Introducció

Vicente Díaz Sáez

PID_00191643

Universitat Oberta de Catalunya

Índex

1. Seguretat en bases de dades i aplicacions web

Les bases de dades són a tot arreu. Tan simple com això; són el magatzem de la informació d'ús diari per a pràcticament tot. Emmagatzemen dades bancàries, mèdiques, el cens de la població, antecedents penals, informació sobre vistes d'ovnis i les dades de la declaració de la renda. No hi ha cap organització o empresa que no faci ús d'una manera o d'una altra d'una base de dades, de manera que és clar que són una peça clau i suculent per a qualsevol atacant, davant el qual qualsevol mesura de protecció és poca.

Això no seria un problema gaire gros si les nostres bases de dades estiguessin guardades en una caixa de seguretat en un lloc desconegut, però llavors tampoc no farien gaire servei. Les bases de dades formen part d'un ecosistema des del qual es permet a certs usuaris accedir a la informació que contenen. Dins d'aquest ecosistema gairebé sempre hi trobem una aplicació web, o girant-ho del revés, gairebé tots els serveis web fan servir d'una manera o d'una altra una base de dades. Per això tots dos mons estan interrelacionats íntimament i per això quan pensem en seguretat no podem considerar aquests elements com a aïllats.

Els serveis web i les aplicacions que els implementen viuen actualment el moment de màxima expansió. El creixement dels serveis 2.0 i la proliferació de la informàtica en núvol (*cloud computing*) han comportat l'explosió de la demanda per a tota mena d'aplicacions que implementen des de la posició GPS d'un usuari d'un telèfon intel·ligent fins a la generació de nombres aleatoris. Sigui com vulgui, i més enllà de les dades que hi hagi al darrere, l'aplicació en si mateixa representa la porta d'entrada per a un potencial atacant a la infraestructura en què es troba. Un servei mal protegit pot acabar amb un atac que aconsegueixi una denegació de servei, que es canviï la pàgina web de la companyia o que s'obtingui un control total de la xarxa interna, és a dir, una amenaça per a tots els clients d'aquesta xarxa.

Hi ha nombrosos exemples d'això. No és que no hi hagi força documentació anterior al 2011, però potser aquest any ha estat especialment significatiu pel que fa a la popularització d'atacs dirigits contra grans companyies i entitats més enllà de l'afany de lucre o d'obtenció de "reputació" habitual entre els pirates (*hackers*) "bons". Grups heterogenis com Anonymous o Lulzsec van aconseguir saltar els controls de seguretat de companyies i entitats de talla mundial, i després van publicar les dades que van obtenir. I una cosa molt interessant: en la majoria dels casos els atacs van ser extremament simples!

La pèrdua o l'accés no autoritzat i, especialment, la publicació de les dades privades d'una entitat poden comportar un mal irreparable per a la reputació d'una organització (recordem el cas d'HBGARY, empresa dedicada a la segure-

tat als Estats Units). Davant una pèrdua es poden adoptar mesures i precaucions, com ara còpies de seguretat, però una intrusió és una cosa més delicada: l'accés a la informació clau d'una companyia pot comportar una pèrdua irreparable.

> Imaginem-nos el cas d'una empresa farmacèutica que dedica molts milions a recerca i desenvolupament de fàrmacs nous. Aquests desenvolupaments duren anys i en molts casos representen la supervivència de la companyia; segur que tots som capaços de trobar un exemple de farmacèutica que es basa en un únic medicament. Un cas d'espionatge industrial en aquest entorn seria fatal.

No es tracta solament de qüestions d'espionatge; hi ha qüestions legals que s'han de tenir ben presents. La Llei orgànica de protecció de dades de caràcter personal (LOPD) a Espanya, en vigor des del 13 de desembre del 1999, fa referència precisament a les dades que emmagatzemen les empreses espanyoles sobre els ciutadans i que són de caràcter personal. Aquesta llei estableix una sèrie de mesures que han de complir obligatòriament les organitzacions que emmagatzemen aquestes dades i segons la informació que emmagatzemin. Estableix tres nivells d'importància de dades i mesures relacionades amb cadascuna d'aquestes dades que s'han de complir obligatòriament, amb les sancions corresponents.

A l'Estat de Califòrnia va entrar en vigor el juliol del 2003 la Llei orgànica SB 1386, que obliga qualsevol organització que tingui dades personals de residents d'aquest estat a notificar a aquests residents la sospita de qualsevol forat de seguretat digital que hagi pogut comprometre les dades, si no estan encriptades. Actualment, hi ha diverses regulacions i lleis que miren de regular aquest problema en diferents àrees de negocis, com Sarbanes Oxley (SOX), Payment Card Industry (PCI) Data Security Standard, Healthcare Services (HIPAA), Financial Services (GLBA) o Data Accountability and Trust Act (DATA).

A cada país hi trobarem un exemple semblant o lleis que es preparen actualment.

Quin impacte pot tenir, però, un atac amb èxit contra una entitat?

És difícil estimar el valor de les dades robades en si; hi ha diverses aproximacions per a donar un valor monetari a aquestes pèrdues. La taula següent n'és un exemple:

Tipus de dada	Valor
Adreça	$0.50
Telèfon	$0.25
Telèfon no publicat	$17.50
Mòbil	$10

Estimació del valor de les dades robades

Tipus de dada	Valor
Data de naixement	$2
Educació	$12
Història creditícia	$9
Detalls de bancarrota	$26.50
Informació judicial	$2.95
Historial laboral complet	$18
Arxiu militar	$35
Targetes de crèdit	$1-6
Identitat completa	$16-18
Disc dur amb transferències del Banc Central de Rússia	$1800
Historial de trucades mitjançant mòbils	$110-200
30.000 correus electrònics	$5

Estimació del valor de les dades robades

Revisarem els aspectes principals relacionats amb la seguretat en les aplicacions web i en les bases de dades que utilitzen, tant a escala conceptual com a escala més tècnica, tenint en compte les arquitectures més populars avui dia.

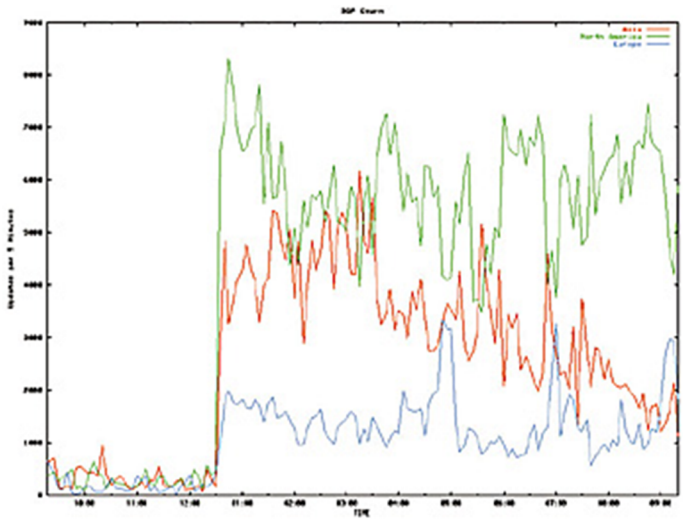
Detallarem els problemes a què hem de fer front per al disseny i la defensa d'una aplicació web, incloent-hi el punt de vista d'un atacant, les tècniques que farà servir contra nosaltres i les eines de què disposa. Explicarem com funcionen, com ens podem protegir, com podem minimitzar riscos i com podem monitorar la infraestructura per detectar qualsevol problema.

2. Evolució dels atacs

El 25 de gener del 2003, un cuc conegut com a *Slammer* va infectar més de setanta-cinc mil màquines en un lapse d'uns deu minuts, propagant-se a una velocitat mai vista fins llavors, i va causar denegació de servei en alguns dominis i lentitud en el trànsit en general.

Aquest cuc explotava una vulnerabilitat de desbordament de memòria intermèdia (*buffer*) en Microsoft SQL Server pública i amb un pedaç disponible des de sis mesos abans de l'atac. Molts administradors no havien apedaçat els seus sistemes de manera adequada, però la difusió tan massiva va ser deguda sobretot al fet que molts usuaris no eren conscients de tenir instal·lat MS SQL Server Desktop Engine (MSDE), que disposa d'un motor d'MS SQL Server.

Figura 1. Augment del trànsit BGP, indicatiu d'embussos a Internet



Courtesy Akamai

Aquest exemple és il·lustratiu en uns quants aspectes:

- D'entrada, és un bon exemple de l'amplíssima difusió de les bases de dades, en aquest cas fins i tot sense el coneixement dels usuaris d'aquestes bases. El motor de la base de dades estava enclòs en un altre producte de programari.

- Una base de dades, a part de qualsevol diferència conceptual que es pugui fer, no és sinó un producte de programari més, susceptible a atacs especialment dirigits per a obtenir les dades que conté, o simplement atacs genèrics que aprofitin una vulnerabilitat no corregida com en qualsevol altre programari.

- La gran presència de bases de dades accessibles per a un atacant potencial. Com que estan connectades a Internet, a vegades donant serveis i a vega-

des de manera inconscient per a l'usuari que les conté, són un objectiu potencial. Encara que la distribució d'aquest cuc va ser indiscriminada, hi ha moltes tècniques per a buscar bases de dades vulnerables a alguna fallada coneguda utilitzant eines tan a l'abast de tothom com ara Google.

- La vulnerabilitat, no perquè fos coneguda ni perquè disposés d'un pedaç des de sis mesos abans de l'atac, va deixar de ser fatal. Això posa en relleu no solament el problema de no disposar d'una política adequada d'actualitzacions, sinó del factor humà. Aquest factor, encara que difícil de remeiar, és determinant en molts casos per a explicar el perquè de molts problemes de seguretat, de manera que la formació i la conscienciació són un tema clau al mateix nivell que qualsevol altre aspecte tècnic.

En tot cas, es tracta únicament d'un exemple d'una vulnerabilitat explotada amb èxit per un atacant. Hi ha desenes de vulnerabilitats, amb diferents dificultats d'explotació i amb diferent grau de divulgació. L'evolució que han seguit les vulnerabilitats i els atacs en bases de dades és semblant a l'evolució que han seguit en qualsevol altre programari a mesura que augmentava de popularitat i de complexitat, amb l'atractiu que si és explotat té una recompensa final molt suculenta. També han crescut, però, una sèrie de vulnerabilitats concretes per a bases de dades com ara els atacs d'injecció SQL, que discutirem més endavant.

Cal contextualitzar els problemes dins de l'entorn en què s'han anat utilitzant les bases de dades, ja que els dissenyadors han anat donant respostes a les necessitats del mercat amb noves funcionalitats. En aquest context, normalment han estat els atacants els primers a pensar nous vectors d'atac i els dissenyadors han anat donant resposta als problemes a mesura que es detectaven.

No obstant això, actualment s'intenta passar d'un escenari reactiu a un escenari proactiu, en què es detectin els possibles problemes abans de treure un producte al mercat. Això és per un canvi en la manera de pensar que tenen les grans companyies, però sobretot en la dels seus clients. La seguretat ha passat de ser un extra a ser una absoluta necessitat, i són els clients mateixos els que la demanen, de manera que els creadors de motors de bases de dades i de programari cada vegada dediquen més recursos a assegurar els seus productes.

Actualment qualsevol producte passa diverses etapes de qualitat relacionades amb seguretat. En general, es passa una auditoria de codi en què es busquen possibles debilitats en el codi font de l'aplicació. També es duen a terme atacs d'intrusió (*penetration test* o *pentest*) en què un auditor té el rol d'un atacant i intenta amb tots els mitjans de què disposa aconseguir el control del programari auditat. D'aquesta manera, es busquen debilitats en què no han pensat els dissenyadors del programari, simulant el paper reactiu que hi han tingut les companyies però en aquest cas abans de treure el producte al mercat.

A part d'intentar que el programari tingui tan poques vulnerabilitats com es pugui, és vital disposar d'un servei de resposta ràpida a incidents que permeti publicar pedaços en resposta a vulnerabilitats tan de pressa com sigui possible. Així i tot, des de l'aparició d'una vulnerabilitat fins a la publicació i la instal·lació del pedaç en el client, hi ha una finestra d'exposició en què el programari és vulnerable al problema detectat. I això suposant que la vulnerabilitat detectada s'informi al venedor i aquest venedor publiqui un pedaç que la solucioni, ja que algunes vulnerabilitats no s'informen mai al venedor i les utilitzen atacants: si algú descobreix una vulnerabilitat amb què pot accedir a un tipus de base de dades per a les seves pròpies finalitats, per què l'ha de fer pública? D'aquestes vulnerabilitats se'n diu *0-day*.

Fins i tot amb l'actual conscienciació i inversió en seguretat, hi continua havent una gran quantitat de problemes.

> El 2006 Oracle va publicar quatre actualitzacions crítiques de seguretat relacionades amb servidors de bases de dades que resolien més de vint vulnerabilitats remotes. El 2007, encara hi ha més de cinquanta vulnerabilitats no apedaçades al servidor de base de dades d'Oracle (*Oracle database server*).

Evidentment això és solament un exemple, i no vol dir ni de bon tros que sigui l'únic motor de bases de dades afectat per aquesta mena de problemes. Això vol dir que no es pot aconseguir un sistema de bases de dades segur al cent per cent, però hi ha una gran quantitat de mesures que es poden implementar per minimitzar la possibilitat i la gravetat d'un atac eventual.

Quins tipus d'atacs són els més usats, però? A continuació fem una llista d'alguns dels més habituals segons OWASP:

- Atacs de força bruta contra contrasenyes.
- Robatori de credencials a la Xarxa.
- Configuracions insegures per defecte.
- Explotació de vulnerabilitats (públiques o no).
- Injecció SQL.
- Ús de cavalls de Troia i d'eines d'instrucció (*rootkits*).
- Robatori de dispositius d'emmagatzematge físics.
- Personal intern.

En realitat, la majoria d'intrusions encara s'aconsegueixen gràcies a una administració deficient. Encara que hi ha una gran quantitat de mètodes, els que produeixen més bons resultats continuen essent els més trivials. Això és per falta de conscienciació o per desconeixement dels potencials atacs. D'altra banda, hi ha una gran quantitat de programari heretat en les empreses que no s'actualitza per por de les actualitzacions o per necessitats d'un altre programari heretat que s'utilitza actualment. Quantes vegades hem sentit que si funciona no ho toquis? Aquesta situació causa molts problemes, ja que en alguns

casos les companyies deixen de donar suport a certes versions de programari després de la publicació de versions més recents (com passa a Microsoft amb Windows NT4).

Per tant, encara que avui dia el nivell de conscienciació respecte a seguretat és molt més alt que fa uns anys, no és suficient. Els atacs cada vegada són més evolucionats i cada vegada hi ha més eines per a evitar-los, però el factor humà continua essent el principal.

Implantar les mesures de seguretat disponibles per a minimitzar el risc d'atac i fer un seguiment mitjançant polítiques de seguretat que assegurin la base de dades al llarg del temps és feina tant de l'administrador com d'un auditor de seguretat.

3. Perspectives

És difícil saber què passarà en un futur, però en aquest apartat farem un petit exercici sobre aquesta qüestió considerant les últimes tendències i les perspectives pel que fa a seguretat per als anys vinents.

3.1. Tipus d'atac

El que sembla clar és que el nombre d'atacs continuarà creixent. Durant els últims anys aquest aspecte s'ha enfilat i han aparegut nous vectors d'atac. És clar que la difusió d'Internet i també dels serveis proporcionats per aquest mitjà facilita la possibilitat de buscar víctimes i informació per a explotar vulnerabilitats. La figura de l'"atacant casual" (script-kiddie[1]) és cada vegada més freqüent.

(1)Usuari que, sense necessitat de tenir coneixements tècnics, utilitza un programari de tercers o una vulnerabilitat coneguda i fàcil d'explotar per a aconseguir una intrusió.

Es pot pensar en això com algú que llegeix com pot aconseguir una intrusió en certs tipus de sistemes de manera senzilla i utilitza aquest coneixement per a atacar un sistema vulnerable sense cap més propòsit que la curiositat. Encara que aquest tipus d'atacants hauria de ser el que preocupi menys perquè no té prou coneixements tècnics, pot ser molt perillós precisament per això: l'atacant pot ser conscient o no del risc que implica la seva acció precisament perquè no té prou coneixements tècnics. Una bona política d'actualitzacions, una configuració correcta i l'ús de mesures de seguretat addicionals haurien de servir perquè aquest tipus d'atacants no tinguessin èxit. El problema és que, si en tenen, la seva falta de coneixements pot provocar un mal més gran del que volien fer precisament perquè no saben ben bé què fan.

Hi ha un altre perfil d'atacant molt capaç tècnicament i que estudia durant un temps un possible objectiu. Actualment hi ha persones que es dediquen a crear programari especialitzat contra un determinat objectiu a partir d'un encàrrec. En general, aquest tipus d'encàrrecs es fan mitjançant l'ús de cavalls de Troia que afecten els clients de determinades entitats o que exploten vulnerabilitats dels sistemes operatius per aconseguir accedir a l'entorn intern de l'entitat i, des d'allà, aconseguir credencials per a accedir als sistemes crítics. No obstant això, pot passar que amb el temps aquest tipus d'atacs tinguin com a objectiu directe les bases de dades. En aquest cas, pot ser que una política d'actualitzacions sigui important per a evitar vulnerabilitats relacionades amb versions antigues, tot i que no necessàriament. Aquest escenari planteja l'ús d'una vulnerabilitat desconeguda per a l'administrador, de manera que pot estar indefens. No obstant això, hi ha una sèrie de mesures que poden pal·liar aquest problema, descobrir-lo mentre s'està produint o fins i tot evitar-lo, com expliquem en aquest curs.

Les aplicacions web s'utilitzen majoritàriament per a generar pàgines web, fòrums, sistemes senzills de gestió, blogs, etc. Creades moltes vegades per usuaris no professionals que en general fan servir solucions predissenyades (amb vulnerabilitats conegudes o no), solen descurar l'actualització. Fins i tot a vegades són els mateixos creadors del programari els que no proporcionen un servei sobre això. Inevitablement, van apareixent vulnerabilitats associades a versions antigues, i una simple consulta en un cercador d'Internet permet l'accés a milers de sistemes vulnerables. Aquest sistema l'utilitzen generalment cucs per a propagar-se de pressa i infectar massivament de manera gairebé instantània.

D'altra banda, els sistemes de bases de dades pròpiament dits, amb dedicació exclusiva i actualitzats degudament, tenen un problema ja que cada vegada són més grans. La gran quantitat de serveis que demanen els usuaris reflecteix una necessitat del mercat a la qual han de donar resposta els fabricants, però l'augment de complexitat també comporta l'aparició de noves possibilitats d'atac.

S'ha de buscar sempre l'equilibri entre funcionalitat, necessitat i risc. La política tradicional en aquest aspecte és evitar instal·lar qualsevol servei innecessari.

No tot són males notícies, però. Els fabricants cada vegada implementen més serveis dedicats a seguretat. Encara que n'hi ha alguns que ja els implementen en alguns dels seus productes actualment, és previsible la implantació massiva. El primer servei és el d'encriptació de dades de manera transparent. Encara que pot comportar una pèrdua de rendiment en certs entorns crítics, en d'altres és molt important tenir totes les dades encriptades de manera automàtica, ja que facilita l'administració i evita errors humans. D'altra banda, gairebé tots els sistemes ja funcionen amb comunicacions també encriptades que eviten la possibilitat que un atacant accedeixi a les dades transmeses entre el client i el servidor. Cal esperar que aquests sistemes s'utilitzaran de manera transparent i que s'evitarà qualsevol altre tipus de comunicació en text pla per a una seguretat més gran del sistema.

3.2. Models de programació

Pel que fa a models de programació que interactuen amb bases de dades, també han seguit una evolució. La utilització de cert tipus d'atacs com la injecció SQL ha comportat un canvi en el paradigma de programació utilitzat fins avui. Els nous entorns de treball (*frameworks*) de creació de programari que interactua amb bases de dades ja inclouen mesures per a evitar aquest tipus de problemes. En aquest marc, la situació és semblant a la de l'ús massiu de motors de bases de dades per programari de tercers, si bé ara el problema no és del motor sinó del programari que hi interactua. La prevenció en aquest escenari passa per l'ús de les mesures que hi ha en el moment de crear el programari,

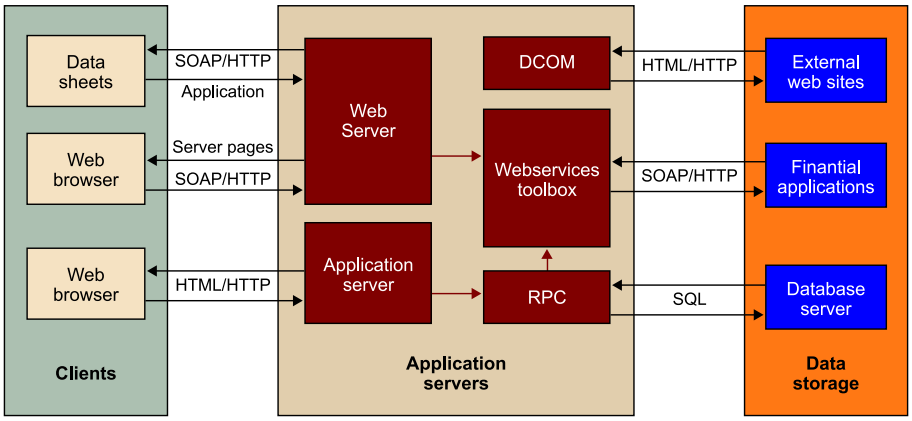
i per l'ús de codi ben estructurat que permeti l'adopció senzilla de modificacions que previnguin nous atacs. També s'ha popularitzat l'ús de filtres, IDS (*intrusion detection system*) i IPS (*intrusion prevention system*). Encara que no es pot preveure l'adopció d'aquestes mesures en la part del motor de la base de dades, és important sempre tenir en compte l'entorn en què es troba i utilitzar les mesures de protecció en tots els nivells. És un error pensar que es pot tenir una base de dades ben protegida sense pensar en tot l'entorn d'aquesta base de dades i a tenir-lo també protegit com cal.

4. Arquitectura d'aplicacions web

4.1. Arquitectura genèrica

L'arquitectura dels sistemes web estableix les directrius que s'han de seguir per a dissenyar els sistemes programari. La figura 2 mostra la típica estructura en tres capes per a aquesta arquitectura i el moviment de dades entre les interfícies d'usuari i les capes d'aplicació i els magatzems de dades.

Figura 2. Arquitectura web general basada en tres fases



Aquesta arquitectura en tres capes permet el desenvolupament modular de les interfícies d'usuari dependents d'un dispositiu específic (un programa determinat) per a comunicar-se mitjançant una estructura d'enviament de serveis web (*Web services*) multicanal. D'altra banda, permet incorporar processos o estratègies de negoci existents o completament nous.

L'enviament de serveis web està ajustat al model de pas de missatges descrit en els protocols *OASIS Web services reliable messaging* (WSRM) proposat per IBM, BEA, TIBCO i Microsoft (d'acord amb W3C).

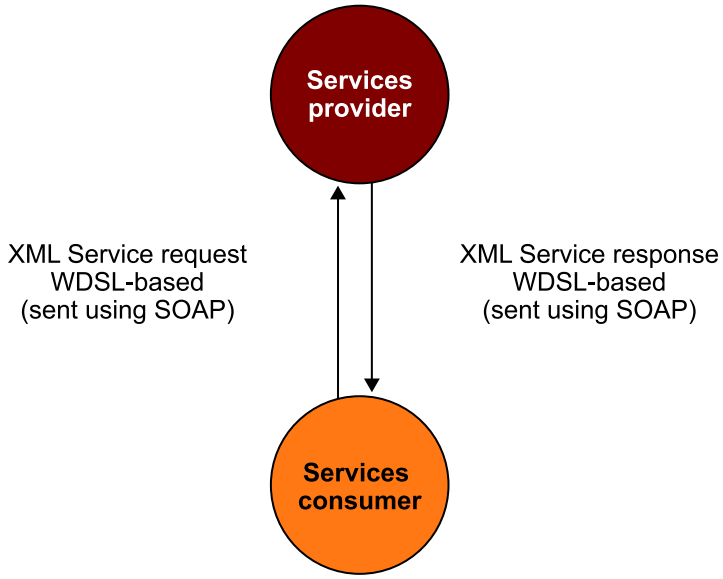
- Per la banda del client, el navegador web proporciona la interfície d'usuari i la lògica de presentació.

- Per la banda del servidor, el servidor web captura totes les peticions HTTP enviades des de l'usuari i les propaga al servidor d'aplicació, que implementa la lògica de l'aplicació.

- Per la banda del magatzem de dades, es processen de manera transicional i històrica les dades de les operacions diàries, i el gestor de bases de dades les emmagatzema en la base de dades del sistema.

El servidor d'aplicació envia una consulta (*query*) al servidor de bases de dades i obté el conjunt de resultats. El servidor prepara la resposta des d'una sèrie de processos i l'envia al servidor web perquè la faci arribar al client. Un usuari registrat pot iniciar la sessió obtenint un perfil en què s'especifiquen els privilegis que té, els drets d'accés i el nivell de complexitat. Una arquitectura basada en aquesta estructura trifàsica pot donar suport tant a una intranet com a un entorn accessible des d'Internet. L'usuari pot utilitzar, per exemple, un navegador web per a accedir al servidor de pàgines web usant HTML i HTTP. El nucli del sistema és al costat dels servidors d'aplicació i proporciona un conjunt de serveis web, que es poden comunicar els uns amb els altres. Aquesta comunicació pot implicar un simple traspàs de dades o pot implicar l'intercanvi d'informació necessari perquè dos serveis o més treballin de manera conjunta per a dur a terme una única activitat.

Figura 3. Arquitectura orientada a serveis bàsica



La figura 3 il·lustra l'arquitectura orientada a serveis bàsica (*services oriented architecture*, SOA). S'hi mostra un consumidor de serveis enviant una petició (missatge de petició o *request*) al proveïdor de serveis, que retorna un missatge de resposta al consumidor (missatge de resposta o *response*). Tant la petició com les connexions de resposta subsegüents estan definides de tal manera que són descodificables pels dos agents que participen en la comunicació. Per a completar aquest esquema tan senzill cal remarcar que un proveïdor de serveis també pot actuar com un consumidor.

En l'exemple que hi ha il·lustrat en la figura 3 tots els missatges estan formatats usant SOAP, que pot fer servir diferents protocols (HTTP, SMTP, etc.). SOAP proporciona l'entorn per a enviar els missatges dels serveis web per Internet o per intranet. Aquest entorn consta de dues parts:

- una **capçalera** opcional en què es pot indicar informació sobre l'autenticació o la codificació de les dades;

- i un **cos** que conté el missatge. Aquests missatges es poden definir usant l'especificació WSDL que fa servir XML per a definir els missatges. XML és un llenguatge d'etiquetatge que pot utilitzar tant el proveïdor de serveis com el consumidor per a intercanviar informació sense haver d'ajustar-se a un protocol en què l'ordre de les dades sigui molt estricte.

La presència dels límits estrictes entre cadascuna de les capes de la figura 2 es trasllada a les aplicacions programari implementades sobre aquesta arquitectura. Una distribució equilibrada dels components de cada capa entre les computadores de la xarxa garanteix un nivell de flexibilitat que no s'aconsegueix amb altres arquitectures alternatives. Com a resultat d'això, els recursos computacionals aconsegueixen un alt rendiment que queda reflectit en l'alta complexitat de la funcionalitat que es pot implementar.

Resum

Les característiques que defineixen aquesta arquitectura són les següents:

La informació final la sol·licita el client i es genera en la capa de servidor. Gràcies a això es prescindeix del fet que l'últim responsable del processament de les dades hagi de ser el client.

Tots els recursos d'informació i la lògica de l'aplicació estan concentrats en el servidor, en contra de la disposició distribuïda que es proposava en arquitectures ara obsoletes (per exemple, el model client-servidor).

Els protocols que s'utilitzen per a intercanviar dades entre la capa del client i la resta de les capes són estàndards, com TCP/IP sobre Internet.

També s'estandarditza el control centralitzat no solament del servidor sinó també dels equips dels clients, almenys en termes de programari. De manera que en cada estació de treball n'hi ha prou de tenir un programa de navegació estàndard.

Aquestes estacions de treball (*workstations*) poden executar programes locals i, de manera remota, els d'altres equips de la xarxa.

Totes aquestes característiques, a excepció de l'última, ajuden a mitigar el problema de seguretat informàtica. La concentració de tots els recursos d'informació i aplicació en capes alienes a la que allotja els clients simplifica el disseny i l'administració del sistema gestor de la seguretat de la informació. A més, el disseny dels sistemes de protecció dels objectes localitzats en un únic lloc és més senzill que si aquesta localització és distribuïda.

L'ús d'estàndards oberts com TCP/IP per a intercanviar dades entre els equips de la xarxa garanteix la unificació de tots els mètodes d'interacció entre estacions de treball i servidors. Gràcies a això es pot proposar una solució per a la protecció dels intercanvis d'informació vàlida per a tots els sistemes. El control centralitzat exercit des dels servidors sobre els equips client redueix la probabilitat d'errors comesos per descuits o falta de formació d'operaris o usuaris. Aquests errors són una de les principals amenaces del sistema i són potencialment molt perillosos.

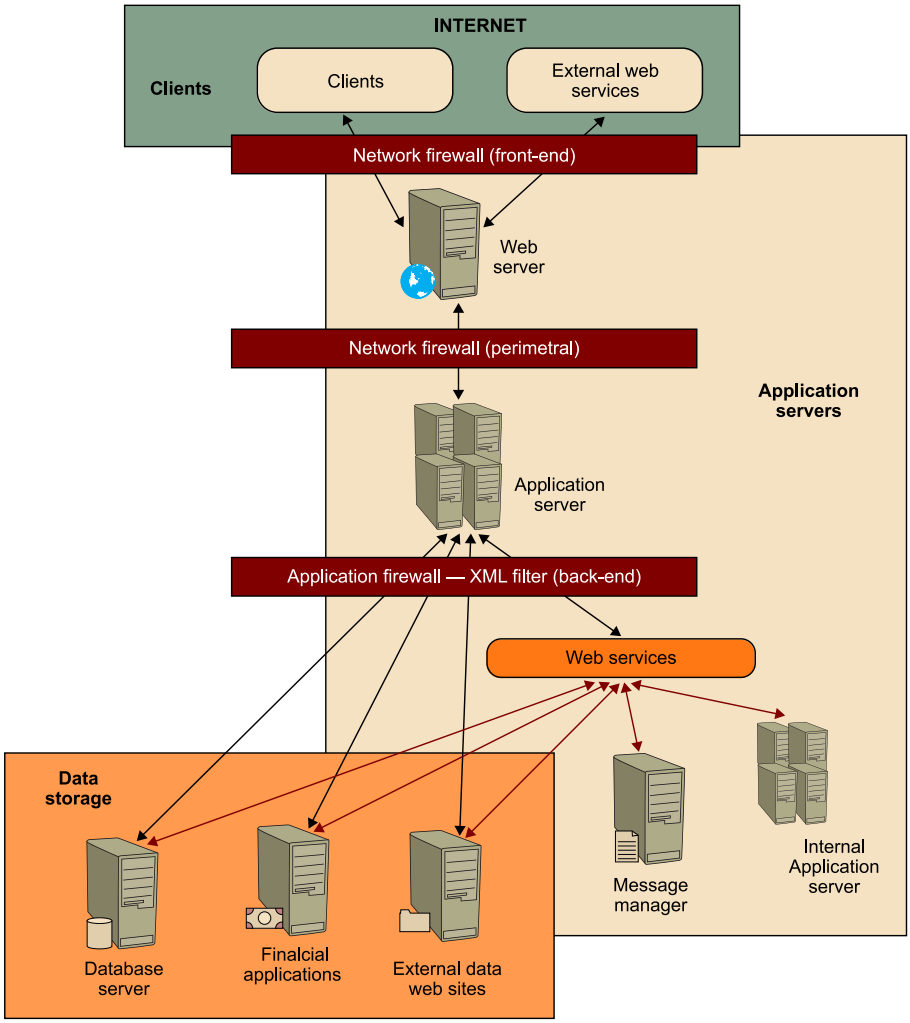
Com hem apuntat més amunt, en aquesta arquitectura, el processament de la informació distribuïda assumeix que els programes obtinguts des dels servidors seran executats en les estacions de treball. Aquest sistema de processa-

ment distribuït permet la concentració de tota la lògica de l'aplicació en la capa de servidor. No obstant això, la possibilitat d'executar programes des del servidor en un client genera noves amenaces que ens obliguen a ser més exigents amb els sistemes de protecció.

Com hem esmentat abans, la implementació més comuna de serveis web és SOAP. Això constitueix un problema potencial ja que SOAP usa el protocol HTTP, que implica que si no s'estableixen mesures de protecció precises es pot arribar, usant HTTP, als sistemes d'emmagatzematge de la figura 2, cosa que pot implicar un risc de seguretat significatiu.

La manera més habitual de solucionar aquest problema és integrar l'arquitectura web amb una arquitectura de comunicacions segura basada en una configuració eficaç de tallafocs (*firewalls*). La figura 4 resumeix les característiques principals d'aquesta arquitectura general protegida.

Figura 4. Estructura segura basada en xarxes perimetrals per a l'arquitectura web de tres capes



Com s'aprecia en la figura, aquesta arquitectura incorpora un tallafoc d'aplicació basat en XML sobre els serveis web interns per a filtrar els missatges que arriben des del servidor d'aplicació.

4.2. Tipus d'aplicacions

A partir de l'arquitectura general proposada en l'apartat anterior es pot establir una classificació d'aplicacions que permet distingir entre tres categories:

- Lloc web públic.
- Intranet.
- Extranet.

A continuació detallarem aquestes tres categories.

1) Lloc web públic, Internet o Web

En aquest escenari l'organització disposa dels recursos per a muntar un sistema de difusió de la informació sense restriccions. No hi ha restriccions aplicables als usuaris que motivin l'habilitació de sistemes arquitecturals per al control d'accessos. Tots els mecanismes d'autenticació depenen de la lògica de l'aplicació i, per tant, els gestiona el servidor d'aplicació.

Les aplicacions típiques són aquelles en què no s'estableix un sistema interactiu de negoci sinó purament informatiu.

2) Intranet

En el sentit més ampli del terme, una intranet és una xarxa de computadors que estableixen les interconnexions usant protocols desenvolupats per a Internet, però que no està integrada en el Web.

No obstant això, des d'una perspectiva purament pràctica, aquesta definició se sol complementar amb l'afirmació que una intranet està formada pels llocs web o aplicacions que es poden trobar a la xarxa i que es poden usar com a recursos i plataformes de comunicacions únicament per a l'organització que els gestiona, no la xarxa de comunicacions en si mateixa sinó els recursos, als quals solament es pot accedir de manera privada.

En un entorn empresarial en què constantment s'incrementa la complexitat de les infraestructures, les organitzacions han de fer front al desafiament que implica adaptar aquestes infraestructures a les noves tecnologies de captura i compartició d'informació amb l'objectiu d'aconseguir una alta efectivitat en la presa de decisions. Hi ha tres categories dins de la tecnologia de la informació que necessiten un entorn basat en intranet per a aconseguir aquest objectiu:

- La planificació de recursos d'empresa a escala transaccional (*enterprise resource planning* o ERP).

- L'administració de la relació amb els clients (*customer relationship management* o CRM).

- El sistema de control de la producció, a temps real (*manufacturing execution system* o MES).

- L'aplicació d'aquestes tres tecnologies sobre un escenari com una intranet persegueix que es compleixin els principis bàsics d'augmentar la producció, aconseguir el *zero downtime* i reduir els costos.

3) Extranet

> Una extranet manté una posició intermèdia entre intranet i Internet. En aquest escenari sorgeixen quan l'ús dels recursos privats d'una intranet s'amplien perquè siguin accessibles per a usuaris externs a l'organització, normalment els clients de l'organització.
>
> Les extranets es mantenen des de la xarxa interna de l'organització, per a garantir als usuaris l'accés als recursos, però utilitzen Internet per a enviar informació sempre que alguns dels usuaris siguin externs a la xarxa interna de l'organització. És a dir, en una extranet, es mantenen recursos privats per a usuaris que poden establir un accés extern a l'organització.

En realitat, la necessitat d'adaptar l'accés als recursos respecte als que es plantejaven en el cas de la intranet no es deu a una alteració en els objectius que es persegueixen. Aquests objectius continuaran essent els mateixos. No obstant això, el que justifica la utilització d'una extranet és la necessitat d'aplicar més flexibilitat a la relació amb els clients. De fet, en una extranet es poden gestionar dos models de negoci que en una intranet és impossible de gestionar: el comerç electrònic d'empresa a empresa (*business to business* o B2B) i el comerç electrònic entre empresa i consumidor (*business to consumer* o B2C).

4.3. Amenaces a l'arquitectura web

4.3.1. Estat de la seguretat web

Guanyant una mica de perspectiva en el compliment dels principis bàsics de productivitat i reducció de costos associats a la gestió de les aplicacions web, es pot concloure que cal abordar una anàlisi de l'estat de la seguretat d'aquests sistemes. Per a això és imprescindible intentar localitzar els punts crítics de les aplicacions esmentades, amb la finalitat d'avaluar quines són les amenaces que planen sobre el sistema. Per aquest motiu s'han d'avaluar dades estadístiques sobre atacs implementats sobre diferents llocs web. A partir d'aquestes dades s'ha d'avaluar a quin tipus d'amenaces corresponen.

La localització de fonts a partir de les quals es puguin extreure dades de valor estadístic és un problema difícil de solucionar. Les implicacions que té sobre la imatge de l'empresa i les pèrdues derivades de la deterioració de l'empresa fan que una de les polítiques freqüents de les empreses sigui l'ocultació d'aquests incidents de seguretat, per a minimitzar els efectes sobre la productivitat i la relació amb els clients. Per això no es pot disposar de prou quantitat de dades reconegudes oficialment per les empreses per a establir resultats amb validesa estadística.

L'única solució per a obtenir informació de quins són els tipus de sistemes que són víctimes d'atacs més sovint, quines són les motivacions dels atacants o si l'atac està enfocat al sistema operatiu o a les aplicacions és localitzar fonts alternatives de dades. De fet, una alternativa per a aquesta recerca d'informació és utilitzar llocs web com www.zone-h.org en què es poden obtenir dades sobre atacs implementats per diferents associacions de pirates i sobre les seves motivacions.

Atenent aquest repositori d'informació, i tenint en compte que no es tracta d'una informació absolutament fiable, sí que es poden extreure una sèrie de conclusions molt interessants.

El nombre d'intrusions web ha anat decreixent cada any per la implantació de noves contramesures i les actuals arquitectures web. Això es pot deure al fet que la implementació dels atacs cada vegada exigeix més professionalització per a contrarestar els alts nivells tècnics introduïts amb les últimes tecnologies de protecció. De fet, durant els tres primers mesos del 2009 el nombre d'atacs registrats per Zone-H ha estat de 54.960, respecte als 108.514 registrats en el primer quadrimestre del 2008.

Els sistemes atacats no tenen un perfil clarament definit. De fet, es pot concloure que s'ataquen totes les tecnologies, des dels sistemes operatius fins a les aplicacions, sense tenir en compte fabricants específics.

Els motius són variats però es poden resumir en els següents:

- Econòmics.
- Per venjança.
- Polítics.
- Per diversió.

A partir d'aquests resultats es pot concloure que, malgrat que sembla que s'està avançant en la direcció adequada pel que fa a la implementació de mesures de protecció, encara s'estan assumint molts riscos a l'hora de plantejar les solucionis web. S'han de reduir els més de cinquanta mil atacs amb èxit per trimestre. Per a això s'ha d'aprofundir més en l'anàlisi de les amenaces definides sobre l'arquitectura i les aplicacions web.

4.3.2. Avaluació de riscos sobre l'arquitectura web

L'avaluació dels riscos que sorgeixen a partir de la proposta d'arquitectura web és tan complexa que cal descompondre l'anàlisi per a determinar sobre quins components de l'arquitectura planen les amenaces potencialment més perilloses.

1) Riscos sobre els clients

En la majoria dels casos els clients accedeixen als recursos de manera local o remota pel navegador d'Internet. Aquest programari executa codi a escala d'usuari, amb el nivell de privilegis que tingui definit l'usuari en el perfil. Els codis que s'executen des del navegador són potencialment perillosos per la gran funcionalitat que es pot implementar amb aquests codis. Els llenguatges més estesos són HTML/DHTML, VBScript, JavaScript o JScript. El navegador web permet incrustar programes que s'han dut a terme com a miniaplicacions (*applets*) de Java, ActiveX o Shockwave de Flash. A més, normalment presenta el codi sense establir cap mesura de protecció; no solen aparèixer encriptats ni ofuscats. Aquesta última característica és especialment sagnant quan hi ha moltes alternatives per a establir un nivell de protecció en el codi (per exemple, Atrise Stealth o HTMLLock).

En realitat, no hi ha cap protecció en el client que sigui bona ja que, a més dels atacs directes o amb descompiladors al navegador web del client, hi ha les tècniques d'intercepció (*man-in-the-middle attack* o MiM) que aparten completament l'usuari de la possibilitat d'evitar l'èxit d'un atac llançat sobre el sistema. Algunes de les eines que permeten avaluar la potencial perillositat dels atacs MiM són les conegudes Achilles , BurpSuite o Odysseus.

2) Riscos sobre la lògica de l'aplicació

El servidor de l'aplicació concentra tota la lògica de l'aplicació i per a això necessita executar codi en contextos privilegiats, de manera que es converteix en un objectiu formidable per a un atac. Aquests codis estan desenvolupats usant potents llenguatges de programació que incrementen molt la criticitat d'un possible atac sobre el sistema. Com hem explicat, aquest component accedeix a la base de dades, envia programes a clients, transfereix fitxers i executa ordres sobre el sistema. A més, proporciona suport a eines per a administrar altres aplicacions i presenta codis d'exemple per a facilitar les tasques de configuració.

Tenint en compte tot això, aquest component és el candidat ideal per a convertir-se en l'objectiu dels atacs o, en la majoria dels casos, servir de vehicle per a implementar atacs sobre altres components –sobre els clients mitjançant atacs

de *cross site scripting*, sobre els magatzems de dades mitjançant atacs d'injecció de codi (SQL Injection, LDAP Injection o XPATH Injection), sobre el sistema amb la injecció de fitxers).

3) Riscos sobre els magatzems de dades

L'emmagatzematge d'informació se sol considerar la clau de negoci ja que un atac a la confidencialitat o la integritat pot implicar enormes pèrdues econòmiques i també, en determinats casos, el trencament de la LOPD. L'emmagatzematge de les dades se sol fer en bases de dades relacionals o jeràrquiques a què s'hi accedeix mitjançant llenguatges de més o menys potència. En el cas de les bases de dades basades en objectes, els llenguatges són molt avançats (tercera generació o quarta), i permeten una altíssima interacció amb el sistema. Un atac amb èxit sobre aquest tipus de sistemes pot resultar molt crític, atesa la facilitat per a explotar-los quan s'han guanyat els privilegis necessaris. De fet, un atac d'aquesta mena podria resultar demolidor si s'ha configurat el sistema perquè es permeti executar programes sobre el sistema.

En el cas de les bases de dades jeràrquiques, la informació emmagatzemada també és important. És freqüent emmagatzemar el catàleg global de dades i proporcionar un llenguatge per a garantir l'accés. Normalment aquests llenguatges són més limitats que els que es defineixen per a altres entorns, però així i tot, si s'ha guanyat un nivell de privilegis suficient, es podran usar per a implementar un atac que permeti extreure tota la informació emmagatzemada.

5. Arquitectura de bases de dades

Encara que és molt difícil conèixer totes les arquitectures a fons, és molt necessari conèixer els principis de totes. Un estudi general permet establir paral·lelismes entre els principals productes i l'evolució que han seguit al llarg del temps. A més de preveure l'escenari en què es crea una base de dades i es compra una llicència d'ús de qualsevol fabricant de l'última versió de programari d'aquesta base de dades, és molt comú treballar amb versions antigues, i per això té aquesta importància conèixer les diferents versions de cada arquitectura.

Hi ha un principi que formula que la base de dades més segura és la que es coneix més bé. L'elecció ha de ser un compromís entre la necessitat i la funcionalitat que ofereix cada arquitectura, i quan s'ha fet l'elecció entre l'oferta del mercat, s'ha d'estudiar a fons per aconseguir un nivell òptim de seguretat.

5.1. Oracle

Probablement, Oracle és la base de dades més coneguda i popular del món. Una mostra de la popularitat que té és que és present en 98 de les 100 indústries incloses en el *top* 100 de *Fortune*. Va ser un dels primers venedors del mercat i s'executa en una gran quantitat de plataformes, a més de proporcionar un molt bon producte, cosa que explica la popularitat que té.

No obstant això, ha tingut un historial de problemes relacionat amb la gran quantitat de funcionalitats que proporciona, especialment la injecció de PL/SQL (el llenguatge *script* proporcionat pel producte) i els desbordaments de la memòria intermèdia en diferents parts de codi.

5.1.1. Història i evolució

Com que és un producte molt veterà en el mercat, hi ha una llarga història de versions d'Oracle. Encara més, quan es parla d'Oracle, es parla d'un terme ambigu que pot donar lloc a confusió per la gran quantitat de productes que hi ha d'aquest venedor.

Les primeres versions tenen interès a escala històrica, encara que avui dia no són vigents en el mercat. Les versions més "actuals" fins avui són aquestes:

- Oracle versió 8i (1999): incorpora millores per a respondre a les necessitats d'Internet (com indica la *i* del nom) i inclou una màquina virtual de Java en el motor de la base de dades (JVM).

- Oracle 9i (2001): té suport per a XML i clusterització (*clustering*), entre quatre-centes noves funcionalitats.

- Oracle 10g (2003): en aquest cas, la *g* del nom posa èmfasi en *grid computing*.

- Oracle 10.2.0.1 (2005): conegut com a Oracle 10g Release 2 (10gR2).

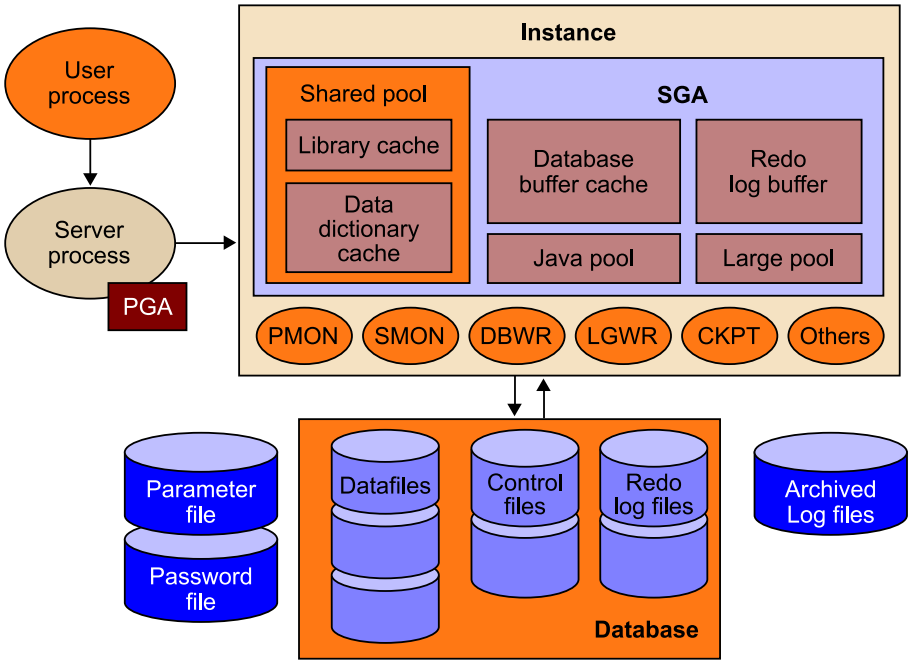
Les diferents versions es proporcionen en diferents productes o edicions, això és, Enterprise Edition, Standard Edition i Standard Edition One, a més de la versió Express per a avaluació i entorns de desenvolupament. Totes utilitzen la mateixa arquitectura, i encara que proporcionen diferents funcionalitats, pel que fa als conceptes i a la seguretat tenen les mateixes característiques.

A part del motor de base de dades, hi ha una gran quantitat de programari que proporciona funcionalitat addicional i interacció amb diferents aspectes de la base de dades, com desenvolupament de programari (Oracle Developer Suite), un servidor d'aplicacions (Oracle Application Server) o clusterització.

5.1.2. Arquitectura

Encara que l'arquitectura específica per a cadascuna de les versions d'Oracle és diferent, hi ha una sèrie de conceptes que constitueixen l'esquelet d'una base de dades Oracle heretats des de la versió 7 fins a les més recents. L'objectiu d'aquest material no és conèixer-ne amb detall tots els aspectes, però sí les idees principals:

Figura 5. Arquitectura d'Oracle



En aquesta imatge s'hi veuen els participants en una base de dades Oracle.

- **Procés de l'usuari**: es connecta al servidor per a interactuar amb la base de dades.

- **Procés del servidor**: escolta els processos d'usuari i fa les peticions a una instància de la base de dades.

- **PGA** (***program global area***): memòria reservada per a un procés d'usuari. S'allibera quan es mor la sessió.

- **Instància**: conjunt de processos tant lògics com de control per a accedir a les dades en si, emmagatzemades en fitxers físics. Accedeix a una base de dades, i només a una. Dins de la instància hi ha una sèrie de subprocessos que detallem a continuació:
  - SMON (*system monitor*): recupera la instància en arrencar la base de dades.
  - PMON (*process monitor*): monitora els processos d'usuari i allibera els recursos que ocupen aquests processos quan acaben.
  - DBWR (*database writer*): escriu les dades físicament en els fitxers de registres (*logs*) de refer (*redo*) quan cal alliberar aquestes memòries intermèdies de la instància.
  - ARCH (*archiver*) (antic LGWR): quan els fitxers de registres de refer són plens, escriu les dades en els fitxers corresponents per alliberar espai.
  - CKPT (*checkpoint process*): cada vegada que es buiden les memòries intermèdies i s'escriuen en fitxers de registres, es produeix un punt de control (*checkpoint*). Aquest procés és l'encarregat de coordinar totes les operacions corresponents a aquest esdeveniment i guardar la coherència de la base de dades mitjançant l'escriptura de les dades en tots els fitxers que correspongui i assegurar que les operacions han tingut èxit.
  - RECO (*recover*): elimina i neteja qualsevol dada incoherent que resulti d'una transacció distribuïda fallida. Aquest procés s'inclou en la versió d'Oracle 10g.

S'ha de tenir en compte que els processos principals es mantenen per mitjà de les diferents versions. No obstant això, és inevitable que n'hi hagi alguns que vagin canviant arran de les noves funcionalitats que s'inclouen en les versions noves, com s'aprecia en l'últim dels processos descrits abans.

- SGA (*system global area*): és l'àrea de memòria creada en arrencar una instància d'Oracle. Té una sèrie de subàrees:

  - *Shared pool* ('bateria compartida'): emmagatzema tant l'estructura i els índexs dels últims objectes a què s'ha accedit en la base de dades com les funcions i els procediments PL/SQL que s'han utilitzat.

  - *Redo log buffer* ('refer el registre de la memòria compartida'): utilitzat per a desfer les accions sobre les dades físiques en cas de fer una acció de repetir (*rollback*) de les accions que s'han dut a terme durant una sessió.

  - *Database buffer cache* ('memòria cau de la memòria intermèdia de la base de dades'): fa referència a totes les dades que memoritza el servidor per evitar accedir a les dades físiques cada vegada que es fa una consulta, de manera que afavoreix el rendiment per a consultes repetitives que fan referència a les mateixes dades.

  - Java *pool* ('bateria de Java')i *large pool* ('bateria gran'): conceptualment són el mateix que la *shared pool*, però es divideixen per a utilitzar-los tant en accés d'entrada i sortida com per a requeriments d'anàlisi (*parcing*) de Java. No tenen un interès especial.

- *Database* ('base de dades'): fa referència als fitxers físics que emmagatzemen les dades. Es divideixen en:
  - **Fitxers de control** (extensió .ctl) que emmagatzemen informació sobre els fitxers que formen part de la base de dades i s'encarreguen de mantenir la consistència de la base de dades.

  - **Fitxers de dades** (extensió .dbf) que emmagatzemen físicament les dades (s'ha de tenir en compte que una única taula es pot emmagatzemar en *N* fitxers).

  - **Fitxers de refer registres (*redo-logs*)** (extensions .rdo i .arc) per a desfer accions en les dades físiques fins a un punt de control, especialment pensats per a recuperar el sistema en un punt segur en cas d'una fallada.

- *Parameter file* ('fitxer de paràmetres'), *password file* ('fitxer de contrasenyes')i *archived log files* ('fitxers de registres arxivats'): fitxers de control que guarden els paràmetres específics per a un servidor Oracle (normalment Init.ora), contrasenyes (en desús en versions actuals del servidor) i arxius de registre del servidor.

Fins aquí hem comentat l'arquitectura bàsica pel que fa a processos i fitxers, és a dir, l'arquitectura física. No obstant això, l'estructura interna lògica pel que fa taules i estructures de dades és fins i tot més important, ja que normalment es tracta el nivell de seguretat dins del context d'un usuari amb accés a la base de dades.

Les dades a Oracle s'organitzen per bases de dades, o contenidors (*tablespaces*). Per a cadascuna d'aquestes bases de dades, els noms són únics. En cada instal·lació d'Oracle hi ha una base de dades per defecte en què es guarda l'esquema (*schema*) de tots els objectes del sistema. Això és, tots els usuaris, bases de dades, taules, vistes, procediments, funcions, etc. Com és normal, els usuaris no tenen tots permisos per a accedir a aquesta base de dades, i ha d'estar ben protegida, ja que conté informació vital, en aquest cas, metadades.

Aquesta base de dades és *sys*, que es correspon amb el catàleg del sistema. A continuació fem una llista d'alguns dels objectes més importants que inclou:

- `all_tables`: conté informació de totes les taules del sistema.
- `dba_users`: conté informació d'usuaris que tenen permisos de DBA.
- `all_users`: conté informació de tots els usuaris de la base de dades.
- `tabs`: conté informació de totes les taules d'usuari.
- `user_tab_columns`: conté informació de columnes de les taules d'usuari.
- `all_db_links`: conté informació dels enllaços amb altres bases de dades.

La diferència entre moltes d'aquestes estructures és el prefix, que especifica quin usuari té permisos per a consultar-les i quin nivell d'informació ofereixen. Les que tenen prefix *dba_* només són accessibles per a usuaris amb aquest nivell de permisos i són les que ofereixen més informació. S'ha de tenir en compte que, per a no tenir dades redundants, aquestes estructures són vistes, de manera que les dades reals no s'hi emmagatzemen.

Per això, en alguns casos, es poden modificar per a fer invisible informació per alguns (o tots) els usuaris.

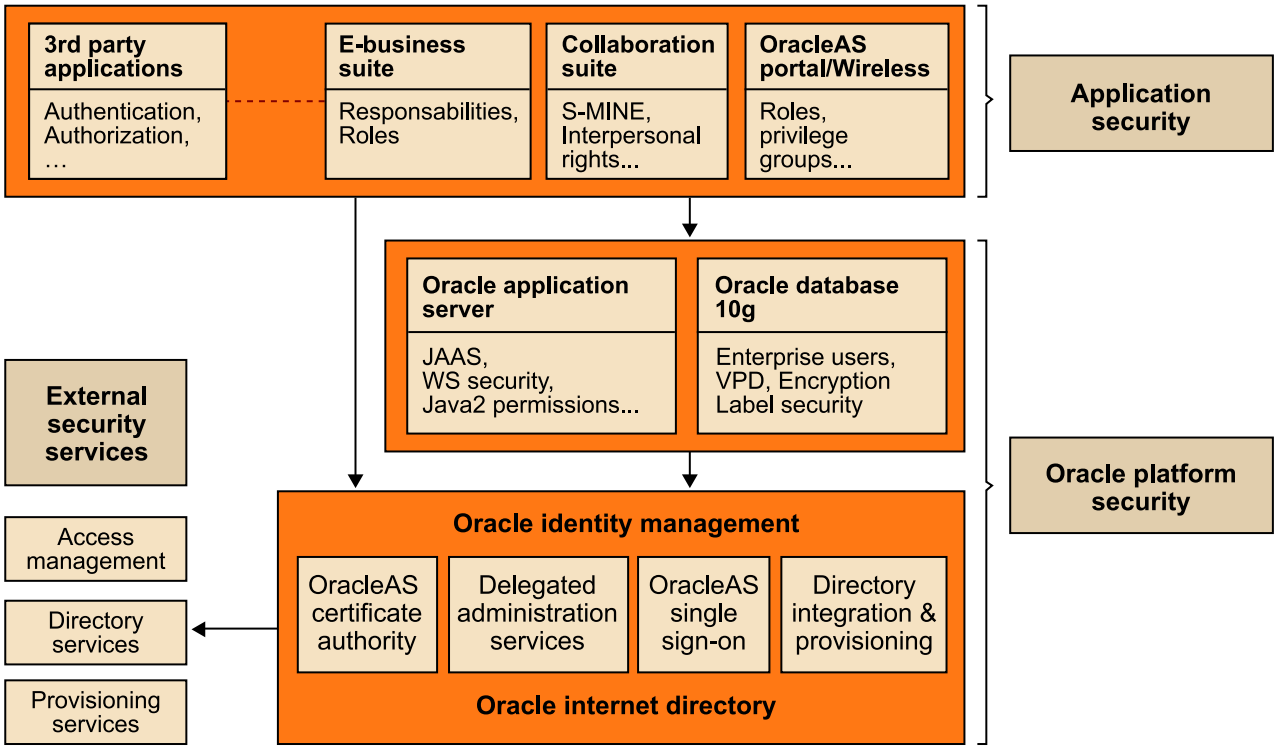
La taula anomenada *dual* és utilitzada com a comodí per Oracle per a fer consultes sense tenir com a objectiu cap taula en concret. Per exemple, es pot seleccionar l'usuari de la sessió de la base de dades amb la consulta següent:

```
select user() from dual
```

En aquest cas no es fa una consulta d'una informació resident en la taula *dual*, sinó que s'utilitza per a usar la sintaxi SQL tenint en compte que aquesta taula en realitat només conté una columna anomenada *dummy* i un únic registre.

Pel que fa a l'autenticació contra la base de dades, Oracle permet autenticar mitjançant el seu propi sistema, mitjançant el sistema operatiu o mitjançant un sistema mixt. A partir de la versió 10g, Oracle implementa l'Oracle Identity Management, que permet integrar l'autenticació de la base de dades amb el sistema operatiu, amb sistemes LDAP, amb sistemes de PKI, etc.

Figura 6. Oracle Identity Management



Una vegada autenticats, els usuaris tenen un rol en la base de dades que els atorga una sèrie de permisos. Aquest apartat és més interessant que el mitjà d'autenticació, ja que segons el rol que tingui assignat l'usuari tindrà uns permisos o uns altres.

Els permisos per a aquests rols s'emmagatzemen en les taules:

- `system_privilege_map`
- `table_privilege_map`

Encara que a partir de la consulta de l'objecte `dba_users_privs` es poden consultar els permisos d'un usuari.

Els rols amb màxims permisos en el sistema són SYS i SYSTEM. Hi ha diversos comptes amb privilegis de DBA (*database administator* 'administrador de base de dades'), que seria el segon compte amb més permisos del sistema, com *ctxsys*, *wksys*, *mdsys* o *sysman*, encara que hi ha llistes amb més de quatre-cents comptes utilitzats per diferents versions d'Oracle. Aquest problema de seguretat és un dels que tenen les instal·lacions d'Oracle per defecte.

A continuació comentarem els processos que executa una base de dades Oracle, i també la funció de cadascun d'aquests processos, i destacarem els més importants.

D'entrada, s'ha de tenir en compte que Oracle es pot executar en una gran quantitat de sistemes operatius, cosa que sumada a les diferents versions de la base de dades dóna un nombre potencialment alt de combinacions. No obstant això, es pot usar el criteri següent com a guia.

En sistemes Windows, el procés principal d'Oracle té el nom d'*oracle.exe*.

En sistemes *nix (Unix i Linux), hi ha un executable per a cadascuna de les funcions que hem destacat abans en aquest apartat. Aquest conjunt de funcions depèn de la versió exacta de la base de dades, però en general hi ha els processos següents:

- `pmon`
- `smon`
- `reco`
- `dbwr`
- `lgwr`
- `ckpt`

Un dels processos més destacables és el *TNSlistener* (*transparent network substrate* 'substrat de xarxa transparent'). Aquest procés té el rol de "procés del servidor" que hem descrit més amunt, és a dir, espera les peticions d'un usuari i les transmet a la base de dades per a retornar-ne la resposta; en aquest cas atén les peticions de xarxa.

Aquí hi ha dos binaris associats: el *tnslnsr*, que és el *listener* mateix, i el *lsnctrl*, que és la utilitat de control del *listener*.

En general, aquest procés escolta pel port 1521 TCP, encara que és un paràmetre que es pot configurar en cada instància de la base de dades. Quan un client vol connectar amb la base de dades, fa una petició al *listener*, que coneix l'estat actual de la base de dades. Si tot és correcte, respon amb un port al qual s'ha de connectar el client per fer les consultes directament contra la base de dades, mitjançant l'autenticació corresponent.

Aquest procés pot estar protegit o no. En les versions anteriors a 10g no ho estava en la configuració per defecte, cosa que comportava un seriós risc de seguretat, ja que es permetia l'administració de *lsnctrl* remotament tret que l'administrador de la base de dades ho evités amb una configuració personalitzada.

En cas de trobar un procés TNS *listener* desprotegit, es pot conèixer informació sobre la instància d'Oracle, i també es poden conèixer els noms de les bases de dades que allotja i més informació del sistema. Això sense tenir en compte un

Vegeu també

En el mòdul "Atacs a aplicacions web" veurem detalladament els atacs de força bruta.

gran nombre de vulnerabilitats més avançades que descriurem més endavant. En cas d'estar protegit, també es pot obtenir informació sobre els noms de les bases de dades que allotja mitjançant atacs de força bruta.

Un altre procés destacable és l'*Oracle intelligent agent* ('agent intel·ligent d'Oracle), que escolta pels ports TCP 1748, 1808 i 1809. Fa diverses funcions, entre les quals hi ha guardar dades relatives a l'administració de les dades i al rendiment que tenen. Aquest procés és interessant perquè es pot consultar de manera remota sense haver de presentar cap mena d'autenticació, de manera que proporciona informació que pot resultar interessant a un atacant, com els processos en execució o l'ús de memòria.

5.2. Microsoft SQL

Microsoft SQL Server és una base de dades molt popular, en bona part gràcies a la gran quantitat de sistemes Microsoft Windows que hi ha arreu del món, cosa que no vol dir que no sigui un bon producte. Aquest motor de base de dades solament és disponible per a sistemes Windows.

Microsoft va començar l'evolució en solitari del seu motor de base de dades basat en el codi desenvolupat per Sybase, que hi va col·laborar en les primeres versions d'SQL Server. No obstant això, a finals dels noranta, Microsoft va adquirir tots els drets d'SQL Server i en va començar a fer el desenvolupament basat en el codi de Sybase, però reescrivint-lo tot.

L'adopció que en va fer el mercat va començar a partir d'empreses petites i mitjanes, en les quals acostumen (o acostumaven) a haver-hi força sistemes Windows, encara que ha anat guanyant quota de mercat en les últimes versions alhora que en millorava el rendiment, la seguretat i l'escalabilitat i s'adoptaven solucions Windows per a servidors de producció, també en grans empreses. Aquesta última conseqüència, presumiblement, es deu a la millora dels sistemes servidors Windows en les últimes versions. S'ha de destacar que en el moment d'entrar en mercat, MS SQL Server no era un obstacle en l'àmbit econòmic tan important com altres productes més consolidats en el mercat, cosa que sens dubte va ajudar a adoptar-lo inicialment. El 2001 era la base de dades més popular en sistemes Windows.

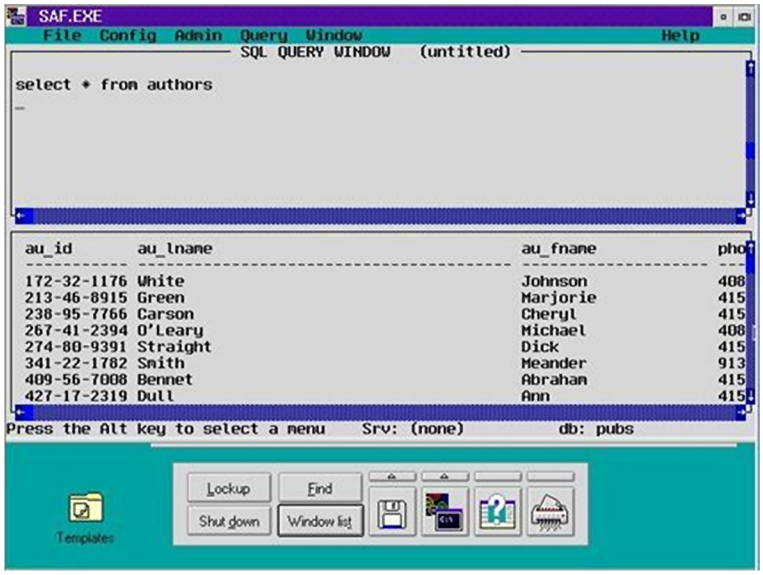
La versió d'SQL implementada per SQL Server es coneix com a Transact-SQL (T-SQL), i és una variant de l'estàndard SQL-92 adoptat per l'empresa, encara que amb petites característiques pròpies, tal com introdueixen tots els venedors de bases de dades.

5.2.1. Història i evolució

Microsoft va començar a desenvolupar a finals dels vuitanta juntament amb Sybase la primera versió d'un motor de base de dades per a competir amb les grans empreses hegemòniques en el mercat en aquella època, com Oracle o

IBM. Fruit d'aquesta feina va sortir SQL Server 1.0 per a OS/2. Després va sortir la versió 3.0 per a Unix, però no va ser fins al 1992 quan va sortir la primera versió de Microsoft SQL Server com a tal, la 4.2.

Figura 7. SQL Server 1.0 en OS/2



La següent versió que va sortir va ser la 6.0, per a arquitectures NT. A partir d'aquesta versió, Microsoft va negociar l'exclusivitat per al codi d'SQL Server en els seus sistemes operatius, de manera que es va acabar l'aventura conjunta amb Sybase.

El 1997 surt SQL Server 7.0, versió reescrita completament a partir del codi original de Sybase, i és la primera versió que incorpora una interfície gràfica. És un gran èxit comercial i es considera la primera versió pròpia completament de Microsoft.

A partir d'aquest punt les versions han estat SQL Server 2000, llançada el 2000 i el 2005, totes dues en diferents paquets amb diferent funcionalitat per a abastar tot l'espectre professional del mercat. Aquestes versions són evolucions "naturals" del producte segons les demandes del mercat, i han aconseguit que actualment sigui un motor de base de dades fiable i amb pocs problemes de seguretat coneguts. Les últimes versions són compatibles amb arquitectures de 64 bits.

5.2.2. Arquitectura

En contrast amb altres bases de dades que funcionen en diferents sistemes operatius, SQL Server només ho fa en Microsoft Windows, cosa que determina moltes de les característiques que té. Les possibilitats de combinació de sistema operatiu – maquinari són molt més petites que per a alguns dels seus competidors, com Oracle, en què aquesta combinació pot arribar a vint-i-sis casos. Per això, en alguns aspectes, aquest fet pot ser que repercuteixi directament en la seguretat, com en alguns cucs que utilitzen desbordament de memòria inter-

mèdia per a aconseguir l'execució de codi maliciós i puguin aprofitar adreces de memòria o trucades a l'API del sistema operatiu sense haver de fer front a una casuística complexa.

En aquest apartat explicarem les característiques principals de l'arquitectura d'SQL Server 2000. Tant la versió anterior (7) com la posterior (2005) són molt semblants i les diferències les comentarem quan caldrà. Tot i que hi ha diferències, no són rellevants, ja que el nostre objectiu és presentar les principals característiques estructurals de les bases de dades pel que fa a seguretat.

El principal procés d'SQL Server és `sqlservr.exe` (servei MSSQLServer), i per defecte escolta les peticions pel port 1433. Es tracta del servei de la base de dades pròpiament dita, que accedeix a les dades i les emmagatzema. La versió d'SQL Server 2000 també escolta pel port 1434 UDP amb el servei SQL Server Monitor, que informa de l'estat del servidor a una petició concreta. Com veurem més endavant, aquest port proporciona possible informació a un atacant, de manera que no se n'aconsella l'accés públic. Aquests ports es poden canviar, la qual cosa és una pràctica recomanable en qualsevol base de dades per a evitar que es desveli de manera immediata el servei que hi ha darrere un port. SQL Server disposa d'una opció per a "ocultar-se" d'escanejos, anomenada *Hide server*, mitjançant la qual automàticament passa a escoltar pel port 2433 i deixa de respondre a peticions *broadcast*, encara que aquesta opció té alguns problemes quan hi ha moltes instàncies del servidor.

Un altre procés important és `sqlmangr.exe` (*SQLmanager*), la utilitat del qual és arrencar i parar el servidor de la base de dades i també l'agent SQL (*SQLagent* , servei SQLServerAgent) que és el procés que s'encarrega de llançar tasques i procediments de manera programada anteriorment, una espècie de planificador per a llançar feines contra la base de dades. En la versió 2005, aquest procés ha tingut millores importants en aspectes de seguretat.

Finalment, hi ha altres serveis en SQL Server:

- Open Data Services: és una interfície entre les biblioteques (*net-libraries*) i les aplicacions que utilitza el servidor. El funcionament d'aquest servei està intrínsecament lligat amb el dels procediments ampliats, que explicarem més endavant en aquest apartat.

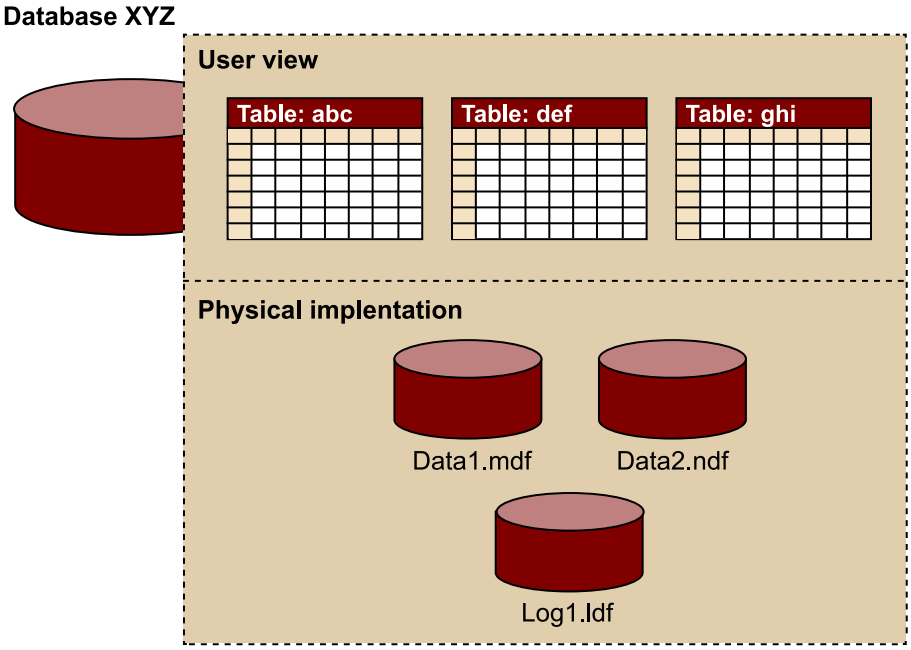
- Microsoft Search Service: proporciona suport per a funcionalitats de recerca en text i d'indexació en taules. Aquesta funcionalitat s'implementava com un servei diferent en SQL Server versió 7, encara que després es va integrar amb el servei MSSQLServer.

- Microsoft Distributed Transaction Coordinator (MS DTC Service): és un servei per a coordinar transaccions entre diferents servidors implicats en una transacció distribuïda.

Les dades físiques es guarden en fitxers amb les extensions següents:

- mdf: son els fitxers primaris de la base de dades i apunten a la resta de fitxers que en formen part.
- ndf: son els fitxers secundaris i guarden tota la informació que no es guarda en els fitxers primaris.
- lgf: guarden tota la informació de registre relativa a la base de dades utilitzada per a recuperar dades en un punt de temps fixat.

Figura 8. Tipus de fitxers



Cada instal·lació d'SQL Server té quatre bases de dades per defecte:

- *master*: és la més important i emmagatzema la informació de catàleg de la resta d'objectes de la base de dades.
- *model*: guarda les plantilles per a crear totes les bases de dades del sistema.
- *tempdb*: guarda informació temporal de taules i de procediments.
- *msdb*: la utilitza el servidor per a llançar alertes i treballs programats amb l'SQL Server Agent.

Pel que fa a les taules més importants del catàleg d'SQL Server, l'esquema s'emmagatzema en la base de dades *master*. Algunes de les taules més destacables són aquestes:

- sysobjects: emmagatzema totes les taules de les bases de dades.
- syscolumns: emmagatzema les columnes de les taules de les bases de dades.
- sysusers: dóna informació sobre els usuaris de la base de dades.
- sysdatabases: indica les bases de dades disponibles.
- sysxlogins: dóna informació sobre els comptes d'usuari.

SQL Server disposa de diversos rols d'usuari predefinits que tenen uns permisos preestablerts. Aquests rols es divideixen en **rols de servidor**, que tenen en compte aspectes de l'administració del servidor de base de dades, i **rols de base de dades**, que tenen en compte aspectes que es refereixen a les bases de dades en si. Aquests rols no es poden canviar, crear ni esborrar. Els més destacables de servidor són aquests:

- dbcreator: permet crear i administrar bases de dades.
- diskadmin: permet accedir a dispositius d'emmagatzematge físics i administrar-los.
- securityadmin: permet crear i administrar rols i usuaris.
- sysadmin: permet tenir el control administratiu total sobre el servidor.

Pel que fa als rols de base de dades, els més destacables són aquests:

- db_datareader: permet fer l'accés de lectura a una base de dades.
- db_datawriter: permet modificar, crear i esborrar dades.
- db_owner: permet fer qualsevol operació en la base de dades.
- db_securityadmin: permet fer l'administració de rols i de permisos d'objectes.

SQL Server permet crear rols d'usuari als quals es permet assignar permisos de manera personalitzada amb qualsevol objecte de la base de dades.

L'autenticació en SQL Server es fa mitjançant dos mecanismes: autenticació nativa de la base de dades i autenticació mitjançant el sistema operatiu. També es proporcionen tres fórmules per a aquest procés:

- nativa,
- sistema operatiu,
- mixta.

El fet de poder utilitzar l'**autenticació del sistema operatiu** és una de les característiques derivades de la relació estreta que hi té, i també de no haver de ser tan flexible per a executar-se en altres sistemes operatius que no siguin de Microsoft.

En el cas de l'**autenticació nativa** d'SQL Server, les dades que es transmeten al servidor no es xifren, sinó que solament s'ofusquen mitjançant una transformació reversible, de manera que si l'autenticació no es fa de manera local o mitjançant una connexió segura, poden ser interceptades i capturades sense dificultat, per la qual cosa és recomanable utilitzar l'autenticació que proporciona mitjançant el sistema operatiu o assegurar que la comunicació sempre és per un canal encriptat.

Versió 2000

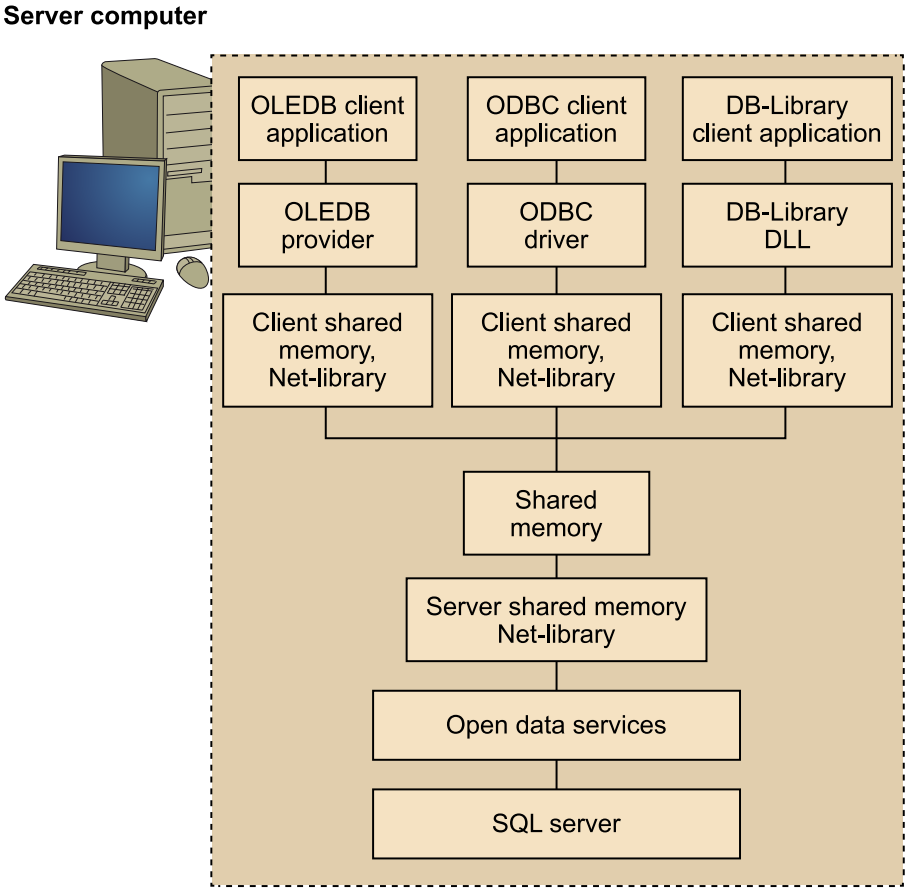
Aquest problema es dóna en la versió 2000 de la base de dades, ja que la 2005 proporciona noves funcionalitats per a solucionar aquest problema, fins i tot proporcionant l'opció de crear certificats digitals per a l'autenticació d'usuaris. A més, subministra funcionalitat per a exigir una mínima complexitat i longitud en les contrasenyes, i també un temps d'expiració per a fer-ne la renovació regular.

L'**autenticació mixta** simplement barreja tots dos sistemes i fa el procés en dues fases: primer comprova les credencials mitjançant el sistema operatiu i després amb les credencials pròpies per a l'usuari de base de dades.

TDS (*tabular data stream* 'flux de dades tabulars') és el protocol que utilitza SQL Server per a la comunicació client–servidor, incloent-hi l'autenticació i les peticions i respostes de consultes a la base de dades. Hi ha implementacions lliures d'aquest protocol més enllà de la que proporciona Microsoft.

Network libraries ('biblioteques de xarxa') són les biblioteques que utilitza SQL Server per a la comunicació per xarxa amb els clients. Per defecte la comunicació es fa mitjançant TPC/IP, encara que també es pot especificar mitjançant *named pipes* (en aquest cas més lenta i solament en el mateix servidor), mitjançant memòria compartida (*shared memory*), en què la comunicació també és local però obté la màxima velocitat disponible, o mitjançant SSL (capa de sòcol segur o *secure socket layer*) per a encriptar la comunicació entre els dos extrems. Hi ha diverses biblioteques addicionals que resisteixen protocols menys comuns, la majoria dels quals són antics i només existeixen per qüestions de compatibilitat.

Figura 9. Protocols compatibles per a comunicació client-servidor



Els procediments emmagatzemats en SQL Server tenen la mateixa funcionalitat que en altres bases de dades, que és la d'ampliar la capacitat de les consultes creant procediments mitjançant el llenguatge de programació TSQL. A més, SQL Server proporciona els procediments emmagatzemats ampliats (*extended stored procedures*, XSP), que permeten crear nova funcionalitat mitjançant procediments en llenguatge d'alt nivell amb C o C++ i emmagatzemar-los en *dll* perquè la base de dades pugui accedir a aquestes funcions mitjançant l'API d'Open Data Services.

Alguns dels procediments emmagatzemats d'SQL Server són el primer objectiu d'un atacant a la base de dades, per la gran potència que tenen, com veurem més endavant. S'ha de destacar que procediments com xp_cmdshell permeten l'execució d'ordres directament contra el sistema operatiu de la màquina que allotja la base de dades, de manera que s'entén de seguida per què aquests procediments són el primer objectiu d'un atac. Per això, és recomanable no donar permisos a cap usuari que no sigui l'administrador, o eliminar-los completament si no són necessaris. En cas d'eliminar-los de la base de dades, una bona idea és esborrar les *dll* que proporcionen la funcionalitat per a evitar que es tornin a incloure dins de les funcionalitats de la base de dades.

SQL Server proporciona un mecanisme d'encriptació dels procediments emmagatzemats per a evitar la consulta del codi d'aquests procediments.

5.3. MySQL

5.3.1. Introducció

MySQL és una base de dades molt popular avui dia malgrat que és una de les últimes de sortir al mercat tot i la falta de moltes funcionalitats fins a versions molt recents. Segons la companyia, hi ha més de deu milions d'instal·lacions del seu producte actualment. De fet, i des d'un punt de vista purista, MySQL no ha tingut algunes de les principals característiques considerades necessàries per a un motor de bases relacional. No obstant això, la seva popularitat es deu a la conjunció del *boom* d'Internet i al fet de ser una base de dades de codi lliure, encara que aquest aspecte el matisarem més endavant.

La falta de característiques funcionals d'integritat no ha estat un obstacle per a la seva popularitat. En un moment d'aparició de gran quantitat d'aplicacions amb llicències lliures de tota mena, la possibilitat de disposar d'una base de dades amb aquestes mateixes condicions i amb un rendiment acceptable per a les operacions menys complexes va ser tota una fita.

A partir d'aquest punt, MySQL va anar cobrant popularitat molt de pressa, i va passar a un nou model de negoci en les versions més recents. Fins llavors els seus ingressos es basaven en les certificacions i el cobrament de llicències per a empreses que volien fer servir el producte amb llicència no GPL. No obstant això, a partir de la versió 5 es va crear un nou concepte amb la versió de MySQL Enterprise, que inclou a més del motor de base de dades relacional un suport continu per a clients, pensat per a un mercat més empresarial en què aquesta mena de serveis són necessaris. Per a no perdre les seves arrels, MySQL continua en paral·lel el desenvolupament de MySQL 5 Community Edition en un model semblant al que ha adoptat Red Hat amb Fedora; la diferència entre les dues versions és la disponibilitat de binaris compilats i optimitzats per a diferents plataformes, que deixen de ser disponibles per a la versió Community, i també el compromís de la publicació de versions noves puntualment cada mes amb tots els errors (*bugs*) corregits per a la versió Enterprise. El codi continua disponible per a baixar-lo amb llicència GPL.

Malgrat la falta d'implementació que té de característiques bàsiques en qualsevol base de dades relacional, com la integritat referencial, MySQL ha evolucionat de mica en mica fins a arribar a un punt en què disposa de les mateixes funcionalitats bàsiques que qualsevol altra base de dades comercial. No obstant això, és important no passar per alt que el camí de la seva popularització va ser la seva llicència lliure, la seva capacitat per a funcionar en pràcticament qualsevol plataforma i el seu funcionament correcte com a base de dades lleugera per a les funcionalitats més bàsiques. En particular, el que es coneix com a *plataforma LAMP* (amb les variacions que té), que ha estat un dels pilars de creació de gestors de contingut a Internet (LAMP = Linux + Apatxe + MySQL + PHP), cosa que ha permès pràcticament a qualsevol persona crear aplicaci-

ons amb una inversió mínima. MySQL va ser el complement ideal a una sèrie d'eines de filosofia semblant que va irrompre en el mercat en el moment adequat. Avui dia n'hi ha molts gestors de contingut lliures, de manera que el seu coneixement pel que fa a seguretat és molt important per la gran presència que té en el mercat. Encara més, a causa de la història que té i l'entorn de popularització, sol ser present en la DMZ de les companyies, de manera que constitueix un risc potencial si no està degudament protegit.

5.3.2. Història i evolució

MySQL és el nom de la base de dades pertanyent a MySQL AB, que és la companyia sueca que la va crear (*AB* és l'equivalent a *SA* en català). Des que es va crear, la companyia ha volgut instaurar una filosofia d'empresa semblant al famós "don't be evil" de Google, destacant una sèrie de valors que es veiessin reflectits en el seu comportament i els seus productes i que es resumeixen en una única frase enunciada per ells mateixos: "To make superior data management software available and affordable to all" ('Crear programari d'administració de dades de qualitat superior, disponible per a tothom'). La companyia la van fundar David Axmark, Allan Larsson i Monty Widenius. El nom de la base de dades és pel fill de Monty, que es diu My (nom suec).

Totes les versions de MySQL són disponibles per a gran quantitat de plataformes, la qual cosa no és sorprenent ja que és un producte de codi lliure. Sempre es pot intentar la compilació per a noves plataformes, però actualment és disponible en la versió Enterprise per a Red Hat Enterprise Linux, Novell SuSE Enterprise Linux, Debian GNU/Linux, Sun Solaris 10 i 9 i 8, HP-UX 11.23, Windows NT/2000, Windows XP, Windows 2003 Server, Apple Mac US X v10.4, IBM AIX, OpenServer 6.0, Fedora, openSUSE, CentOS i Ubuntu. La versió Community és disponible encara per a més plataformes, de manera que queda patent la flexibilitat del motor de base de dades.

El codi inicial el va crear Monty a finals dels vuitanta i principis dels noranta per a ús propi. A mesura que va créixer el projecte i s'hi van afegir noves funcionalitats, va començar a pensar a fundar una petita companyia per a comercialitzar el producte. A mitjan anys noranta funda MySQL AB i busquen inversors de capital de risc per a invertir en el desenvolupament del producte, cosa que dóna pas a les successives versions cada vegada amb més funcionalitats, si bé fins a la versió 5 no s'han implementat característiques considerades molt importants com ara els procediments emmagatzemats, els gallets (*triggers*) o el compliment de les propietats ACID (atomicitat, consistència, isolació i durabilitat), considerades bàsiques per a bases de dades relacionals. És en aquesta època en què la companyia es trasllada a Califòrnia.

L'adquisició de la companyia per part d'Oracle va posar tota aquesta filosofia en grans dificultats. Des de llavors hi ha una "community edition" de codi obert i una altra versió solament de pagament i que implementa les últimes característiques. Això ha fet que hagi perdut part del públic que tenia, que ha

decidit migrar a altres solucions de codi obert. Un dels fundadors de MySQL va decidir abandonar-la per crear una nova base de dades que seguís la mateixa filosofia inicial de MySQL.

5.3.3. Arquitectura

El principal programa de la base de dades és *mysqld*, encara que depenent de l'entorn pot ser *mysqld-nt*, que és l'equivalent per a entorns Windows, o *mysqld-max*, en cas que incorpori la base de dades MaxDB. Es tracta d'un suport a un altre tipus de base de dades que també desenvolupa MySQL AB, amb un impacte en el mercat molt marginal i que no tractarem. Aquest programa és el que s'encarrega de l'accés i el tractament del motor de la base de dades; la resta són programes client.

Hi ha diferents mètodes per a la comunicació client-servidor:

- TCP/IP, tant per a connexions locals com remotes.
- *Named pipes*, en connexions remotes només en sistemes Windows.
- Sòcols (*sockets*), en connexions remotes només en sistemes Unix.

En connexions TPC/IP, el servidor escolta per defecte pel port 3306. MySQL no utilitza cap protocol "estrany" de xarxa, i a partir de la versió 4.0 resisteix l'ús d'SSL en connexions de xarxa.

Hi ha diferents interfícies per a interactuar amb la base de dades, com la biblioteca C anomenada *libmysqlclient*, el connector ODBC o el connector JDBC.

API de MySQL

A part d'aquestes interfícies, que són les que resisteix oficialment MySQL, hi ha moltes API desenvolupades de manera independent compatibles amb PHP, Perl, Python, Ruby, Pascal i Tcl.

Pel que fa al tractament dels arxius per part de MySQL, s'associa cada base de dades amb un subdirectori sota el directori d'emmagatzematge de dades, especificat en el fitxer de configuració de MySQL (normalment en el fitxer *my.cnf*). Cada subdirectori té el nom de la base de dades que representa. Una base de dades pot ser que sigui buida i no ha de contenir cap subdirectori. Cada taula dins de la base de dades es representa en un fitxer amb extensió .frm que conté la definició de la taula. Pel que fa a l'emmagatzematge de les dades pròpiament dites, depèn del tipus de taules, i aquí és quan entra en joc una de les característiques que criden més l'atenció de MySQL.

A l'hora de crear una taula, se'n pot especificar el tipus, de manera que el motor de base de dades corresponent és el que s'encarrega de tractar-la. Això implica que, segons aquest tipus, les taules tenen diferents característiques, cosa que pot resultar confusa. L'aproximació més bona és pensar com si hi haguessin diferents bases de dades en una i que, a l'hora de definir les taules, l'usuari tria el motor amb què vol treballar. Segons aquest tipus, també es guarden els fitxers en disc amb un format o un altre. MySQL resisteix els tipus de taula següents:

- MyISAM: és el tipus de taula més popular i el tipus de taula "original" de MySQL disponible des de les primeres versions. El motor que s'encarrega de tractar aquest tipus no té moltes de les característiques desitjables per a bases de dades relacionals, de manera que no disposa de funcionalitat com a integritat referencial. Aquestes taules es representen en disc com a fitxers amb extensió .MYD per a emmagatzemar les dades i com a fitxers amb extensió .MYI per a emmagatzemar els índexs.

- InnoDB: aquest motor és el que, en bona part, ha permès que MySQL hagi evolucionat fins a ser una base de dades "seriosa" implementant característiques que no tenen els altres motors. És compatible amb les propietats ACID i permet transaccions atòmiques, de manera que es poden confirmar o refutar amb Commit i Rollback, respectivament. També implementa integritat referencial. En aquest cas, les taules no s'emmagatzemen en disc usant diferents fitxers, sinó que es guarden totes les dades en un únic espai d'emmagatzematge que utilitza el motor d'InnoDB i que pot consistir en un fitxer o diversos fitxers, a mesura que creix de mida. Per defecte, el nom de l'espai en què InnoDB emmagatzema les dades és *ibdata1*, i el de l'espai en què emmagatzema els registres, *ib_logfile0* i *ib_logfile1*.

Tant InnoDB com MyISAM són els principals motors de MySQL; els que hi ha en la llista següent tenen una utilitat més marginal:

- Merge: les taules d'aquest motor són una col·lecció de taules MyISAM idèntiques, representades en disc amb l'extensió .frm per al format de la taula i .MRG per a llistar totes les taules MyISAM que formen part de la taula Merge. En essència, es tracta de crear una entitat que aglutini un conjunt de taules MyISAM idèntiques amb la finalitat de sobrepassar les limitacions d'espai de les primeres.

- BDB (Berkeley DB): les taules en disc tenen una extensió .frm per al format i .db per a emmagatzematge de dades i índexs. Aquest motor implementa les propietats ACID i resisteix transaccions atòmiques, i també bloqueig de pàgina, cosa que pot provocar problemes de bloqueig com a preu per a un augment de la concurrència.

- Memory tables: són taules que, encara que tenen la definició en disc en un fitxer amb extensió .frm, guarden les dades i els índexs a la memòria, i resulten en el tipus de taules més ràpid disponible. Com que usen un recurs escàs, no és un tipus factible per a taules grans. Utilitza bloqueig de taula, cosa que baixa la concurrència i evita bloquejos.

En la versió 5 de MySQL s'introdueixen nous motors addicionals a aquests. S'ha de destacar que es pot crear un motor propi i utilitzar-lo en MySQL mitjançant un procés relativament senzill, cosa que afavoreix l'aparició de nous motors amb el pas del temps especialitzats en diferents funcionalitats. Els nous motors disponibles són Example, Federated, Archive, CSV i Blackhole.

Com hem vist, el motor de MySQL permet treballar amb diferents tipus de bases de dades amb característiques pròpies. No obstant això, s'ha de destacar que no fa gaire dues de les companyies que treballaven juntament amb MySQL en el desenvolupament d'aquestes bases de dades han estat adquirides per la competència. En concret, Oracle va comprar el 2005 Innobase OY, la companyia finlandesa encarregada de desenvolupar el motor InnoDB. Tenint en compte que aquest motor era el que implementava un nombre més gran de funcionalitats necessàries per a una base de dades relacional, implementant la integritat referencial des de les primeres versions, pot representar un revés important per a MySQL, encara que han arribat a un acord per a continuar el desenvolupament conjunt durant un temps.

Ha passat un cas semblant amb l'adquisició, també per Oracle, de Sleepycat Software, la companyia creadora de la Berkeley DB i també usada per MySQL.

Pel que fa a l'esquema de les bases de dades, des de la versió 5, MySQL incorpora la base de dades Information Schema, que conté les taules en què s'emmagatzema la informació respecte a la resta de bases de dades i taules (metadades). Algunes de les taules més interessants són aquestes:

- Tables: informació sobre les taules d'altres bases de dades.
- Columns: camps que componen les taules.
- Views: vistes disponibles.
- User_privileges: permisos de què disposen els usuaris.
- Routines: emmagatzematge de procediments i funcions.

En versions anteriors a la 5, aquesta base de dades no existia, de manera que aquestes dades s'havien de consultar en la base de dades anomenada *mysql*. Encara que no s'hi recollia informació respecte a les taules que conformaven altres bases de dades, s'hi guardava informació respecte als usuaris, incloent-hi els coixinets (*hashes*) de les contrasenyes. En concret, aquesta informació era disponible en la taula *user*.

Altres taules

Altres taules interessants eren *db*, *tables_priv* i *host*.

MySQL utilitza un sistema propi per a l'autenticació, consistent en l'ús d'un parell usuari-contrasenya. Permet especificar juntament amb el nom d'usuari l'amfitrió (*host*) des del qual es permet fer-ne l'autenticació, de manera que un intent de connexió des d'un altre dispositiu no permetria entrar en el sistema encara que se sabés tant l'usuari com la contrasenya correctes. No permet cap interacció amb el sistema operatiu per a l'autenticació.

Històricament, hi ha hagut problemes amb l'autenticació en MySQL. En concret, en versions anteriors a la 4.1, n'hi havia prou de saber el coixinet de la contrasenya per a aconseguir l'autenticació, sense haver de piratejar res. En versions anteriors a la 3.23.11, hi havia una debilitat que permetia aconseguir l'autenticació amb un únic caràcter, de manera que l'historial de la base de da-

des és dolent pel que fa a problemes d'autenticació, cosa que se suma al fet que en versions antigues no s'utilitzava cap tipus d'encriptació en l'autenticació per xarxa, de manera que les dades les podia interceptar qualsevol persona.

Les funcions definides per l'usuari (*user definded functions*, UDF) són extensions que permeten crear noves funcionalitats mitjançant la programació en llenguatges d'alt nivell (C, C++) i implementar-les des de la base de dades. Encara que d'aquest punt en tractarem amb més detall en pròxims capítols, s'ha de dir que la potència que aporta és enorme, i també els riscos en cas de no ser acurats.

El tractament en l'emmagatzematge de dades que s'utilitza en MySQL és bastant particular; no és freqüent trobar la correspondència tan directa entre bases de dades i taules amb fitxers en disc. Això implica que cal una cura especial a l'hora de protegir els permisos sobre aquests fitxers, perquè l'accés d'un usuari no autoritzat implica la disponibilitat directa de les dades, i fins i tot pot ser que les modifiqui sense gaire dificultat.

Pel que fa als permisos en la base de dades, no hi ha rols predefinits, sinó que els permisos s'assignen directament als usuaris. S'ha de destacar que, quan s'han canviat els privilegis, s'ha d'especificar de manera explícita que s'apliquin mitjançant l'ordre `FLUSH PRIVILEGES`, ja que fins llavors no es produeixen els canvis. Els permisos poden ser en dos nivells:

- El primer nivell es refereix als **objectes de la base de dades**, cosa que és habitual en altres bases de dades.

- El segon nivell es refereix a **permisos** d'executar certes accions **amb l'usuari** en qüestió, com accedir al disc per a carregar dades d'un arxiu concret o per a guardar els resultats d'una consulta en un altre. En els mòduls següents explicarem que aquesta característica és una de les que criden més l'atenció i que s'ha de tenir molt en compte a l'hora de protegir la base de dades correctament.

5.4. DB2

DB2 Universal Database, creat per IBM, és considerada per algunes persones la primera base de dades que va utilitzar SQL. És una de les bases de dades més veteranes en la indústria.

Encara que aquesta base de dades té un percentatge de mercat important, fa l'efecte de tenir una presència més petita que la que mostren les estadístiques, potser perquè no està exposada habitualment a entorns més "externs" com Internet sinó a entorns més fortificats i interns de l'estructura de xarxa de la companyia. Malgrat la presència que té, és difícil trobar-la habitualment com passa amb altres bases de dades. Possiblement l'escenari més habitual

en què es troba aquesta base de dades és en conjunció amb altres productes d'IBM, sobretot en els famosos servidors AS/400, que van adquirir una gran popularitat a mitjan anys noranta en l'entorn empresarial i en què la base de dades d'IBM estava instal·lada per defecte.

IBM es distingeix de la resta del mercat perquè proporciona algunes característiques (almenys pel que fa a màrqueting) avançades que queden lluny d'altres motors de bases de dades, com ara gestor de dades (*data warehousing*) o mineria de dades (*data mining*). És possible que el motor de DB2 per a z/OS en arquitectures de 64 bits sigui el més potent per a aquest tipus d'operacions del mercat, amb un rendiment molt bo i reconegut tant pel mercat com per la competència. És, per tant, un motor de base de dades una mica "exclusiu" i amb unes característiques bastant concretes que potser l'allunyen d'un tipus de client més generalista. S'ha de considerar que les solucions d'IBM normalment combinen tant programari com maquinari, i constitueixen una solució integral a causa del model de negoci de l'empresa i que el poden allunyar d'un altre tipus de motors de bases de dades. Així i tot, IBM ha proporcionat versions per a plataformes generalistes en les últimes versions, i mostra així un canvi en l'estratègia empresarial.

Pel que fa a seguretat, DB2 té els mateixos problemes que qualsevol altra base de dades. No obstant això, sempre ha tingut una bona reputació perquè, d'una banda, proporciona una funcionalitat més reduïda en comparació d'altres bases de dades, cosa que implica una superfície d'atac més petita i minimitza les possibilitats d'atac, i de l'altra, IBM ha tingut una política molt eficaç pel que fa a solució de problemes enviats i ha proporcionat solucions de manera ràpida i eficaç. Tot plegat, sumat al poc impacte, sabut almenys, pel que fa a problemes de seguretat greus, ha donat una bona imatge de robustesa d'aquest motor de base de dades.

5.4.1. Història i evolució

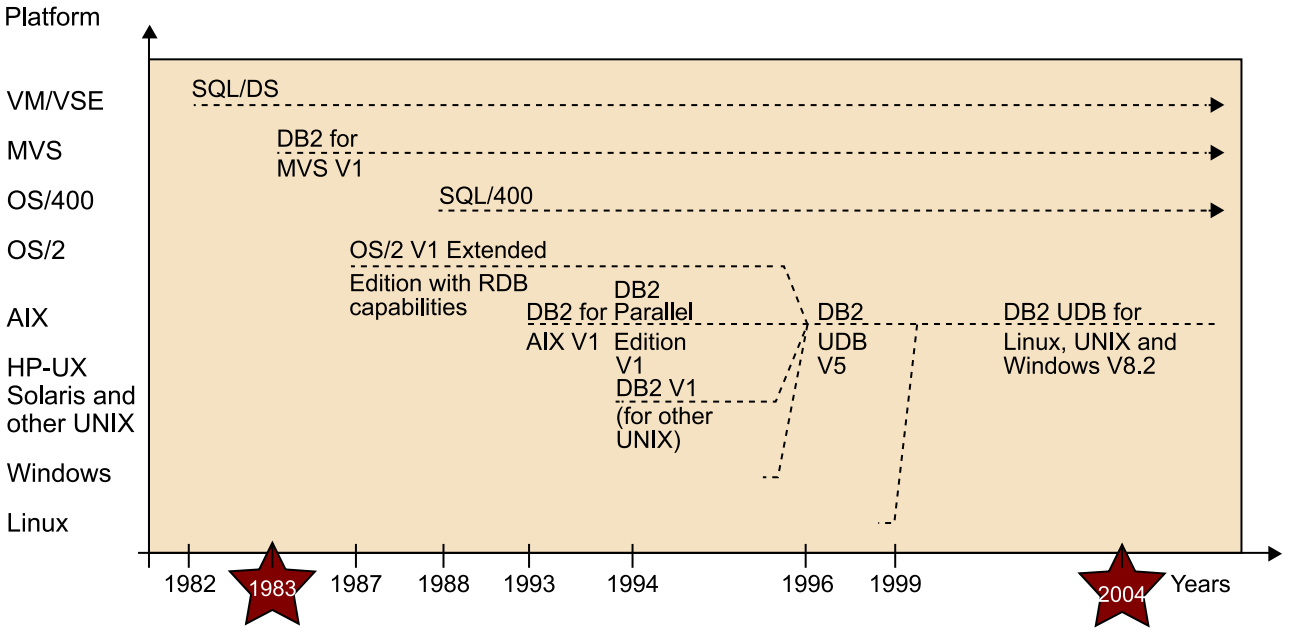
Encara que va ser el 1978 quan IBM va llançar el famós System R, no va ser fins al 1982 que va llançar al costat de la seva plataforma d'ordinador central (*mainframe*) els productes SQL/DS i DB2. Encara que DB2 és una evolució del System R, fins a l'any esmentat no es va crear com a producte propi amb el nom amb què ha arribat fins avui.

Inicialment, i seguint la filosofia d'IBM, la base de dades era disponible solament per als ordinadors centrals de la seva mateixa marca. Durant la dècada dels noranta es va produir un canvi en l'estratègia de la companyia, que va ampliar la disponibilitat de la base de dades per a altres plataformes, concretament per a servidors Windows i Unix, encara que més endavant es va llançar per a plataformes Linux. Durant aquest recorregut, l'alternança al capdavant de quota de mercat ha estat entre DB2 i Oracle de manera històrica.

DB2 és disponible en tres edicions amb diferents funcionalitats, conegudes com a Express, Workgroup i Enterprise (de menys a més). També permet llicenciar una versió reduïda en què es pot abaratir el cost eliminant serveis que no requereix el client, i també una versió avançada coneguda com a Data Warehouse Enterprise Edition. Hi ha una versió gratuïta coneguda com a DB2 9 Express C, que va sortir el 2006 com a resposta als productes que oferia la competència.

Les versions que hi ha en el mercat actualment oscil·len des de la versió 7 fins a la 9, que va sortir al mercat el 2006. La característica principal anunciada per IBM de l'última versió de DB2 és l'emmagatzematge natiu d'XML.

Figura 10. Línia cronològica de DB2



Es pot dividir el producte en tres branques principals segons les plataformes en què funcionen, cadascuna de les quals té un nucli comú, encara que amb algunes peculiaritats i característiques pròpies:

- DB2 per a z/OS.
- DB2 per a Windows, Unix i Linux (també conegut com a LUW).
- DB2 per a iSeries (AS/400).

El 2001 Oracle va adquirir Informix, i va incorporar diversa funcionalitat d'aquest motor de base de dades en DB2, sobretot la relacionada amb el concepte de base de dades orientada a objectes.

5.4.2. Arquitectura

Processos

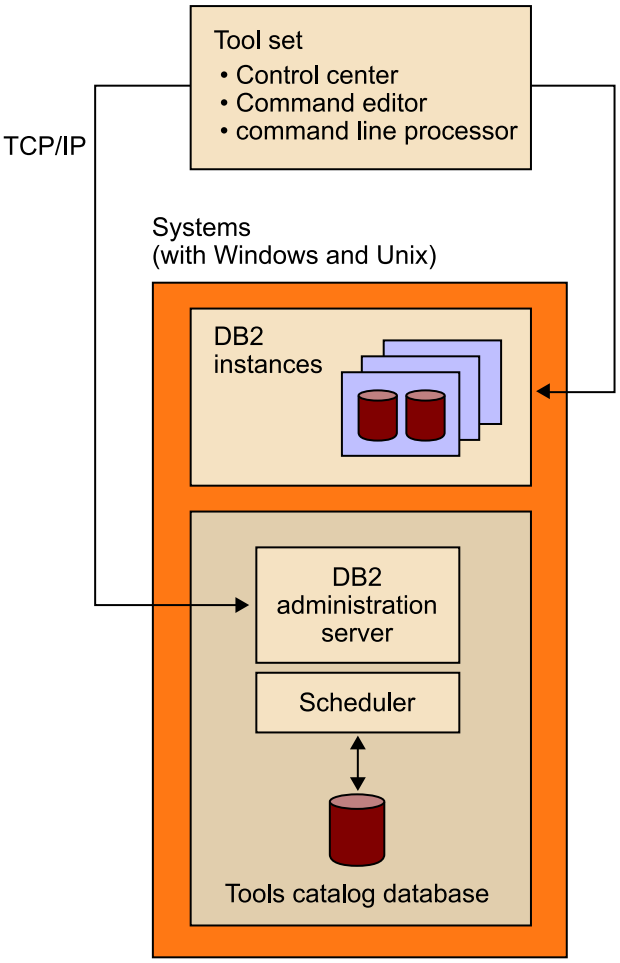
En cadascun dels equips en què s'instal·la un servidor de base de dades DB2 hi poden funcionar diverses instàncies, i també en cada instància hi pot haver diferents bases de dades. En una instal·lació per defecte, funcionen en el servidor dos processos que es corresponen amb dues instàncies del servidor, anomenats DB2 i DB2CTLSV. En cas d'instal·lar la base de dades d'exemple disponible en la instal·lació del producte, la trobarem en la instància anomenada DB2.

A part de les instàncies, hi ha el servei DB2 Database Administration Server (DAS), que s'encarrega d'administrar la base de dades.

El nom amb què funcionen els processos en Windows és DB2SYSCS per a les instàncies i les bases de dades. El nom d'aquest procés en Linux és DB2SYSC. El procés d'administració de la base de dades o DAS es diu DB2DASRRM en tots dos entorns.

Hi ha un altre procés anomenat DB2FMP, en què les rutines definides es carreguen en la base de dades. Es carreguen en un procés separat perquè, en cas d'haver-hi algun problema o en cas d'executar-se alguna rutina defectuosa, no caigui el procés principal de la base de dades.

Figura 11. Relació del DAS amb la resta de components



Cadascuna de les instàncies escolta per ports diferents; per defecte, DB2 escolta pel port 50000, i DB2CTLSV, pel port 50001. En entorns Windows es pot utilitzar *namedpipes* per a la comunicació. Aquests ports són configurables.

Pel que fa al DAS, el port pel qual escolta per defecte és el 523 (tant TCP com UDP).

Per a la comunicació entre el servidor i el client, DB2 usa DRDA (*distributed relational database architecture*), que és un protocol obert, encara que no ha tingut gaire difusió. Aquest protocol es transmet sobre TCP/IP i dins de cadascun d'aquests paquets hi ha empaquetat un paquet DSS (*datastream structures*) o més d'un, en què hi ha les ordres de base de dades juntament amb els paràmetres corresponents.

En l'autenticació, el protocol fa la transmissió de credencials en text pla (encara que pot canviar segons la configuració del servidor), si bé en cas de robar els paquets de xarxa sembla que això no és evident perquè no es transmeten en ASCII sinó que DB2 utilitza caràcters codificats mitjançant EBCDIC (*extended binary coded decimal interchange code*), codi propietat d'IBM. En vista d'aquest

escenari, es pot fer un mapatge entre aquests dos conjunts de caràcters i accedir a les dades d'autenticació en cas d'accedir al trànsit entre client i servidor durant el procés d'autenticació.

Fitxers

La instal·lació de la base de dades per defecte es fa en el directori *c:/Arxius de programa/IBM/SQLLIB* en Windows, en què hi ha els executables i també algun fitxer de registre. A partir d'aquest directori, es crea un subdirectori per a cadascuna de les instàncies que s'instal·len del servidor. A Linux s'instal·la per defecte a */opt/IBM/db2*.

Les bases de dades s'instal·len a l'arrel de la unitat en el cas de Windows, mitjançant la creació d'un directori per a cadascuna de les bases de dades (per exemple, *c:/DB2*). A partir de cadascun d'aquests subdirectoris, se'n crea un altre d'anomenat NODE0000 i, a sota, una sèrie de subdirectoris amb un rang de noms des de SQL00001 fins a SQL0000X, i també un altre d'anomenat SQLDBDIR. En aquesta estructura, DB2 crea la sèrie de fitxers que utilitza la base de dades i en els quals hi ha la informació que inclouen pròpiament aquestes dades. L'usuari amb què funciona a Windows la base de dades és *db2admin*.

Pel que fa a Linux, l'estructuració de la informació està molt lligada amb la creació d'una sèrie de comptes. Mitjançant la instal·lació per defecte es creen tres comptes:

- *dasusr1*: és l'encarregat d'executar el servei d'administració i és el que conté els fitxers executables que hi estan relacionats i també els fitxers de registre.

- *db2fenc1*: és l'encarregat d'executar el servei DB2FMP per a les rutines i les funcions que utilitza la base de dades.

- *db2inst1*: és l'encarregat d'executar el servei de la instància de la base de dades, i en el directori */home* hi ha el directori / *db2inst1*, que al seu torn conté els directoris *sqllib*, en què hi ha fitxers de configuració de la instància, i *db2inst1*, en què hi ha els fitxers que contenen els arxius amb les dades de la base de dades.

Estructura lògica

Els objectes de la base de dades es guarden en esquemes, cosa que inclou taules, vistes, procediments, etc. Els esquemes principals són aquests:

- SYSIBM: informació respecte a les taules.
- SYSCAT: informació respecte a les vistes.
- SYSFUN: informació respecte a les funcions.
- SYSPROC: informació respecte als procediments.

Autenticació

L'autenticació en DB2 es fa solament mitjançant el sistema operatiu. Això proporciona certs avantatges en comparació d'altres bases de dades, com ara blindatge respecte a configuracions incorrectes, com les relacionades amb les instal·lacions per defecte d'MS SQL Server de l'usuari *sa* sense cap contrasenya, o les relacionades amb la gran quantitat de comptes heretats a Oracle amb les contrasenyes corresponents i que en moltes instal·lacions els administradors no canvien. Així i tot, en versions antigues de DB2, els comptes que es creaven relacionats amb el motor de la base de dades usaven contrasenyes per defecte, cosa que comportava el mateix problema. Aquest comportament implica que no hi ha informació relacionada amb els usuaris en DB2 en la qual s'emmagatzemin les contrasenyes d'autenticació.

Es pot configurar el nivell d'encriptació entre el client i el servidor a l'hora de transmetre les dades d'autenticació, ja que com hem vist més amunt les dades es poden enviar sense autenticar en cas d'una configuració incorrecta.

Quan els usuaris estan autenticats, els seus permisos s'especifiquen mitjançant una sèrie de rols, igual que en altres motors de bases de dades, encara que en DB2 es coneixen com a *autoritats* (*authorities*). Les autoritats poden tenir un abast dels permisos respecte a la instància o de la base de dades. Aquesta informació es guarda en taules en la base de dades dins de l'esquema SYSCAT:

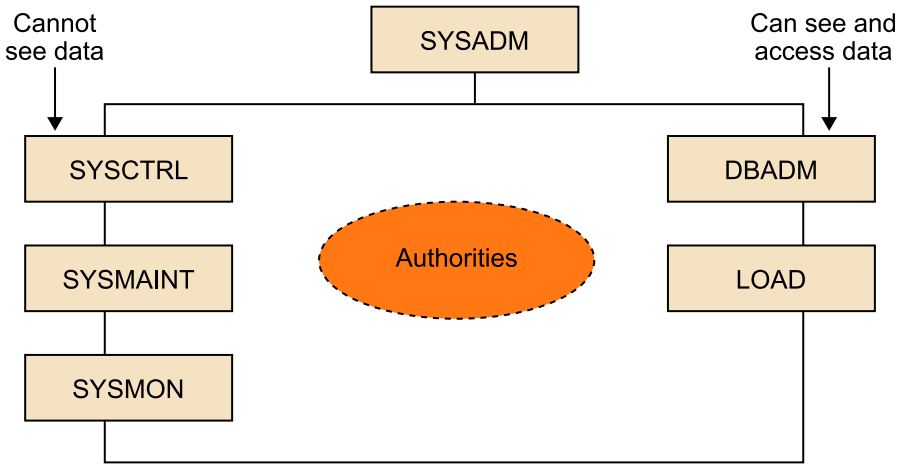
- DBAUTH: informació respecte a les autoritats de bases de dades i les accions permeses.

- TABAUTH: informació respecte a permisos sobre accions sobre taules de la base de dades.

- ROUTINEAUTH: informació respecte a qui té permisos d'execució de les rutines.

En DB2 el rol amb més permisos és SYSADM, i el grup a què pertany és SYSADM_GROUP.

Figura 12. Esquema de les autoritats principals



Taules destacables

Les taules més interessants que contenen informació respecte als objectes de la base de dades són les següents:

- SysIBM.tables: informació respecte a les taules de la base de dades.
- SysIBM.views: informació respecte a les vistes de la base de dades.
- SysIBM.columns: informació respecte a les columnes de les taules de la base de dades.
- SysIBM.usernames: informació respecte a connexions externes de la base de dades.

En DB2 la taula comodí és SYSIBM.Sysdummy1 (equivalent a la taula *dual* d'Oracle).

Bibliografia

http://attrition.org/dataloss/

http://math.hws.edu/vaughn/cpsc/343/2003/history.html

http://en.wikipedia.org/wiki/SQL_slammer_(computer_worm)

http://www.oracle.com/technology/deploy/security/alerts.htm