizatorului, instanțele de apache împart secțiunea dedicată modulelor adiționale în două directoare separate: **/etc/apache2/mods-available**, respectiv **/etc/apache2/mods-enabled**. Ori de câte ori utilizatorul va instala un modul adițional prin intermediul sistemului apt, acesta va adăuga unul sau două fişiere în directorului **mods-available**, pentru a marca încărcarea modulelor

respective în instanță server-ului. Aceste fişiere vor avea extensia **.load** şi vor specifica prin intermediul unor directive standard încărcarea modulelor. Opțional, pot fi create şi fişiere cu extensia **.conf** prin intermediul cărora vor fi specificați parametri adiționali.

După instalarea unor module adiționale, şi implicit şi crearea automată a fişierelor în directorul **mods-available**, acestea nu vor fi active. Pentru a le activa, utilizatorul va trebui să folosească comanda **a2enmod [nume_modul]**. Acest utilitar va crea legături simbolice în cadrul directorului **mods-enabled**, care vor arăta spre fişierele corespunzătoare (inclusiv fişierul **.conf** dacă acesta există) din directorul **mods-available**. Pentru continuarea instalării modulelor necesare, utilizatorul va trebui să lanseze comanda: **a2enmod ssl rewrite suexec include** după care acesta va trebui să restarteze server-ul apache: **/etc/init.d/apache2 restart**

Pentru dezactivarea anumitor module se poate folosi comanda:

a2dismod [nume_modul]

Din punctul de vedere al site-urilor pe care server-ul apache le va deservi, utilizatorul trebuie să fie familiar cu două directoare importante ce pot fi localizate în directorul implicit **/etc/apache2**. Acestea sunt **sites-available** şi **sites-enabled**, în mod similar cu directoarele dedicate modulelor adiționale, acestea conțin fişiere de configurare pentru site-urile găzduite. Deoarece, ca şi în cazul modulelor **available** (disponibile) nu înseamnă şi **enabled** (activate), utilizatorul trebuie să folosească alte două utilitare incluse pentru activarea, respectiv dezactivarea site-urilor:

a2ensite [nume_fişier_configurare] – folosit pentru activarea unui site, al cărui fişier de configurare poartă numele specificat că parametru şi poate fi găsit în directorul implicit **/etc/apache2/sites-available**. Similar cu utilitarul **a2enmod**, acesta va crea legături simbolice în directorul **sites-enabled**.

a2dissite [nume_fişier_configurare] – folosit în mod similar, pentru dezactivarea unui site.

În acest moment, instalarea server-ului apache, cât şi activarea modulelor adiționale este gata, acesta putând fi testat printr-un browser web, folosind ca adresă, IP-ul sistemului (sau **localhost**)pe care rulează (pagina implicită conține doar textul „**It works!**”). În continuare, vom analiza posibilitățile de configurare ale server-ului apache folosind şi o serie de exemple. Împărțirea configurației în mai multe fişiere mai mici permite o gestionare mai simplă şi mai rapidă – pe de altă parte, cât de mare ar fi fost fişierul de configurație dacă ar fi conținut direct, şi nu prin intermediul directivei **Include** toate fişierele de configurare (întregul conținut al tuturor fişierelor din directoarele **mods-enable, mods-available, conf.d** şi restul fişierelor de configurație)? Vom parcurge astfel, urmărind structura fişierelor de configurare, majoritatea opțiunilor importante. Este important de ştiut că toate fişierele de configurare apache folosesc o sintaxă destul de simplă, bazată pe directive, despre care vom discuta în detaliu. Pentru o dezactivare mai uşoară a anumitor directive,

fără ştergerea acestora, se poate folosi sistemul de comentare, care presupune adăugarea caracterului „#” la începutul liniei dorite. În urmă acestui proces, directiva respectivă va fi ignorată.

Fişierul ports.conf

Server-ul apache, ca orice alt server HTTP aşteaptă cereri în mod implicit pe port-ul 80, acesta fiind port-ul standard pentru acest protocol. Apache nu este însă limitat doar la acest port, şi poate fi configurat să aştepte conexiuni în mod implicit pe alte porturi (poate chiar să asculte simultan pe mai multe port-uri diferite).

Server-ul pus să aştepte conexiuni pe un port folosind directiva **Listen**, după cum se poate observa în fişierul **ports.conf**:

```
NameVirtualHost *:80
Listen 80


<IfModule mod_ssl.c>
    # SSL name based virtual hosts are not yet supported, therefore no
    # NameVirtualHost statement here
    Listen 443
</IfModule>
```

Datorită configurării implicite, site-ul de bază oferit este considerat un site virtual. În poza anterioară putem observa folosirea a două directive importante: **NameVirtualHost** şi **Listen**. Prima directivă poate fi folosită pentru identificarea unui site virtual, însă poate semnala şi o configurare selectivă prin care se asociază un site virtual cu un port – în exemplul de față, directiva poate fi tradusă astfel: toate site-urile virtuale (caracterul „*”) vor fi accesibile pe port-ul 80. Directiva **Listen** este folosită pentru configurarea server-ului astfel încât acesta să asculte pe un port anume. Se pot folosi mai multe directive **Listen** pentru a face server-ul să aştepte conexiuni pe mai multe port-uri. Ca mențiune, în cazul în care un site virtual este configurat să fie disponibil pe un port anume, administratorul trebuie să se asigure că server-ul este configurat la rândul său să accepte conexiuni pe port-ul respectiv.

Ultima secțiune delimitată prin **tag-uri** (structuri de tipul **<nume_tag> conținut … </nume_tag>**) reprezintă o configurare selectivă a modulului adițional **ssl**, responsabil pentru comunicările securizate pe protocolul **HTTPS**. Acesta este configurat în mod implicit să accepte conexiuni doar pe port-ul 443.

OBSERVAȚIE: Site-ul implicit, care este configurat odată cu prima instalare a server-ului **apache** are fişierul de configurație în **/etc/apache2/sites-enabled/000-default**. Acesta este configurat să fie disponibil pe portul 80, ceea ce forțează administratorul (în cazul în care doreşte să schimbe port-ul implicit) să modifice atât acest fişier (prin schimbarea portului din directiva de pe prima linie - ***:[port]**) cât şi fişierul **ports.conf**. O soluție mai elegantă la această problemă este definirea port-ului exclusiv din cadrul fişierului de

configurare pentru un site – acest proces implică comentarea directivei **NameVirtualHost *:80** din fişierul **ports.conf** prin adăugarea caracterului „**#**” înaintea directivei – dar şi activarea portului respectiv printr-o directivă **Listen** în fişierul **ports.conf**. Exemplul următor va sublinia schimbările făcute în fişierele **000-default**, respectiv **ports.conf** pentru schimbarea port-ului implicit din 80 în 801.

fisierul /etc/apache2/sites-enabled/000-default

```
<VirtualHost *:801>
        ServerAdmin webmaster@localhost

        DocumentRoot /var/www/
```

fisierul /etc/apache2/ports.conf

```
#NameVirtualHost *:80
Listen 801


<IfModule mod_ssl.c>
    # SSL name based virtual hosts are not y
```


Schimbarea anterioară nu va deveni activă decât după ce server-ul apache va fi repornit prin comanda **/etc/init.d/apache2 restart**. În urma acestei schimbări, site-ul implicit cu textul „**It works!**” va fi accesibil doar pe port-ul 801 al server-ului.

Fişierul httpd.conf va fi inițial un fişier gol. Acesta era folosit drept principalul fişier de configurare în versiunile precedente de apache, în prezent fiind păstrat din motive de compatibilitate. Administratorii pot adăuga directive suplimentare în acest fişier, fiind inclus în configurația globală prin directiva **Include**.

Fişierul apache2.conf este principalul fişier de configurare, şi conține câteva din setările esențiale ale server-ului. Acest fişier este folosit şi pentru includerea fişierelor secundare de configurație, folosind directiva **Include**. Fişierele incluse prin intermediul acestei directive sunt:

- Toate fişierele din directoul **mods-enabled**
- Toate fişierele din directorul **sites-enabled**
- Toate fişierele din directorul **conf.d**
- Fişierele **httpd.conf** şi **ports.conf**

Pe lângă includerea fişierelor de configurație secundare, **apache2.conf** mai are rolul setării parametrilor server-ului apache, prin intermediul unor directive speciale. În continuare vom prezenta pe scurt principalele directive folosite pentru modificarea parametrilor server-ului:

- Directiva **ServerRoot („/etc/apache2”)** – specifică locația fişierelor de configurare ale server-ului apache. Directiva este case-sensitive.
- Directiva **Timeout (300)** este folosită pentru specificarea timpului (măsurat în secunde) pe care server-ul îl va aştepta înainte să trimită un mesaj de timeout.
- Directiva **KeepAlive (On/Off)** specifică dacă server-ul ar trebui să accepte conexiuni persistente, mai exact să permită primirea mai multor cereri în intervalul unei singure conexiuni.
- Directiva **MaxKeepAliveRequests (100) –** numărul maxim de cereri raportate la intervalul unei singure conexiuni persistente
- Directiva **KeepAliveTimeout (15)** – numărul de secunde pe care server-ul îl va aştepta pentru următoarea cerere de la acelaşi client, în intervalul unei conexiuni.
- Directiva **AccessFileName (.htaccess)** – menționează fişierul după care l să se uite server-ul în fiecare director pentru configurarea modului de acces restricționat.
- Directiva **HostnameLookups (Off)** - menționează dacă server-ul va include în jurnalizare numele (nu adresa IP) tuturor clienților.
- Directiva **ErrorLog (/var/log/apache2/error.log**) – specifică locația jurnalului de eroare
- Directiva **LogLevel (warn)** – specifică nivelul de mesaje care vorfi înregistrate în jurnal
- Directivele **Include (nume_fişier)** sunt folosite pentru încărcare

Crearea unui nou site

Pentru crearea unui nou site, utilizatorii trebuie să plaseze un fişier de configurare specific site-ului pe care-l doresc (ai cărui parametri îi vom discuta în continuare) şi apoi să-l activeze. În continuare vom analiza atât principalele directive folosite pentru configurarea site-urilor virtuale, cât şi paşii necesari. În scopul acestui exercițiu, vom parcurge un exemplu în care vom configura un site virtual numit **exemplu.ro** (pentru care vom folosi intrarea din DNS creată în secțiunea anterioară), disponibil pe port-ul 2009. La finalul exercițiului vom introduce şi o secțiune opțională dedicată accesării cu modulul de autentificare.

Pasul 1. Crearea fişierului de configurație este primul demers. Acesta va fi alcătuit exclusiv din directive apache, prin care vom descrie diferiți parametri funcționali. Structura fişierelor de configurare este una ierarhică, bazată pe o serie de structuri. Server-ul apache instantiaza site-urile într-un mod similar cu modulele: foloseşte două directoare numite **sites-available** respectiv **sites-enable** şi două script-uri pentru activarea şi dezactivarea acestora. Orice site trebuie să aibă fişierul său propriu de configurare în cadrul directorului **sites-available**, pentru a putea fi activat cu utilitarul **a2ensite [nume_fişier_configurare]** (care va crea o legătură simbolică către fişierul de configurare în directorul **sites-enable**).

Fişierul de configuraţie al unui site nou poate fi creat de la zero, însă se recomandă folosirea unui şablon – site-ul implicit. Fişierul de configuraţie al site-ului implicit poate fi găsit tot în directorul **sites-available** şi poartă numele **default**. Pentru a-l folosi pe acesta ca şablon, trebuie creată o copie pe care o vom modifica:

```
cp /etc/apache2/sites-available/default /etc/apache2/sites-available/www.exemplu.ro
```

Urmează să deschidem fişierul cu un editor de text pentru a face modificările necesare:

```
gedit /etc/apache2/sites-available/www.exemplu.ro
```

Încă din momentul deschiderii fişierului respectiv, putem observa natura ierarhică în care sunt organizate datele. Putem observa structuri ce încep cu directive compuse, al căror rol este gruparea unor directive ce se raportează la grupul respectiv că un întreg. Principala directivă compusă se întinde pe tot conținutul fişierului şi se numeşte **<VirtualHost>** - sfârşitul acetei directive este marcat prin **</VirtualHost>**. Toate directivele conținute în această structură se vor raporta în mod direct la host-ul virtual astfel definit. Se pot observa o serie de alte structuri marcate prin directivele **<Directory (nume_director)>** şi **</Directory>**. Acestea sunt folosite pentru restrângerea efectului directivelor conținute exclusiv la nivelul directorului specificat şi al subdirectoarelor sale – în general această directivă se foloseşte pentru configurarea directoarelor site-urilor conform unor cerințe specifice, de exemplu limitarea accesului sau includerea unor opțiuni speciale.

Pasul 2. Interpretarea fişierului de configurare se face de la prima linie, progresiv până la ultima, urmând directivele apache:

- **<VirtualHost adresă> … </VirtualHost>** se foloseşte pentru gruparea directivelor ce vor specifica parametrii site-ului respectiv. În acest caz adresa poate fi fie un nume de domeniu, fie o adresă ip.
- **ServerAdmin adresă_mail** setează adresa administratorului site-ului respectiv, adresa ce va fi inclusă în orice mesaj de eroare.
- **DocumentRoot director** stabileşte locația de unde server-ul apache va putea servi fişierele ce alcătuiesc site-ul. Acest director va reprezenta directorul root al site-ului.
- **ErrorLog nume_jurnal** oferă posibilitatea alegerii unui jurnal de erori separat de cel implicit. Se foloseşte împreună cu directiva **CustomLog**
- **Directivele Alias** respectiv **ScriptAlias** permit folosirea unor resurse care nu se află în directorul specificat prin directiva **DocumentRoot**. În general, se folosesc pentru activarea script-urilor de tip **cgi** şi pentru includerea unor resurse externe site-ului.
- **<Directory director> … </Directory>** grupează o serie de directive, pentru restrângerea efectului acestora la nivelul directorului specificat. În cadrul fişierului aferent exemplului, aceste structuri sunt folosite pentru descrierea individuală a setărilor fiecărui subdirector inclus în site.

- **Options** este una dintre cele mai importante directive, prin intermediul căreia se poate controla disponibilitatea tuturor funcțiilor server-ului la nivel de directoare. Directiva poate primi mai multe opțiuni, separate între ele prin spații. Printre opțiunile cele mai folosite se numără (acestea sunt case-sensitive):
    - **All** - toate opțiunile activate cu excepția **Multiviews**. Aceasta este şi setarea implicită a directivei **Options**.
    - **FollowSymLink** – permite server-ului apache să urmeze legăturile simbolice în directorul curent.
    - **ExecCGI** – permite executarea scripturilor CGI prin intermediul modulului aferent.
    - **Indexes** - în cazul în care directorul la care se referă nu conține nici un fişier de tip index (exemplu **index.html**), atunci server-ul va genera o pagină al cărei conținut va fi o listă cu toate fişierele directorului respectiv. Acest comportament va fi activat numai dacă fişierul index nu există, iar URL-ul folosit în cerere de către client nu se referă în mod direct la un fişier din directorul curent.
- **AllowOverride tip** – atunci când server-ul găseşte un fişier **.htaccess** în directorul curent, această directivă îi specifică ce opțiuni pot înlocui orice directive precedente. Tipurile disponibile sunt:
    - **None** – va duce la ignorarea fişierului **.htaccess**.
    - **All** – orice directivă care se referă la directorul respectiv va avea acces la fişierul **.htaccess.**
    - **AuthConfig** – permite folosirea directivelor de autorizare
    - **FileInfo** – permite folosirea directivelor ce controlează tipul documentelor
    - **Indexes** – permite folosirea directivelor ce controlează procesul de indexare al directoarelor
- **Directiva Order** specifică ordinea în care vor fi interpretate directivele **Allow şi Deny** prin intermediul cărora se poate stabili un sistem de acces bazat pe trei iterații. Opțiunile posibile sunt:
    - **Allow,Deny** – se începe prin evaluarea directivelor **Allow** (dacă orice cerere nu se potriveşte cu nici o directivă, aceasta va fi respinsă), urmate apoi de directivele **Deny**.
    - **Deny,Allow** – se începe prin evaluarea directivelor **Deny**, iar în cazul unei potriviri, cererea este respinsă.

      Exemple:
      Order Deny,Allow
      Deny from all
      Allow from www.exemplu.ro
      În cazul anterior, toate cererile vor fi respinse, cu excepția celor provenite de la domeniul **www.exemplu.ro**.
      Order Allow, Deny
      Allow from exemplu.ro
      Deny from ns.exemplu.ro

În exemplul de mai sus, deşi ultima directivă specifică faptul că tot ceea ce vine de la domeniul **ns.exemplu.ro** va fi respins, datorită ordinii alese, directiva **Allow** va avea prioritate mai mare, şi va suprascrie directiva **Deny**, pentru întregul domeniu **www.exemplu.ro** (inclusiv subdomenii). Directivele **Allow** şi **Deny** pot lua ca parametri fie cuvântul cheie **all**, fie un nume de domeniu, fie o adresă ip.

Pasul 3. Modificarea fişierului de configurare al site-ului se poate face în felul următor:

```
gedit /etc/apache2/sites-available/www.exemplu.ro
```

Modificarea acestui fişier trebuie reflecte site-ul pe care îl vom crea şi ca urmare trebuie să satisfacă următoarele condiții:

- DocumentRoot-ul va fi directorul /var/www.exemplu.ro
- Host-ul virtual se va numi 192.168.2.102 (adresa IP a server-ului, oricare ar fi aceasta)

Fişierul de configurare al site-ului ar trebui să arate în felul următor după modificare:

```
<VirtualHost 192.168.2.102:2009>
        ServerAdmin webmaster@exemplu.ro

        DocumentRoot /var/www.exemplu.ro/

        <Directory /var/www.exemplu.ro/>
                Options Indexes FollowSymLinks MultiViews
                AllowOverride None
                Order allow,deny
                allow from all
        </Directory>

        ErrorLog /var/log/apache2/error.log
        CustomLog /var/log/apache2/access.log combined
</VirtualHost>
```


Aceasta este o variantă simplificată de site, care va avea conținutul propriu-zis în directorul **/var/www.exemplu.ro** (este o practică comună pentru adminsitratori să păstreze site-urile în directoare cu acelaşi nume pentru o localizare mai uşoară). A se observa că am redus numărul de directive față de fişierul original şi că am configurat site-ul virtual să fie disponibil pe port-ul 2009.

Pasul 4. Crearea conținutului site-ului va fi un simplu fişier html, pentru demonstrarea conceptului. Va trebui creat mai întâi directorul corespunzător, apoi adăugat fişierul html:

```
mkdir /var/www.exemplu.ro
mcedit /var/www.exemplu.ro/index.html
```

Fişierul pe care îl vom crea se va numi **index.html** şi reprezintă unul dintre fişierele index implicite. Când server-ul va înregistra o cerere către adresa

sa, pe port-ul 2009, fără vreun fişier anume specificat în URL, acesta va trimite ca răspuns fişierul **index.html**.

Conținutul acestui fişier va respecta standard-ul html şi este la alegerea utilizatorului, însă un exemplu elocvent este următorul:

```
<html>
<body>
        Salut! Acesta este un site.
</body>
</html>
```

Pasul 5. Activarea server-ului pe port-ul 2009 este necesară din moment ce am stabilit că noul site va fi disponibil pe acest port. Pentru acest fapt, trebuie adăugată directiva „**Listen 2009”** în cadrul fişierului **/etc/apache2/ports.conf**, conform pozei de mai jos:

```
#NameVirtualHost *:80
Listen 801
Listen 2009
```

Pasul 6. Activarea site-ului este un pas esențial, şi poate fi realizat prin executarea comenzii:

```
a2ensite www.exemplu.ro
```

După activarea efectivă a fişierului de configurare, şi crearea automată a legăturii simbolice în directorul **sites-enabled**, trebuie executată comanda:

```
/etc/init.d/apache2 reload
```

pentru a forța server-ul apache să reîncarce fişierele de configurare şi implicit şi noul site. După procesul de reîncărcare al fişierelor de configurare, site-ul va fi accesibil la adresa IP a server-ului, pe portul 2009. Dacă au fost urmați şi paşii anteriori acestui exemplu, utilizatorul va putea observa că site-ul virtual implicit (**/etc/apache2/sites-available/default**) este accesibil, însă pe portul 801. În mod similar, server-ul poate fi configurat în aşa fel încât să deservească un site diferit pentru fiecare interfața de rețea de care beneficiază.

În continuare vom modifica noul site creat, pentru a limita accesul la acesta, numai pe baza unei autentificări prealabile a clientului. O greşeală comună este concepția că autentificările trebuie limitate numai în cadrul fişierelor **.htaccess**, ceea ce este o greşeală conform recomandărilor autorilor server-ului apache. Prin urmare, vom adăuga facilitatea de autentificare direct în fişierul de configurare al site-ului.

Pasul 1. Crearea fişierului de parolă se face prin intermediul utilitarului inclus **htdigest**. Acest utilitar va folosi un fişier pentru a stoca parolele individuale ale utilizatorilor, aceştia putând fi adăugați sau şterşi din listă. Fişierul de parole, împreună cu o primă înregistrare se poate crea apelând comanda:

htdigest –c /etc/apache2/parole „Zonă restricționată” user1

O locație bună pentru stocarea fişierului de parole este un loc în care acesta să nu fie accesibil prin server-ul de web (directorul **/etc/apache/** este o locație potrivită). Parametrul **–c** specifică faptul că se doreşte crearea fişierului **parole**¸ urmând ca utilizatorul să introducă parola pentru contul **user1**. Dacă se doreşte adăugarea unui utilizator nou, într-un fişier de parole deja existent, sintaxa este aceeaşi, dar va trebui omis parametrul **–c**, şi evident, specificat numele utilizatorului în loc de **user1.** Penultimul parametru (în acest caz „Zonă restricționată”) se numeşte **realm** şi este folosit pentru identificare – acesta trebuie să se potrivească cu valoarea alea pentru directiva **AuthName**.

Pasul 2. Încărcarea modulului de autentificare Digest se poate realiza folosind comanda

a2enmod auth_digest

Încărcarea oricăror module adiționale necesită de asemenea şi repornirea server-ului:

/etc/init.d/apache2 restart

Opțional se poate folosi autentificarea de tip „Basic” în loc de „Digest” - va trebui folosit utilitarul **htpasswd** (în loc de **htdigest**) pentru generarea fişierului de parole şi folosirea directivei **AuthType Basic** – însă această metodă trimite parolele în clar, fără nici un fel de criptare, prin urmare nefiind recomandată.

Pasul 3. Adăugarea directivelor de autentificare se va face în structura descrisă de directiva **<Directory>…</Directory>** pentru directorul în cauză. Urmărind exemplul anterior, vom alege să implementăm directivele de autentificare în cadrul directorului **/var/www.exemplu.ro**. Vom modifica aşadar, conținutul fişierului **/etc/apache2/sites-available/www.exemplu.ro** în felul următor:

```
<Directory /var/www.exemplu.ro/>
        Options Indexes FollowSymLinks MultiViews
        AllowOverride None
        Order allow,deny
        allow from all
        AuthType Digest
        AuthName "Zona restrictionata"
        AuthUserFile /etc/apache2/parole
        Require user user1
</Directory>
```

Pasul 4. Repornirea server-ului este necesară şi se face prin comanda

/etc/init.d/apache2 restart

În urma accesării site-ului, clientul va fi rugat să introducă datele de autentificare – utilizator şi parola. În configurația actuală, site-ul disponibil pe port-ul 2009 va fi disponibil doar pentru utilizatorul **user1**. Se pot adăuga noi utilizatori separați prin spații (de exemplu):

Require user user1 user2 user3 ….

Pe lângă includerea numelor utilizatorilor în cadrul secțiunii de autentificare, aceştia vor trebui să fie adăugați şi în fişierul de parole, cu ajutorul utilitarului **htdigest**.

Exemplul 2: Configurarea modulului UserDir

Modulul **UserDir** permite fiecărui cont de utilizator să poată avea o pagină web predefinita. Sistemul funcționează prin traducerea URL-ului primit de la client, în baza directivei **UserDir**, într-o adresă locală. Dacă modulul **UserDir** este încărcat, fiecare utilizator va putea avea o pagină proprie de web, în funcție de modul în care este configurat server-ul. Paginile individuale ale utilizatorilor sunt accesibile prin următoarea metodă de adresare:

http://adresă_server:port/~nume_utilizator

Directiva **UserDir** poate fi inclusă în cadrul fişierelor de configurare apache, şi are următoarea sintaxă:

UserDir adresă

Adresa specificată va fi folosită pentru a traduce URL-ul utilizatorului (**~nume_utilizator**) într-o cale locală. Folosit fără cale absolută, se va referi la calea menționată relativă la directorul home al fiecărui utilizator. Mai jos se pot observa câteva exemple de folosire:

- Exemplu 1: date fiind directiva **UserDir www,** calea **www.exemplu.ro/~user1/1.html** s-ar traduce automat în **/home/user1/www/1.html**
- Exemplu 2: date fiind directiva **UserDir /var/www,** calea **www.exemplu.ro/~user1/1.html** s-ar traduce automat în **/var/www/user1/1.html**

Această directivă se raportează în mod implicit la toți utilizatorii cu cont din sistem, însă la nevoie poate fi restrâns prin intermediul opțiunilor **enabled** şi **disabled**:

- Exemplu 1: **UserDir disabled user1 root** ar dezactiva modulul pentru utilizatorii **user1** şi **root**
- Exemplu 2: **UserDir disabled**
  UserDir enabled root user1

Secventa anterioară ar dezactiva modulul pentru toți utilizatorii, cu excepția conturilor **root** şi **user1**.

Vom încerca în exemplul ce urmează să configurăm modulul **UserDir** pentru toți utilizatorii sistemului, astfel încât paginile individuale ale acestora să se afle în directorul **www** din directorul **home** al fiecărui utilizator.

Pasul 1. Activarea modulului se face prin comanda:

a2enmod userdir

Pasul 2. Adăugarea directivei UserDir poate fi făcută în orice fişier de configurare, însă se recomandă crearea unui fişier nou, în cadrul directorului **/etc/apache2/conf.d**. În mod implicit toate fişierele de configurație din acest

director vor fi încărcate automat prin directiva **Include**. Pentru rezolvarea cerinței, se va crea fişierul **/etc/apache2/conf.d/userdir** al cărui conținut va fi următorul:

```
UserDir www
```

Pasul 3. Repornirea server-ului este necesară deoarece a fost încărcat un modul nou. În urma repornirii, toți utilizatorii vor putea beneficia de pagini web proprii, în directorul **/home/(nume_utilizator)/www**.

Pentru mai multe informații referitoare la directivele apache, consultați documentația instalată, sau accesați **httpd.apache.org/docs/2.2**/.

Instalarea şi configurarea serviciului de Email

Serviciul de poştă electronică a revoluționat comunicarea când a apărut, oferind o alternativă ieftină şi foarte rapidă de schimbare a mesajelor digitale între utilizatori. Serverele ce oferă acest serviciu funcționează pe principiul **store-and-forward**, adică acestea primesc mesajele şi le stochează individual pentru toate conturile existente în sistem. Utilizatorii acestui serviciu nu trebuie decât să se conecteze la server-ul de email şi să îşi acceseze mesajele.

Deşi original mesaje erau limitate doar la conținut text, suportul pentru mesaje cu ataşamente a fost extins prin intermediul standardului MIME (Multipurpose Internet Mail Extensions). Orice mesaj email poate fi împărțit în două secțiuni: antetul mesajului, care conține informații de control (adresele destinatarului şi receptorului, subiect, etc) şi corpul mesajului care reprezintă conținutul efectiv al mesajului.

Unul dintre cele mai folosite servere de email la ora actuală este **Postfix**, un server licențiat conform standardelor GNU şi a fost gândit de la început ca o alternativă sigură, rapidă şi uşor de instalat şi configurat la server-ul **Sendmail**. În cazul serviciului de email, există mai multe entități: **MTA** reprezintă un **Mail transfer agent**, adică un program care se ocupă atât de recepționarea cât şi de transmiterea mesajelor digitale, şi reprezintă server-ul de mail. **MUA** (sau **Mail user agent**) este programul folosit pentru interacționarea cu server-ul de mail şi poate fi fie un client de email, fie chiar un server de mail care are rolul de **relay** (adică transmite un mesaj mai departe), însă de cele mai multe ori, când se menționează termenul respectiv, se face referire la un client.

În continuare vom vedea paşii necesari înlocuirii clientului implicit de mail ce vine instalat cu distribuția Debian, procedura de instalare a server-ului **Postfix**, dar şi nişte exemple de configurare a acestuia. De asemenea, vom adăuga o înregistrare de tip **MX** în cadrul server-ului DNS pentru a putea identifica server-ul de mail în cadrul domeniului fictiv **www.exemplu.ro.** În mod implicit, server-ul de mail instalat în cadrul distribuției Debian 5.0 este **exim4**, acesta urmând să fie dezinstalat odată cu instalarea server-ului **Postfix** (din moment ce nu pot exista două servere de mail pe acelaşi sistem).

Instalarea server-ului Postfix se face urmând următorii paşi:

Pasul 1. Adăugarea intrării MX în cadrul server-ului DNS se face modificând fişierul de zonă ce reprezintă domeniul **exemplu.ro**. În secțiunea dedicată instalării şi configurării serverului **bind9** am folosit server-ul **/etc/bind/db.exemplu.ro** pentru a descrie domeniul respectiv. Pentru adăugarea intrării **MX,** acesta trebuie modificat prin adăugarea liniilor:

```
        IN    MX    10    mail.exemplu.ro
@       IN    A           (adresă_ip_server)
```

Amintim pe această cale, că fişierul de zonă trebuie să se termine neapărat cu o linie nouă, goală. Adăugarea celor două linii de mai sus va avea ca

efect înregistrarea subdomeniului **mail** ca o intrare de tip **MX**, cu prioritatea 10**,** şi asocierea subdomeniului **mail.exemplu.ro** cu adresa ip a server-ului.

Pasul 2. Repornirea server-ului DNS este necesară pentru forțarea reîncărcării zonei

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

Pasul 3. Instalarea pachetului postfix se poate face folosind sistemul apt:

```
apt-get install postfix
```

În urma executării liniei de mai sus, utilizatorul va fi atenționat că server-ul implicit de mail (**exim4**) urmează să fie dezinstalat. Mai mult, utilizatorul va urma un fel de wizzard pentru alegerea configurației de bază. Utilizatorul poate naviga butoanele interfeței folosind fie tastele de direcție, fie butonul tab, şi poate confirma o selecție folosind Enter. Vom parcurge toate întrebările pe care programul de instalare le va pune, pentru a clarifica orice neînțelegere ce poate să apară pe parcurs:

- **Întrebarea 1:** „Please select the mail server configuration type that best meets your needs”

  Odată cu întrebarea, sunt descrise pe scurt toate opțiunile disponibile. Vom presupune pentru acest exercițiu că vom alege configurația **Internet Site** prin care se certifică faptul că server-ul va folosi protocolul **SMTP** (Simple Mail Transfer Protocol).
- **Întrebarea 2:** „System mail name”

  Conform descrierii, **mail name** se referă la numele de domeniu care va fi folosit pentru calificarea tuturor mesajelor primite. În exemplul de față, vom alege **exemplu.ro**. Este important ca server-ul DNS să funcționeze în momentul instalării, astfel încât domeniul ales să fie disponibil.

Pasul 4. Configurarea server-ului Postfix

În acest moment, server-ul Postfix este instalat, şi rulează conform configurației implicite. Fişierul de configurare pentru acesta este un fişier text, şi se poate găsi în directorul **/etc/postfix/main.cf**. Pentru alterarea configurației, acesta trebuie deschis cu un editor text:

```
mcedit /etc/postfix/main.cf
```

Opțiunile de configurare sunt prezentate în formatul **opțiune = valoare** şi sunt relativ uşor de interpretat (liniile care încep prin caracterul „#” sunt comentarii):

- **myorigin = nume** – reprezintă numele de la care mail-urile locale vor origina. Se pot folosi alți parametri deja definiți în fişier folosind în loc de valoarea propriu-zisă textul **$opțiune**. Dacă am folosi **myorigin = $mydomain**, atunci s-ar folosi valoarea opțiunii **mydomain**.
- **mydomain = nume –** reprezintă numele de domeniu al server-ului de mail.
- **myhostname = nume** – reprezintă numele de sistem al server-ului de mail.

- **smtpd_banner = $myhostname ESMTP $mail_name** – reprezintă textul de salutare primit după codul de confirmare 220 în cadrul protocolului SMTP. Valoarea **$myhostname** trebuie specificată la începutul liniei, fiind o cerință specifică a protocolului SMTP.
- **biff = valoare yes/no** – specifică dacă se doreşte folosirea serviciului local **biff**, care trimite notificările de mesaje noi tuturor utilizatorilor care s-au înregistrat folosind comanda „**biff y**”.
- **append_dot_mydomain = valoare yes/no** – în cazul mesajelor locale, specifică dacă se va adăuga textul „$mydomain” la adresele care nu conțin această informație.
- **delay_warning_time = valoare h** – specifică intervalul măsurat în ore după care destinatarul va primi antentul mesajelor care încă aşteaptă.

Vom descrie pe scurt câteva din opțiunile mai importante:

- **inet_interfaces = valoare** – se foloseşte pentru specificarea interfețelor de rețea ale sistemului pe care server-ul va primi mail. Valorea implicită este **all**, ceea ce înseamnă că server-ul va primi mesaje pe toate interfețele configurate ale sistemului
- **mynetworks = listă** – lista clienților SMTP care vor avea privilegii sporite. Mai exact, aceştia vor putea folosi server-ul de mail pentru procesul de **mail relay**.
- **smtpd_sasl_auth_enable = yes/no** – activează sau dezactivează autentificarea de tip SASL în server-ul **Posfix.** În mod implicit, acesta nici măcar nu foloseşte autentificare.
- **smtpd_sasl_security_options = valoare –** specifică lista de opțiuni de securitate:
    - **noplaintext –** interzice orice metodă ce foloseşte parole în clar
    - **noactive** – interzice orice metodă de atac activ
    - **nodictionary** – interzice orice de metodă de atac pasiv
    - **noanonymous** – interzice orice metodă de autentificare anonimă
    - **forward_secrecy** – se folosesc doar metode care suportă forward secrecy
    - **mutual_auth** – se folosesc doar metode care implică autentificarea mutuală.
- **smtpd_sasl_local_domain = valoare –** specifică zona de autentificare SASL a server-ului.
- **broken_sasl_auth_clients = yes/no** – asigură compatibilitatea între clienți SMTP mai vechi care folosesc o variantă depreciată de autentificare.
- **smtpd_recipient_restrictions** = listă – specifică restricții pe care server-ul le aplică comenzii **RCPT TO**.
- **home_mailbox = director** – specifică calea opțională către directorul de

mail, relativ la adresa home a utilizatorilor.

- **alias_maps = valoare –** baza de date de alias-uri folosită pentru trimiterea locală de mesaje.
- **relayhost = nume** – server-ul iterației următoare (în cazul mail-urilor externe)
- **stmpd_recipient_limit = valoare –** specifică numărul limită de recipienți acceptați pentru fiecare mesaj.

Pentru familiarizarea în cadrul procesului de configurare a server-ului **Postfix,** se recomandă crearea unei copii de siguranță a fişierului **/etc/postfix/main.cf.**

Pentru mai multe informații despre opțiunile existente în cadrul fişierului de configurare accesați **www.postfix.org/postconf.5.html.**

```
cp /etc/postfix/main.cf /etc/postfix/main.cf.backup
```

Unul din avantajele majore ale server-ului **Postfix** este flexibilitatea sa, acesta mergând în majoritatea cazurilor fără nici un fel de configurare prealabilă. Se recomandă însă modificarea următoarelor două linii pentru asigurarea funcționalității server-ului (pentru domeniul fictiv **exemplu.ro**):

```
myhostname = mail.exemplu.ro
myorigin = exemplu.ro
mydestination = exemplu.ro, stație.stație, localhost
```

Este bine de ştiut că server-ul **Postfix** va respinge în mod automat orice fel de mesaje al căror destinar sau expeditor nu poate fi determinat. Datorită legăturii strânse cu server-ul DNS cât şi faptului că a fost legat de domeniul **exemplu.ro**, server-ul va respinge orice mesaje în care adresa destinatarului (sau expeditorului la trimitere) nu reprezintă un nume de domeniu calificabil. Prin urmare, pentru a oferi utilizatorilor posibilitatea folosirii atât a numelui de domeniu cât şi a adresei IP a server-ului pentru trimiterea, respectiv primirea mesajelor electronice, server-ul **Postfix** trebuie configurat să poată interpreta „domeniile numerice” prin adăugarea opțiunii următoare în cadrul fişierului **/etc/postfix/main.cf**:

```
resolve_numeric_domain = yes
```

Ca urmare a activării acestei opțiuni, server-ul va accepta ca adrese de destinatar sau expeditor atât adresele IP (**nume_cont@adresă_ip**), cât şi adresele de tipul **nume_cont@domeniu**.

De asemenea, opțiunea **mydestionation** trebuie să conțină toate numele de sistem pentru care server-ul **Postfix** se va considera ca adresă finală. În cazul în care domeniul **exemplu.ro** nu ar fi inclus în lista de opțiuni, server-ul **Postfix** ar considera că toate mesajele electronice pe care le primeşte nu îi sunt destinate direct şi prin urmare ar juca rolul de **relay host** (adică ar încerca să trimită mesajul mai departe). Acest mod de funcționare devine evident atunci când server-ul nu este configurat ca un **relay host**, caz în care expeditorul va fi refuzat, iar mesajul întors.

Pasul 5. Instalarea server-ului IMAP/POP3

În momentul în care un utilizator trimite un mesaj electronic cu un **MUA** (Mail user agent), sau mai exact client de mail, acesta este trimis prin protocolul **SMTP** entității **MTA** (server-ul). Acesta verifică apoi adresa recipientului, face o cerere către DNS-ul sau pentru a afla **MTA-**ul corespunzător domeniului recipientului şi în final transmite acelui server mesajul, tot prin intermediul protocolului **SMTP**. În final, după ce mesajul a ajuns la server-ul corespunzător, utilizatorul îşi poate verifica mesajele folosind protocolul **POP3** sau protocolul **IMAP**.

Pentru a oferi o interfață de acces la directoarele de mail din sistem prin protocolul IMAP sau POP3, trebuie instalat un server care să poată oferi acest serviciu. **Dovecot** este un astfel de server, dezvoltat în mod special pentru această facilitate şi poate fi configurat să funcționeze în paralel cu alte servere de mail tradiționale precum **Qmail, Exim,** sau în cazul nostru **Postfix.** Server-ul **Dovecot** nu joacă niciun rol în primirea efectivă a mesajelor de la alte entități **MTA,** ci doar oferă o metodă prin care utilizatorii pot accesa mesajele deja stocate pe server. Instalarea server-ului se face prin comanda:

```
apt-get install dovecot-imapd dovecot-pop3d dovecot-common
```

După instalare, server-ul **Dovecot** trebuie configurat în vederea selectării protocoalelor IMAP şi POP3 prin modificarea fişierului de configurare **/etc/dovecot/dovecot.conf** şi adăugarea liniilor (sau decomantarea acestora acolo unde este cazul):

```
protocols = pop3 imap
disable_plaintext_auth = no
pop3_uidl_format = %08Xu%08Xv
```

După reconfigurarea server-ului **Dovecot**, este necesară o repornire a acestuia prin intermediul comenzii

```
/etc/init.d/dovecot restart
```

În acest moment, toate conturile de utilizator locale se vor putea conecta folosind unul din protocoalele menționate mai sus pentru a primi sau a trimite (local) mesaje electronice. Server-ul de email se poate testa folosind orice fel de client de mail local sau extern (dar în cadrul aceleaşi rețele) capabil să comunice pe unul dintre cele două protocoale menționate.

Deşi avem de-a face cu un server de email perfect funcțional, acesta este momentan capabil să trimită şi să primească doar mesaje locale. În cazul în care se încearcă trimiterea unui mesaj către un server extern, utilizatorul va primi următorul mesaj de eroare: „**Relay access denied”**. Pentru remedierea acestei probleme, configurația server-ului trebuie alterată pentru a permite autentificarea prin protocolul **SASL** (Simple authentication security layer).

Protocolul **TLS** (Transport Layer Security) este un protocol criptografic ce oferă comunicații sigure peste rețele nesigure – precum Internet-ul.

Combinația **SALS + TLS** este foarte folosită, oferind o metodă de autentificare a utilizatorilor înaintea trimiterii efective către servere externe. Prin această implementare, se restricționează sensibil funcția de **relay**; este esențial ca server-ul de email să fie protejat în acest fel, pentru a nu permite accesul neautorizat la acesta.

Pasul 6. Instalarea modulului SASL se face prin intermediul comenzii:

```
apt-get install sasl2-bin libsasl2-2 libsasl2-modules
```

După instalarea modulelor necesare, fişierul de configurare al server-ului **Postfix** trebuie modificat, pentru a implementa noul mediu. Pentru a configura server-ul **Postfix** astfel încât acesta să suporte autentificarea **SASL**, trebuie adăugate următoarele linii în fişierul **/etc/postfix/main.cf**:

```
smtpd_sasl_auth_enable = yes
smtpd_sasl_local_domain = exemplu.ro
smtpd_recipient_restrictions =
permit_mynetworks,permit_sasl_authenticated,reject_unauth_
destination
smtpd_sasl_security_options = noanonymous
```

Pasul 7. Ajustarea daemon-ului saslauthd astfel încât acesta să poată comunica cu server-ul **Postfix**, care foloseşte un concept de securitate foarte eficient numit **chroot** (mai multe detalii pot fi găsite în secțiunea dedicată securității). Pentru a permite server-ului **Postfix** comunicarea cu daemon-ul SASL, trebuie executate următoarele comenzi:

```
rm -r /var/run/saslauthd/
mkdir -p /var/spool/postfix/var/run/saslauthd
ln -s /var/spool/postfix/var/run/saslauthd /var/run
chgrp sasl /var/spool/postfix/var/run/saslauthd
adduser postfix sasl
cp /etc/sasldb2 /var/spool/postfix/etc/sasldb2
chmod a+r /var/spool/postfix/etc/sasldb2
```

Ultimele două linii se referă la fişierul ce va conține datele de autentificare pentru serviciul **saslauthd**. Postfix (rulând implicit **chrooted**) nu va putea accesa direct fişierul **/etc/sasldb2**. Ca urmare acesta trebuie copiat într-o locație unde server-ul **Postfix** să-l poată accesa. De asemenea, în momentul creării, acesta nu conține nicio înregistrare. Pentru a adăuga utilizatori în acest fişier, trebuie folosit utilitarul **saslpasswd2** în felul următor:

```
saslpasswd2 –f /var/spool/postfix/etc/sasldb2 –u exemplu.ro –a
smtpauth (utilizator)
```

Comanda anterioară poate fi interpretată astfel: parametrul **–f** este folosit pentru specificarea fişierului de parole (în acest caz **sasldb2**), **-u** se foloseşte pentru specificarea domeniului iar **-a** se foloseşte pentru specificarea numelui aplicației folosite. Ultimul parametru reprezintă numele contului de utilizator pentru care se va adăuga înregistrarea. După executarea comenzii, administratorul va fi rugat să introducă parola pentru numele de cont specificat.

Pasul 8. Ajustarea server-ului Dovecot trebuie făcută pentru a specifica socket-ul deamon-ului de autentificare. În acest scop trebuie modificat fişierul **/etc/dovecot/dovecot.conf** prin localizarea liniei care începe cu „**auth default”** şi inserarea următoarelor linii înaintea acesteia:

```
auth default {
mechanisms = plain login
passdb pam {
}
userdb passwd {
}
socket listen {
client {
path = /var/spool/postfix/private/auth
mode = 0660
user = postfix
group = postfix
}
}
}
```

Sintaxa prezentată mai sus trebuie respectată la caracter, orice abatare ducând la incapacitatea server-ului **Dovecot** de a fi pornit.

Pasul 9. Repornirea tuturor componentelor ce alcătuiesc sistemul email:

```
/etc/init.d/saslauthd restart
/etc/init.d/postfix restart
/etc/init.d/dovecot restart
```

Testarea serviciului de email presupune folosirea unui client de email configurat corespunzător. Server-ul a fost configurat astfel încât să nu permită trimiterea mesajelor electronice fără o autentificare în prealabil, limitând astfel posibilitatea ca acesta să fie folosit în moduri nedorite. Pentru a putea autentifica un utilizator al sistemului, acesta trebuie neapărat să posede o înregistrare în fişierul **sasldb2** pentru a putea fi validat.

Fiind strâns legat de funcționarea server-elor DNS, administratorul trebuie să se asigure că domeniul de care răspunde este înregistrat corect şi conține cel puțin o înregistrare de tip **MX** pentru identificarea server-ului de email, cât şi să fie localizabil prin procedura de **reverse lookup**. Multe servicii de email cunoscute (precum **Gmail** sau **Yahoo!**) au serverele configurate astfel încât să refuze orice fel de mesaje de la servere a căror adresă nu o pot identifica în mod corespunzător, în încercarea de a limita trimiterea nesolicitata de email-uri (**spam**).

De asemenea se recomandă ca administratorul să supravegheze starea server-ului **Postfix** prin intermediul fişierelor de jurnalizare (localizate în mod implicit la adresa **/var/log/mail.log**), cât şi starea cozii de mesaje a server-ului. Coada de mesaje (în curs de trimitere) se poate vizualiza prin executarea comenzii: **postqueue –p** iar ştergerea tuturor mesajelor din acesta (în cazul în care acest lucru se doreşte) se poate realiza prin comanda: **postsuper –d ALL.**

Instalarea unei platforme webmail

Opțional, adminstratorul poate instala şi un modul de **webmail** (oferind în acest fel acces la server-ul de mail din orice locație, prin intermediul serviciului **HTTP**). Modulul **webmail** se numeşte **SquirrelMail** şi este una dintre cele mai populare platforme de acest gen. **SquirrelMail** foloseşte tehnologia **PHP** (deci necesită un server web cu modulul PHP activat), cu suport nativ pentru protocoalele **IMAP** şi **POP3**. **SquirrelMail** este foarte uşor de configurat şi oferă o gamă largă de facilități precum suport **MIME,** agenda de contacte şi suport pentru directoare. Un alt avantaj major al platformei îl reprezintă posibilitatea extinderii funcționalităților de baza prin intermediul unor module adiționale (**plugins**).

Pasul 1. Instalarea aplicației SquirrelMail se poate realiza prin executarea comenzii:**apt-get install squirrelmail**

Vom presupune că server-ul deja rulează o instanță a server-ului web **apache2**, având modulul php activat. În cazul în care server-ul web nu este instalat, puteți urma exemplul de instalare pentru serviciul **HTTP**, unde este descris în detaliu întregul proces.

În mod implicit, aplicația poate fi accesată prin orice browser web la adresa:

http://www.exemplu.ro/squirrelmail/

Versiunile mai vechi de **SquirrelMail** necesitau includerea fişierului de configurație pentru server-ul **apache2**, însă în prezent această operație nu mai este necesară. În momentul instalării se va crea o legătură simbolică în directorul **/etc/apache2/conf.d** către fişierul **/etc/squirrelmail/apache.conf**, iar aplicația va fi accesibilă prin intermediul host-ului virtual principal.

Pasul 2. Configurarea aplicației se poate realiza foarte uşor prin intermediul script-ului **/usr/sbin/squirrelmail-configure**. Acesta oferă o interfață de consolă destul de uşor de utilizat, prin care se pot modifica parametrii funcționali ai aplicației, de la titlu şi logo până la modulele adiționale folosite (**plugins**). Deşi aplicația respectă o serie de protocoale standard şi va funcționa aproape sigur fără nici un fel de configurare prealabilă, se recomandă modificarea unor parametri funcționali prin intermediul script-ului **squirrelmail-configure**. Se va selecta din meniul principal opțiunea **2** (**Server Settings),** urmată de tastă ENTER, după care se va selecta opțiunea **A** (**IMAP settings)**. În final se va alege opțiunea **8**, prin intermediul căreia se va specifica server-ul **IMAP** folosit, afişându-se pe ecran o scurtă descriere urmată de o listă de servere **IMAP** cunoscute. În cazul de față utilizatorul trebuie să tasteze **dovecot** şi să apese tastă ENTER pentru a selecta server-ul **IMAP.** Această procedură va asigura compatibilitatea totală cu server-ul **IMAP** folosit, eliminând astfel o serie de probleme ce pot să apară în timpul trimiterii şi primirii mesajelor. **SquirrelMail** este o platformă cu destulă vechime şi foarte stabilă, şi ca atare poate fi implementată pe sisteme în producție, însă se recomandă o configurare detaliată, în acest caz. Sistemul de jurnalizare va fi comun cu cel al server-ului de mail, toate mesajele înregistrate putând fi accesate în fişierul **/var/log/mail.log**.

Instalarea şi configurarea server-ului de DHCP

DHCP (Dynamic Host Configuration Protocol) este un protocol folosit pentru atribuirea dinamică a unor adrese IP unor dispozitive conectate într-o rețea. Serverele **DHCP** sunt folosite pentru automatizarea procesului de desemnarea a parametrilor de rețea mai multor dispozitive, fiind foarte uşor ca administratorul să adauge în timp foarte scurt noi sisteme în rețea. Clienții **DHCP** vor transmite o cerere server-ului **DHCP** şi în final, vor primi informațiile cerute. Server-ul este responsabil de o serie de adrese IP, precum şi informații despre conexiune precum **gateway**, **numele de domeniu, servere DNS** şi aşa mai departe, şi, în urma unei cereri din partea unui client, server-ul va desemna acestuia o adresă IP (precum şi alte detalii necesare conexiunii – **subnet mask, gateway, servere DNS**) precum şi durata autorizată (**lease time**). Având în vedere faptul că mare parte din proces se întâmplă fără că clientul să aibă stabilită o adresă IP, acesta începe de obicei imediat după procesul de **boot**, pentru a putea permite comunicația bazată pe adrese IP între dispozitive. Serverele **DHCP** au două moduri funcționale principale: modul **dinamic**, în care clienții vor primi adrese IP în baza unor cereri, din gama de adrese de care dispune server-ul (în funcție de disponibilitate, sau preferențial, în funcție de adresa precedentă a clientului) şi modul **static** în care clienții vor primi adresele IP într-o manieră semi-statica, bazată pe asocierea cu adresa MAC a clientului (în acest mod, clienții configurați în mod explicit vor avea mereu aceeaşi adresă IP, furnizată de server în baza adresei MAC a clientului).

Alegerea unui server **DHCP** care să ruleze pe o platformă Debian trebuie aşadar să țină de o serie de factori: acesta trebuie să fie stabil, uşor de instalat şi configurat, şi nu în ultimul rând să corespundă cerințelor rețelei în care va rula.

Distribuția Debian 5.0 include în lista sa şi pachetul **dhcp3-server**. Acesta reprezintă implementarea **ISC** (Internet Software Consortium) a server-ului **DHCP**, şi poate lucra cu mai multe interfețe de rețea simultan. Vom prezenta în continuare procesul de instalare şi configurare al unui server **DHCP** în baza pachetului menționat anterior.

Pasul 1. Instalarea server-ului DHCP se poate realiza prin instalarea pachetului menționat anterior, prin executarea comenzii:

```
apt-get install dhcp3-server
```

Instalarea pachetului se va realiza printr-o aplicația de tip wizzard, şi în mod implicit va atentiona administratorul asupra câtorva facilități de care server-ul dispune. Primul mesaj specifică faptul că server-ul ce urmează a fi instalat este o versiune **non-authoritive**, adică server-ul va transmite tuturor clienților să nu mai utilizeze adrese pentru care acesta a primit o cerere din partea unui client, şi de care acesta nu răspunde.

Odată instalat, server-ul trebuie configurat prealabil pornirii sale, nefiind setat să asculte cereri pe nicio interfață în mod implicit. Dacă administratorul încearcă să lanseze daemon-ul **DHCP** prin intermediul comenzii

„**/etc/init.d/dhcp3-server start**”, acesta va primi un mesaj de eroare conform căruia este atenționat că serviciul **DHCP** nu a putut fi pornit. În general, este recomandată vizualizarea jurnalelor unui serviciu în cazul în care administratorul întâmpină dificultăți cu acesta, iar server-ul **DHCP** nu este o excepție, fişierul **/var/log/syslog** fiind locația în care server-ul îşi va salva toate mesajele de eroare sau avertizare.

Pasul 2. Configurarea server-ului DHCP

Configurarea server-ului poate fi realizată în mod similar cu alte servicii, prin modificarea fişierului de configurare **/etc/dhcp3/dhcpd.conf**. Structura acestuia este una cunoscută, fiind folosită o sintaxă strictă pentru descrierea parametrilor funcționali. Ca şi în alte fişiere de configurare, liniile care încep cu caracterul „#” reprezintă comentarii şi vor fi ignorate din punct de vedere funcțional de către server. În continuare vom folosi un exemplu de fişier de configurare de bază, parcurgând toți parametrii funcționali esențiali pentru funcționarea server-ului, însă înainte de orice modificare a fişierului de configurare se recomandă crearea unei copii de siguranță a acestuia.

În scopul acestui exemplu, vom configura server-ul **DHCP** să răspundă de domeniul **exemplu.ro**, să cunoască server-ele **DNS** incluse, să răspundă de o anumită gamă de adrese IP şi nu în ultimul rând, să funcționeze şi în modul static – pentru anumite adrese MAC. Astfel, trebuie adăugată următoarele linii de configurare în cadrul fişierul **/etc/dhcp3/dhcpd.conf:**

```
option domain-name "exemplu.ro";
option domain-name-servers 192.168.2.102;
option routers 192.168.2.1;
default-lease-time 36000;
subnet 192.168.2.0 netmask 255.255.255.0 {
range 192.168.2.150 192.168.2.200;
}
```


Liniile de mai sus vor descrie funcționarea server-ului după cum urmează:

- **option domain-name ”exemplu.ro”** – defineşte numele domeniului curent; urmând contextul prezentat până în acest moment, vom folosi domeniul **exemplu.ro**.
- **option domain-name-servers 192.168.2.102** – defineşte serverele **DNS** ce vor fi transmise clienților pentru rezolvarea numelor de domenii în adrese IP. În acest caz, singurul server DNS este **192.168.2.102**, aceeaşi adresă folosită şi la configurarea server-ului DNS în secțiunile anterioare. Această opțiune permite înşiruirea mai multor servere, separate prin virgule, acestea urmând să fie folosite de către clienți în ordinea în care apar: un exemplu bun pentru mai multe servere DNS ar fi **option domain-name-servers 192.168.2.102, 192.168.2.103;**
- **option routers 192.168.2.1 –** specifică care va router-ul folosit de către clienți. La fel ca şi în cazul precedent, pot fi specificate mai multe routere, delimitate prin virgule, acestea urmând să fie folosite în ordinea în care sunt menționate.

- **default-lease-time 36000** – specifică durata de folosire (măsurată în secunde) pentru clienții care nu cer în mod specific o durată de închiriere. În acest caz, durata implicită este de 10 ore, sau 36000 secunde.

Aceste opțiuni reprezintă parametri **globali**, care de regulă descriu detalii generale, valabile pentru o organizație întreagă (de exemplu servere DNS sau numele de domeniu). În continuare vom examina şi celelalte opțiuni, care se referă în mod specific la subretele individuale:

- **subnet 192.168.2.0 netmask 255.255.255.0** – foloseşte o structură delimitată prin acolade, pentru a descrie o subretea pe care server-ul o va deservi. Adresa IP menționată este folosită pentru a permite server-ului să determine dacă o adresă aparține sau nu unei subretele şi pentru a transmite clienților configurații particulare. În cazul de față, am definit subreteaua 192.168.2.* cu masca 255.255.255.0.
- **range 192.168.2.150 192.168.2.200** – specifică gama de adrese disponibile pentru atribuirea automată de către server-ul **DHCP**, şi în cazul exemplului curent, opțiunea este inclusă în cadrul structurii subnet menționate. Cele două adrese IP menționate că parametri specifică adresa minimă, respectiv adresa maximă de care server-ul **DHCP** va răspunde pentru subreteaua actuală. Astfel, server-ul va fi configurat să atribuie adrese IP clienților săi, începând cu adresa **192.168.2.150** şi terminând cu adresa **192.168.2.200**

OBSERVAȚIE: Sintaxa prezentată în poza anterioară trebuie respectată întocmai, mai ales în cazul caracterului „**;**” folosit pentru a specifica sfârşitul unei opțiuni.

Pasul 3 (Opțional). Configurarea adreselor statice se poate realiza în cazul sistemelor mai importante unde se vrea totuşi o configurație bazată pe **DHCP** – cazul unui server DNS local de exemplu – dar cu o serie de adrese statice. Pentru adăugarea unor adrese statice trebuie adăugate următoarele linii în cadrul fişierului de configurare **/etc/dhcp3/dhcpd.conf**:

```
host server {
hardware ethernet 00:0c:29:87:a3:4f;
fixed-address 192.168.2.199;
}
```

Liniile de mai sus definesc o nouă entitate numită „**server”**, prin intermediul directivei **host**. Ca şi în cazul subretelelor, directiva **host** delimitează prin acolade o structură ce va conține parametri funcționali pentru o entitate specifică.

- **hardware ethernet 00:0c:29:87:a3:4f –** specifică adresa MAC a interfeței de rețea asociată cu sistemul „**server”**, cât şi tipul de rețea (ethernet / token-ring).
- **fixed-address 192.168.2.199** – specifică o adresă IP care va fi atribuită unui client. Se pot menționa mai multe adrese separate prin virgule, din care clientul va alege adresa potrivită în funcție de subreteaua din care face parte. Putem afirma, drept concluzie, că am alocat adresa statică

192.168.2.199 sistemului denumit **server,** ale cărui interfețe de rețea are adresa MAC **00:0c:29:87:a3:4f**.

Pasul 4. Repornirea serviciului DHCP este necesară pentru că noua configurație să devină activă. Server-ul poate fi repornit prin intermediul comenzii:

```
/etc/init.d/dhcp3-server restart
```

Respectarea sintaxei menționate în cadrul fişierului de configurare este esențială, utilizatorul fiind atenționat în cazul în care server-ul a detectat o eroare de sintaxă.

Instalarea şi configurarea serviciului Lightweight Directory Access Protocol (LDAP)

Protocolul **LDAP** este un protocol cât şi o arhitectură de organizare a datelor în directoare. Creat ca o inițiativă academică la începutul anului 1995, este o alternativă simplificată a protocolului **DAP** (Data Access Protocol) cu avantajul de a putea rula pe protocolul **TCP/IP.** Principalul rol **LDAP** este structurarea ierarhică a datelor referitoare la rețele (dispozitive, aplicații, utilizatori, adrese şi aşa mai departe) într-o manieră foarte bine definită, în cadrul unor baze date optimizate pentru operații de citire.

În contextul unui server Debian, vom studia în continuare procesul de instalare şi configurare al serviciului **LDAP**, folosind server-ul **OpenLDAP**. Acesta este o implementare open-source a protocolului respectiv, şi va fi folosit pentru a oferi un punct central de autentificare a utilizatorilor - aceştia beneficiand de directoare home create într-o manieră automată – cât şi pentru a servi meta-date despre aceştia. În cadrul procesului de instalare, nu vom vorbi despre securizarea comunicării prin protocolul **LDAP**, însă administratorul trebuie să fie conştient că în mod implicit datele sunt trimise pe mediul de comunicație necriptate.

Privit exclusiv din punct de vedere tehnic, un director **LDAP** este compus dintr-un set de înregistrări organizate ierarhic, unde fiecare înregistrare aparține unei **clase** şi conține perechi de tipul **cheie = valoare** numite **atribute.** Fiecare înregistrare va conține un identificator unic **Distinguished name** (prescurtat „**DN**”), format dintr-o serie de componente delimitate prin virgule, prin intermediul cărora se va specifica adresa relativă (**full path**) către înregistrarea respectivă, având ca punct de referință rădăcina grafului. Clasele, atributele, regulile cât şi sintaxa sunt încărcate în momentul inițializării server-ului **LDAP** din cadrul fişierelor tip **schemă**.

Protocolul nu este asociat în mod direct cu conturile sau datele aleatoare prezente pe un sistem, ci are ca scop stocarea informațiilor obişnuite ce pot fi găsite în cadrul sistemelor Unix, precum fişierele **/etc/passwd** sau **/etc/group**, oferind astfel o interfață de autentificare centralizată într-o rețea, iar datele referitoare la conturile utilizatorilor vor fi asociate cu datele prezente în arborele **LDAP**. Server-ul poate fi configurat astfel încât să conțină şi parolele utilizatorilor pentru facilitarea procesului de autentificare a utilizatorilor în vederea obținerii unor privilegii sporite. – în momentul în care un utilizator se conectează la server-ul **LDAP** pentru a vizualiza conținutul directorului **LDAP**, parola pe care acesta o va furniza va fi folosită pentru a determina atât identitatea sa, cât şi privilegiile de care acesta beneficiază.

Pasul 1. Instalarea server-ului OpenLDAP se poate face prin executarea următoarei comenzi:

```
apt-get install slapd libdb4.6
```

În urma acestei comenzi, utilizatorul va fi rugat să aleagă o parolă pentru înregistrarea **admin** din directorul **LDAP**. Dacă procesul de instalare decurge în mod normal, serviciul **LDAP** va fi pornit la final, folosind configurația implicită.

Pasul 2. **Configurarea server-ului OpenLDAP**

Pentru a păstra un oarecare grad de consecventă cu celelalte servicii prezentate până în acest moment, vom considera instalarea serviciului **LDAP** pentru domeniul **exemplu.ro**. Pentru configurarea server-ului vom folosi una din facilitățile oferite de aplicația **dpkg-reconfigure**, care se ocupă de schimbarea configurației unui pachet deja instalat prin intermediul unui wizzard. Procesul de reconfigurare trebuie lansat prin intermediul comenzii:

```
dpkg-reconfigure slapd
```

În continuare utilizatorul va trebui să furnizeze câteva informații esențiale pentru funcționarea serviciului:

- „**Omit OpenLDAP server configuration**” – **No**
- **„DNS domain name” – exemplu.ro**
- **„Organization name” – exemplu.ro**
- **„Administrator password” – la alegerea utilizatorului**
- **„Confirm password” – la alegerea utilizatorului**
- **„Database backend to use” – BDB**
- **„Do you want the database to be remove when slapd is purged?”–No**
- **„Move old database” – Yes**
- **„Allow LDAPv2 protocol” – No**

Aceste răspunsuri vor garanta funcționarea server-ului în condițiile în care domeniul specificat este susținut de un server DNS. Parametrul **BDB** specifică faptul că se va folosi sistemul de cache **Berkeley DB Cache**, iar ultima opțiune va forța clienții să comunice folosind ultima versiune a protocolului. Penultima întrebare este folosită pentru crearea unui back-up în cazul în care se reconfigurează pachetul după instalare, baza de date cât şi fişierele de configurare urmând să fie mutate din directorul **/etc/ldap**.

Urmează editarea fişierului de configurare **/etc/ldap/slapd.conf** pentru verificare cât şi adăugarea unor parametri adiționali. Administratorul trebuie să se asigure că următoarele linii (care sunt folosite pentru includerea fişierelor schemă) există:

```
include         /etc/ldap/schema/core.schema
include         /etc/ldap/schema/cosine.schema
include         /etc/ldap/schema/nis.schema
include         /etc/ldap/schema/inetorgperson.schema
```

În continuare, trebuie modificați următorii parametri:

- **„loglevel 256” –** este folosit pentru determinarea căror mesaje de jurnal vor fi înregistrate prin **syslogd**. Nivelul este stabilit ca o sumă de puteri ale lui 2, reprezentând biții de configurare. Valoarea 256 specifică că se doreşte jurnalizarea evenimentelor de tip conexiuni/operații/rezultate. Consultați **man slapd.conf** pentru mai multe detalii.
- Localizarea liniei „**index objectClass eq”** şi inserarea liniei „**index uid eq”** imediat după acestea. Această modificare va permite indexarea în vederea căutărilor. Pentru că noile schimbări de index să devine active, trebuie executate următoarele comenzi:

/etc/init.d/slapd stop
slapindex
chown openldap:openldap /var/lib/ldap/*
/etc/init.d/slapd start

În acest moment, server-ul **LDAP** este configurat, şi gata de a fi folosit; acesta nu conține însă date importante, acestea urmând să fie adăugate.

Pasul 3. Testarea server-ului

Testarea server-ului **LDAP** se poate realiza prin efectuarea unor operații de citire, numite în terminologia protocolului LDAP „**search**”. Căutările se pot realiza folosind o serie de utilitare în linie de comandă precum **ldapsearch** sau **slapcat**. Diferența dintre cele două este aceea ca **ldapsearch** (şi restul utilitarelor ce încep cu „**ldap”)** efectuează operații în mod **online**, adică prin intermediul server-ului **OpenLDAP** folosind protocolul specific, pe când **slapcat** (şi restul utilitarelor ce încep cu **„slap**”) citesc fişierele locale – motiv pentru care trebuie rulate local pe sistemul care rulează server-ul **LDAP** cu privilegii de root.

Pentru folosirea utilitarelor din gama **ldap**, trebuie instalat pachetul

apt-get install ldap-utils

şi modificat fişierul de configurare specific tuturor clienților **LDAP** cu un editor text. Modificările vor specifica o serie de parametri generali astfel încât utilitarul să ştie domeniul şi server-ul pentru care urmează să realizeze căutarea. În acest scop, trebuie adăugate următoarele două linii în cadrul fişierului **/etc/ldap/ldap.conf**:

BASE dc=exemplu, dc=ro
URI ldap://192.168.2.102

Domeniul este descris de prima linie, prin atribute de tipul **dc** (Domain Component) unde fiecare subdomeniu este definit, iar adresa server-ului **LDAP** este descrisă de cea de-a doua linie.

În final, trebuie lansat utilitarul:

ldapsearch –x

Parametrul **–x** specifică faptul că se doreşte o autentificare simplă în locul metodei preferate (**SASL)**:

```
# exemplu.ro
dn: dc=exemplu,dc=ro
objectClass: top
objectClass: dcObject
objectClass: organization
o: exemplu.ro
dc: exemplu

# admin, exemplu.ro
dn: cn=admin,dc=exemplu,dc=ro
objectClass: simpleSecurityObject
objectClass: organizationalRole
cn: admin
description: LDAP administrator
```

Rezultatul va fi afişat pe ecran, utilizatorul putând să identifice două înregistrări diferite, una pentru nivelul de top al arborelui LDAP, şi una pentru înregistrarea administratorului.

Pe lângă **ldapsearch,** se mai poate folosi şi comanda **slapcat** fără nici un parametru, care va afişa un rezultat de genul:

```
dn: dc=exemplu,dc=ro
objectClass: top
objectClass: dcObject
objectClass: organization
o: exemplu.ro
dc: exemplu
structuralObjectClass: organization
entryUUID: 8124d13a-83ab-102e-940e-511d1aaa1f55
creatorsName:
createTimestamp: 20091223010903Z
entryCSN: 20091223010903.300875Z#000000#000#000000
modifiersName:
modifyTimestamp: 20091223010903Z

dn: cn=admin,dc=exemplu,dc=ro
objectClass: simpleSecurityObject
objectClass: organizationalRole
cn: admin
description: LDAP administrator
userPassword:: e2NyeXB0fVB4dkNUWEJmWUFLSlU=
structuralObjectClass: organizationalRole
entryUUID: 81258d1e-83ab-102e-940f-511d1aaa1f55
creatorsName:
createTimestamp: 20091223010903Z
entryCSN: 20091223010903.305814Z#000000#000#000000
modifiersName:
modifyTimestamp: 20091223010903Z
```


Diferența dintre cele două utilitare devine aparentă în urma examinării rezultatelor afişate: **slapcat** va afişa şi atribute pe care **ldapsearch** le va omite, pe motive de securitate – unul dintre acestea fiind câmpul **userPassword** care nu ar trebui dezvălui utilizatorilor anonimi.

Pasul 4. Crearea structurii arborelui

Bazele de date **LDAP** sunt structurate ca arbori, având rădăcina (elementul de top al arborelui fiind de multe ori chiar numele domeniului pe care îl reprezintă). Coborând mai jos în ierarhia arborelului, organizația poate fi împărțită în mai multe entități precum oameni, grupuri, servicii, rețele, etc. În cazul exemplului de față, vom crea două entități diferite numite **Oameni** şi **Grupuri**, cele două având corespondență la fişierele tipice Unix **/etc/passwd** respectiv **/etc/group**.

Comenzile şi datele **LDAP** se schimba între client şi server folosind un format text numit **LDIF**, în care sunt incluse comenzile şi datele aferente comunicării. În acest scop, vom crea un fişier temporar numit **/var/tmp/ldap** care va avea următorul conținut:

```
dn: ou=Oameni,dc=exemplu,dc=ro
ou: Oameni
objectClass: organizationalUnit

dn: ou=Grupuri,dc=exemplu,dc=ro
ou: Grupuri
objectClass: organizationalUnit
```

Acest fişier va fi folosit pentru descrierea celor două entități **Oameni** şi **Grupuri** ce urmează a fi adăugate în baza de date **LDAP**. Pentru că schimbările să fie activate, acest fişier va trebui încărcat (folosind utilitarul **slapadd**) după ce server-ul a fost oprit:

```
/etc/init.d/slapd stop
slapadd –c –v –l /var/tmp/ldap
/etc/init.d/slapd start
```

Seria de parametri pentru utilitarul **slapadd** sunt: **-c,** va forța ignorarea erorilor, **-v,** va activa modul „verbose” iar **–l** este folosit pentru specificarea unui fişier de tip **LDIF**. După repornirea server-ului, este recomandată executarea comenzii **ldapsearch –x** pentru a verifica dacă cele două înregistrări de mai sus au fost într-adevăr adăugate în baza de date.

Pasul 5. Crearea conturilor de utilizator

Crearea conturilor de utilizator se face în aceeaşi manieră ca şi crearea entităților precedente, prin intermediul fişierelor LDIF. Astfel, trebuie creat un fişier temporar care va fi încărcat în baza de date **LDAP** cu ajutorul utilitarului **ldapadd**. Fişierul temporar în acest caz va fi **/var/tmp/user** şi va conține următoarele linii:

```
dn: cn=user10,ou=Grupuri,dc=exemplu,dc=ro
cn: user10
gidNumber: 5000
objectClass: top
objectClass: posixGroup

dn: uid=user10,ou=Oameni,dc=exemplu,dc=ro
uid: user10
uidNumber: 5000
gidNumber: 5000
cn: user10
sn: user10
objectClass: top
objectClass: person
objectClass: posixAccount
objectClass: shadowAccount
loginShell: /bin/bash
homeDirectory: /home/user10
```

Încărcarea fişierului LDIF în directorul **LDAP** se va face cu ajutorul comenzii:

```
ldapadd –c –x –D cn=admin,dc=exemplu,dc=ro –W –f
/var/tmp/user
```

Parametrii au următoarele semnificații:

- **-c** – va forța continuarea chiar şi în cazul erorilor
- -**x** – foloseşte autentificare simplă în loc de SASL
- -**D [text]** – foloseşte textul drept Distinguished Name
- **-W –** cere parolă pentru autentificare în loc să fie specificată ca parametru
- -**f –** specifică un fişier LDIF drept sursă.

În cazul conținutului fişierului LDIF menționat anterior, putem identifica următoarele tipuri de atribute:

- cn – common name
- ou – Organizațional Unit
- dc – domain component
- sn – surname
- uid – user ID

În final, trebuie creată o nouă parolă pentru utilizatorul **user10** prin intermediul utilitarului **ldappasswd**:

```
ldappasswd –x –D cn=admin,dc=exemplu,dc=ro –W –S
uid=user10,ou=Oameni,dc=exemplu,dc=ro
```

În urma executării acestei comenzi, administratorul va trebui să aleagă o nouă parolă pentru utilizatorul **user10**, şi în final să introducă parola pentru **admin**.

OBSERVAȚIE: Crearea parolelor nu mai este necesară în cazul în care se foloseşte **Kerberos,** din moment ce acesta va fi responsabil pentru stocarea parolelor şi autentificarea utilizatorilor.

Pasul 6. Instalarea şi configurarea modulului NSS

Pentru a permite sistemului să recunoască noul utilizator **user10**, trebuie instalate următoarele pachete:

```
apt-get install libnss-ldap
```

Wizzard-ul de instalare va cere o serie de informații referitoare la server-ul LDAP, la care administratorul trebuie să răspundă în felul următor:

- „**LDAP server Uniform Resource Identifier" – ldap://192.168.2.102** – adică adresa server-ului
- **„Distinguished name of the search base"** – **dc=exemplu,dc=ro** – adică componentele domeniului de care răspunde server-ul
- „**LDAP version to use" – 3**
- **„Does the LDAP database require login" – No**
- **„Special LDAP privileges for root" – No**
- **„Make the configuration file readable/writeable by its owner only" – No**
- **„Make local root Database admin" – No**
- **„Does the LDAP database require login" – No**
- **„Local crypt to use when changing passwords" – crypt**

Modulul **NSS** trebuie configurat mai departe astfel încât să funcționeze corespunzător prin adăugarea liniilor următoare în fişierul **/etc/libnss-ldap.conf** (în cazul în care acestea nu există deja):

```
base dc=exemplu,dc=ro
uri    ldap://192.168.2.102
```

Deşi vă fi activ, modulul **NSS** nu va intra în mod implicit în acțiune datorită modului în care este configurat sistemul. Ca atare, trebuie modificat fişierul **/etc/nsswitch.conf** pentru a schimba ordinea în care se verifică sursele pentru autentificare. În acest scop trebuie modificate următoarele două linii din cadrul fişierului **nsswitch.conf**:

```
passwd:         files ldap
group:          files ldap
```

În acest moment utilizatorul **user10** este activ pentru sistem şi poate fi folosit. Verificarea se poate face prin comanda „**id user10”** care ar trebui să afişeze informații despre contul respectiv.

OBSERVAȚIE: Dacă utilizatorul **user10** nu este vizibil, motivul cel mai probabil este serviciul **nscd** care oferă posibilitatea de caching pentru fişiere precum **passwd, group** sau **hosts**. Pentru verificarea noului utilizator acest serviciu trebuie oprit de regulă temporar, prin executarea comenzii **/etc/init.d/nscd stop**. În acest fel se va asigura citirea directă a fişierelor importante.

Pasul 7. Instalarea şi configurarea modulului PAM

Aşa cum am menționat în secțiunile anterioare, **PAM** (Pluggable Authentication Modules) este în esență un mecanism de autentificare a utilizatorilor. Problema principală a mecanismelor noi de autentificare este aceea că toate daemon-urile care depind de acestea trebuie rescrise pentru a suporta noile facilități. **PAM** oferă o alternativă de dezvoltare a programelor independentă de modul de autentificare; aceste programe folosesc module de autentificare pentru a îndeplini funcțiile acestora.

Instalarea modulului anterior a avut drept consecință imediată şi instalarea pachetului **libpam-ldap**, însă în cazul în care acesta nu a fost instalat din diverse motive se poate instala folosind platforma **apt**. De asemenea, în cadrul procesului de instalare, utilizatorul a trebuit să răspundă la o serie de întrebări de configurare prin intermediul wizzard-ului de instalare – acestea sunt ultimele 3 întrebări de la punctul anterior.

Pentru a asigura configurația corectă a modulului **PAM**, trebuie verificată existența următoarelor două linii în fişierul **/etc/pam_ldap.conf**:

```
base dc=exemplu,dc=ro
uri    ldap://192.168.2.102
```

Liniile anterioare urmăresc ideea de configurare prezentată până în acest punct. Vom modifica în continuare o serie de fişiere de configurare ale modulului **PAM** – cu atenție însă, pentru ca sintaxă este una foarte strictă şi **orice** fel de greşeală poate schimba radical configurația. În acest scop, trebuie modificate următoarele fişierele în modurile descrise:

- Fişierul **/etc/pam.d/common-account** trebuie să arate în felul următor:

```
accountsufficient  pam_unix.so
accountrequired  pam_ldap.so
```

- Fişierul **/etc/pam.d/common-auth** trebuie să arate în felul următor:

```
auth  [success=1  default=ignore] pam_unix.so  nullok_secure
auth  required  pam_ldap.so  use_first_pass
auth  required  pam_permit.so
```

- Fişierul **/etc/pam.d/common-session**

```
session required  pam_unix.so
session required  pam_mkhomedir.so  skel=/etc/skel  umask=0022
```

OBSERVAȚII: Modulul **PAM** necesită existența datelor despre un cont fie în fişierul **/etc/passwd** fie în directorul **LDAP**, pentru ca procesul de autentificare să poată fi finalizat. Reamintim pe această cale, faptul că prin natura lor, conexiunile **LDAP** nu sunt criptate, deci pot reprezenta un risc de securitate. Pentru securizarea procedurii de autentificare, consultați secțiunea dedicată de securitate. De asemenea, sistemele de operare Microsoft, precum **Windows XP/2000** nu vor şti să se autentifice folosind altceva în loc de Microsoft Active Directory.

Există posibilitatea configurării stațiilor ce folosesc MS **Windows XP/2000** astfel încât acestea să poată folosi server-ul **OpenLDAP** pentru autentificare, însă este nevoie de înlocuirea modulului de autentificare înglobat în cadrul sistemului de operare (pentru fiecare stație în parte ce va folosi protocolul **LDAP** pentru autentificare) prin urmărirea pasului următor.

Pasul 8. (OPȚIONAL) Înlocuirea modulului de autentificare Windows XP/2000

Acest pas se poate realiza folosind aplicația **pGina**, cu site-ul disponibil la adresa **http://www.pgina.org**. Arhitectura **pGina** este modulară, fiind folosite **plug-in-uri** pentru sarcini diferite. Administratorul va trebui să descarce atât programul **pGina** cât şi colecția de **plug-in-uri** asociată şi să le instaleze în parte pe toate sistemele ce rulează Microsoft Windows ce vor urma să folosească protocolul **LDAP** pentru autentificare. Pachetele necesare acestui pas pot fi descărcate de la adresa **http://sourceforge.net/projects/pgina/files/** şi sunt enumerate mai jos:

- **pGina Core -> pGina 1.8.8**
- **Plugin Bundle -> Plugin Bundle 12-30-2006**

La momentul scrierii acestui suport, versiunile pachetelor menționate anterior erau ultimele versiuni oficiale. În cazul în care link-urile prezentate anterior nu mai sunt valide, noile locații de descărcare ale aplicației pot fi descoperite accesând site-ul principal. După descărcarea şi instalarea pachetelor de mai sus, folosind utilitarul **pGina Configuration Tool** vom configura plug-in-ul **ldapauth_plus.dll** folosind următorii parametri:

- **LDAP Server – adresa ip sau numele de sistem al server-ului LDAP (**conform scenariului urmărit până acum aceasta va fi **192.168.2.102** sau **exemplu.ro)**.
- **Port – „389”** este port-ul standard
- **Admin user – „cn=admin,dc=exemplu,dc=ro”**
- **Admin pass – parola utilizatorului admin de pe server**
- **PrePend – „uid=”**
- **Append – „ou=Oameni,dc=exemplu,dc=ro”**

După configurarea tuturor parametrilor, trebuie bifat butonul **Map Mode**, şi apăsat butonul **OK**. Odată cu repornirea sistemului pe care a fost instalat, dialogul de autentificare normal va fi schimbat cu unul nou, care va permite atât autentificarea utilizatorilor locali ai sistemului, cât şi conform înregistrărilor din cadrul directorului **LDAP.** Este important de ştiut însă că această modificare va dezactiva procesul **Fast User Switching** şi că modulul original de autentificare al sistemului de operare poate fi restaurat prin dezinstalarea aplicației **pGina**.