SECURING THREE-TIER SYSTEMS WITH ORACLE8*I*

John H. Heimann, Oracle

BACKGROUND

This paper focuses on security issues associated with building a three-tier system for access to an Oracle database. It discusses the problems which companies building three-tier systems face, and the solutions which Oracle offers through Oracle8*i*.

WHY THREE-TIER?

Three-tier architectures have been in existence for many years; for example, transaction processing (TP) monitors which allows dumb terminals or PCs to access data on a mainframe are very common. The primary reasons for deploying a three-tier system are: efficient resource management, improved scalability, and security. In a three-tier system, the middle tier can act as a concentrator, allowing many user devices to share a relatively few connections to the back-end system. Moreover, in a three-tier system the middle tier can focus on presentation of data to the user, allowing the back end tier to focus on management and processing of data. Since databases are optimized for efficient data management, moving responsibility for data presentation and connection management to the middle tier can improve system efficiency and scalability. Moreover, application logic in the middle tier can limit access of users, and provide another layer of isolation to sensitive data maintained in database. This improves system security.

The Internet has greatly increased the need for three-tier systems. Increasing numbers of companies are allowing users to access their enterprise data processing systems through the Internet. They typically do so by deploying web servers, which act as front ends to their enterprise, and mediate access to enterprise systems. Using web servers in the middle tier allows use of thin clients (i.e., web browsers), which are easy to use and can greatly reduce overall system cost for systems which support large user communities. Web-enabled access to companies 'enterprise systems is the fastest growing type of three-tier deployment.

SECURITY CHALLENGES

Although security is one of the reasons for moving to a three-tier architecture for enterprise access, there are many practical security challenges which arise in designing a three-tier-system. These include assuring user authentication, controlling user access, auditing user actions, protecting data security between tiers, limiting privilege of the middle tier, managing identities across tiers, and building scalable systems.

AUTHENTICATING THE USER

Three-tier architectures increase the complexity of user authentication and access control. In a two-tier client-server architecture, where client users connect directly to the server, the database can authenticate users at connect time. The database can associate any data, query, or transaction which is sent over the user's connection with that user, can grant or deny the user access to sensitive database resources, and audit user actions accordingly.

In a typical three-tier architecture, the middle tier is responsible for authenticating users 'identities. Moreover, data which a user sends to the middle tier is processed by the middle tier before it is submitted to the back end database as a query, update, or similar transaction, and may bear little resemblance to the data which the user originally sent to the middle tier. In addition, multiple client

to middle-tier connections may be multiplexed over a single middle-tier to database connection. For these reasons, the database must delegate some responsibility for authenticating users to the middle tier. The database must also rely on the middle tier to properly associate users 'identities with data sent to the database on behalf of those users. Clearly, designing a secure authentication strategy in a three-tier system is much more complicated than in a two-tier system.

CONTROLLING USER ACCESS

Once a user's identity is authenticated to a three-tier system, the system must control what data, applications, and resources the user may access within the system. This is particularly true for Internet-enabled systems. Giving customers and business partners access to mission-critical systems over the Internet may yield reduced cost, better service, and more timely information to those who need it, but increases the need for effective access control within the system. Organizations must not only keep data safe from intruders, but also must segregate data from legitimate users appropriately, often to the level of individual customers or users.

Prior to Oracle8*i*, access control within the database was enforced by determining whether a specific user could have access to a specific table, but not to specific data or rows within the table. When organizations needed to enforce finer-grained access control for data within tables, they had to rely on database views, or program the access logic into stored procedures or applications. There are several limitations with these approaches. Among them is that users who can gain direct SQL access to tables can bypass security constraints imposed by applications, stored procedures or views. Another is that programming access logic into views or applications is often cumbersome and difficult to manage. Creating a separate view for each user is clearly impractical for more than a small community of users. If access logic is programmed into applications, then these applications must be rewritten if security policies change.

In addition to controlling what users may access, the system must also enforce how they access it. A typical security requirement in a three-tier system is that each user be limited to running specific applications on the middle tier, depending on the user's identity or role within the organization. For example, in a business e-commerce system, customers may be allowed access to ordering or account information, whereas partners would have access to fulfillment applications, and employees have access to system internal functions. It is also usually the case that users accessing an enterprise through a middle tier application should not be able to gain direct access to the back-end database, nor issue arbitrary SQL commands.

PROTECTING USER DATA

Data exchanged between tiers must be protected against unintended disclosure or modification. Encryption is the standard mechanism for this purpose. Middle tier servers must implement an encryption protocol which is supported by the clients which communicate with the middle tier; in the case of web browser clients this protocol is SSL. Middle tiers must also support an encryption protocol to communicate with the database. In addition to protecting data in the network when exchanged between tiers, encryption protocols such as SSL can provide cryptographic user authentication.

Note that in a three-tier architecture it is usually not desirable to encrypt data end-to-end between client and database. End-to-end encryption implies that data is not decrypted in the middle tier, which prevents the middle tier from doing significant processing on the data. This would eliminate much of the value of a three-tier architecture. If data is in fact decrypted in the middle tier, however, then client encryption only authenticates the client to the middle tier, and not the database.

TRACKING USER ACTION

Accountability through auditing is a basic principle of information security. Auditing is complementary to access control, and helps ensure that privileged users do not abuse their access privileges. Three-tier systems increase the complexity of tracking and correlating user activities which may be security sensitive, and thus make security auditing more difficult. Even deciding what events to audit can be a challenge in a three-tier system.

In a three-tier system, users do not normally have direct access to the database. Instead, a user performs some action at the middle tier, and that action may cause the middle tier to request that the database take some corresponding action on behalf of the user. Unless the database records the identity of the user on behalf of whom the middle tier performed some action at the database, it may be difficult for a system auditor to correlate related actions performed at the middle and database tiers, and determine which user was responsible.

LIMITING PRIVILEGE OF MIDDLE TIER

Another challenge in designing a secure three-tier system is for the database to delegate an appropriate degree of trust to the middle tier. In some three-tier architectures, the database allows the middle tier to connect as any user, and assume any role or privilege which that user would have if the user were connected directly to the database. This approach may be acceptible in some situations, such as when there is little or no sensitive data in the database, or where users do not differ greatly in privilege. In most cases, however, it is desirable for the database to be able to limit the privilege of the middle tier, so that the database can restrict whether a specific middle tier can act on behalf of a specific user, and can assume specific user privileges. Middle tier servers which do not implement strong authentication mechanisms, or which are relatively vulnerable to attack (e.g., because they are outside the corporate firewall and are exposed to hacking by users coming in through the internet), could thus be granted less privilege than middle tier servers which implemented strong authentication or were within the corporate firewall.

MANAGING IDENTITIES AND ACCOUNTS ACROSS TIERS

A significant problem when building three-tier systems is associating middle tier user identities with database user identities. It is often the case that the way in which a user's identity is represented will differ in different components of a three-tier system. E.g., users accessing a system through a web server may authenticate themselves by exchanging X.509 certificates with the web server through the Secure Sockets Layer (SSL) protocol. X.509 certificates contain the user's identity in X.500 Distinguished Name (DN) format; e.g.:

C="US", O="Oracle", OU="Server Technologies", CN="Mary Ann Smith".

This format looks nothing like a typical Oracle database username, which might be *MASMITH*. Translating from middle tier user identity to database user identity may present a significant challenge to the design of the system security architecture.

A related problem is the management of user identities and accounts in a three-tier system. When a new user is added the system, identities, accounts, and privileges for that user must be created at both the middle tier and database. If X.509 certificates are used to authenticate the client, then each client user must request and install an X.509 certificate from a certificate authority (CA) recognized by the middle tier. If identities are represented differently in each tier, then they should be correlated in some way, so that (for example) the middle tier can correctly associate a client user's X.509 identity with the identity associated with that user's account on the middle tier. The middle tier must also be able to translate the user's identity to the appropriate database identity when requesting that some action be performed at the database on the user's behalf.

Scaling to Internet User Communities

Although it is desirable, for security reasons, to be able to track and control the activities of each user at both middle tier and database, it is undesirable for each user to have a separate account and user schema on the database. It may be practical to create and manage many hundreds or even thousands of user accounts on a database, but there are customers who wish to build three-tier, Internet-enabled applications which support millions of users. It is not practical to create and manage this many traditional user accounts on a database. Moreover, in many applications, each user's information consists of a single row within a table, and creating a separate user account and schema for each of these users is excessive overheard. To support Internet-scale user communities, the database must enforce some means for managing user access, and tracking user activity, which does not require each user to have an account and schema on the database.

Oracle's Three-Tier Strategy

Although three-tier architectures have a number of benefits, there are a number of security challenges which may arise in the design of a three-tier system. Since no two system designs are identical, the security issues which arise in the design of any specific system may differ. Oracle offers a number of security features in Oracle8*i* which are designed to address the security requirements of three-tier systems.

Security Features of Oracle8*i*

OCI Lightweight User Session

Perhaps the most useful security feature in Oracle8*i* for supporting three-tier systems is the Oracle Call Interface (OCI) lightweight user session. OCI is Oracle's C-language client API for accessing an Oracle database. The OCI lightweight user session feature was initially released in Oracle8, and allowed a database client to set up, within a single database connection, a number of "lightweight" user sessions, each of which is associated with a different database user. This feature has been enhanced in Oracle8*i* specifically to address the security requirements of three-tier systems.

With Oracle8*i*, a middle tier server can access the database through OCI on behalf of a client user by establishing a lightweight session for that user. The feature is designed so that a specific middle tier can be restricted to acting on behalf of a specified set of users. Once the middle tier has authenticated itself to the database, it can establish a lightweight session on behalf of those users without submitting user-specific authentication information such as passwords. Moreover, Oracle8*i* can be configured so that a specific middle tier can assume a specific set of database roles when acting at the database on behalf of a specific user. In other words, the database uses both middle tier identity and client user identity when determining what privileges to grant a middle tier acting for a user through a lightweight session. Middle tiers are granted the ability to connect as a particular user by altering the user's definition as follows:

```
ALTER USER john GRANT CONNECT THROUGH appsrvr;
```

After successfully authenticating itself to the database, user appsrvr (an application server) is able to create a lightweight user session on behalf of John.

Oracle8*i*'s OCI lightweight user feature addresses a number of security problems associated with three-tier systems. Since each middle tier can be delegated ability to authenticate and act on behalf of a specific set of users, and with a specific set of roles, OCI lightweight user supports a limited trust model for the middle tier server, and avoids the problem of an all-privileged middle tier. It is possible to give more privilege to a trusted middle tier (e.g., one that is within the corporate firewall) than to a less-trusted middle tier (e.g., one that is outside the firewall and thus more vulnerable to compromise). Moreover, because the identity of both middle tier and user are passed to the database

through an OCI lightweight user session, this feature makes it easier to audit the actions of users in a three-tier system, and thus improves accountability.

SSL

Oracle8*i* implements the SSL protocol for encryption of data exchanged between database clients and the database. This includes data in Net8, LDAP, and IIOP format. SSL encryption provides users with an alternative to the native Net8 encryption protocol which has been supported in Oracle Advanced Security (formerly known as Advanced Networking Option) since Oracle7. A benefit of SSL is that it is a de facto Internet standard, and can be used with clients which use protocols other than Net8.

In a three-tier system, SSL support in the database means that data exchanged between the middle tier and the database can be encrypted using SSL. The SSL protocol has gained confidence of users since it is perhaps the most widely deployed and well-understood encryption protocol in use today. Oracle8*i* ŝ implementation of SSL supports the three standard modes of authentication, including anonymous (Diffie-Hellman), server-only authentication using X.509 certificates, and mutual client-server authentication with X.509.

Oracle Applications Server also supports SSL encryption between thin clients and the Oracle Applications Server, as well as between Oracle Applications Server and Oracle8*i*. As in Oracle8*i*, anonymous, server-only, and client-server authentication via X.509 are supported.

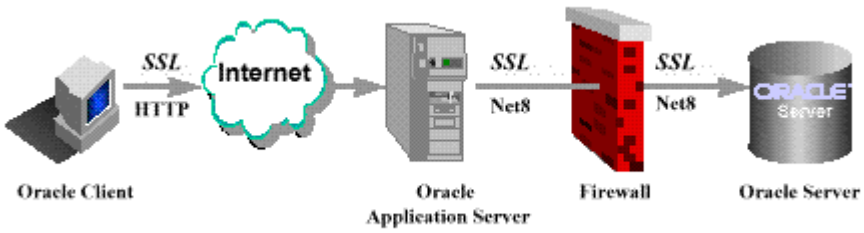

Figure 1: SSL Secures Internet and Oracle Communications

SSL addresses the problem of protecting user data exchanged between tiers in a three-tier system. By providing strong, standards-based encryption, SSL provides system developers and users with confidence that data will not be compromised in the Internet. Note also that unlike password-based authentication, which authenticates client to server only, SSL can authenticate server to client as well as client to server. This is a useful feature when building a web-based three tier system, since users often insist on authenticating the identity of a web server before they will provide the server sensitive information such as credit card numbers.

PUBLIC KEY INFRASTRUCTURE

Public Key Infrastructure (PKI) has emerged as the authentication technology which is most appropriate for securing Internet and e-commerce applications. There are a number of reasons for this. One is that PKI is highly scalable. Since users maintain their own certificates, and certificate authentication involves exchange of data between client and server only (i.e., no third party authentication server needs to be online), there is no limit to the number of users which can by supported using PKI. Moreover, PKI allows delegated trust. That is, a user who has obtained a certificate from a recognized and trusted CA can authenticate himself to a server the very first time he connects to that server, without that user having previously been registered with the system.

As noted in the section on SSL, Oracle8*i* supports certificate-based authentication. More specifically, Oracle8*i* supports standard X.509 version 3 certificates and the PKCS certificate request and installation standards. It allows users to request certificates from any certificate authority (CA) which also supports these standards. It also allows users to install trusted root certificates from their choice of CAs, allowing the server to recognize and validate certificates issued by those CAs. Oracle is working with leading PKI service and product vendors to ensure that their CA trusted roots will be pre-installed in Oracle8*i*, allowing customers to install Oracle8*i* and immediately integrate it into a PKI based on these vendors 'certificates.

VIRTUAL PRIVATE DATABASE

Oracle8*i* sets a new standard in database security with the introduction of Virtual Private Database: server-enforced, fine-grained access control, together with secure application context, enabling multiple customers and partners to have secure direct access to mission-critical data. The Virtual Private Database enables, within a single database, per-user or per-customer data access with the assurance of physical data separation. For Internet access, the Virtual Private Database can ensure that online banking customers see only their own orders. Web hosting companies can maintain multiple companies 'data in the same Oracle8*i* database, while allowing each company to see only its own data.

Within the enterprise, the Virtual Private Database results in lower cost of ownership in depoying applications. Security can be built once, in the data server, rather than in each application which access data. Security is stronger, because it is enforced by the database, no matter how a user accesses data. Security is no longer bypassed by a user accessing an ad hoc query tool or new report writer.

The Virtual Private Database is enabled by associating one or more security policies with tables or views. Direct or indirect access to a table with an attached security policy causes the database to consult a function implementing the policy. The policy function returns an access condition known as a predicate (a WHERE clause) which the database appends to the user's SQL statements, thus dynamically modifying the user's data access. A secure application context enables access conditions to be based on virtually any attributes an application deems significant, such as organization, cost center, account number, or position. For example, a human resources application may base its security on "organization," "employee number," and "position;" that is, a user in a "manager" position can see (but not update) the employee records of all employees in his organization while a user in the "employee" position can see and update only those records matching his own "employee number."

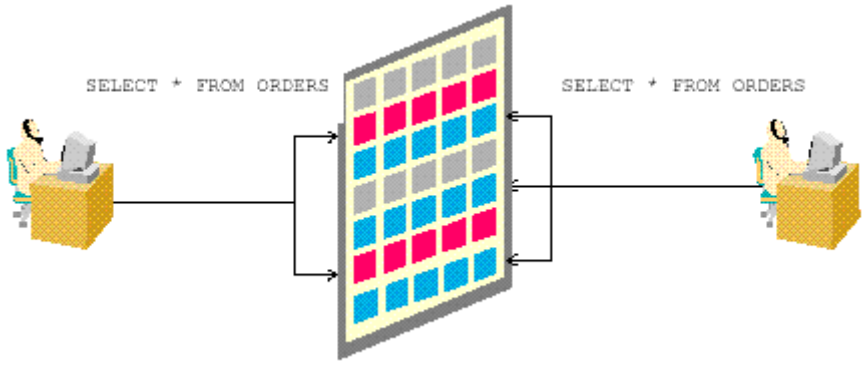

Figure 2: Virtual Private Database:Customers See Only Their Own Orders

The Virtual Private Database ensures that, no matter how a user gets to the data (through an application, a report writing tool, or SQL*Plus®) the same strong access control policy is enforced. The Virtual Private Database can help banks ensure that customers see their own accounts (and nobody else's), that telecommunications firms can keep customer records safely segregated, and that human resources applications can support their complex rules of data access to employee records. The Virtual Private Database is a key enabling technology in building three-tier systems which expose mission-critical resources to customers and partners.

DIRECTORY INTEGRATION

An inherent challenge of any distributed system, including three-tier systems, is that common application information is often fragmented across the enterprise, leading to data that is redundant, inconsistent, and expensive to manage. Directories are being viewed by an increasing number of Oracle and third-party products as the best mechanism to make enterprise information available to multiple different systems within an enterprise. Directories also make it possible for organizations to access or share certain types of information over the Internet, for example, through a virtual private network. The trend towards directories has been accelerated by the recent growth of the Lightweight Directory Access Protocol (LDAP).

A specific type of enterprise information which is commonly proposed for storage in a directory is privilege and access control information. Both user privileges, represented as roles, and object constraints, represented as Access Control Lists (ACLs) listing those users who may access an object, may be stored in a directory. Oracle Application Server is currently capable of storing and accessing, in one or more LDAP directories, ACLs for objects on the application server.

Directory information which specifies users' privileges or access attributes is sensitive, since unauthorized modification of this information can result in unauthorized granting or denial of privileges or access to users. A directory which maintains this information on behalf of the enterprise must ensure that only authorized system security administrators can modify privilege or access information maintained in the directory. Oracle Internet Directory supports attribute level access control and optional strong user authentication through SSL, and can be configured so that only specific users who are strongly authenticated are allowed to update directory information about user privileges or access.

Oracle8*i* supports enterprise roles: centrally-administered privilege sets, maintained in Oracle Internet Directory, or in directories from selected partners which meet Oracle's security criteria. Enterprise roles enable strong, centralized authorization of users. Also, an administrator can add capabilities to enterprise roles (granted to multiple users) without having to update the authorizations of each user independently. Oracle Enterprise Security Manager provides one tool to centrally manage user definitions and assign roles, resulting in a lower cost of user administration throughout the enterprise. Another benefit of single station administration is that if security is easy to administer, organizations are more likely to implement strong security throughout the enterprise.

SCHEMALESS USERS

The schemaless user feature of Oracle8*i* extends the benefits of directory integration by allowing the database to delegate administration of user identity, as well as privilege, to the directory. A schemaless user is a database user whose identity is maintained in a central LDAP repository; specifically, Oracle Internet Directory. When a schemaless user connects to the database, the

database queries the directory to determine if the user is registered there, and if so to what database schema the user should be mapped, and what roles the user should obtain.

Suppose, for example, that there are 500 users of an application HRAPP, who require access to data on several database servers in the enterprise. Instead of maintaining 500 different user accounts on each database, Oracle8*i* allows the system administrator to create a single shared schema HRAPPUSER, with appropriate privileges, on each database, and then create 500 enterprise users in an Oracle Internet Directory. When they connected to any specific database these users would be mapped to the HRAPPUSER schema, and would inherit the privileges associated with HRAPPUSER, but would also obtain any additional privileges that are associated with the roles granted to them in the directory. Although these users share a common schema, individual schemaless users 'identities are associated with their sessions by the database, and could be used for access control or auditing purposes.

The schemaless user feature has a number of benefits. It reduces the administrative burden associated with managing users in an enterprise, and should allow effective management of much larger communities of users than was previously possible. Moreover, in the future it will provide a mechanism for integrating user account and privilege management across tiers in a multi-tier system, as long as the middle tier also supports management of user identities and privileges in the directory. In such a system, new users and their privileges would be registered once in a directory, and this would give them appropriate access to the middle tier as well as any databases in the enterprise that they would need to access. In the future, it should be possible to build three-tier systems (e.g., web storefronts) in which new users can register themselves with a web server, and the web server then creates an entry for these users in the directory, giving them access to information in appropriate databases which pertain to them.

SECURE APPLICATION ROLES

As organizations open their mission-critical systems to the Internet, it is imperative that they control not only what a user may access, but how a user may access it. For example, an organization deploying a web storefront application must ensure that any privileges the user employs in the back-end order entry database are only enabled within the web storefront application. To address this need, Oracle has developed the secure application role, a role which can be enabled only through an application. Although not included in the current release of Oracle8*i*, application roles are expected to be available in the near future.

Oracle's secure application roles enable security in thin client or web-based applications by ensuring that the privileges enabled within the middle tier are only enabled within the application. Application roles are defined in Oracle such that an application can validate a SET ROLE command (using any desired criteria) prior to allowing SET ROLE to succeed. Oracle ensures that it is a trusted package - a secure program element - enabling the SET ROLE command. For example, Oracle can ensure that the user is connected through an application server proxying the user's identity to the database, and that the user is not connected to the database directly (and thus invoking the role outside of the application). All database privileges the user needs may be granted to the application role; as a result, the user can't access any data except within the application. Prior to secure application roles, system security officers who wanted to ensure that users only enabled roles through applications (and not directly in the data server) had to rely on "security by obscurity" to obtain this functionality. You could embed a role enabled by password within an application (for which the users who are granted the role did not know the password), but the password needed to be supplied by the application in some way, e.g., by burying the password within the application. Secure application roles are key enabling technology in extending the enterprise to the Internet, because they can ensure that users only access data through your Internet applications.

THREE-TIER AUDIT ENHANCEMENTS

The OCI lightweight user feature of Oracle8i enables system administrators to audit actions taken by a middle tier on behalf of a user. For example, suppose an application server *hrappserver* creates multiple lightweight sessions for users Ajit and Jane. A DBA could enable auditing for SELECTs on the bonus table that *hrappserver* initiates for Jane as follows:

AUDIT SELECT ON bonuses BY hrappserver ON BEHALF OF Jane;

Alternatively, the DBA could enable auditing on behalf of multiple users (in this case, both Jane and Ajit) connecting through a middle tier as follows:

AUDIT SELECT ON bonuses BY hrappserver ON BEHALF OF ANY;

This auditing option only audits SELECT statements being initiated by *hrappserver* on behalf of other users. A DBA can enable separate auditing options to capture SELECTs against the bonus table from clients connecting directly to the database:

AUDIT SELECT ON bonuses;

SUMMARY

Three-tier architectures are becoming increasingly common, fueled by the growth of the Internet and by the need for companies to expand their enterprises to support customers and partners through the Internet. Whether companies employ client-server or three-tier architectures, they will continue to store and manage their data in database servers. While protecting this data and controlling access to it becomes more challenging when moving from a two- to a three-tier architecture, Oracle8*i* provides many new security features which are specifically designed to address the needs of three-tier systems. Building on almost 20 years of Oracle client-server expertise, Oracle8*i* provides robust, industry-standard security mechanisms to meet the challenges of securing the Internet-enabled enterprise.