Introducere în securitatea sistemului *Oracle*

Cerinţele de securitate a bazelor de date provin din necesitatea protejării datelor, fie de pierderea sau coruperea accidentală a lor, fie de încercările neautorizate deliberate de a accesa sau a modifica datele. Acest din urmă aspect include protecţia împotriva întârzierilor în accesarea şi utilizarea datelor, sau împotriva interferenţei care poate conduce chiar la refuzul unui serviciu. Costurile globale cauzate de astfel de „breşe" de securitate se ridică la miliarde de dolari anual, iar costurile companiilor individuale pot fi foarte severe.

Cerinţele referitoare la securitate sunt dinamice. Tehnologii şi practici noi furnizează în continuu noi „arene" pentru exploatarea neautorizată, dar şi modalităţi noi de utilizare greşită, accidentală sau deliberată care afectează chiar şi produse sau medii stabile. Evoluţia actuală implică un mediu tehnologic şi cultural în schimbare la nivel global, în care problemele de securitate afectează atât utilizarea soluţiilor existente, cât şi dezvoltarea unora noi.

Pe măsură ce cerinţele de securitate sunt înţelese mai clar, anumite principii generale pot fi dezvoltate pentru satisfacerea lor şi pentru anularea ameninţărilor împotriva acestora, ce derivă din vulnerabilităţile Internetului. Aceste principii pot varia din punct de vedere al eficienţei. Implementările lor variază, de obicei, în privinţa costului: achiziţie şi mentenanţă *hardware* şi *software*, personal administrativ şi de programare, precum şi impactul măsurilor de securitate asupra timpilor de procesare şi răspuns. Costul total include, de asemenea, pe cel al gestiunii *hardware*, *software*, personalului, eficienţei, timpului de răspuns etc. Aceste costuri pot escalada odată cu creşterea volumului de utilizatori, tranzacţii şi tipuri de date.

Elementele şi operaţiile de bază ale mediilor de baze de date includ conexiunea la un server sau la o schemă, accesul şi modificarea tabelelor, precum şi utilizarea aplicaţiilor. Securizarea acestora împotriva utilizării eronate, accidentale sau deliberate, sunt sarcinile responsabililor de securitate (*security officers*), a administratorilor şi a programatorilor de aplicaţii. Aceştia trebuie, de asemenea, să administreze şi să protejeze drepturile utilizatorilor interni ai bazei de date, şi să garanteze confidenţialitatea atunci când clienţii accesează bazele de date de oriunde de pe Internet.

În epoca Internetului, spectrul de riscuri asupra datelor, asociate accesului şi confidenţialităţii utilizatorului, este mai larg decât oricând. Figura 1 arată un scenariu clasic de

mediu de calcul, suficient de relevant pentru a exemplifica complexitatea mediilor pe care planurile de securitate trebuie să le protejeze.

Diagrama arată câteva aspecte importante ale securităţii, ilustrând comunităţile de clienţi, conexiunile, bazele de date şi serverele, toate acestea trebuind să fie securizate împotriva accesului sau utilizării inadecvate. Aceste arii pot necesita diferite tehnici pentru realizarea unei bune securităţi şi trebuie să se integreze astfel încât să excludă sau să minimizeze breşele de securitate sau vulnerabilităţile.

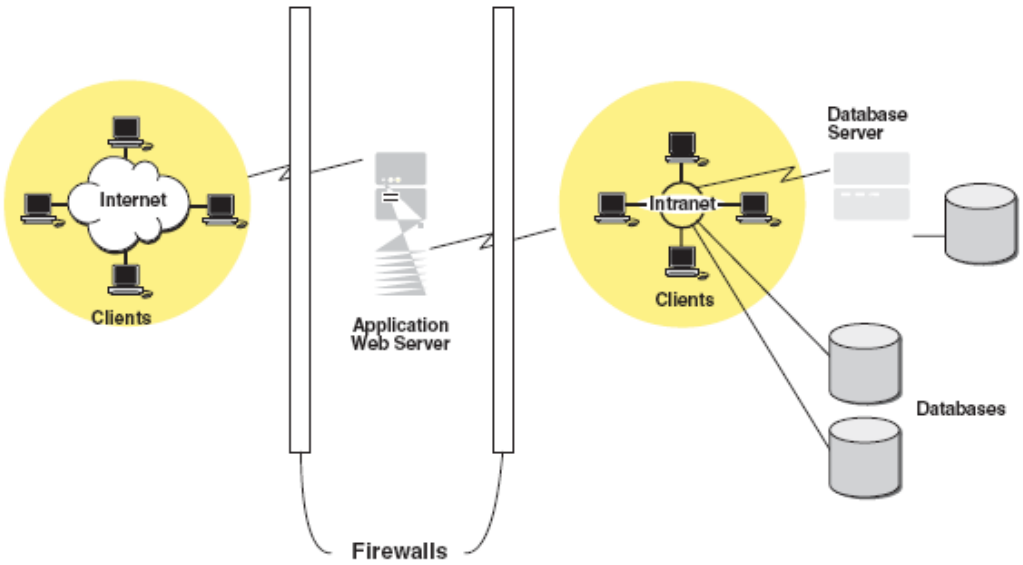

Figura 1. Domenii de securitate în lumea Internetului

Tabelul de mai jos arată categoriile care se disting în concentrarea eforturilor pentru a crea aplicaţii sigure în medii sigure. Atunci când aceste componente necesare sunt consistente în privinţa cerinţelor de securitate, coerente în privinţa modalităţilor în care conlucrează şi complete prin închiderea tuturor canalelor cunoscute de atac sau utilizare eronată, securitatea este cât se poate de bună.

Dimensiune	Probleme de securitate
Fizică	Computerele trebuie să fie inaccesibile fizic utilizatorilor neautorizaţi prin păstrarea lor într-un mediu fizic sigur.
Personal	Persoanele responsabile de securitatea fizică, administrarea sistemului şi securitatea datelor trebuie să fie de încredere. Efectuarea unor verificări înaintea luării deciziei de angajare a unui *DBA* este o măsură de protecţie.

Procedural	Procedurile şi politicile folosite în operarea sistemului trebuie să asigure date de încredere. Adesea, este bine să separăm rolurile funcţionale ale utilizatorilor în gestiunea datelor. De exemplu, o persoană poate fi responsabilă de *backup*-urile bazei de date, având doar rolul de a se asigura că baza de date este pornită şi activă. Altă persoană poate fi responsabilă de generarea rapoartelor aplicaţiei asupra datelor privind plăţile şi vânzările, având rolul de a examina datele şi de a le verifica integritatea. Mai departe, pot fi stabilite politici care protejează tabelele şi schemele împotriva utilizării neautorizate, accidentale sau răuvoitoare.
Tehnic	Stocarea, accesul, prelucrarea şi transmiterea datelor trebuie securizate prin tehnologii care impun politici particulare de control al informaţiei.

1. Gestiunea identităţilor: securitate în medii complexe, cu volume mari de date

În plus faţă de problemele generale şi categoriile de securitate, numărul de utilizatori şi activităţi ce necesită siguranţă adaugă o altă dimensiune a complexităţii.

Deoarece numărul de utilizatori, baze de date, aplicaţii şi reţele este în continuă creştere, complexitatea interacţiunilor lor creşte exponenţial. În aceeaşi măsură cresc riscurile, dar şi *task*-urile de gestiune necesare pentru a menţine securitatea şi eficienţa.

De exemplu, numărul de interacţiuni pentru 10 utilizatori care accesează 5 baze de date poate fi egal cu 50. Dacă adăugăm 90 de utilizatori şi 45 de baze de date, obţinem 100 de utilizatori care accesează 50 de baze de date, având un număr de 5000 de interacţiuni posibile. Dacă adăugăm acestui exemplu un număr de aplicaţii şi reţele, avem o perspectivă a complexităţii extreme, cu riscuri de securitate direct proporţionale. O breşă de securitate în reţea poate ameninţa securitatea bazelor de date şi a utilizatorilor săi, dar şi pe cea a altor reţele conectate, având la rândul lor baze de date şi utilizatori.

Acest tip de mediu complex necesită viteză şi flexibilitate în acordarea sau revocarea drepturilor de acces pentru orice utilizator sau resursă. Întârzierile din procesele administrative sau cele de implementare a drepturilor în bazele de date corespunzătoare implică fie un acces legitim întârziat, fie acordarea accesului atunci acesta când ar trebui refuzat.

De exemplu, atunci când un angajat cu acces la mai multe aplicaţii şi conturi pe mai multe baze de date părăseşte compania, accesul său la acele aplicaţii şi conturi trebuie oprit imediat. Realizarea acestui lucru poate fi dificilă atunci când controlul administrativ şi responsabilitatea sunt distribuite în nodurile din reţea şi între diferiţi administratori şi grupuri.

Dar dacă un depozit central inteligent ar putea controla eficient datele cu privire la identităţi, conturi, autentificare şi autorizare, şi comunica rapid orice informaţie solicitată de orice nod sau aplicaţie? Apoi, una sau mai multe modificări într-un loc pot modifica toate drepturile de acces şi privilegiile unui angajat care părăseşte compania. Presupunerea care se face este că inteligenţa imputată acestui depozit central şi aplicaţiilor sale ţine cont de toate aceste consideraţii şi conexiuni. Desigur, toate rutinele dependente, la nivel de sistem, baze de date şi aplicaţii va trebui să se ajusteze astfel încât să se bazeze pe acel depozit central ca fiind „singura sursă de adevăr" în ceea ce priveşte astfel de informaţii.

Aceste consideraţii constituie baza pentru o soluţie integrată de gestiune a identităţilor.

1.1 Beneficii dorite ale gestiunii identităţilor

Scopul gestiunii identităţilor este acela de a crea:

- O securitate mai mare, deoarece un singur punct de control este în mod inerent mai uşor de securizat şi poate răspunde mai repede decât mai multe astfel de puncte;
- O eficienţă mai bună, deoarece un singur punct de control elimină în mod automat duplicarea şi întârzierile inerente într-un sistem în care responsabilităţile administrative dispersate necesită multiple acţiuni pentru aceleaşi conturi.

Costul şi complexitatea rămân măsurile relevante ale viabilităţii oricărei soluţii care, în mod ideal, ar trebui să furnizeze următoarele reduceri în alocarea resurselor:

- Cost *one-time*: planificarea şi implementarea infrastructurii de gestiune a identităţilor trebuie să aibă un cost *one-time*, şi să nu constituie o parte necesară fiecărei desfăşurări a unei aplicaţii *enterprise*. Astfel, aplicaţii noi pot fi desfăşurate rapid, beneficiind automat de infrastructură, dar nefiind nevoie să o re-creeze. Astfel de exemple includ portaluri, aplicaţii *J2EE* şi aplicaţii *e-business*.
- Management centralizat cu instrumente distribuibile: Provizionarea şi gestiunea identităţilor trebuie să fie făcută central, chiar dacă este administrată în mai multe locuri utilizând instrumente care pot gestiona orice modificare de cont necesară.

- Distribuire fără perioade de indisponibilitate. Modificările la conturile de utilizatori, profiluri sau privilegii trebuie să fie disponibile instant tuturor aplicaţiilor de tip *enterprise* şi comunicate rapid bazelor de date distribuite.
- *Single sign-on* utilizator: Infrastructura centralizată de securitate trebuie să facă funcţionalitatea *single sign-on* posibilă între aplicaţiile de tip *enterprise*. Această funcţionalitate face să nu mai fie necesar ca utilizatorii să-şi amintească mai multe parole, iar ca administratorii de securitate să protejeze mai multe depozite de parole şi mecanisme de provizionare.
- Punct singular de integrare: Infrastructura centralizată de gestiune a identităţilor trebuie să furnizeze un punct singular de integrare între mediul *enterprise* şi alte sisteme de gestiune a identităţilor. Astfel, trebuie să elimine orice necesitate de soluţii de integrare *point-to-point* personalizate.

O soluţie de gestiune a identităţilor, care întruneşte toate aceste criterii la un nivel înalt, furnizează un sistem de tip *enterprise* cu un nivel înalt de disponibilitate, localizare a informaţiei şi administrare delegată a componentelor. Mai departe, fiecare aplicaţie suplimentară desfăşurată în acel sistem va fortifica infrastructura partajată pentru serviciile de gestiune a identităţilor.

De exemplu, infrastructura *Oracle Identity Management* utilizează componente integrate pentru a furniza beneficiile care vor fi descrise în secţiunea următoare.

1.2 Componente ale infrastructurii *Oracle Identity Management*

Infrastructura *Oracle Identity Management* include următoarele componente şi funcţionalităţi:

- *Oracle Internet Directory* este un serviciu scalabil, robust, compatibil cu LDAP (*Lightweight Directory Access Protocol*), implementat în baza de date *Oracle*.
- *Oracle Directory Integration and Provisioning* este parte a *Oracle Internet Directory*, care permite sincronizarea între *Oracle Internet Directory* şi alte directoare şi depozite ale utilizatorului. Acest serviciu furnizează servicii de provizionare automată pentru componentele şi aplicaţiile *Oracle*, prin interfeţe standard, pentru aplicaţii *third-party*.
- *Oracle Delegated Administration Service* este parte a lui *Oracle Internet Directory*, care furnizează administrare pe bază de *proxy* a informaţiei din director prin intermediul utilizatorilor şi administratorilor de aplicaţii.

- *Oracle Application Server Single Sign-On* furnizează acces *single sign-on* la *Oracle* şi aplicaţii *web third-party*.
- *Oracle Application Server Certificate Authority* generează şi publică certificate PKI (*Public Key Infrastructure*) X.509 versiunea 3 pentru a veni în sprijinul metodelor avansate de autentificare, transmitere sigură a mesajelor etc.

Pe lângă utilizarea SSL (*Secure Socket Layer*), *Oracle Application Server Containers for J2EE* şi *Oracle HTTP Server*, infrastructura *Oracle Identity Management* are un suport *built-in* pentru *Oracle AS Single Sign-On* şi *Oracle Internet Directory*. Când *OracleAS Certificate Authority* este folosit, el publică fiecare certificat valid într-o intrare din director pentru numele distinct care este utilizat. *Server*-ul *single sign-on* şi celelalte componente se pot baza pe aceste intrări în director din cauza faptului că autoritatea emitentă de certificate şterge certificatele revocate şi expirate din director în mod regulat. Utilizatorii autentificaţi prin *server*-ul *single sign-on* cărora le lipseşte un certificat pot obţine rapid unul direct de la *OracleAS Certificate Authority*, care le permite să se autentifice pe orice componentă *Oracle* sau aplicaţie care este configurată să autentifice utilizatori cu ajutorul *server*-ului *single sign-on*.

Într-o desfăşurare tipică de aplicaţii *enterprise*, este desfăşurată o singură instanţă a infrastructurii *Oracle Identity Management*, ce constă în mai multe instanţe specifice *server*-ului sau componentelor. O astfel de configuraţie furnizează beneficiile menţionate anterior, referitoare la gradul înalt de disponibilitate, localizarea informaţiei şi administrarea delegată a componentelor.

2. Verificări şi recomandări de securitate

Acest capitol oferă o perspectivă mai largă a multiplelor tipuri de *task*-uri care trebuie realizate pentru a construi o bună securitate. Înţelegerea diverselor categorii de *task*-uri îmbunătăţeşte probabilitatea de a fi prevenite vulnerabilităţile de securitate. Astfel de vulnerabilităţi, dacă sunt exploatate accidental sau în mod intenţionat, pot submina sau depăşi reguli stricte de securitate care au fost create în alte domenii.

În continuare, prezentăm cerinţele pentru obţinerea unei bune securităţi, ameninţările pe care le putem întâmpina, şi conceptele care s-au dovedit folositoare în crearea metodelor practice pentru dezvoltarea şi susţinerea acesteia.

Vom identifica şi descrie pe scurt diferitele categorii de *task*-uri folositoare în întâmpinarea acestor cerinţe şi ameninţări. O bună securitate necesită controlul accesului fizic,

personal de încredere, o instalare şi proceduri de configurare sigure, comunicaţii sigure, şi controlul operaţiilor asupra bazelor de date (selectarea, vizualizarea, actualizarea sau ştergerea înregistrărilor). Deoarece unele dintre aceste cerinţe implică aplicaţii sau proceduri stocate, dar şi acţiunea factorului uman, procedurile de securitate trebuie să prevadă modul în care aceste programe sunt dezvoltate şi abordate.

Următoarele preocupări practice trebuie, de asemenea, avute în vedere:

- minimizarea costului echipamentelor, personalului şi instruirii;
- minimizarea întârzierilor şi erorilor;
- maximizarea responsabilităţii rapide şi detaliate.

Scalabilitatea este un criteriu practic important şi independent care ar trebui dobândit pentru fiecare soluţie propusă.

2.1 Lista de verificare a controlului accesului fizic

Nu ar trebui să fie uşoară pătrunderea într-o incintă fără o cheie sau un *badge*, sau fără a se cere dovada identităţii sau a autorizării. Controlul accesului fizic este primul lucru care trebuie luat în considerare, prin protejarea datelor (şi a personalului) împotriva celor mai simple intruziuni şi interferenţe accidentale sau maliţioase.

Lipsa acestui control poate simplifica observarea, copierea sau furtul altor controale de securitate, inclusiv ale cheilor interne, ale codurilor de chei, ale numerelor de *badge* sau a *badge*-urilor ş.a.m.d. Securitatea acestor măsuri depinde şi de cât de alert sau conştient de problemele de securitate este personalul, dar controlul accesului fizic opreşte o mulţime de potenţiale probleme.

Fiecare organizaţie trebuie să evalueze propriile riscuri şi bugete. Elaborarea unor măsuri poate să nu fie necesară, în funcţie de mai mulţi factori: mărimea companiei, riscurile de pierderi, controalele accesului intern, cantitatea şi frecvenţa vizitatorilor externi ş.a.m.d. Pregătirea responsabilităţii şi a recuperării sunt consideraţii suplimentare, prin instalarea de alarme sau sisteme de supraveghere video. Vizibilitatea acestor pregătiri pot acţiona şi în scopul descurajării.

2.2 Lista de verificări a personalului

Prima problemă în privinţa personalului o constituie selecţia, intervievarea, observarea şi verificarea referinţelor. Realizate bine, aceste lucruri pot preveni angajarea unor oameni care

sunt (sau pot deveni) inadecvaţi pentru *task*-uri sau medii care depind de stabilirea şi menţinerea securităţii. Până la un anumit nivel, securitatea depinde de indivizi.

A doua problemă este cât de conştient şi alert este personalul în privinţa preocupărilor şi considerentelor de securitate. Această conştiinţă este doar o parte a *background*-ului, iar mediul şi instruirea care sunt oferite sunt influenţele majore, dacă există onestitate şi intenţia de a coopera.

2.3 Lista de verificări pentru instalarea şi configurarea sigură

Pentru bazele de date, stabilirea unei configuraţii sigure este o primă problemă foarte importantă, fiind realizată prin utilizarea celor mai bune practici din standardele industriale privind desfăşurările operaţionale ale bazelor de date. Următoarea listă de astfel de practici este foarte generală şi prezintă recomandările a căror implementare furnizează baza pentru o configuraţie sigură.

- Instalarea doar a componentelor necesare.

  Se recomandă instalările de tip *custom,* evitându-se opţiunile de instalare şi produsele care nu sunt necesare. Este bine să se aleagă instalarea doar a produselor şi opţiunilor suplimentare, pe lângă server-ul de baze de date, care sunt necesare. Sau, dacă este aleasă o instalare *typical*, se recomandă îmbunătăţirea securităţii după terminarea procesului de instalare, prin ştergerea opţiunilor şi produselor care nu sunt necesare.

- Blocarea şi expirarea conturilor de utilizator care nu sunt necesare.

  Oracle Database se instalează cu multe conturi implicite (prestabilite) de utilizatori ai bazei de date. După crearea cu succes a unei instanţe a *server*-ului bazei de date, *Database Configuration Assistant* blochează şi expiră, în mod automat, majoritatea conturilor implicite de utilizator.

  După ce baza de date este instalată, se recomandă blocarea lui *SYS* şi *SYSTEM* şi utilizarea *AS SYSDBA* pentru accesul administratorului. Parolele administrative trebuie specificate individual.

  Acest cont (*AS SYSDBA*) urmăreşte numele de utilizator al sistemului de operare, permiţând menţinerea responsabilităţii. Dacă este necesar accesul doar pentru pornirea şi oprirea bazei de date, atunci trebuie utilizat *AS SYSOPER*. Acest cont are mai puţine privilegii administrative decât *SYS*, dar suficient încât să poată fi efectuate operaţii de bază, cum ar fi pornirea, oprirea, montarea, *backup*-ul, arhivarea şi recuperarea.

Utilitarul *Database Configuration Assistant* nu este utilizat în timpul unei instalări manuale, prin urmare toţi utilizatorii impliciţi ai bazei de date rămân deblocaţi şi pot obţine accesul neautorizat la date sau pot întrerupe operaţiile asupra bazei de date. Prin urmare, după o instalare manuală, trebuie utilizat *SQL* pentru a bloca şi expira toate conturile implicite de utilizator, mai puţin *SYS*, *SYSTEM*, *SCOTT* şi *DBSNMP*. Aceste acţiuni pot fi realizate şi asupra lui *SCOTT*, cu excepţia cazului în care este utilizat activ. De asemenea, pot fi blocate *SYS* şi *SYSTEM*, aşa cum a fost menţionat mai devreme. Dacă un cont blocat este necesar mai târziu, atunci un utilizator al bazei de date îl poate debloca şi activa cu o nouă parolă.

- Modificarea parolelor utilizatorilor impliciţi.

Securitatea este compromisă cel mai uşor în situaţia în care un cont de utilizator implicit al *server*-ului de baze de date are o parolă implicită chiar şi după instalare. Următorii paşi rezolvă această situaţie:

- Trebuie modificate parolele implicite ale utilizatorilor administrativi imediat după instalarea serverului bazei de date. În orice mediu *Oracle* (de producţie sau de test) trebuie atribuite parole puternice, sigure pentru conturile *SYS* şi *SYSTEM* imediat după instalarea cu succes a serverului bazei de date.
- Trebuie modificate parolele implicite ale tuturor utilizatorilor imediat după instalare. Toate conturile implicite trebuie blocate şi expirate după instalare. Dacă vreun astfel de cont este activat ulterior, atunci parola sa trebuie modificată.
- Trebuie stabilite reguli pentru gestiunea parolelor, cum ar fi reguli referitoare la lungimea parolei, istoric, complexitate. Se recomandă solicitarea utilizatorilor de a-şi modifica parolele în mod regulat.

  Dacă este posibil, se recomandă utilizarea *Oracle Advanced Security*, o opţiune a *Oracle Database Enterprise Edition*, cu servicii de autentificare în reţea (de exemplu, Kerberos), token cards, smart cards sau certificate X.509. Aceste servicii permit autentificarea mai sigură a utilizatorilor şi o mai bună protecţie împotriva accesului neautorizat.

- Activarea protecţiei dicţionarului datelor.

  Se recomandă protecţia dicţionarului datelor pentru a împiedica utilizatorii care au privilegiul sistem *ANY* de la a-l folosi pe dicţionarul datelor. *Oracle Database* setează parametrul O7_DICTIONARY_ACCESSIBILITY la valoarea *FALSE*. Această setare previne utilizarea privilegiului sistem *ANY* pe dicţionarul datelor, cu excepţia

utilizatorilor autorizaţi care realizează conexiuni cu privilegii *DBA* (de exemplu *CONNECT / AS SYSDBA*).

- Se recomandă practicarea principiului „celui mai mic privilegiu”.
  Următoarele practici implementează acest principiu:
  - Acordarea doar a privilegiilor necesare: restricţionarea numărului de privilegii sistem şi obiect acordate utilizatorilor bazei de date, restricţionarea, cât mai mult posibil, a numărului de conexiuni cu privilegii *SYS*. De exemplu, în general nu este necesară acordarea privilegiului *CREATE ANY TABLE* oricărui utilizator fără privilegii *DBA*.
  - Retragerea privilegiilor şi rolurilor care nu sunt necesare grupului de utilizatori *PUBLIC*. Acest rol implicit, acordat fiecărui utilizator dintr-o bază de date *Oracle*, permite utilizarea nerestricţionată a privilegiilor sale, cum ar fi *EXECUTE* asupra unor diferite pachete *PL/SQL*. Dacă privilegiile şi rolurile care nu sunt necesare nu sunt retrase din *PUBLIC,* atunci un utilizator cu privilegii minime ar putea accesa şi executa pachete, altfel inaccesibile lui.
  - Restricţionarea permisiunilor asupra facilităţilor *run-time*. Nu trebuie atribuite toate permisiunile oricărei facilităţi *run-time* a serverului bazei de date, cum ar fi *Oracle Java Virtual Machine* (*OJVM*).

- Impunerea eficientă a controalelor de acces.
  Clienţii trebuie autentificaţi corespunzător. Deşi autentificarea la distanţă poate fi pornită, instalarea este mai sigură cu ea oprită (*remote_os_authentication = FALSE*, care este valoarea implicită). Dacă autentificarea la distanţă este pornită, baza de date acordă implicit încredere fiecărui client, deoarece presupune că fiecare a fost identificat de către sistemul de autentificare distant. Însă clienţii, în general (de exemplu, un calculator distant) nu prezintă certitudinea efectuării autentificării corespunzătoare în sistemul de operare, deci pornirea acestei caracteristici nu este o practică bună.

- Restricţionarea accesului sistemului de operare, prin anumite practici: limitarea numărului de utilizatori ai sistemului de operare, limitarea privilegiilor conturilor sistemului de operare (administrative, cu privilegii *root* sau *DBA*) pe calculatorul pe care se află *Oracle Database* la cele mai puţine şi mai puţin puternice privilegii necesare fiecărui utilizator.

- Restricţionarea accesului în reţea.

- Aplicarea *patch*-urilor de securitate.

2.4 Lista de verificări pentru asigurarea securităţii în reţea

SSL este protocolul *Internet* standard pentru comunicaţia sigură, ce furnizează mecanisme pentru integritatea şi criptarea datelor. Aceste mecanisme pot proteja mesajele trimise şi primite de către utilizator sau de către aplicaţii şi servere, având suport pentru autentificarea sigură, autorizarea şi transmiterea mesajelor cu ajutorul certificatelor şi, dacă este necesar, al criptării. Pentru utilizarea cu succes a *SSL*, trebuie acordată atenţie anumitor detalii, dintre care menţionăm: fişierele de configurare (pentru clienţi şi listeneri) folosesc portul corect pentru *SSL*, cel configurat la instalare; *tcps* trebuie să fie specificat ca protocol în parametrul *ADDRESS* din fişierul *tnsnames.ora* şi în *listener.ora*.; modul *SSL* să fie consistent la ambele capete ale comunicaţiei etc.

Deoarece autentificarea calculatoarelor client pune probleme pe *Internet*, în locul acesteia se efectuează autentificarea utilizatorului. Această abordare evită probleme ale sistemului client, cum ar fi adrese IP falsificate, sisteme de operare sau aplicaţii compromise de *hacker*-i şi identităţi falsificate sau furate ale sistemelor client. Pe lângă aceasta, următoarele măsuri îmbunătăţesc securitatea conexiunilor client: configurarea conexiunii pentru a utiliza *SSL*, autentificarea prin certificate a clienţilor şi serverelor.

Pentru că *listener*-ul acţionează ca gateway al bazei de date către reţea, este importantă limitarea consecinţelor interferenţei maliţioase, prin: restricţionarea privilegiilor listener-ului, astfel încât acesta să nu poată citi sau scrie fişiere în baza de date sau în spaţiul de adrese al server-ului Oracle; administrarea sigură (protejarea listener-ului cu parolă, prevenirea administrării online, utilizarea SSL la administrarea listener-ului, ştergerea configurării procedurilor externe din fişierul listener.ora, dacă nu se vor utiliza astfel de proceduri); monitorizarea activităţii listener-ului.

2.5 Practici de securitate recomandate pentru proiectarea aplicaţiilor

- Activarea şi dezactivarea promptă a rolurilor, constând în activarea unui rol numai când aplicaţia porneşte şi dezactivarea lui de îndată ce aceasta se termină. Pentru aceasta, fiecare aplicaţie trebuie să aibă roluri distincte, care să conţină toate privilegiile necesare pentru utilizarea cu succes a respectivei aplicaţii. Dacă este necesar, se vor stabili câteva roluri suplimentare care conţin numai câteva dintre aceste privilegii, pentru a furniza un

nivel de securitate mai restrictiv sau dimpotrivă, pentru anumiţi utilizatori sau utilizări ale aplicaţiei. Fiecare rol al bazei de date trebuie să fie protejat printr-o parolă.
Instrucţiunea *SET ROLE*, la lansarea aplicaţiei, permite activarea unui rol asociat acelei aplicaţii.

- Încapsularea privilegiilor în proceduri stocate, pentru a restricţiona instrumentele de interogare ad hoc de a exercita privilegii la nivel de aplicaţie. Se recomandă acordarea de privilegii utilizatorului asupra acestor proceduri, în locul acordării directe de privilegii, astfel încât privilegiile nu pot fi utilizate în afara procedurii corespunzătoare.

  Utilizatorii pot exercita apoi privilegii doar în contextul aplicaţiilor bine formate. De exemplu, trebuie luată în considerare autorizarea utilizatorilor de a actualiza un tabel numai prin executarea unei proceduri stocate, şi nu prin actualizarea directă a tabelului. În acest mod, se evită situaţia în care utilizatorul deţine privilegiul *SELECT* şi îl utilizează în afara aplicaţiei.

- Utilizarea de parole necunoscute utilizatorului pentru roluri. Pentru privilegiile care trebuie exercitate numai în cadrul aplicaţiei, se recomandă activarea rolului printr-o parolă cunoscută numai de creatorul rolului. Aplicaţia lansează o instrucţiune *SET ROLE*, iar parola trebuie să fie încapsulată în aplicaţie sau recuperabilă dintr-un tabel al bazei de date cu ajutorul unei proceduri stocate. Deoarece utilizatorii răuvoitori pot decompila codul clientului şi recupera parolele încapsulate, această metodă nu este foarte sigură. Recuperarea parolei dintr-un tabel al bazei de date este mai sigură, deoarece necesită ca utilizatorul să descopere ce procedură stocată să folosească, să dobândească permisiunea de *EXECUTE* pe acea procedură, să o execute şi să regăsească parola. Doar în aceste condiţii utilizatorul ar putea să folosească rolul în afara aplicaţiei.

- Utilizarea autentificării prin *proxy* şi a unui rol sigur pentru aplicaţie. Pentru activarea unui rol în sistemele cu o arhitectură pe trei niveluri, utilizatorul trebuie să acceseze baza de date printr-o aplicaţie *middle-tier* care necesită autentificare prin *proxy* şi un rol de aplicaţie sigur. Autentificarea prin *proxy* face distincţie între un *middle tier* care creează o sesiune în numele unui utilizator şi utilizatorul care se conectează direct. Atât informaţia referitoare la utilizatorul *proxy* (acel *middle tier*) cât şi cea referitoare la utilizatorul real sunt captate în sesiunea utilizatorului.

  Un rol sigur de aplicaţie este implementat printr-un pachet, care realizează validarea dorită înainte de a permite unui utilizator să îşi asume privilegiile din rol. Atunci când o aplicaţie utilizează autentificarea prin *proxy*, pachetul corespunzător acelui rol sigur

validează faptul că sesiunea utilizatorului a fost creată prin *proxy*. Dacă utilizatorul se conectează la baza de date printr-o aplcaţie, rolul poate fi stabilit, ceea ce nu se întâmplă dacă utilizatorul se conectează direct.

De exemplu, considerăm situaţia în care dorim restricţionarea utilizării unui rol de administrare asupra lui *HR* doar pentru utilizatorii care accesează baza de date (prin *proxy*) prin *middle-tier*-ul *HRSERVER*. Se poate crea următorul rol:

CREATE ROLE admin_role IDENTIFIED USING hr.admin;

Pachetul *hr.admin* efectează validarea dorită, permiţând rolului să fie setat numai dacă determină că utilizatorul este conectat prin *proxy*. Pachetul poate utiliza subprogramul *SYS_CONTEXT*, care returnează identificatorul şi numele utilizatorului *proxy* (*HRSERVER*). Dacă utilizatorul încearcă să se conecteze direct la baza de date, pachetul *hr.admin* nu va permite setarea rolului.

- Utilizarea rolurilor la nivel de aplicaţie pentru verificarea adreselor IP.

  Pachetul corespunzător rolului aplicaţiei poate utiliza informaţii suplimentare din sesiunea utilizatorului pentru a restricţiona accesul, cum ar fi adresa IP de origine a utilizatorului. Adresele IP nu trebuie folosite niciodată ca prim criteriu pentru a lua decizii asupra controlului accesului, deoarece aceste adrese pot fi falsificate. Totuşi, adresa IP poate fi utilizată pentru a creşte restricţiile în privinţa accesului, după utilizarea altor criterii ca prime decizii asupra controlului accesului.

  De exemplu, dorim să ne asigurăm că sesiunea unui utilizator a fost creată prin *proxy* pentru un utilizator *middle-tier* care se conectează de la o anumită adresă IP. Nivelul din mijloc va trebui, desigur, să se autentifice către baza de date înaintea creării unei sesiuni *lightweight*, iar baza de date asigură că nivelul din mijloc are privilegiul de a crea o sesiune în numele utilizatorului.

  Pachetul poate valida adresa IP a conexiunii. Înainte de a lansa *SET ROLE* cu succes, ne putem asigura că acea conexiune *HRSERVER* (sau sesiunea *lightweight*) porneşte de la adresa adecvată utilizând *SYS_CONTEXT*. Acest lucru furnizează un nivel suplimentar de securitate.

- Utilizarea contextului aplicaţiei şi a controlului de acces *fine-grained*. În acest scenariu, sunt combinate controlul accesului *fine-grained*, impus de către server şi, prin contextul aplicaţiei, a atributelor sesiunii.

3. Limbajul de control al datelor

Pentru că informaţia reprezintă valoare, securitatea datelor constituie o problematică importantă a sistemelor de gestiune a bazelor de date.

La nivelul datelor, pe lângă crearea de vizualizări, un mecanism de asigurare a securităţii datelor îl constituie acordarea şi revocarea de permisiuni (privilegii). Vizualizările pot fi utilizate în conjuncţie cu sistemul de acordare de privilegii, făcând posibil accesul selectiv la submulţimi de date.

3.1 Acordarea privilegiilor şi a *role*-urilor

Accesul tuturor utilizatorilor la anumite obiecte este reglementat prin acordarea de privilegii şi *role*-uri. Un privilegiu este permisiunea de a accesa un obiect într-o manieră predefinită. Un *role* este un grup de privilegii care poate fi acordat utilizatorilor bazei de date sau altor *role*-uri.

Privilegii sistem sau *role*-uri nu pot atribui decât utilizatorii cărora acestea le-au fost acordate cu opţiunea *WITH ADMIN OPTION* a comenzii *GRANT* sau cei care deţin privilegiul sistem *GRANT ANY ROLE*.

Comanda *GRANT* care permite atribuirea de privilegii sistem sau *role*-uri unui utilizator are următoarea sintaxă:

GRANT {*privilegiu_sistem* | *role*} [, {*privilegiu_sistem* | *role*}...]
TO {*nume_utilizator* | *role* | ***PUBLIC***}
[, {*nume_utilizator* | *role* | ***PUBLIC***}...]
[***WITH ADMIN OPTION***];

Acordarea privilegiilor sistem tuturor utilizatorilor bazei de date este posibilă cu ajutorul opţiunii *PUBLIC*. Prin specificarea clauzei *WITH ADMIN OPTION*, utilizatorilor le este permis să acorde altor utilizatori sau *role*-uri privilegiile sistem şi *role*-urile respective.

Acordarea de privilegii obiect nu poate fi realizată împreună cu acordarea de privilegii sistem sau *role*-uri, în cadrul aceleiaşi comenzi *GRANT*. Privilegiile obiect pot fi acordate de proprietarul obiectului sau de un utilizator căruia i s-a acordat privilegiul obiect respectiv, cu opţiunea *WITH GRANT OPTION*. Comanda folosită pentru acordarea unui privilegiu obiect este:

GRANT {*privilegiu_obiect* [(*listă_coloane*)]
[, *privilegiu_obiect* [(*listă_coloane*)...]
| ***ALL*** [***PRIVILEGES***] }
ON [*nume_schemă*.]*obiect*
TO {*nume_utilizator* | *role* | ***PUBLIC***}

[, {*nume_utilizator* | *role* | ***PUBLIC***}...]
[***WITH GRANT OPTION***];

Clauza *ON* [*nume_schemă*.]*obiect* precizează obiectul relativ la care este acordat privilegiul.

Coloanele unui tabel sau ale unei vizualizări pentru care se acordă privilegiul se enumeră în opţiunea *listă_coloane*. Această opţiune poate fi folosită atunci când sunt acordate privilegii obiect de tip *INSERT*, *REFERENCES* sau *UPDATE*. Acordarea tuturor privilegiilor obiect pentru care utilizatorul ce iniţiază comanda *GRANT* are opţiunea *WITH GRANT OPTION* se poate realiza cu ajutorul opţiunii *ALL*.

Clauza *PUBLIC* permite acordarea privilegiilor obiect tuturor utilizatorilor bazei de date. Utilizatorii pot acorda privilegii obiect altor utilizatori dacă se precizează clauza *WITH GRANT OPTION*.

Privilegiile referitoare la comenzile *LMD* pot fi acordate pentru operaţiile *DELETE*, *INSERT*, *SELECT* şi *UPDATE* asupra unui tabel sau unei vizualizări, doar utilizatorilor sau *role*-urilor care trebuie să interogheze sau să prelucreze datele respective.

Permisiunile *INSERT* şi *UPDATE* se pot restricţiona pentru anumite coloane ale tabelului. Un privilegiu *INSERT* restricţionat pentru anumite coloane permite inserarea de valori doar pentru coloanele autorizate. Coloanele restricţionate primesc valori implicite sau *null*. Un privilegiu *UPDATE* restricţionat permite doar modificarea coloanelor pentru care utilizatorul are acest drept.

Un utilizator care încearcă să execute o comandă *LDD* trebuie să aibă anumite privilegii sistem sau obiect. Acordarea privilegiilor *ALTER, INDEX* şi *REFERENCES* permite executarea de operaţii *LDD* asupra unui tabel. Privilegiul *REFERENCES* poate fi acordat unei anumite coloane a unui tabel. Astfel, tabelul respectiv este folosit ca tabel „părinte“ pentru orice cheie externă care trebuie creată.

Exemplu. Să se acorde utilizatorilor *student* şi *profesor* privilegiile obiect *SELECT* şi *INSERT* asupra coloanelor tabelului *organizator*, cu posibilitatea ca aceştia să acorde privilegiile şi altor utilizatori sau *role*-uri. Să se acorde utilizatorului *invitat* privilegiul obiect *INSERT* doar pentru coloanele *cod_organizator* şi *denumire.*

```
GRANT SELECT, INSERT
ON    organizator TO student, profesor
WITH GRANT OPTION;
GRANT INSERT (cod_organizator, denumire)
ON    organizator TO invitat;
```

Exemplu. Să se acorde utilizatorului *student* permisiunea de a defini constrângeri care referă tabelul *publicitate* din schema *profesor*.

```
GRANT REFERENCES
ON profesor.publicitate TO student;
```

3.2 Revocarea privilegiilor şi a *role*-urilor

Revocarea de privilegii sau *role*-uri se poate realiza prin intermediul comenzii *REVOKE*. Un utilizator care are opţiunea de administrare sau de acordare a unui privilegiu sau *role*, le poate revoca pe acestea oricărui *role* sau utilizator al bazei de date. Un utilizator care deţine privilegiul *GRANT ANY ROLE* poate revoca orice *role*.

Comanda de revocare a unui privilegiu sistem sau *role* are următoarea sintaxă generală:

REVOKE {*privilegiu_sistem* | *role*} [, {*privilegiu_sistem* | *role*}...]
FROM {*nume_utilizator* | *role* | ***PUBLIC***};

Revocarea privilegiilor sistem sau a *role*-urilor tuturor utilizatorilor bazei de date este posibilă prin specificarea opţiunii *PUBLIC*.

Comanda de revocare a unui privilegiu obiect are sintaxa generală:

REVOKE {*privilegiu_obiect* [, *privilegiu_obiect* ...]
| ***ALL*** [***PRIVILEGES***] }
ON [*nume_schemă*.]*obiect*
FROM {*nume_utilizator* | *role* | ***PUBLIC***}
[, {*nume_utilizator* | *role* | ***PUBLIC***}...]
[***CASCADE CONSTRAINTS***];

Revocarea tuturor privilegiilor obiect acordate utilizatorului este posibilă cu ajutorul opţiunii *ALL*. Obiectul la care se referă privilegiul ce trebuie revocat se identifică prin clauza *ON*. Clauza *FROM* specifică utilizatorii sau *role*-urile pentru care este revocat privilegiul obiect. Opţiunea *PUBLIC* determină revocarea unor privilegii obiect tuturor utilizatorilor bazei de date.

Suprimarea constrângerilor de integritate referenţială definite folosindu-se privilegiile *REFERENCES* sau *ALL* se realizează prin intermediul clauzei *CASCADE CONSTRAINTS*.

Revocarea privilegiilor poate afecta definiţiile obiectelor care depind de privilegii *LMD* sistem sau obiect. De exemplu, dacă privilegiul sistem *SELECT ANY TABLE* a fost acordat unui utilizator care apoi a creat vizualizări ce folosesc un tabel din altă schemă, atunci revocarea acestui privilegiu determină invalidarea vizualizărilor respective.

Privilegiile revocate nu afectează definiţiile obiectelor pentru care sunt necesare privilegiile obiect *ALTER* şi *INDEX*. De exemplu, dacă privilegiul *INDEX* este revocat unui utilizator care a creat un index asupra unui tabel al altui utilizator, indexul respectiv va continua să existe şi după revocare.

Prin revocarea unui privilegiu obiect poate apărea efectul de revocare în cascadă a acestuia. De exemplu, dacă utilizatorului *profesor* i se acordă privilegiul obiect *SELECT* asupra unui tabel, cu opţiunea *WITH GRANT OPTION*, iar acesta îl acordă utilizatorului *student*, atunci

revocarea privilegiului utilizatorului *profesor* va determina automat revocarea acestui privilegiu şi pentru *student*.

Exemplu. Să se revoce utilizatorului *student* permisiunea de a defini constrângeri care referă tabelul *publicitate* din schema *profesor*, eliminând automat constrângerile de acest tip care au fost deja definite de către *student*.

```
REVOKE REFERENCES
ON profesor.publicitate
FROM student
CASCADE CONSTRAINTS;
```

Informaţii despre privilegii şi *role*-uri se pot obţine din vizualizările existente în dicţionarului datelor. Dintre acestea, se pot enumera: *DBA_TAB_PRIVS* (privilegiile acordate asupra obiectelor bazei), *DBA_COL_PRIVS* (privilegiile acordate asupra coloanelor), *DBA_ROLES* (*role*-urile definite în baza de date) etc.