De application server in het middelpunt

John Zuidweg

Steeds meer organisaties maken gebruik van *application servers* om informatie uit databases te ontsluiten. Dit heeft grote consequenties voor de manier waarop de netwerkarchitectuur wordt ingericht. Wat betekent dit voor de IT-auditor?
In dit artikel wordt eerst ingegaan op het begrip application server en wordt daarna duidelijk gemaakt op welke manier de application server-architectuur tot stand gekomen is. Vervolgens wordt uitvoeriger ingegaan op een aantal centrale componenten, waarbij het concept *Identity Management* een belangrijke plaats inneemt. Vanuit deze beschrijving wordt ten slotte een aantal aandachtspunten voor de IT-auditor geïdentificeerd.

N.J. (John) Zuidweg heeft Information Security Technology gestudeerd aan de Technische Universiteit Eindhoven. In het kader van zijn afstuderen heeft hij bij EDP Audit Pool onderzoek gedaan naar de betrouwbaarheid van Oracle Application Server. Inmiddels is hij als EDP-auditor werkzaam bij Ernst & Young Advisory. Dit artikel is geschreven op persoonlijke titel. Het bevat een samenvatting van enkele aspecten uit het afstudeerverslag [ZUID08].

Het begrip 'application server' is niet in enkele woorden volledig te definiëren. In essentie is het een computerprogramma waarmee applicaties aangeboden kunnen worden via het netwerk. Deze applicaties zijn doorgaans van het type webpagina of webservice. Met behulp hiervan kunnen gebruikers of andere application servers toegang krijgen tot gegevens uit bijvoorbeeld een database. De application server fungeert dus als intermediair tussen een databron en gebruikers of services en is daarmee een typisch voorbeeld van *middleware*. In de praktijk omvat een application server echter meer functionaliteit dan alleen een computerprogramma voor het aanbieden van applicaties. In veel gevallen gaat achter dit begrip een complete architectuur van diverse componenten schuil. Deze architectuur biedt functionaliteit die voorheen verzorgd werd door het DBMS[1] of door programma's op de pc's van eindgebruikers. In het kader van een onderzoek naar de technische infrastructuur is deze architectuur daarom voor IT-auditors zonder meer het analyseren waard, als het gaat om de exclusiviteit, integriteit en beschikbaarheid van informatie. Bij een technische IT-audit is het mechanisme voor autorisatie en authenticatie van gebruikers een cruciaal aspect. Ook wat dit betreft kan de application server een belangrijke rol spelen: verschillende application servers bieden een vorm van Identity Management waarbij alle identiteitsgegevens van gebruikers centraal kunnen worden vastgelegd.

Van mainframe tot middleware

Vanaf de jaren tachtig van de vorige eeuw begonnen veel organisaties hun *mainframes* en *terminals* te vervangen door servers en pc's. In veel gevallen konden bedrijfsprocessen hiermee op een goede manier worden ondersteund. Toch had ook deze *client-server-architectuur* te kampen met beperkingen: binnen organisaties zag men de prestaties van de diverse applicaties afnemen naarmate meer *clients* aan de server werden gekoppeld. Ook was het voor systeembeheerders een forse klus om alle applicaties op de verschillende PC's te beheren en te zorgen voor een zorgvuldige toekenning van autorisaties.
Het is daarom niet verwonderlijk dat er veel belangstelling was voor de nieuwe *drielaagsarchitectuur* die in de jaren negentig van de vorige eeuw werd gepresenteerd. Deze architectuur beloofde namelijk de beperkingen van de client-server-architectuur op te kunnen lossen. Evenals in de client-

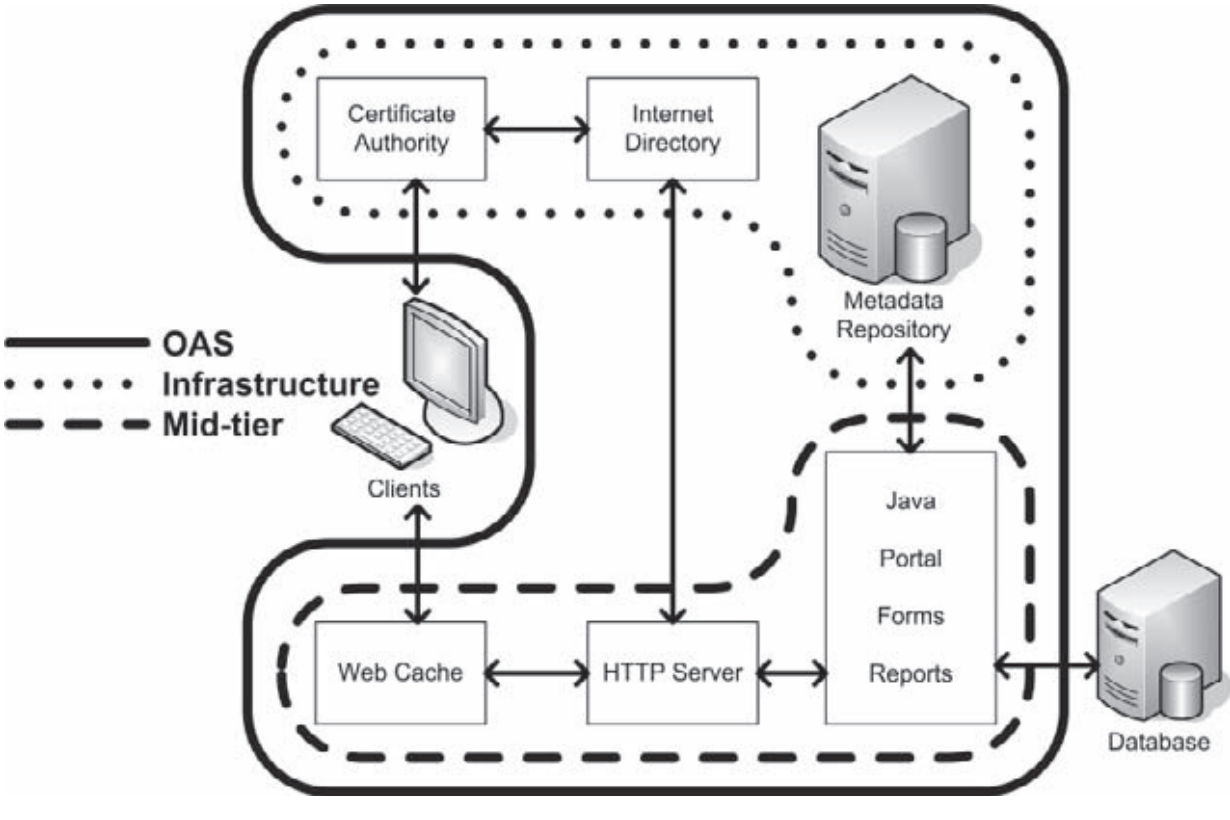
server-architectuur stonden de data op een centrale server en werden applicaties vanaf pc's aangestuurd. De applicaties met bedrijfslogica verschoven echter van de clients naar een nieuwe component: de application server. Informatie uit de database werd op deze manier ontsloten via deze tussenliggende component. Kenmerkend voor de manier waarop dit gebeurde, is het gebruik van *webtechnologie*. Application servers zijn namelijk bij uitstek geschikt voor het aanbieden van webapplicaties: *webpagina's* die door eindgebruikers benaderd kunnen worden via een *browser* op hun pc en *webservices* die met andere applicaties kunnen communiceren door middel van XML[2]-berichten (zie [LEAN00]).

Component of architectuur

Het product application server is inmiddels door verschillende softwarefabrikanten uitgewerkt en op de markt gebracht. Bekende leveranciers zijn Adobe (met ColdFusion), Red Hat (met JBoss), Oracle (met Oracle Application Server) en IBM (met WebSphere Application Server). Tussen de diverse application servers zijn verschillen te ontdekken wanneer de implementatie in ogenschouw genomen wordt. Over het gehele spectrum genomen, valt echter wel te concluderen dat bijna alle application servers meer functionaliteit bevatten dan alleen het aanbieden van applicaties: de term application server is in veel gevallen een soort paraplubegrip geworden waaronder een complete architectuur van middleware schuilgaat.

Een duidelijk voorbeeld hiervan is te vinden bij de application server van Oracle (hierna: OAS, zie ook [GREE04]). Bij nadere beschouwing blijkt namelijk dat de application server in dit geval uit minstens zes subcomponenten bestaat (zie figuur 1). Verder valt op dat deze subcomponenten in twee categorieën onderverdeeld kunnen worden. De *mid-tier* bevat de kernfunctionaliteit van de application server, terwijl de *infrastructure* een meer ondersteunende rol vervult.

De *web cache* fungeert als *proxy*: alle binnenkomende verzoeken komen binnen via deze component. Statische (gedeeltes van) webpagina's worden direct door de web cache aangeboden aan gebruikers, zodat de *HTTP-server* minder wordt belast. Aangezien de application server met gebruikers en applicaties communiceert via webservices en webpagina's, nemen deze componenten een centrale plaats in binnen de application server. Communicatie met deze webapplicaties verloopt namelijk doorgaans via het *hyper text transfer protocol* (HTTP) of, met gebruikmaking van certificaten, via de beveiligde versie van dit protocol (HTTPS). Overigens is de HTTP-server binnen OAS een aangepaste versie van de *open source Apache webserver*. Bij het auditen van de web cache en de HTTP-server is vooral de beoordeling van *patch*-management van groot belang. Omdat deze componenten zich aan de 'voorkant' van de architectuur bevinden, worden ze door een eventuele aanvaller namelijk als eerste benaderd. Via internet zijn diverse bekende beveiligingslekken voor deze producten eenvoudig te vinden. Wanneer niet accuraat



Figuur 1 Conceptweergave architectuur Oracle Application Server

wordt gepatched, is hierdoor voor een aanvaller de eerste barrière al snel geslecht.

Bij een application server gaat het uiteindelijk om de applicaties die aangeboden worden. Binnen OAS kunnen vier soorten applicaties worden onderscheiden: *Forms*, *Reports, Portal en Java*. Forms en Reports zijn typische Oracle-producten uit het *client-server*-tijdperk. Oracle Forms is een soort omgeving waarmee binnen een bepaalde opzet applicaties gemaakt kunnen worden, waarmee informatie uit de database kan worden opgevraagd en gewijzigd.
Oracle Reports is geschikt voor het (periodiek) opbouwen van rapportages over gegevens uit de database. Oracle heeft deze producten geconverteerd naar een drielaagsarchitectuur zodat Forms- en Reports-applicaties uitgevoerd kunnen worden op de application server, waardoor gebruikers deze applicaties kunnen benaderen via een webbrowser. Oracle Portal is ook een standaardomgeving van Oracle, alleen is deze wel vanaf het begin specifiek toegesneden op een drielaagsarchitectuur. Oracle Portal is een verzameling van webpagina's waarmee verschillende databronnen (bijvoorbeeld databases of andere webpagina's) via één centrale plek ontsloten kunnen worden. Ook het inrichten en beheren van dit portal gebeurt 'gewoon' via een webbrowser. Naast al deze standaard Oracle-applicaties, bestaat binnen OAS ook de mogelijkheid om 'maatwerksoftware' aan te bieden. De programmeertaal waarin deze software geschreven dient te worden is Java. Met behulp van deze programmeertaal kunnen stukjes code uitgevoerd worden in HTML-pagina's. Ook is het mogelijk om een applicatie te schrijven die een webpagina, een webservice of zelfs een complete verzameling van webpagina's of webservices genereert. Deze applicaties kunnen communiceren met de database, zodat het mogelijk is via een webpagina of webservice informatie uit de database op te vragen of te modificeren (zie ook kader Java EE).

De veiligheid van deze applicaties is voor IT-auditors een belangrijk aspect: de applicaties zijn gemakkelijk via een webbrowser te benaderen en geven direct toegang tot de database. Zelfs indien de architectuur goed beveiligd is, kan het complete systeem onderuit gehaald worden als applicaties lekken bevatten of niet op een goede manier zijn ingericht. Een succesvolle c*ross site scripting* (XSS)[3] aanval kan bijvoorbeeld leiden tot een 'gekaapte' SSO-sessie (zie kader *Single sign-on*), waarmee een ongeautoriseerde kwaadwillende gebruiker aan de slag kan met alle applicaties die gebruikmaken van dit mechanisme. Een onzorgvuldig geprogrammeerde of geconfigureerde applicatie kan dan ook grote gevolgen hebben. Het uitvoeren van beveiligingstesten op deze applicaties is daarom van groot belang. De enige echte oplossing is *security by design*: applicatieveiligheid zal al tijdens het ontwerp- en implementatieproject een belangrijke plaats moeten krijgen. Alleen achteraf testen op beveiligingsproblemen vergroot het risico dat programmeerfouten over het hoofd gezien worden, die wellicht in een later stadium door een kwaadwillende kunnen worden misbruikt om toegang te krijgen tot een functionaliteit waarvoor deze persoon niet is geautoriseerd.

Java EE

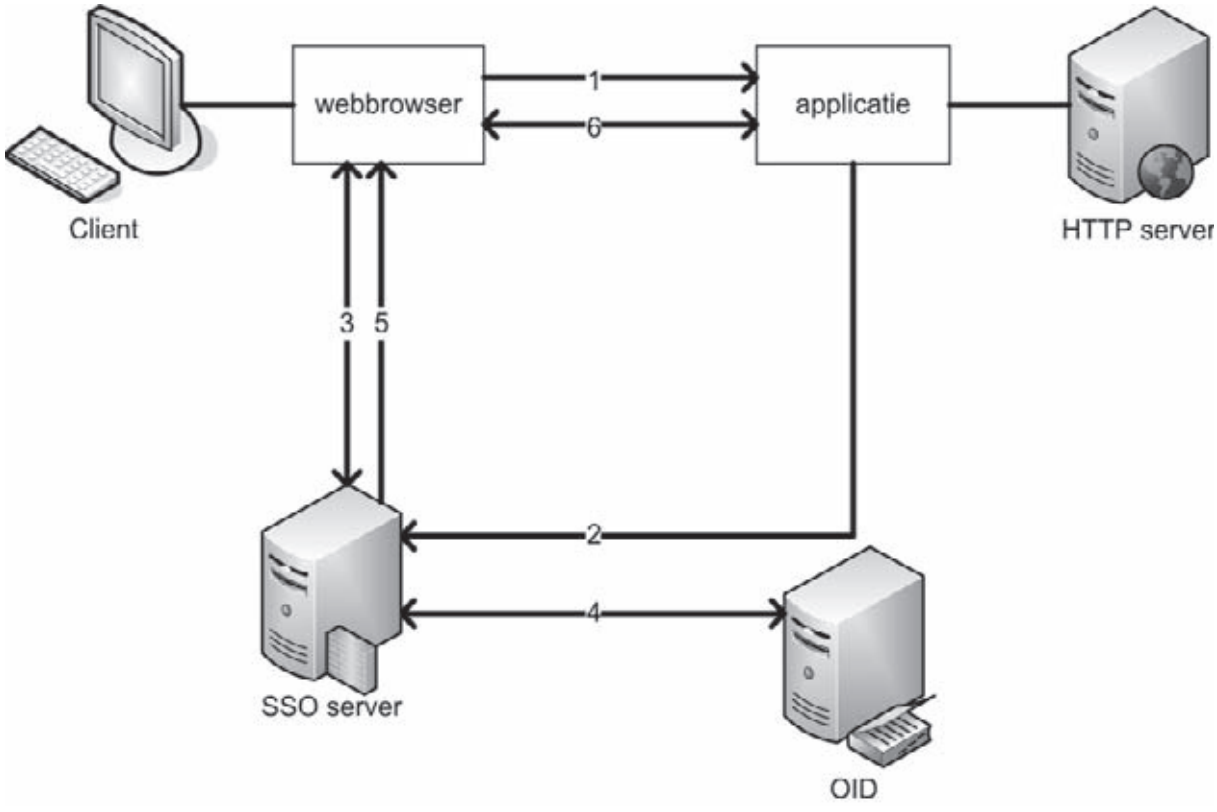
Java Enterprise Edition (Java EE, voorheen J2EE) is een versie van de objectgeoriënteerde programmeertaal Java. Java EE is speciaal gericht op de ontwikkeling van *server-side* software zoals webapplicaties. De meegeleverde bibliotheken bevatten een verzameling van functionaliteiten die gebruikt kan worden in deze applicaties. De bibliotheek die ontwikkeling van *Enterprise Java Beans* (EJB's) mogelijk maakt, speelt een belangrijke rol binnen de context van de application server als intermediair tussen het DBMS en de gebruiker. Met behulp van deze EJB's kunnen gegevens uit de database op een voor de programmeur eenvoudige wijze vertaald worden naar zogenaamde Java-beans. Een Java-bean is een virtueel object dat een *record* in de database representeert. Elk veld uit het database record wordt vertaald naar een attribuut van dit object, zodat de waarden van het record corresponderen met de inhoud van de attributen. Door deze Java-beans op een efficiënte manier te synchroniseren met de database kan de systeembelasting van het DBMS in veel gevallen significant worden verlaagd. In vergelijking met 'klassieke' Java-programma's maken deze EJB's het werk voor softwareontwikkelaars gemakkelijker, aangezien zij zich niet langer hoeven te bekommeren over de implementatie van de communicatie met de database. Indien EJB's gebruikt worden, behoeven applicaties in principe niet langer SQL-code te bevatten voor de communicatie met het DBMS. Dit kan zowel de veiligheid als de interoperabiliteit ten goede komen. (Zie bijvoorbeeld [GREE04] voor meer informatie.)

De *Certificate Authority* (zie [MISR05]) is verantwoordelijk voor het uitgeven en intrekken van certificaten van gebruikers en componenten. Elk certificaat bevat een publieke sleutel die toebehoort aan de eigenaar van het certificaat. Deze sleutels kunnen gebruikt worden voor het verifiëren van de identiteit van gebruikers en componenten; tevens bieden deze sleutels de mogelijkheid om gegevens voor veiligheidsdoeleinden te versleutelen voordat deze worden verzonden.[4] Een publieke sleutel kan ook weer gebruikt worden voor het certificeren van een andere publieke sleutel, zodat een hiërarchie van certificaten kan ontstaan. Het gaat hier dus om beveiliging van gegevens (exclusiviteit) en om authenticatie van gebruikers en componenten. Voor IT-auditors is het daarom van belang vast te kunnen stellen dat de Certificate Authority zorgvuldig is ingericht en goed wordt beheerd. Belangrijk is hierbij dat een certificaat uitsluitend wordt toegekend aan een geautoriseerde entiteit, zodat uit het certificaat kan worden opgemaakt op welke entiteit het betrekking heeft. Ook is van belang dat certificaten ingetrokken worden als de onderliggende sleutel of een bovenliggend certificaat is gecompromitteerd.
In de *Metadata Repository* is een groot deel van alle configuratiegegevens van de componenten uit de mid-tier opgeslagen. Het centraal vastleggen van deze gegevens maakt de

application server schaalbaar, aangezien hierdoor op relatief eenvoudige wijze meerdere mid-tiers te koppelen zijn aan één *infrastructure*. Voor IT-auditors is de integriteit van deze configuratiedata een belangrijk aspect, omdat dit weerslag heeft op alle componenten uit de mid-tier. Hierbij is het van belang dat alleen application servers toegang hebben tot de *repository*, zodat manuele modificatie van gegevens uitgesloten is.

Oracle *Internet Directory* (zie ook [DESM05]) bevat gegevens die betrekking hebben op identiteiten en autorisaties

Single sign-on



Figuur 2 Oracle Application Server single sign-on proces voor interne applicaties

Door gebruik te maken van *single sign-on* (SSO) kan voorkomen worden dat een gebruiker voor meerdere applicaties apart moet inloggen (met diverse gebruikersnamen en wachtwoorden). Bij het benaderen van een applicatie binnen OAS werkt dit mechanisme als volgt (zie ook [WISH06]):

1 Een gebruiker probeert met een webbrowser een applicatie binnen OAS te benaderen. De applicatie 'controleert' of een geldig applicatie-*cookie*[5] aanwezig is in de browser van de gebruiker. De gebruiker krijgt direct toegang tot de applicatie indien een geldig applicatie-cookie aanwezig is.

2 Als geen geldig applicatie-cookie aanwezig is, wordt de gebruiker doorgestuurd naar de SSO-server die bepaalt of de gebruiker reeds is ingelogd door te controleren of een geldig SSO-cookie aanwezig is in de browser van de gebruiker.

3 Indien dit niet het geval is, wordt de gebruiker gevraagd een geldige gebruikersnaam en wachtwoord in te voeren in de webbrowser. (*Oracle Internet Directory* (OID) wordt geraadpleegd voor het controleren van deze gegevens.) Wanneer de gebruiker is ingelogd wordt een SSO-cookie opgeslagen in de webbrowser van de client.

4 Wanneer een geldig SSO-cookie aanwezig is, controleert de SSO-server via OID de autorisaties van de gebruiker op de applicatie.

5 De gebruiker krijgt toegang tot de applicatie indien deze hiervoor geautoriseerd blijkt te zijn. De toegang wordt verkregen doordat de webbrowser van de gebruiker wordt doorverwezen naar een speciale URL die naast de locatie van de applicatie ook een versleuteld *authenticatie-token* bevat, zodat de applicatie kan vaststellen dat de gebruiker hiertoe geautoriseerd is.

6 Zodra de gebruiker toegang verkregen heeft tot de applicatie wordt een applicatie-cookie weggeschreven in de browser van de gebruiker, zodat de gebruiker gedurende de actieve sessie de applicatie kan benaderen, zonder dat hiervoor de SSO-server geraadpleegd hoeft te worden.

SSO kan binnen OAS ook gebruikt worden voor het benaderen van externe bronnen zoals websites. Het mechanisme werkt dan als een systeem dat de gebruiker automatisch 'inlogt' bij de externe bron nadat de gebruiker zich succesvol heeft geauthentiseerd bij de SSO-server.

van gebruikers. Door alle autorisatiegegevens in deze LDAP-directory te centraliseren, kunnen identiteiten en autorisaties van gebruikers gemakkelijker beheerd en gecontroleerd worden. Alle identiteiten, rollen en permissies die eerst én deels in de database én deels in diverse applicaties waren vastgelegd kunnen nu op één plaats worden opgevraagd en gewijzigd. Zeker voor grote organisaties die te maken hebben met veel medewerkers die in dienst treden, van functie veranderen of de organisatie verlaten kan een dergelijke vorm van identity management veel toegevoegde waarde hebben. Voor beheerders is het weliswaar een grote uitdaging om alle identiteiten, rollen en permissies uit databases en applicaties op een juiste manier te centraliseren, maar als dit eenmaal gerealiseerd is, kan dit veel tijdsbesparing opleveren. Ook voor IT-auditors kan dit voordelig zijn, omdat ze zich in dit geval bij het onderzoek naar autorisaties in principe kunnen beperken tot de internet directory. Ook voor gebruikers kan een dergelijke opzet voordelen met zich meebrengen: door gebruik te maken van single sign-on (zie kader) hoeft een gebruiker slechts één enkele keer in te loggen, waarna authenticatie automatisch plaatsvindt voor alle applicaties waarvoor de betreffende gebruiker geautoriseerd is. Dit zou zelfs voordelen kunnen hebben voor de veiligheid van de IT-omgeving, aangezien gebruikers minder wachtwoorden behoeven te onthouden, waardoor men minder snel geneigd zal zijn tot het noteren van wachtwoorden op 'spiekbriefjes' aan bijvoorbeeld de onderkant van het toetsenbord. Voor IT-auditors was informatie over authenticatie en identificatie van gebruikers altijd al een belangrijk aspect. Het risico bestaat wel dat, indien het wachtwoord van een gebruiker bekend is bij een onbevoegde, hij direct toegang heeft tot alle applicaties van die gebruiker.

Het is mogelijk alle componenten van OAS op één en dezelfde computer te installeren. In productieomgevingen worden de componenten doorgaans verdeeld over verschillende fysieke componenten, om zo de benodigde *performance* en schaalbaarheid te kunnen bieden. Door deze hardware met elkaar te verbinden via een standaard TCP/IP-netwerk kunnen de diverse componenten samenwerken, zodat de application server kan functioneren. Indien kritieke componenten redundant zijn uitgevoerd, kunnen calamiteiten worden opgevangen. Hiermee hangt de beschikbaarheid van de application server in belangrijke mate af van de manier waarop de diverse componenten zijn verdeeld over de gebruikte hardware. Tevens speelt het (interne) netwerk een grote rol.
Oracle heeft de naam application server aangepast, omdat deze vlag de lading niet langer dekt. Daarom is nu de naam 'Oracle Fusion Middleware' in het leven geroepen. Onder deze naam worden nu diverse pakketten (zoals 'Collaboration Suite' en 'Business Intelligence') aangeboden waar een application server deel van uitmaakt.

Aandachtspunten voor de IT-auditor

In het voorgaande werd duidelijk dat een application-architectuur een verzameling is van diverse samenwerkende componenten. Per component is reeds kort aangegeven welke aspecten bij een IT-audit van belang zijn. Bovendien is het uitschakelen van standaardaccounts (of in elk geval het wijzigen van de wachtwoorden ervan) een essentiële stap bij het beveiligen van deze architectuur. Hierbij is ook het afschermen van allerlei (webgebaseerde) beheer*interfaces* en het verwijderen van meegeleverde demoapplicaties van groot belang. De diverse componenten communiceren intensief met elkaar via het netwerk. Aangezien de beveiliging van de individuele componenten nooit geheel gegarandeerd kan worden, is het raadzaam het netwerk zodanig te segmenteren dat alleen die componenten met elkaar kunnen communiceren waarvoor geldt dat een verbinding noodzakelijk is. Feitelijk gaat het hier om *security in depth*: door het *hardenen* van de diverse componenten en de onderliggende systemen, het implementeren van een goed patchbeleid en het segmenteren van het netwerk, kan de architectuur als geheel veiliger worden gemaakt. Aan de IT-auditor de taak zekerheid te verschaffen over de kwaliteit van deze maatregelen. Controlewerkprogramma's en diverse applicaties (zie [ZUID08]) kunnen hierbij goede hulpmiddelen zijn. Een gefundeerd oordeel over de exclusiviteit, integriteit en beschikbaarheid van de beschikbaar gestelde informatie kan echter niet gegeven worden, voordat de complete architectuur is geanalyseerd in de context van de organisatie.
Daarnaast bestaat de mogelijkheid dat databasebeheerders (DBA's) door de aanschaf van een application server te maken kunnen krijgen met een forse uitbreiding van hun functie en verantwoordelijkheden. Naast de database moet de beheerder bijvoorbeeld ook webservers, webapplicaties en een *Certificate Authority* beheren. In sommige gevallen kan dit zelfs leiden tot schending van het beginsel van functiescheiding. Afhankelijk van het type applicaties kan het daarom noodzakelijk zijn dat, naast de DBA, een *applicatiebeheerder* wordt aangesteld, die verantwoordelijk is voor het technische beheer van de applicaties. Hiervan kan bijvoorbeeld sprake zijn wanneer een DBA zowel het technische beheer uitvoert van financiële programma's als van de onderliggende database.
Ten slotte is het van belang zich te realiseren dat er een verschuiving optreedt in de plaats waar de autorisaties van gebruikers zijn opgeslagen. Waar dit vroeger vastlag in de database en in sommige applicaties, wordt dit in de application server-architectuur in principe in een LDAP-directory vastgelegd, waarbij er slechts één of hooguit enkele database accounts worden gebruikt. In de praktijk zijn echter allerlei mengvormen mogelijk, zodat de autorisaties alsnog verspreid over de systemen kunnen liggen waardoor er niet eenvoudig uitspraken gedaan kunnen worden over de inrichting van deze autorisaties.

Conclusie

Implementatie van een application server kan leiden tot een veiligere en robuustere architectuur die schaalbaar is. Voor zowel beheerders als IT-auditors introduceert deze architec-

tuur echter ook een aantal extra aandachtspunten. Hierbij zijn hardening van de complete keten, applicatieveiligheid, een doordachte functie-indeling en een duidelijk vastgelegde autorisatiestructuur van groot belang. Dit alles kan niet met een standaard controlewerkprogramma worden afgedicht: een grondige analyse van het complete systeem in de context van de organisatie is vereist, alvorens conclusies getrokken kunnen worden over de betrouwbaarheid van het systeem. ■

Literatuur

[BLEI98] Bleichenbacher, D., *Chosen Ciphertext Attacks Against Protocols Based on the RSA Encryption Standard PKCS #1*, Lecture Notes in Computer Science 1462 (1998).

[DESM05] Desmond, E., *Oracle Identity Management Concepts and Deployment Planning Guide*, Oracle Corporation, 2005.

[GREE04] Greenwald, R., R. Stackowiak en D. Bales, *Oracle Application Server 10g Essentials*, O'Reilly, 2004.

[LEAN00] Leander, R., *Building Application Servers*, Cambridge University Press, 2000.

[MISR05] Misra, V.H., *Oracle Application Server Certificate Authority*, Oracle Corporation, 2005.

[WISH06] Wishbow, N., *Oracle Application Server Single Sign-On*, Oracle Corporation, 2006.

[ZUID08] Zuidweg, N.J., *IT Auditing of Oracle Application Server*, Technische Universiteit Eindhoven, 2008, http://tinyurl.com/zuidweg2008-pdf.

Noten

1 DBMS staat voor Database Management System, dit is de software die het gebruik van een digitale database mogelijk maakt.
2 XML staat voor eXtensible Markup Language.
3 XSS is een techniek waarbij een aanvaller een speciale programmacode laat uitvoeren in de browser van een gebruiker wanneer deze een webpagina benadert. Met behulp van deze techniek zou bijvoorbeeld de inhoud van een sessie *cookie* 'gestolen' kunnen worden.
4 AI is het aan te raden om verschillende publieke sleutels te gebruiken voor authenticatie enerzijds en versleuteling anderzijds, zie [BLEI98].
5 Een cookie is een klein tekstbestandje dat opgeslagen kan worden in een webbrowser. Cookies bevatten vaak gegevens die ervoor dienen de gebruiker te identificeren.