Lowering the Cost of Enterprise Security

An Oracle Business White Paper

November 1998

Lowering the Cost of Enterprise Security

INTRODUCTION

Internet and extranet-based applications, and new technologies such as CORBA and Java, can reduce the total cost of ownership for enterprises, the former through more timely, efficient, and closer interactions with customers and partners, and the latter through economies of code reuse and/or "write once, run anywhere." Security has usually been viewed only as a necessary cost of enabling these new technologies. However, security can actually reduce the total cost of ownership of information technology in multiple ways:

- lower development costs through embedding security in the database (once), rather than implementing security multiple ways, across multiple applications
- higher user productivity through single sign-on
- reduced administration cost through centralized user management
- efficiencies in accessing centrally-managed application metadata

REDUCING COST OF OWNERSHIP THROUGH SERVER-ENFORCED SECURITY

Problem: The Need for Data-Driven, User-Based Security

The adoption of the Internet, not only for sharing information with customers, but to incorporate online transactions, has dramatically increased the need for security. Giving customers and partners direct access to mission-critical systems may yield reduced cost, better service, and more timely information, but it also offers new challenges. Organizations must not only keep data safe from prying eyes, but they must segregate data appropriately, often to the level of individual customers or users. Companies want the revenue benefits of increased sales and improved customer service, without jeopardizing these revenues through loss or compromise of valuable customer data.

Another trend sweeping the corporate community is an increased focus on core competencies and the outsourcing of tasks such as human resources, customer support, online ticketing, and Web storefronts. Many companies are interested in providing "hosting" environments, with a well-designed and well-managed computing infrastructure, but face tremendous challenges in designing systems that keep the data of each "hosted" corporation separate and secure from each other, while allowing personalizations and data access methods which best meet their individual needs.

One of the main needs within an enterprise for server-enforced security is a long-standing access control challenge: the "ad hoc query problem." When access control is embedded in an application, users who have access to ad-hoc queries or reporting tools bypass the security of the application. Therefore, organizations must spend considerable sums of money reimplementing data security in every application which accesses data, a practice which is not only wasteful and duplicative, but one that often results in insecure implementations.

Solution: The Virtual Private Database

Oracle8*i* addresses these diverse security needs by introducing the Virtual Private Database: server-enforced, flexible, fine-grained access control, based on security "contexts." The combination of very granular, server enforced security with security "contexts" for making access control decisions enables, within a single Oracle8*i* database, per-customer or per-user data access with the assurance of physical data separation. Each user accesses his own data as if that data were stored and managed in a physically separate system.

Fine-grained access control is enabled by associating one or more security policies (implemented by functions) with tables or views. Direct or indirect access to a table with an attached security policy causes the Oracle8*i* data server to consult the policy function. The policy function returns an access condition known as a predicate (a WHERE clause) which the data server appends to the SQL statements, dynamically modifying the user's data access. For example, if your security policy is that customers can see their own orders, a user issuing the following query:

```
        SELECT * FROM orders;
```

could have her query transparently and dynamically rewritten by Oracle8*i* as follows:

```
        SELECT * FROM orders
        WHERE cust_num =
        (SELECT custnum FROM customers
        WHERE cust_name = USERENV(‘user’));
```

As a result, each user only accesses her own data, and is not even aware that other records exist.

Fine-grained access control enables organizations to dynamically modify data access, transparently to both users and applications, based on almost any criteria, even to the degree of having different access conditions per user, per group of users, or per application.

Flexible Implementation The Virtual Private Database offers flexible policy implementation, to allow customers to fine-tune security policies based on their specific needs:

- Attach security policies to tables or views. Many applications already use views for security reasons, or to enforce business rules. Allowing security policies to be attached to either views or tables allows organizations to add fine-grained access to their existing applications, without completely rewriting them, providing better security in a cost-effective way. This also avoids having to build multiple views to implement security.
- Add security policies to only those tables or views where it is needed. For example, to implement the policy ‘customers can see only their own orders,’ you need only add security policies to the ORDERS and ORDER_LINES table. This prevents potentially expensive ‘security overkill’ solutions offered by more complex granular access control systems, such as multilevel security.
- Enable different policies for different types of access, e.g. select, insert, delete, and update. For example, you could implement a policy on the EMP table that enables users to query name and address information for any employee, but allows them to update only their own records. This reduces the cost of implementing fine-grained access control, since it need not apply to every type of statement on every table or view.
- Add multiple policies per table. For example, you may have a hosting application which allows different companies' HR systems to enable different access control conditions. Companies can add additional security policies on top of the base HR application security policy (e.g. that data access is restricted by company), without affecting the security enforcement and data separation policies of the base application: "customization without code rewrite."

Context-based Security Enforcement Most access control is based on some attribute of the user who will be accessing the data, such as his position or organizational unit. To make the Virtual Private Database easy to implement, Oracle8*i* provides security 'application contexts,' user-definable, secure attributes on which access control can be based.

For example, a typical human resources application may base its access control on 'organization,' 'employee number,' and 'position'; that is, a user in the 'manager' position can see the employee records of all employees in his 'organization,' while a user in the 'employee' position can only see and update records matching his own 'employee number.' Alternatively, a general ledger application may base its security on 'set of books,' and 'cost center.' Application contexts can be accessed within policy functions to determine the correct access condition (predicate) to return, or within an access condition itself, to provide the correct employee number for a logged-in user, for example. Oracle8*i* ensures that application contexts are secure by enforcing that only trusted packages implementing them set context values. Because application contexts offer such flexible security policy implementation, it's now simpler and more cost-effective to write applications (such as human resource applications) using database security.

The basic value proposition for the Virtual Private Database is that it enables you to truly build security policies—even complex, multi-faceted security policies—in the data server, which are enforced no matter how users access data. This removes the need to re-implement security in multiple applications. Most applications need not be security-aware, but may completely rely upon underlying data server security enforcement.

The Virtual Private Database is key enabling technology for Internet-based customer and partner access to production systems. Many organizations seeking to do this have had to implement multiple, complex layers of security to achieve data separation, at considerable cost. The Virtual Private Database offers a simple, more cost-effective and secure way of effecting data-based data separation, with the assurance of physical data separation. Better security at lower cost thus affords greater profit margins from 'Web-icizing' the enterprise.

Problem: The Need for More Flexible Privilege Models

The desire to encapsulate business logic in well-defined transactions is often enabled via stored procedures: program elements, stored in compiled form in the database, which typically execute using a "definer's rights" model. That is, users who have EXECUTE permission on Chuck's (the definer's) procedure access Chuck's data with Chuck's privilege set, for the duration of a well-formed transaction only. "Definer's rights" procedures are most useful for encapsulation of privileges within a business context; that is, users need not have direct privilege on objects, merely the privilege to execute a procedure which accesses objects in a defined way.

However, object-oriented technology and the use of new programming languages such as Java require a more flexible privilege model, in which business logic is separate from data and the privileges required to access an object. For example, an Enterprise Java Bean that updates a bank account balance should update Jane's account balance if Jane accesses the bean, but John's account balance if John accesses the bean. Furthermore, the Enterprise Java Bean may be deployed in a bean store, and the bean may actually act upon different databases, or different schemas within the same database. Alternatively, developers of data cartridges wish to deploy application libraries, in which business logic *must* remain independent of specific users' privileges. For example, the developer of a time series cartridge has no knowledge of whose data the cartridge will ultimately act upon.

Solution: Invoker's Rights Procedures

To support the above requirements, Oracle8*i* extends its privilege model by offering "invoker's rights" procedures, available in both PL/SQL and Java, which execute with an invoker's privilege set, on an invoker's schema.

Invoker’s rights procedures enable organizations to lower their cost of deploying applications, since business logic—for example, a procedure which updates account balances—is not tied to a particular user’s privilege set or a particular schema, and thus can be used (and reused) by many applications and users. For example, an organization may have a common set of applications which multiple divisions use, but the data upon which the applications act are separated from one another. Division 1 employees never access Division 2’s data, and vice versa. One approach to this problem would be to physically separate data on different servers, which is more expensive, and makes it difficult to do necessary summarizations at a corporate level. Another approach is to maintain the data of each application in a separate schema, and have the application reside in an application-owned schema. Using invoker’s rights procedures, users from each division can access the same application to act upon only their own data. Invoker’s rights procedures thus enhance the ability of organizations to deploy common applications which nonetheless “act” differently for different sets of users. The result is stronger security at a lower cost of deployment.

REDUCING THE COST OF USER MANAGEMENT

The problem of user authentication and user management has become acute in virtually every organization. Users, in attempting to contend with far too many passwords, either write them down, or choose the same easily-guessed password for all accounts. Organizations attempting to manage multiple accounts for each user devote significant resources to user administration, or invest in network authentication services, many of which promise centralized user management and single sign-on. “Solving the single sign-on problem” has become the Holy Grail of security administration.

Problem: Too Many Passwords, Too Many User Accounts

Single sign-on (SSO) solutions can lower the cost of ownership by reducing the amount of time users have to spend logging in to multiple applications, waiting for locked accounts to be unlocked, and making up new passwords. Of course, having one password to remember also makes for a pleasanter user experience than remembering twenty-odd passwords, and almost always provides better security, because there is one complex password to remember at most instead of twenty easy passwords. Also, there are significant cost savings to implementing sign-on, such as greater worker productivity, and less administrator time spent unlocking accounts.

However, even the best single sign-on solution usually does not address the core problem for organizations: too many user *accounts*. Many SSO solutions mask the true problem from users by either having them connect to a central server, where they obtain a ticket that can be used to authenticate them to other applications (in which they still need an account), by storing usernames and passwords centrally which are accessed by a master password, or by “checking out” a generic user account used for other applications. Of course, having a user connect to an application as “Public User A” is not conducive to either good user accountability or separation of privilege, two other important security considerations. Storing usernames and passwords in a central data store is also an imprudent security practice. In short, single sign-on alone is no panacea for an organization’s user management ailments.

Solution: Integrated Security and Directory Services

There is a solution to the ‘too many accounts’ problem, and it lies in the domain of directory services. The rapid growth of the Lightweight Directory Access Protocol (LDAP) is leading more organizations to store common application information—particularly common user information—in a directory server. Implementation of a directory-based repository of user information (e.g. user name, organization, mailstop, etc.) provides immediate benefits to applications that share the same metadata (for example, mail and human resources). Additionally, these directories can be extended to include roles and privileges the users have, and even their authentication credentials, such as X.509 certificates and suitably-protected private keys. In a classic example of the sum being greater than the parts, the integration of security and directory services provides greater benefits than either function alone can offer.

Many organizations implement central user administration principally to achieve better control over system security. When users' 'identities' are scattered across multiple applications, organizations typically cannot react quickly enough to organizational change; e.g. if a user changes jobs or leaves the company, it may be days or weeks before his privilege set changes. As a result, users either can't access what they need to access, or can access information they should no longer be allowed to see. However, if the user 'definition' is stored in a directory, together with his privileges, an organizational change can be effected in one place—the directory—and immediately reflected in privilege changes.

Ultimately, if *all* applications can rely on a centralized user definition, with centralized privileges, then this removes the need to create user accounts in multiple locations. What was originally a 'better security management' solution can then become a means of actually lowering the cost of user management. While realistically, the move to directory-driven user definition must be implemented over time, even a ten to twenty percent reduction in the number of user accounts would yield a substantial cost savings for a typical organization. One can only imagine the cost savings to organizations of managing a single account per user instead of the twenty or thirty accounts per user so prevalent today.

Oracle8*i* addresses the above needs for strong security and centralized management of user information by offering integrated security and directory services, including storing and managing user information in a directory. Multiple Oracle applications can rely on a common, centralized definition of a user to determine which applications, services, and data servers a user may access, and with what privileges.

Oracle8*i*'s common security and directory services include:

- single sign-on to multiple services throughout the enterprise
- single station administration (SSA) of users
- a single enterprise user, instead of multiple accounts per user
- well-integrated, standards-based public key infrastructure (PKI)

Single Sign-On Oracle8 provided single sign-on to Oracle users through X.509 (version 1) digital certificates and a proprietary authentication protocol. The advantage of X.509 certificates is that they may be used to uniquely identify an individual within an organization and thus enable strong authentication. Also, instead of remembering multiple passwords, a user need only remember the password that unlocks his Oracle 'wallet.' The certificate and private key contained in the wallet are used to authenticate the user to multiple services, including application servers and data servers, which need no longer store and manage local passwords for users.

Oracle8*i* offers enhanced PKI-based single sign-on through use of interoperable X.509 (version 3) certificates for authentication over Secure Sockets Layer (SSL), the standard for Internet authentication. In addition to strong user authentication, SSL also provides network data confidentiality and data integrity for multiple types of connections: LDAP, IIOP (Internet Intra-ORB Protocol), and Net8. Oracle wallets can accept certificates issued by Oracle Certificate Authority or other X.509-compliant CAs, which enables organizations to further leverage their existing public key infrastructure.

Single Station Administration (SSA) As discussed above, managing thousands of user accounts is one of the largest administration challenges facing large organizations. Creating user accounts and assigning privileges is often a multi-step process, requiring multiple tools. Significant new functionality has been added in Oracle8*i* to address this need. Oracle Security Manager (an extension to Oracle Enterprise Manager) provides single station administration: from a single console, an administrator can perform all the following:

- create a user in Oracle Internet Directory (an LDAP-compliant directory server)
- create a user in multiple Oracle8*i* databases
- create enterprise roles that span multiple databases
- assign one or more enterprise roles to a user

Having one tool to centrally manage user definitions—in the directory itself, as well as in multiple databases—results in a lower cost of user administration throughout the enterprise. Of course, the main benefit of single station administration is that if security is easy to administer, organizations are more likely to implement security well throughout the enterprise.

Single Enterprise User The ultimate payoff for investing in integrated security and directory services will be the "single enterprise user": a user who is created once for the enterprise—along with his enterprise roles, privileges, and access rights—in a directory server accessible over LDAP. Users will no longer need to be Oracle8*i* database users nor have identified schemas; therefore, organizations can manage far few user accounts. Security becomes easier to enforce (and arguably, less costly to implement): should a user change jobs or leave the organization, one can alter or remove all his privileges merely by changing his user entry in Oracle Internet Directory. Organizations need no longer worry about "orphan" accounts or out-of-date privileges, which consume valuable system resources as well as being a target for hackers. Organizations save significant resources by managing a single user account and assigning enterprise roles once, instead of creating multiple user accounts with multiple passwords, having multiple authorizations.

The Single Enterprise User will offer organizations deploying Internet applications the greatest benefit. These organizations want to have identified "users," but cannot afford the management or storage overhead of creating potentially hundreds of thousands of users, certainly not in the applications and databases behind their Internet front-end. However, directories excel at management, search, and retrieval of hundreds of thousand of user records. Creating "Internet application users" as directory entries gives organizations the benefit of keeping track of their users (and their users' preferences), with high security and low overhead.

LOWER COSTS, BETTER SECURITY

No organization has unlimited funds to spend to implement security. Typically, the cost of security measures has been weighed against the value of data protected. However, Oracle8*i* takes security from being a cost of doing business to a cost *savings* of doing business. Oracle8*i* "makes security pay" for both Internet and enterprise applications by:

- enabling organizations to build security once, in the data server
- lowering the cost of deploying applications
- enabling single sign-on
- centralizing user management through single station administration

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
+1.650.506.7000
Fax +1.650.506.7200
http://www.oracle.com/

This document is provided for informational purposes only, and the information herein is subject to change without notice. Please report any errors herein to Oracle Corporation. Oracle Corporation does not provide any warranties covering and specifically disclaims any liability in connection with this document.