Oracle9*i*

Security Overview

Release 2 (9.2)

March 2002

Part No. A96582-01

ORACLE®

Oracle9*i* Security Overview, Release 2 (9.2)

Part No. A96582-01

Primary Author: Jeff Levinger

Contributing Authors: Rita Moran, Kristy Browder, Mary Ann Davidson, John Heimann, Paul Needham, David Saslav, Uppili Srinivasan

Contributors: Mike Cowan, Sudha Iyer, Richard Smith, Deborah Steiner, Daniel Wong

Graphic Designer: Valarie Moore

Contents

Send Us Your Comments xiii

Preface xv

Audience xvi
Organization xvi
Related Documentation xviii
Conventions xix
Documentation Accessibility xxi

Part I Security Challenges

1 Data Security Challenges

Top Security Myths 1-2
Understanding the Many Dimensions of System Security 1-3
Fundamental Data Security Requirements 1-5
Confidentiality 1-5
Privacy of Communications 1-5
Secure Storage of Sensitive Data 1-5
Authenticated Users 1-6
Granular Access Control 1-6
Integrity 1-6
Availability 1-7
Security Requirements in the Internet Environment 1-8
Promises and Problems of the Internet 1-8

Increased Data Access 1-9
Much More Valuable Data 1-9
Larger User Communities 1-10
Scalability 1-10
Manageability 1-11
Interoperability 1-11
Hosted Systems and Exchanges 1-11
A World of Data Security Risks 1-12
Data Tampering 1-12
Eavesdropping and Data Theft 1-12
Falsifying User Identities 1-13
Password-Related Threats 1-13
Unauthorized Access to Tables and Columns 1-14
Unauthorized Access to Data Rows 1-14
Lack of Accountability 1-14
Complex User Management Requirements 1-15
Multitier Systems 1-15
Scaling the Security Administration of Multiple Systems 1-15
A Matrix of Security Risks and Solutions 1-16
The System Security Team 1-18

Part II Technical Solutions to Security Risks

2 Protecting Data Within the Database

Introduction to Database Security Concepts 2-2
System and Object Privileges 2-2
System Privileges 2-2
Schema Object Privileges 2-3
Managing System and Object Privileges 2-3
Using Roles to Manage Privileges 2-4
Database Roles 2-4
Global Roles 2-5
Enterprise Roles 2-5
Secure Application Roles 2-6
Using Stored Procedures to Manage Privileges 2-6

Using Network Facilities to Manage Privileges ... 2-7
Using Views to Manage Privileges ... 2-7
Row Level Security ... 2-8
Complex and Dynamic Views ... 2-9
Application Query Rewrite: Virtual Private Database ... 2-9
Label-Based Access Control ... 2-9
Encrypting Data on the Server ... 2-10
Selective Encryption of Stored Data ... 2-10
Industry Standard Encryption Algorithms ... 2-11
Database Integrity Mechanisms ... 2-11
System Availability Factors ... 2-12
Secure Configuration Practices ... 2-13

3 Protecting Data in a Network Environment

Introduction to Data Protection in a Network Environment ... 3-2
Protecting Data During Transmission ... 3-3
Controlling Access Within the Network ... 3-3
Middle-Tier Connection Management ... 3-3
Native Network Capabilities (Valid Node Checking) ... 3-3
Database Enforced Network Access ... 3-4
Encrypting Data for Network Transmission ... 3-4
Encryption Algorithms ... 3-5
Data Integrity Checking ... 3-6
Secure Sockets Layer (SSL) Protocol ... 3-6
Firewalls ... 3-7
Ensuring Security in Three-Tier Systems ... 3-8
Proxy Authentication to Ensure Three-Tier Security ... 3-8
Java Database Connectivity (JDBC) ... 3-8
JDBC-Oracle Call Interface Driver ... 3-9
JDBC Thin Driver ... 3-9

4 Authenticating Users to the Database

Introduction to User Authentication ... 4-2
Passwords for Authentication ... 4-2
Strong Authentication ... 4-3

Kerberos and CyberSafe 4-4
RADIUS 4-4
Token Cards 4-5
Smart Cards 4-6
Distributed Computing Environment (DCE) 4-7
Biometrics 4-7
PKI and Certificate-Based Authentication 4-7
Proxy Authentication and Authorization 4-8
Single Signon 4-10
Server-Based Single Signon 4-10
Middle Tier Single Signon 4-11

5 Using and Deploying a Secure Directory

Introduction 5-2
Centralizing Shared Information with LDAP 5-3
Securing the Directory 5-5
Directory Authentication of Users 5-5
Password Protection in a Directory 5-6
Directory Access Controls and Authorization 5-7
Directory-Based Application Security 5-8
Authorization of Users 5-8
Authorization of Administrators 5-8
Administrative Roles in the Directory 5-12

6 Administering Enterprise User Security

Introduction 6-2
Enterprise Privilege Administration 6-3
Shared Schemas 6-4
Password-Authenticated Enterprise Users 6-5
Enterprise Roles 6-5
Multitier Authentication and Authorization 6-5
Single Sign-On 6-6

7 Auditing to Monitor System Security

Introduction 7-2
Fundamental Auditing Requirements 7-2
Robust, Comprehensive Auditing 7-2
Efficient Auditing 7-3
Customizable Auditing 7-3
Fine Grained, Extensible Auditing 7-3
Auditing in Multitier Application Environments 7-4

8 The Public Key Infrastructure Approach to Security

Introduction 8-2
Security Features of PKI 8-2
Components of PKI 8-3
Advantages of the PKI Approach 8-3
Public Key Cryptography and the Public Key/Private Key Pair 8-4
Secure Credentials: Certificate-Based Authentication in PKI 8-5
Certificates and Certificate Authorities 8-5
Certificate Authorities 8-5
Certificates 8-6
Authentication Methods Used with PKI 8-7
Secure Sockets Layer Authentication and X.509v3 Digital Certificates 8-7
Entrust/PKI Authentication 8-8
Storing Secure Credentials with PKI 8-8
Single Sign-On Using PKI 8-9
Network Security Using PKI 8-9

Part III Oracle9*i* Security Products

9 Oracle9*i* Security Products and Features

Oracle9*i* Standard Edition 9-2
Integrity 9-3
Data Integrity 9-3
Entity Integrity Enforcement 9-3
Referential Integrity 9-3

Authentication and Access Controls in Oracle9*i* 9-4
Privileges 9-4
Roles 9-5
Auditing 9-5
Views, Stored Program Units, Triggers 9-5
Data Encryption 9-6
High Availability 9-6
User Profiles 9-6
Online Backup and Recovery 9-7
Advanced Replication 9-7
Data Partitioning 9-7
Very High Availability with Real Application Clusters 9-8
Proxy Authentication in Oracle9*i* 9-9
Introduction 9-9
Support for Additional Protocols 9-10
Expanded Credential Proxy 9-10
Application User Proxy Authentication 9-11
Oracle9*i* Enterprise Edition 9-12
Internet Scale Security Features 9-12
Deep Data Protection 9-12
Internet-Scale Security 9-13
Secure Hosting and Data Exchange 9-13
Application Security 9-13
Virtual Private Database in Oracle9*i* 9-14
Virtual Private Database in Oracle8*i* and Oracle9*i* 9-14
How Virtual Private Database Works 9-15
Application Context in Oracle9*i* 9-16
How Application Context Facilitates VPD 9-17
Application Context Accessed Locally 9-17
Application Context Initialized Externally 9-17
Application Context Initialized Globally 9-18
Application Context Accessed Globally 9-18
How Partitioned Fine-Grained Access Control Facilitates VPD 9-19
User Models and Virtual Private Database 9-20
Oracle Policy Manager 9-20

Secure Application Role 9-21
Fine-Grained Auditing 9-21
Oracle Auditing for Three-Tier Applications 9-23
Java Security Implementation in the Database 9-23
Class Execution 9-23
SecurityManager Class 9-23
Oracle Advanced Security 9-24
Introduction to Oracle Advanced Security 9-25
Network Security Services of Oracle Advanced Security 9-27
Oracle Net Services Native Encryption 9-27
Data Integrity Features of Oracle Advanced Security 9-29
Secure Sockets Layer (SSL) Encryption Capabilities 9-29
Oracle Advanced Security Support for SSL 9-29
Checksumming in Oracle Advanced Security SSL 9-29
Oracle9*i* Application Server Support for SSL 9-30
Java Encryption Features of Oracle Advanced Security 9-30
JDBC-OCI Driver 9-30
Thin JDBC 9-31
Secure Connections for Virtually Any Client 9-32
Oracle Java SSL 9-32
Strong Authentication Methods Supported by Oracle Advanced Security 9-33
Oracle Public Key Infrastructure-Based Authentication 9-34
Kerberos and CyberSafe with Oracle Advanced Security 9-36
RADIUS with Oracle Advanced Security 9-36
Token Cards with Oracle Advanced Security 9-37
Smart Cards with Oracle Advanced Security 9-37
Biometric Authentication with Oracle Advanced Security 9-37
Distributed Computing Environment (DCE) with Oracle Advanced Security .. 9-38
Single Sign-On Implementations in Oracle Advanced Security 9-38
Single Sign-On Configuration with Third-Party Products 9-38
PKI-Based Single Sign-On Configuration 9-38
Enterprise User Security Features of Oracle Advanced Security 9-39
Password-Authenticated Enterprise Users 9-40
Tools for Enterprise User Security 9-40
Shared Schemas in Oracle Advanced Security 9-41

Current User Database Links ... 9-41
Directory Integration ... 9-41
PKI Implementation in Oracle Advanced Security ... 9-42
Components of Oracle Public Key Infrastructure-Based Authentication ... 9-42
Secure Sockets Layer ... 9-42
Oracle Call Interface ... 9-42
Trusted Certificates ... 9-42
X.509 Version 3 Certificates ... 9-43
Oracle Wallets ... 9-43
Oracle Wallet Manager ... 9-43
Oracle Enterprise Login Assistant ... 9-43
Oracle Internet Directory ... 9-43
Oracle Enterprise Security Manager ... 9-44
PKI Integration and Interoperability ... 9-44
PKCS #12 Support ... 9-45
Wallets Stored in Oracle Internet Directory ... 9-45
Multiple Certificate Support ... 9-45
Strong Wallet Encryption ... 9-45
Oracle PKI Implementation Summary ... 9-46
Oracle Label Security ... 9-47
Oracle Internet Directory ... 9-48
Introduction to Oracle Internet Directory ... 9-49
LDAP Compliance ... 9-51
How Oracle Internet Directory is Implemented ... 9-52
How Oracle Internet Directory Organizes Enterprise User Management ... 9-53
Enterprise User Administration with Oracle Internet Directory ... 9-53
Shared Schemas with Oracle Internet Directory ... 9-53
Oracle Net Services ... 9-54
Components of Oracle Net Services ... 9-54
Oracle Net on the Client ... 9-54
Oracle Net on the Database Server ... 9-54
Oracle Protocol Support ... 9-55
Oracle Connection Manager ... 9-55
Protocol Conversion ... 9-55
Access Control ... 9-55

Session Multiplexing 9-56
Firewall Support with Oracle Net Services 9-56
Firewalls Using Oracle Connection Manager in an Intranet Environment 9-56
Firewalls Using Oracle Net Firewall Proxy in an Internet Environment 9-57
Valid Node Checking in Oracle Net Services 9-58
Database-Enforced VPD Network Access 9-58
Oracle9*i* Application Server 9-59
Oracle HTTP Server 9-60
Oracle Portal 9-61
Single Sign-On in Oracle9*i* Application Server 9-61
Web SSO Technology 9-61
Login Server 9-62
LDAP Integration 9-62
PKI Support 9-62
Multitier Integration 9-63
Oracle Single Sign-On Summary 9-63

Index

Send Us Your Comments

Oracle9*i* Security Overview, Release 2 (9.2)

Part No. A96582-01

Oracle Corporation welcomes your comments and suggestions on the quality and usefulness of this document. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most?

If you find any errors or have any other suggestions for improvement, please indicate the document title and part number, and the chapter, section, and page number (if available). You can send comments to us in the following ways:

- Electronic mail: infodev_us@oracle.com
- FAX: (650) 506-7227 Attn: Server Technologies Documentation Manager
- Postal service:
  Oracle Corporation
  Server Technologies Documentation
  500 Oracle Parkway, Mailstop 4op11
  Redwood Shores, CA 94065
  USA

If you would like a reply, please give your name, address, telephone number, and (optionally) electronic mail address.

If you have problems with the software, please contact your local Oracle Support Services.

Preface

Oracle9i Security Overview presents the basic concepts of data security in an Internet environment. It outlines fundamental data security requirements and explains the risks that threaten the integrity and privacy of your data. Several chapters introduce the rich array of technology that can contribute to system security. The book concludes with a survey of the Oracle features and products that implement these technologies.

Together, these products have the potential to control access to all the vulnerable areas of your system. They can help users and administrators to perform their tasks without jeopardizing the security plan you have put in place.

This preface contains these topics:

- Audience
- Organization
- Related Documentation
- Conventions
- Documentation Accessibility

Audience

Oracle9i Security Overview is intended for database administrators (DBAs), application programmers, security administrators, system operators, and other Oracle users who perform the following tasks:

- Analyze application security requirements
- Create security policies
- Implement security technologies
- Administer enterprise user security

To use this document, you need general familiarity with database and networking concepts.

Organization

This document introduces the basic concepts of system security in an Internet environment. It outlines the data security risks which are prevalent today, and the industry-standard technologies available to address them. It then presents the carefully integrated suite of Oracle products you can use to implement these security technologies.

Part I: Security Challenges

This part explains the wide range of security risks to the integrity and privacy of data.

Chapter 1, "Data Security Challenges"

This chapter introduces the fundamental concepts of data security, and outlines the threats against which data and systems must be defended.

Part II: Technical Solutions to Security Risks

This part introduces the technology available to meet data security challenges.

Chapter 2, "Protecting Data Within the Database"

This chapter describes the fundamental elements of database security.

Chapter 3, "Protecting Data in a Network Environment"

This chapter explains how data can be protected while being transmitted over a network. It covers network access control, encryption, Secure Sockets Layer, and firewalls, as well as security in a three-tier environment.

Chapter 4, "Authenticating Users to the Database"

This chapter describes the wide range of technology available to verify the identity of database, application, and network users.

Chapter 5, "Using and Deploying a Secure Directory"

It can be advantageous to centralize storage and management of user-related information in a directory. This chapter describes how to protect such a directory, and how access can be controlled by using a directory.

Chapter 6, "Administering Enterprise User Security"

This chapter describes the elements which make up a strong enterprise user management facility.

Chapter 7, "Auditing to Monitor System Security"

This chapter describes technology available to monitor the effectiveness of your security policies.

Chapter 8, "The Public Key Infrastructure Approach to Security"

This chapter introduces the Public Key Infrastructure (PKI) approach to security. It describes the components of PKI, and explains why this has become an industry standard.

Part III: Oracle9*i* Security Products

This part presents the suite of Oracle security products which can meet your data security requirements.

Chapter 9, "Oracle9i Security Products and Features"

This chapter presents the major security-related products available with Oracle9*i*, and specifies the way in which each of them implements the kinds of security technologies described in Part II of this book.

Related Documentation

For more information, see these Oracle resources:

- *Oracle9i Database Concepts*
- *Oracle9i Application Developer's Guide - Fundamentals*
- *Oracle9i Database Administrator's Guide*
- *Oracle Advanced Security Administrator's Guide*
- *Oracle Internet Directory Administrator's Guide*
- *Oracle Label Security Administrator's Guide*
- *Oracle9i Net Services Administrator's Guide*
- *Single Sign-On Administrator's Guide*
- *Oracle9i Java Developer's Guide*
- *Oracle9i JDBC Developer's Guide and Reference*
- *Oracle Enterprise Manager Concepts Guide*

Many books in the documentation set use the sample schemas of the seed database, which is installed by default when you install Oracle. Refer to *Oracle9i Sample Schemas* for information on how these schemas were created and how you can use them yourself.

In North America, printed documentation is available for sale in the Oracle Store at

`http://oraclestore.oracle.com/`

Customers in Europe, the Middle East, and Africa (EMEA) can purchase documentation from

`http://www.oraclebookshop.com/`

Other customers can contact their Oracle representative to purchase printed documentation.

To download free release notes, installation documentation, white papers, or other collateral, please visit the Oracle Technology Network (OTN). You must register online before using OTN; registration is free and can be done at

`http://otn.oracle.com/admin/account/membership.html`

If you already have a username and password for OTN, then you can go directly to the documentation section of the OTN Web site at

http://otn.oracle.com/docs/index.htm

To access the database documentation search engine directly, please visit

http://tahiti.oracle.com

Conventions

This section describes the conventions used in the text and code examples of this documentation set. It describes:

- Conventions in Text
- Conventions in Code Examples

Conventions in Text

We use various conventions in text to help you more quickly identify special terms. The following table describes those conventions and provides examples of their use.

Convention	Meaning	Example
Bold	Bold typeface indicates terms that are defined in the text or terms that appear in a glossary, or both.	When you specify this clause, you create an **index-organized table**.
Italics	Italic typeface indicates book titles or emphasis.	*Oracle9i Database Concepts* Ensure that the recovery catalog and target database do *not* reside on the same disk.
UPPERCASE monospace (fixed-width font)	Uppercase monospace typeface indicates elements supplied by the system. Such elements include parameters, privileges, datatypes, RMAN keywords, SQL keywords, SQL*Plus or utility commands, packages and methods, as well as system-supplied column names, database objects and structures, usernames, and roles.	You can specify this clause only for a NUMBER column. You can back up the database by using the BACKUP command. Query the TABLE_NAME column in the USER_TABLES data dictionary view. Use the DBMS_STATS.GENERATE_STATS procedure.

Convention	Meaning (Cont.)	Example
`lowercase monospace (fixed-width font)`	Lowercase monospace typeface indicates executables, filenames, directory names, and sample user-supplied elements. Such elements include computer and database names, net service names, and connect identifiers, as well as user-supplied database objects and structures, column names, packages and classes, usernames and roles, program units, and parameter values. **Note:** Some programmatic elements use a mixture of UPPERCASE and lowercase. Enter these elements as shown.	Enter `sqlplus` to open SQL*Plus. The password is specified in the `orapwd` file. Back up the datafiles and control files in the `/disk1/oracle/dbs` directory. The `department_id`, `department_name`, and `location_id` columns are in the `hr.departments` table. Set the `QUERY_REWRITE_ENABLED` initialization parameter to `true`. Connect as `oe` user. The `JRepUtil` class implements these methods.
`lowercase monospace (fixed-width font) italic`	Lowercase monospace italic font represents placeholders or variables.	You can specify the *`parallel_clause`*. Run `U`*`old_release`*`.SQL` where *`old_release`* refers to the release you installed prior to upgrading.

Conventions in Code Examples

Code examples illustrate SQL, PL/SQL, SQL*Plus, or other command-line statements. They are displayed in a monospace (fixed-width) font and separated from normal text as shown in this example:

```
SELECT username FROM dba_users WHERE username = 'MIGRATE';
```

The following table describes typographic conventions used in code examples and provides examples of their use.

Convention	Meaning	Example
[ ]	Brackets enclose one or more optional items. Do not enter the brackets.	`DECIMAL (`*`digits`*` [ , `*`precision`*` ])`
{ }	Braces enclose two or more items, one of which is required. Do not enter the braces.	`{ENABLE | DISABLE}`
|	A vertical bar represents a choice of two or more options within brackets or braces. Enter one of the options. Do not enter the vertical bar.	`{ENABLE | DISABLE}` `[COMPRESS | NOCOMPRESS]`

Convention	Meaning	Example
. . .	Horizontal ellipsis points indicate either: ■ That we have omitted parts of the code that are not directly related to the example ■ That you can repeat a portion of the code	`CREATE TABLE ... AS subquery;` `SELECT col1, col2, ... , coln FROM employees;`
. . .	Vertical ellipsis points indicate that we have omitted several lines of code not directly related to the example.	
Other notation	You must enter symbols other than brackets, braces, vertical bars, and ellipsis points as shown.	`acctbal NUMBER(11,2);` `acct CONSTANT NUMBER(4) := 3;`
Italics	Italicized text indicates placeholders or variables for which you must supply particular values.	`CONNECT SYSTEM/system_password` `DB_NAME = database_name`

UPPERCASE	Uppercase typeface indicates elements supplied by the system. We show these terms in uppercase in order to distinguish them from terms you define. Unless terms appear in brackets, enter them in the order and with the spelling shown. However, because these terms are not case sensitive, you can enter them in lowercase.	`SELECT last_name, employee_id FROM employees;`
		`SELECT * FROM USER_TABLES;`
		`DROP TABLE hr.employees;`

lowercase	Lowercase typeface indicates programmatic elements that you supply. For example, lowercase indicates names of tables, columns, or files.
	Note: Some programmatic elements use a mixture of UPPERCASE and lowercase. Enter these elements as shown.
		`SELECT last_name, employee_id FROM employees;`
		`sqlplus hr/hr`
		`CREATE USER mjones IDENTIFIED BY ty3MU9;`

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Standards will continue to evolve over time, and Oracle Corporation is actively engaged with other

market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For additional information, visit the Oracle Accessibility Program Web site at

`http://www.oracle.com/accessibility/`

Accessibility of Code Examples in Documentation JAWS, a Windows screen reader, may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, JAWS may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation This documentation may contain links to Web sites of other companies or organizations that Oracle Corporation does not own or control. Oracle Corporation neither evaluates nor makes any representations regarding the accessibility of these Web sites.

Part I

Security Challenges

Part I explains the wide range of security risks to the integrity and privacy of data.

- Chapter 1, "Data Security Challenges"

1

Data Security Challenges

This chapter presents an overview of data security requirements, and examines the full spectrum of data security risks that must be countered. It then provides a matrix relating security risks to the kinds of technology now available to protect your data. This chapter contains the following sections:

- Top Security Myths
- Understanding the Many Dimensions of System Security
- Fundamental Data Security Requirements
- Security Requirements in the Internet Environment
- A World of Data Security Risks
- A Matrix of Security Risks and Solutions
- The System Security Team

Note: As far as possible, this overview of security technology attempts to present issues independent of the way the technology is implemented. In some instances, however, a technology may only be provided by products from Oracle Corporation. In such cases, the conceptual discussion is from the point of view of the Oracle solution.

Refer to Chapter 9, "Oracle9i Security Products and Features" for a complete discussion of security solutions available from Oracle Corporation.

Top Security Myths

The field of data security is rife with mistaken beliefs which cause people to design ineffective security solutions. Here are some of the most prevalent security myths:

- *Myth: Hackers cause most security breaches.*

  In fact, 80% of data loss is to caused by insiders.

- *Myth: Encryption makes your data secure.*

  In fact, encryption is only one approach to securing data. Security also requires access control, data integrity, system availability, and auditing.

- *Myth: Firewalls make your data secure.*

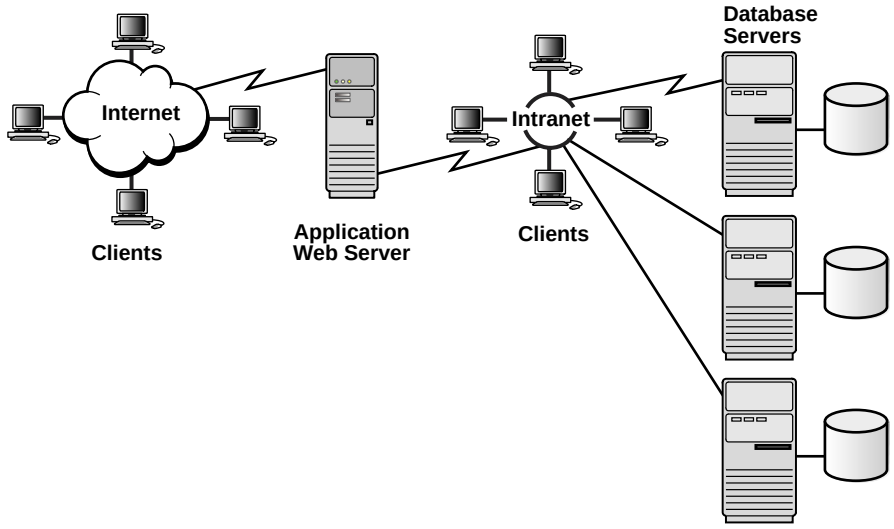
  In fact, 40% of Internet break-ins occur in spite of a firewall being in place.

To design a security solution that truly protects your data, you must understand the security requirements relevant to your site, and the scope of current threats to your data.

Understanding the Many Dimensions of System Security

In an Internet environment, the risks to valuable and sensitive data are greater than ever before. Figure 1–1 presents an overview of the complex computing environment which your data security plan must encompass.

Figure 1–1 Scope of Data Security Needs



You must protect databases and the servers on which they reside; you must administer and protect the rights of internal database users; and you must guarantee the confidentiality of ecommerce customers as they access your database. With the Internet continually growing, the threat to data traveling over the network increases exponentially.

To protect all the elements of complex computing systems, you must address security issues in many dimensions, as outlined in Table 1–1:

Table 1–1 Dimensions of Data Security

Dimension	Security Issues
Physical	Your computers must be physically inaccessible to unauthorized users. This means that you must keep them in a secure physical environment.
Personnel	The people responsible for system administration and data security at your site must be reliable. You may need to perform background checks on DBAs before making hiring decisions.
Procedural	The procedures used in the operation of your system must assure reliable data. For example, one person might be responsible for database backups. Her only role is to be sure the database is up and running. Another person might be responsible for generating application reports involving payroll or sales data. His role is to examine the data and verify its integrity. It may be wise to separate out users' functional roles in data management.
Technical	Storage, access, manipulation, and transmission of data must be safeguarded by technology that enforces your particular information control policies.

Think carefully about the specific security risks to your data, and make sure the solutions you adopt actually fit the problems. In some instances, a technical solution may be inappropriate. For example, employees must occasionally leave their desks. A technical solution cannot solve this physical problem: the work environment must be secure.

Fundamental Data Security Requirements

The following sections describe the basic security standards which technology must ensure:

- Confidentiality
- Integrity
- Availability

Confidentiality

A secure system ensures the confidentiality of data. This means that it allows individuals to see only the data which they are supposed to see. Confidentiality has several different aspects, discussed in these sections:

- Privacy of Communications
- Secure Storage of Sensitive Data
- Authenticated Users
- Granular Access Control

Privacy of Communications

How can you ensure the privacy of data communications? Privacy is a very broad concept. For the individual, it involves the ability to control the spread of confidential information such as health, employment, and credit records. In the business world, privacy may involve trade secrets, proprietary information about products and processes, competitive analyses, as well as marketing and sales plans. For governments, privacy involves such issues as the ability to collect and analyze demographic information, while protecting the confidentiality of millions of individual citizens. It also involves the ability to keep secrets that affect the country's interests.

Secure Storage of Sensitive Data

How can you ensure that data remains private, once it has been collected? Once confidential data has been entered, its integrity and privacy must be protected on the databases and servers where it resides.

Authenticated Users

How can you designate the persons and organizations who have the right to see data? Authentication is a way of implementing decisions about whom to trust. Authentication methods seek to guarantee the identity of system users: that a person is who he says he is, and not an impostor.

Granular Access Control

How much data should a particular user see? Access control is the ability to cordon off portions of the database, so that access to the data does not become an all-or-nothing proposition. A clerk in the Human Relations department might need some access to the `emp` table—but he should not be permitted to access salary information for the entire company. The granularity of access control is the degree to which data access can be differentiated for particular tables, views, rows, and columns of a database.

Note the distinction between authentication, authorization, and access control. Authentication is the process by which a user's identity is checked. When a user is authenticated, he is verified as an authorized user of an application. Authorization is the process by which the user's privileges are ascertained. Access control is the process by which the user's access to physical data in the application is limited, based on his privileges. These are critical issues in distributed systems. For example, if `JAUSTEN` is trying to access the database, authentication would identify her as a a valid user. Authorization would verify her right to connect to the database with Product Manager privileges. Access control would enforce the Product Manager privileges upon her user session.

Integrity

A secure system ensures that the data it contains is valid. Data integrity means that data is protected from deletion and corruption, both while it resides within the database, and while it is being transmitted over the network. Integrity has several aspects:

- System and object privileges control access to application tables and system commands, so that only authorized users can change data.
- Referential integrity is the ability to maintain valid relationships between values in the database, according to rules that have been defined.
- A database must be protected against viruses designed to corrupt the data.
- The network traffic must be protected from deletion, corruption, and eavesdropping.

Availability

A secure system makes data available to authorized users, without delay. Denial-of-service attacks are attempts to block authorized users' ability to access and use the system when needed. System availability has a number of aspects:

Table 1–2 System Availability Aspects

Availability Aspect	Description
Resistance	A secure system must be designed to fend off situations, or deliberate attacks, which might put it out of commission. For example, there must be facilities within the database to prohibit runaway queries. User profiles must be in place to define and limit the resources any given user may consume. In this way the system can be protected against users consuming too much memory or too many processes (whether maliciously or innocently), lest others be prevented from doing their work.
Scalability	System performance must remain adequate regardless of the number of users or processes demanding service.
Flexibility	Administrators must have adequate means of managing the user population. They might do this by using a directory, for example.
Ease of Use	The security implementation itself must not diminish the ability of valid users to get their work done.

Security Requirements in the Internet Environment

The Internet environment expands the realm of data security in several ways, as discussed in these sections:

- Promises and Problems of the Internet
- Increased Data Access
- Much More Valuable Data
- Larger User Communities
- Hosted Systems and Exchanges

Promises and Problems of the Internet

Information is the cornerstone of e-business. The Internet allows businesses to use information more effectively, by allowing customers, suppliers, employees, and partners to get access to the business information they need, when they need it. Customers can use the Web to place orders which can be fulfilled more quickly and with less error, suppliers and fulfillment houses can be engaged as orders are placed, reducing or eliminating the need for inventory, and employees can obtain timely information about business operations. The Internet also makes possible new, innovative pricing mechanisms, such as online competitive bidding for suppliers, and online auctions for customers. These Internet-enabled services all translate to reduced cost: there is less overhead, greater economies of scale, and increased efficiency. The greatest promise of e-business is more timely, more valuable information accessible to more people, at reduced cost of information access.

The promise of e-business is offset by the security challenges associated with the disintermediation of data access. Cutting out the middleman—removing the distributors, wholesalers and retailers from the trading chain—too often cuts out the information security the middleman provides. Likewise, the user community expands from a small group of known, reliable users accessing data from the intranet, to thousands of users accessing data from the Internet. Application hosting providers and exchanges offer especially stringent—and sometimes contradictory—requirements of security by user and by customer, while allowing secure data sharing among communities of interest.

While putting business systems on the Internet offers potentially unlimited opportunities for increasing efficiency and reducing cost, it also offers potentially unlimited risk. The Internet provides much greater access to data, and to more

valuable data, not only to legitimate users, but also to hackers, disgruntled employees, criminals, and corporate spies.

Increased Data Access

One of the chief e-business benefits of the Internet is disintermediation. The intermediate information processing steps which employees typically perform in traditional businesses, such as typing in an order received over the phone or by mail, are removed from the e-business process. Users who are not employees and are thus outside the traditional corporate boundary (including customers, suppliers, and partners) can have direct and immediate online access to business information which pertains to them.

In a traditional office environment, any access to sensitive business information is through employees. Although employees are not always reliable, at least they are known, their access to sensitive data is limited by their job function, and access is enforced by physical and procedural controls. Employees who pass sensitive information outside the company contrary to policy may be subject to disciplinary action. The threat of punishment thus helps prevent unauthorized access.

Making business information accessible by means of the Internet vastly increases the number of users who may be able to access that information. When business is moved to the Internet, the environment is drastically changed. Companies may know little or nothing about the users (including, in many cases, employees) who are accessing their systems. Even if they know who their users are, it may be very difficult for companies to deter users from accessing information contrary to company policy. It is therefore important that companies manage access to sensitive information, and prevent unauthorized access to that information before it occurs.

Much More Valuable Data

E-business relies not only on making business information accessible outside the traditional company, it also depends on making the best, most up-to-date information available to users when they need it. For example, companies can streamline their operations and reduce overhead by allowing suppliers to have direct access to consolidated order information. This allows companies to reduce inventory by obtaining exactly what they need from suppliers when they need it. Companies can also take advantage of new pricing technology, such as online competitive bidding by means of exchanges, to obtain the best price from suppliers, or offer the best price to consumers.

Streamlining information flow through the business system allows users to obtain better information from the system. In the past, data from external partners,

suppliers, or customers was often entered into the system through inefficient mechanisms that were prone to error and delay. For example, many companies accepted the bulk of their orders by phone, letter, or fax, and this information was typed in by clerks or sales people. Even when electronic data interchange mechanisms existed, they were typically proprietary and difficult to integrate with companies' internal data infrastructure. Now, businesses that allow other businesses and consumers to submit and receive business information directly through the Internet can expect to get more timely, accurate, and valuable information, at less expense than if traditional data channels were used.

Formerly, when information was entered into a business system, it was often compartmentalized. Information maintained by each internal department, such as sales, manufacturing, distribution, and finance, was kept separate, and was often processed by physically separate and incompatible databases and applications—so-called "islands of information". This prevented businesses from taking full advantage of the information they already had, since it was difficult for different departments to exchange information when it was needed, or for executives to determine the latest and most accurate status of the business. Companies have found that linking islands of information and consolidating them where possible, allows users to obtain better information, and to get more benefit from that information. This makes the information more valuable.

Improving the value of data available to legitimate users generally improves its value to intruders as well. This increases the potential rewards to be gained from unauthorized access to that data, and the potential damage that can be done to the business if the data were corrupted. In other words, the more effective an e-business system is, the greater the need to protect it against unauthorized access.

Larger User Communities

The sheer size of the user communities which can access business systems by way of the Internet not only increases the risk to those systems, but also constrains the solutions which can be deployed to address that risk. The Internet creates challenges in terms of scalability of security mechanisms, management of those mechanisms, and the need to make them standard and interoperable.

Scalability

Security mechanisms for Internet-enabled systems must support much larger communities of users than systems which are not Internet-enabled. Whereas the largest traditional enterprise systems typically supported thousands of users, many Internet-enabled systems have millions of users.

Manageability

Traditional mechanisms for identifying users and managing their access, such as granting each user an account and password on each system she accesses, may not be practical in an Internet environment. It rapidly becomes too difficult and expensive for system administrators to manage separate accounts for each user on every system.

Interoperability

Unlike traditional enterprise systems, where a company owns and controls all components of the system, Internet-enabled e-business systems must exchange data with systems owned and controlled by others: by customers, suppliers, partners, and so on. Security mechanisms deployed in e-business systems must therefore be standards-based, flexible, and interoperable, to ensure that they work with others' systems. They must support thin clients, and work in multitier architectures.

Hosted Systems and Exchanges

The principal security challenge of hosting is keeping data from different hosted user communities separate. The simplest way of doing this is to create physically separate systems for each hosted community. The disadvantage of this approach is that it requires a separate computer, with separately installed, managed, and configured software, for each hosted user community. This provides little in the way of economies of scale to a hosting company.

Several factors can greatly reduce costs to hosting service providers. These factors include mechanisms which allow multiple user communities to share a single hardware and software instance; mechanisms which separate data for different user communities; and ways to provide a single administrative interface for the hosting provider.

Exchanges have requirements for both data separation and data sharing. For example, an exchange may ensure that a supplier's bid remains unviewable by other suppliers, yet allow all bids to be evaluated by the entity requesting the bid. Furthermore, exchanges may also support communities of interest in which groups of organizations can share data selectively, or work together to provide such things as joint bids.

A World of Data Security Risks

The integrity and privacy of data are at risk from unauthorized users, external sources listening in on the network, and internal users giving away the store. This section explains the risky situations and potential attacks that could compromise your data.

- Data Tampering
- Eavesdropping and Data Theft
- Falsifying User Identities
- Password-Related Threats
- Unauthorized Access to Tables and Columns
- Unauthorized Access to Data Rows
- Lack of Accountability
- Complex User Management Requirements

Data Tampering

Privacy of communications is essential to ensure that data cannot be modified or viewed in transit. Distributed environments bring with them the possibility that a malicious third party can perpetrate a computer crime by tampering with data as it moves between sites.

In a data modification attack, an unauthorized party on the network intercepts data in transit and changes parts of that data before retransmitting it. An example of this is changing the dollar amount of a banking transaction from $100 to $10,000.

In a replay attack, an entire set of valid data is repeatedly interjected onto the network. An example would be to repeat, one thousand times, a valid $100 bank account transfer transaction.

Eavesdropping and Data Theft

Data must be stored and transmitted securely, so that information such as credit card numbers cannot be stolen.

Over the Internet and in Wide Area Network (WAN) environments, both public carriers and private network owners often route portions of their network through insecure land lines, extremely vulnerable microwave and satellite links, or a number of servers. This situation leaves valuable data open to view by any

interested party. In Local Area Network (LAN) environments within a building or campus, insiders with access to the physical wiring can potentially view data not intended for them. Network sniffers can easily be installed to eavesdrop on network traffic. Packet sniffers can be designed to find and steal user names and passwords.

Falsifying User Identities

You need to know your users. In a distributed environment, it becomes more feasible for a user to falsify an identity to gain access to sensitive and important information. How can you be sure that user Pat connecting to Server A from Client B really is user Pat?

In addition, malefactors can hijack connections. How can you be sure that Client B and Server A are what they claim to be? A transaction that should go from the Personnel system on Server A to the Payroll system on Server B could be intercepted in transit and routed instead to a terminal masquerading as Server B.

Identity theft is becoming one of the greatest threats to individuals in the Internet environment. Criminals attempt to steal users' credit card numbers, and then make purchases against the accounts. Or they steal other personal data, such as checking account numbers and driver's license numbers, and set up bogus credit accounts in someone else's name.

Nonrepudiation is another identity concern: how can a person's digital signature be protected? If hackers steal someone's digital signature, that person may be held responsible for any actions performed using their private signing key.

Password-Related Threats

In large systems, users must remember multiple passwords for the different applications and services that they use. For example, a developer can have access to a development application on a workstation, a PC for sending e-mail, and several computers or intranet sites for testing, reporting bugs, and managing configurations.

Users typically respond to the problem of managing multiple passwords in several ways:

- They may select easy-to-guess passwords—such as a name, fictional character, or a word found in a dictionary. All of these passwords are vulnerable to dictionary attacks.
- They may also choose to standardize passwords so that they are the same on all machines or Web sites. This results in a potentially large exposure in the event

of a compromised password. They can also use passwords with slight variations that can be easily derived from known passwords.

- Users with complex passwords may write them down where an attacker can easily find them, or they may just forget them—requiring costly administration and support efforts.

All of these strategies compromise password secrecy and service availability. Moreover, administration of multiple user accounts and passwords is complex, time-consuming, and expensive.

Unauthorized Access to Tables and Columns

The database may contain confidential tables, or confidential columns in a table, which should not be available indiscriminately to all users authorized to access the database. It should be possible to protect data on a column level.

Unauthorized Access to Data Rows

Certain data rows may contain confidential information which should not be available indiscriminately to users authorized to access the table.

You need granular access control—a way to enforce confidentiality on the data itself. For example, in a shared environment businesses should only have access to their own data; customers should only be able to see their own orders. If the necessary compartmentalization is enforced upon the data, rather than added by the application, then it cannot be bypassed by users.

Systems must therefore be flexible: able to support different security policies depending on whether you are dealing with customers or employees. For example, you may require stronger authentication for employees (who can see more data) than you do for customers. Or, you may allow employees to see all customer records, while customers can only see their own records.

Lack of Accountability

If the system administrator is unable to track users' activities, then users cannot be held responsible for their actions. There must be some reliable way to monitor who is performing what operations on the data.

Complex User Management Requirements

Systems must often support thousands of users, or hundreds of thousands of users: thus they must be scalable. In such large-scale environments, the burden of managing user accounts and passwords makes your system vulnerable to error and attack. You need to know who the user really is—across all tiers of the application—to have reliable security.

Multitier Systems

This problem becomes particularly complex in multitier systems. Here, and in most packaged applications, the typical security model is that of One Big Application User. The user connects to the application, and the application (or application server) logs on and provides complete access for everyone, with no auditing and unlimited privileges. This model places your data at risk—especially in the Internet, where your Web server or application server depends upon a firewall. Firewalls are commonly vulnerable to break-ins.

Scaling the Security Administration of Multiple Systems

Administration of hundreds of thousands of users is difficult enough on a single system. This burden is compounded when security must be administered on multiple systems.

To meet the challenges of scale in security administration, you should be able to centrally manage users and privileges across multiple applications and databases by using a directory based on industry standards. This can reduce system management costs and increase business efficiency.

Further, creating and building separate databases for multiple application subscribers is not a cost-efficient model for an application service provider. While technically possible, the separate database model would quickly become unmanageable. To be successful, a single application installation should be able to host multiple companies—and be administered centrally.

A Matrix of Security Risks and Solutions

Table 1–3 relates security risks to the technologies which address them, and to the corresponding Oracle products.

Table 1–3 Matrix of Security Risks and Solutions

Problem	Solution	Security Technology	Oracle Products and Features
Unauthorized users	Know your users	Authentication	Oracle9*i* Standard Edition, and Oracle9*i* Enterprise Edition: Passwords, Password management Oracle Advanced Security: Tokens, smart cards, Kerberos, and so on. PKI: X.509 Certificates
Unauthorized access to data	Limit access to data	Access control	Oracle9*i* Standard Edition Oracle9*i* Enterprise Edition: Virtual Private Database feature
	Dynamic query modification	Fine-grained access control	Oracle9*i* Enterprise Edition: Virtual Private Database feature
	Limit access to data rows and columns	Label-based access control	Oracle Label Security
	Encrypt data	Data encryption	Oracle9*i* Standard Edition, and Oracle9*i* Enterprise Edition
	Limit privileges	Privilege management	Oracle9*i* Standard Edition: Roles, Privileges Oracle9*i* Enterprise Edition: Secure Application Roles Oracle Advanced Security: Enterprise Roles
Eavesdropping on communications	Protect the network	Network encryption	Oracle Advanced Security: Encryption Secure Sockets Layer
Corruption of data	Protect the network	Data integrity	Oracle Advanced Security: Checksumming PKI: Checksumming (as part of SSL)

Table 1–3 Matrix of Security Risks and Solutions(Cont.)

Problem	Solution	Security Technology	Oracle Products and Features
Denial of service	Control access to resources	Availability	Oracle9*i* Standard Edition and Oracle9*i* Enterprise Edition: User Profiles
Complexity to user	Limit number of passwords	Single signon	Oracle Advanced Security: Kerberos, DCE, Enterprise User Security Login Server: Web-Based SSO
Complexity to administrator	Centralize management	Enterprise user security	Oracle Advanced Security: Directory Integration Oracle Internet Directory
Lack of accountability	Monitor users' actions	Auditing	Oracle9*i* Standard Edition: Auditing Oracle9*i* Enterprise Edition: Standard Auditing, Fine-Grained Auditing.
Overly broad access to data	Dynamic query modification	Fine-grained access control	Oracle9*i* Enterprise Edition: Virtual Private Database Oracle Label Security
Too many accounts	Centralize management	Directory services, LDAP-compliant directory services	Oracle Internet Directory
Operating system break-in	Encrypt sensitive data	Stored data encryption	Oracle9*i* Standard Edition and Oracle9*i* Enterprise Edition: Data encryption

The System Security Team

Complex data security systems require a team of people to ensure security at a particular site. Table 1–4 introduces the types of administrators who may be involved.

Table 1–4 The System Security Team

Person	Responsibilities
User	Responsible for using the system for legitimate purposes, protecting sensitive data to which she has access, and managing her passwords securely.
Database Administrator	Responsible for creating and administering database users, granting system and object privileges, and assigning local roles to users.
Operating System Administrator	Responsible for maintaining the underlying security of the operating system.
Network Administrator	Responsible for ensuring the security of data in transmission.
Application Administrators	Responsible for deploying applications in such a way as to ensure security.
Trusted Application Administrator	Responsible for creating and administering users of trusted applications, and their associated privileges.
Enterprise Security Manager	Responsible for maintaining the security of the directory, and for implementing centralized enterprise user security.

Part II

Technical Solutions to Security Risks

Part II introduces the technology available to meet data security challenges.

- Chapter 2, "Protecting Data Within the Database"
- Chapter 3, "Protecting Data in a Network Environment"
- Chapter 4, "Authenticating Users to the Database"
- Chapter 5, "Using and Deploying a Secure Directory"
- Chapter 6, "Administering Enterprise User Security"
- Chapter 7, "Auditing to Monitor System Security"
- Chapter 8, "The Public Key Infrastructure Approach to Security"

2

Protecting Data Within the Database

Data is vulnerable at many points in any computer system, and many security techniques and types of functionality can be employed to protect it. This chapter provides a systematic introduction to security features that can protect the memory, files, and processes residing on the server. It contains the following sections:

- Introduction to Database Security Concepts
- System and Object Privileges
- Managing System and Object Privileges
- Row Level Security
- Encrypting Data on the Server
- Database Integrity Mechanisms
- System Availability Factors
- Secure Configuration Practices

Note: As far as possible, this overview of security technology attempts to present issues independent of the way the technology is implemented. In some instances, however, a technology may only be provided by products from Oracle Corporation. In such cases, the conceptual discussion is from the point of view of the Oracle solution.

Refer to Part III, "Oracle9i Security Products" for a complete discussion of security solutions available from Oracle Corporation.

Introduction to Database Security Concepts

Confidentiality, integrity, and availability are the hallmarks of database security. Who should have the right to access data? What portion of all the data should a particular user be able to access? What operations should an authorized user be able to perform on the data? Can authorized users access valid data when necessary?

Authorization is permission given to a user, program, or process to access an object or set of objects.The type of data access granted to a user can be read-only, or read / write. Privileges specify the type of Data Manipulation Language (DML) operations which the user can perform upon data.

This chapter introduces these and other fundamental concepts of database security.

System and Object Privileges

A privilege is permission to access a named object in a prescribed manner; for example, permission to query a table. Privileges are granted to users at the discretion of other users (administrators). Privileges can be granted to enable a particular user to connect to the database (create a session); create a table in his own schema; select rows from someone else's table; or execute someone else's stored procedure.

The following sections describe the two distinct categories of privileges within a database:

- System Privileges
- Schema Object Privileges

See Also: "Privileges" on page 9-4

System Privileges

System privileges allow users to perform a particular systemwide action or a particular action on a particular type of schema object. For example, the privileges to create a tablespace or to delete the rows of any table in the database are system privileges. Many system privileges are available only to administrators and application developers because the privileges are very powerful.

Schema Object Privileges

Access to data is most commonly controlled on the level of access to the database itself, or to specific tables. Schema object privileges allow users to perform a particular action on a specific schema object. For example, the privilege to delete rows of a specific table is an object privilege.

Schema object privileges for tables allow table security at the level of data manipulation language (DML) and data dictionary language (DDL) operations. For example, an administrator can grant individual users the privileges to use the DML operations `DELETE`, `INSERT`, `SELECT`, and `UPDATE` on a table or view, or to use the `ALTER`, `INDEX`, and `REFERENCES` privileges to perform DDL operations on a table.

Privileges can be specified at the column level. It is possible to restrict a user's `INSERT` and `UPDATE` privileges for a table to individual columns of the table. Likewise, privileges can be specified at the row level. It is possible to restrict a user's `SELECT`, `INSERT`, `UPDATE`, and `DELETE` privileges for a table to specific rows of the table.

As a general rule, object privileges can only be granted by the object owner. However, an owner can also specify that a particular user has the right to grant a privilege to others. The full range of privileges for any action on any object in the schema is typically granted by default to the administrator. Even this complete set can be delegated by the administrator to other users, that is, selectively granted to or revoked from any user or group. In particular, an administrator can grant an application developer or DBA the privilege to `GRANT ANY OBJECT PRIVILEGE`. Having this privilege can make it easier for the developer to do the security configuration tasks he faces, and aid the DBA in resolving access control problems as they arise.

Managing System and Object Privileges

The user's ability to supply a valid username and password can be used as a first level of authorization for a user to access a database or specific database tables. The following sections discuss several additional techniques that can be used to further manage system and object privileges:

- Using Roles to Manage Privileges
- Using Stored Procedures to Manage Privileges
- Using Network Facilities to Manage Privileges
- Using Views to Manage Privileges

Using Roles to Manage Privileges

A role mechanism can be used to provide authorization. A single person or a group of people can be granted a role or a group of roles. One role can be granted in turn to other roles. By defining different types of roles, administrators can manage access privileges much more easily.

This section contains these topics:

- Database Roles
- Global Roles
- Enterprise Roles
- Secure Application Roles

See Also: "Roles" on page 9-5

Database Roles

Privileges enable users to access and modify data in the database. Database roles are named groups of privileges relating to a specific job function that are granted to users or other roles. Because roles allow for easier and better management of privileges, privileges are normally granted to roles and not to specific users. You can selectively enable or disable the roles granted to a user. This allows specific control of a user's privileges in any given situation. For example, you can protect role use with a password. Applications can be created specifically to enable a role when supplied the correct password; that way, users cannot enable the role if they do not know the password.

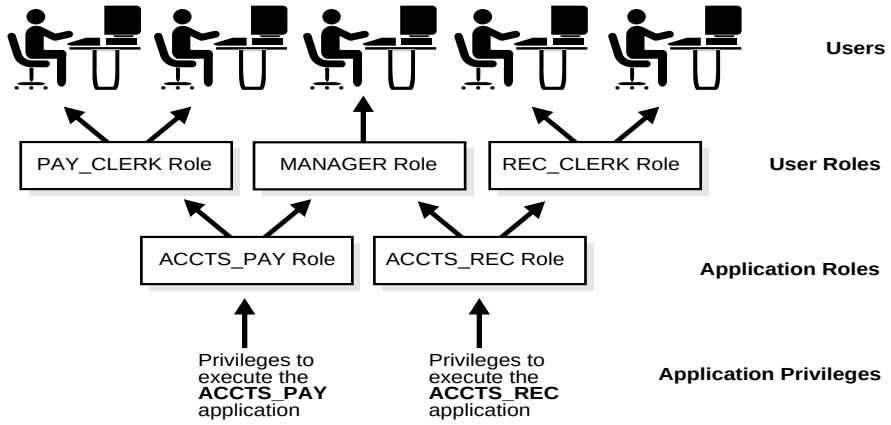
The following properties of roles allow for easier privilege management:

- Reduced granting of privileges: Rather than explicitly granting the same set of privileges to many users, a database administrator can grant the privileges for a group of related users to a role. The database administrator can then grant the role to each member of the group.
- Dynamic privilege management: When the privileges of a group must change, only the privileges of the role need to be modified. Security domains of all users who are granted the group role automatically reflect the changes made to the role.
- Selective availability of privileges: The roles granted to a user can be selectively enabled (available for use) or disabled (not available for use). This allows specific control of a user's privileges in any given situation.

- Application awareness: A database application can be designed to enable and disable selective roles automatically when a user attempts to use the application.

By using various levels of roles and privileges, you can achieve increased granularity of access controls and adhere to the principle of least privilege, as illustrated in Figure 2–1. Here, each individual has only the privileges necessary to perform his or her job.

Figure 2–1 Common Uses for Roles



Global Roles

Global roles are one component of enterprise user security. A global role only applies to one database, but it can be granted to an enterprise role defined in the enterprise directory. Although a global role is managed in a directory, its privileges are contained within a single database—the database in which it is defined.

You define the global role locally in the database by granting privileges and roles to it, but you cannot actually grant the global role to any user or to any other role in the database. When an enterprise user attempts to connect to the database, the directory is queried to obtain any global roles associated with the user.

Enterprise Roles

An enterprise role is a directory structure which can contain global roles on multiple databases, and which can be granted to enterprise users. By storing and

managing enterprise roles in an LDAP-based directory service, you can centralize management of user-related information, including authorizations.

For example, the enterprise role `clerk` could contain the global role `hrclerk` with its unique privileges on the Human Resources database, and the `analyst` role with its unique privileges on the Payroll database.

An enterprise role can be granted to or revoked from one or more enterprise users. For example, you could grant the enterprise role `clerk` to a number of enterprise users who hold the same job. This information is protected in the directory, and only you, as the administrator, can manage users and grant and revoke their roles.

A user can be granted local roles and privileges in a database, in addition to enterprise roles.

See Also: Chapter 6, "Administering Enterprise User Security"

Secure Application Roles

A long-standing security problem has been that of limiting how users access data, to prevent users from bypassing application logic to access data directly. For example, in web-based applications, even if users are known to the database, it may not be desirable to allow them to have direct access to data. To date, this has been a very difficult security problem to solve, because there has been no secure way to validate which application is used to access data. For example, a malicious user could write a program that *appears* to be a valid human resources application.

One way to address this challenge is through a secure application role: a role implemented by a package. The package can perform any desired validation to ensure that the appropriate conditions are met before the user can exercise privileges granted to the role in the database. The database ensures that it is only the trusted package implementing the role that determines the correct access conditions.

A secure application role is used by an application, can only be enabled by the application, and does not need a password.

See Also: "Secure Application Role" on page 9-21

Using Stored Procedures to Manage Privileges

Through stored procedures you can restrict the database operations that users can perform. You can allow them to access data only through procedures and functions that execute with the definer's privileges. For example, you can grant users access to a procedure that updates a table, but not grant them access to the table itself. When

a user invokes the procedure, the procedure executes with the privileges of the procedure's owner. Users who have only the privilege to execute the procedure (but not the privileges to query, update, or delete from the underlying tables) can invoke the procedure, but they cannot manipulate table data in any other way.

See Also: "Views, Stored Program Units, Triggers" on page 9-5

Using Network Facilities to Manage Privileges

Database roles can potentially be mapped to external services (such as DCE groups and RADIUS authorizations) so that you can centrally manage and administer privileges for all network resources—of which databases are only one piece.

See Also: Chapter 3, "Protecting Data in a Network Environment"

"Oracle Net Services" on page 9-54

Using Views to Manage Privileges

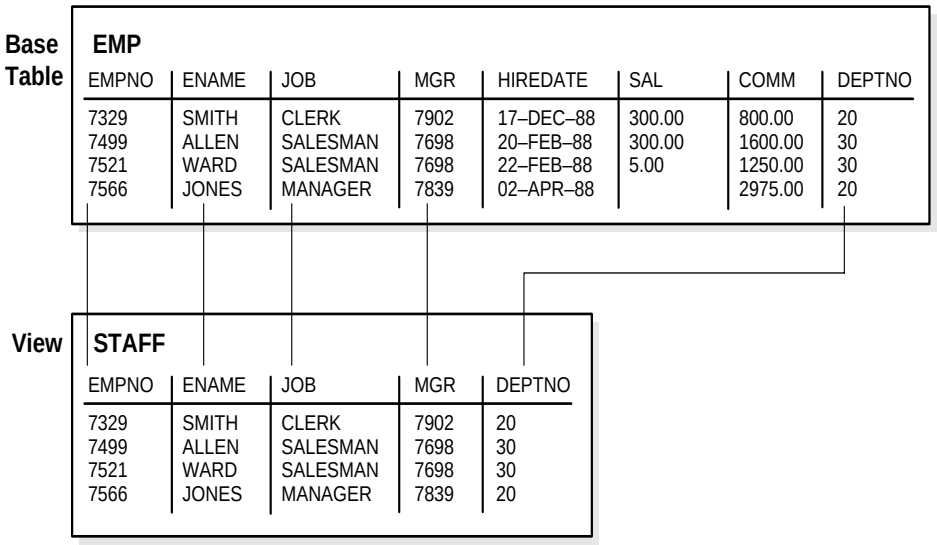
Rather than granting users privileges on a particular table, you can give them access to a view of the table. Views add two more levels of security:

- A view can limit access to only selected columns of the base table.
- A view can provide value-based security for the information in a table. Thus a `WHERE` clause in the definition of a view can display only selected rows of a base table.

To use a view requires appropriate privileges only for the view itself. The user need not be given privileges on base objects underlying the view.

Figure 2–2 shows an example of a view called `staff` derived from the base table `emp`. Notice that the view shows only five of the columns in the base table.

Figure 2–2 An Example of a View



See Also: "Views, Stored Program Units, Triggers" on page 9-5

Row Level Security

A much more granular form of data access is row level access. For any table with data, access to particular rows can be based on such considerations as the department to which employees belong, their job responsibility or title, or other significant factors. In the past, complex and dynamic views have been used to implement row level security. There are, however, two more effective approaches to this problem: Virtual Private Database (VPD), in which you create your own implementation of row level security; and label-based access control, in which you customize a ready-made VPD policy to accomplish this. This section describes these alternative approaches.

- Complex and Dynamic Views

- Application Query Rewrite: Virtual Private Database
- Label-Based Access Control

Complex and Dynamic Views

Complex views and dynamic views are among the historical approaches to row level security. Complex view definitions result when application designers build their own user security tables and join the application tables with the new security table based on the name of the application user. This approach usually requires many complex view definitions which must be maintained as security requirements change. Another approach is dynamic view creation. This approach uses dynamic DDL execution utilities to define new view definitions based on the identity of the application user. Using dynamic views, however, is costly and time consuming.

Application Query Rewrite: Virtual Private Database

Virtual Private Database is the ability to perform query modification based on a security policy you have defined in a package, and associated with a table, view, or synonym. Virtual private database provides fine-grained access control which is data-driven, context-dependent, and row-based. It is a key enabling technology in building three-tier systems which expose mission-critical resources to customers and partners.

See Also: "Virtual Private Database in Oracle9i" on page 9-14

Label-Based Access Control

Label-based access control allows organizations to assign sensitivity labels to data rows, control access to data based on those labels, and ensure that data is marked with the appropriate sensitivity label. The most familiar example of this is perhaps the security classification system used by the United States and other governments. In this model, hierarchical classification labels such as CONFIDENTIAL, SECRET, or TOP SECRET are assigned to data based on the sensitivity level of the information. In addition, formal security compartments are defined, such as NATO or CRYPTO, and assigned to the data. Access to data labeled at a certain level (such as SECRET) is restricted to those users who have been granted that level of access or higher. Access to data in a specific compartment (such as CONFIDENTIAL NATO) is restricted to those who have access to the appropriate level, as well as explicit permission to access the compartment in question.

While e-businesses do not typically have label data classification systems, they almost always have data labeling requirements. For example, an e-business may differentiate between Company Confidential information and Public information. Further, there may be some Company Confidential information that can be shared with partners, under a Confidential Disclosure Agreement or other legal document, while other information is only accessible by certain groups within the company (such as Finance or Sales divisions). The ability to natively manage labeled data is a tremendous advantage for e-businesses in providing the right information to the right people at the right level of secure data access.

See Also: "Oracle Label Security" on page 9-47

Encrypting Data on the Server

Encryption is a technique of encoding data, so that only authorized users can understand it. Encryption alone, however, is not sufficient to secure your data. Protecting data in the database includes access control, data integrity, encryption, and auditing. This section includes:

- Selective Encryption of Stored Data
- Industry Standard Encryption Algorithms

Selective Encryption of Stored Data

For certain applications, you may decide to encrypt data as an additional measure of security. Most issues of data security can be handled by appropriate authentication and access control, ensuring that only properly identified and authorized users can access data. Data in the database, however, cannot normally be secured against the database administrator's access, since a DBA has all privileges. Likewise, organizations may have concerns about securing sensitive data stored offline, such as backup files stored with a third party. They may want to guard against intruders accessing the data where it is physically stored on the database.

Although encryption is not a substitute for effective access control, you can obtain an additional measure of security by selectively encrypting sensitive data before it is stored in the database. Information which may be especially sensitive and warrant encryption could include credit card numbers, national identity numbers in countries with strict privacy laws, or trade secrets, such as industrial formulas. Applications for which a user is authenticated to the application, rather than to the database, may also use encryption to protect the application user password or cookie.

Industry Standard Encryption Algorithms

A number of industry-standard encryption algorithms are useful for the encryption and decryption of data on the server. Two of the most popular are:

Data Encryption Standard (DES)	Provides standards-based encryption for data privacy
Triple DES (3DES)	Encrypts message data with three passes of the DES algorithm

Note that the RC4 encryption algorithm is a stream cipher, and therefore not suitable for encryption in the database. It is useful for network encryption.

See Also: "Encryption Algorithms" on page 3-5

"Java Security Implementation in the Database" on page 9-23

"Oracle Net Services" on page 9-54

Database Integrity Mechanisms

Database integrity ensures that data in the database is correct and consistent. Database integrity mechanisms can be divided into those mechanisms that support system integrity, and those that enforce relational database integrity properties (like entity integrity, referential integrity, transaction integrity, and business rules).

Traditional system integrity involves ensuring that the data inserted into the system is the same as the contents of that data when it is subsequently retrieved. Further, data must not be altered or deleted by a user who is not authorized to do so.

A database must ensure that data adheres to certain business rules, as determined by the database administrator or application developer. For example, assume that a business rule says that no employee in the `emp` table can receive a raise greater than 20% of the value in the `salary` column. If an `insert` or `update` statement attempts to violate this integrity rule, then the statement must fail. Integrity constraints and database triggers can be used to manage a database's data integrity rules.

Referential integrity provides that a rule defined on a column or set of columns in one table, match the values in a related table (the referenced value). Referential integrity also includes the rules that dictate what types of data manipulation are allowed on referenced values and how these actions affect dependent values. Referential integrity rules can be used to provide for these relationships.

See Also: "Integrity" on page 9-3

System Availability Factors

Data security also involves the accessibility of the data to authorized users, as needed. Availability is often thought of as continuity of service, ensuring that a database is available 24 hours a day, 7 days a week. However, there are security aspects to availability. For example, if a user is able to manipulate system resources in order to deny their availability to other users, he is breaching security. This is referred to as **denial of service**.

System availability can be protected by factors such as these:

Table 2–1 System Availability Factors

Factor	Description
Storage quotas	The administrator should be able to direct and limit the use of disk space allocated to the database for each user, including default and temporary tablespaces and tablespace quotas.
Resource limits	Each user should be assigned a profile that limits the system resources available to the user, including the number of concurrent sessions the user can establish, the CPU processing time available to the user's session, the amount of logical I/O available to the user, and so on.
Hot backups	Data should be copied as a safeguard against unexpected data loss and application errors. If you lose the original data, then you can reconstruct it by using a backup.
Resistance to attack	Software should be written according to secure coding standards.
Secure configuration	The system should be set up in such a way as to avoid exposing any vulnerabilities which could be exploited by a malicious intruder.
Parallel systems	Clusters can be used to ensure highly available access to queue data. Queues can be implemented by using database tables. In case of an instance failure, messages managed by the failed instance can be processed immediately by one of the surviving instances.

See Also: "High Availability" on page 9-6

Secure Configuration Practices

Finally, any techniques you employ to assure data security can be rendered useless unless the database administrator follows good security practices. For example, the DBA should always:

- Revoke privileges from public accounts
- Lock unused accounts
- Change any default passwords after installation
- Set the proper permissions. You may implement roles to manage privileges, but if permissions are incorrectly set in the operating system, there may be a security loophole.

See Also: *Oracle9i Database Administrator's Guide*

Oracle9i Application Developer's Guide - Fundamentals

3

Protecting Data in a Network Environment

This chapter explains how data can be protected while being transmitted over a network. It contains these sections:

- Introduction to Data Protection in a Network Environment
- Protecting Data During Transmission
- Ensuring Security in Three-Tier Systems

Introduction to Data Protection in a Network Environment

Security issues become more complex in a network environment. You must ensure that access to the network is controlled, and that data is not vulnerable to attack during transmission across the network. Many technologies are available to encrypt data and thus help to ensure its privacy and integrity. They ensure that:

- Data remains confidential.
- Data cannot be modified.
- Data cannot be replayed.
- Lost packets can be detected.

When multitier systems are involved, network access becomes even more complex. Users may access the network from a middle tier, in which case only the middle tier may be known to the database: the individual user's authorizations may be lost. To ensure confidentiality, the database must be able to identify the actual user who is accessing it from a middle tier.

> **See Also:** "Network Security Services of Oracle Advanced Security" on page 9-27
>
> "Data Integrity Features of Oracle Advanced Security" on page 9-29
>
> "Oracle Net Services" on page 9-54

Protecting Data During Transmission

The following sections describe the technology available to ensure data privacy and integrity during transmission:

- Controlling Access Within the Network
- Encrypting Data for Network Transmission
- Secure Sockets Layer (SSL) Protocol
- Firewalls

Controlling Access Within the Network

This section describes different ways to control access within the network.

Middle-Tier Connection Management

You can configure a middle tier that manages the connections of very large user populations. To support a large number of users, you can configure multiple instances of Oracle Connection Manager. This product multiplexes multiple client network sessions through a single network connection to the database, increasing the total number of connections.

It is also possible to filter on source, destination, and host name. Thus you can ensure that connections only come from a physically secure terminal, or from an application Web server with a known IP address. (IP address alone is not enough for authentication, since it can be faked.) In this way you could allow connections from IP address `foo`, connecting to host `bar` for `payroll`.

See Also: "Oracle Connection Manager" on page 9-55

Native Network Capabilities (Valid Node Checking)

In the case of a sensitive database, you may want to ensure that connections only come from certain points in the network. For example, a company might have a security policy saying that user `jausten` can access the payroll database, but only when she is present at work.

See Also: "Valid Node Checking in Oracle Net Services" on page 9-58

Database Enforced Network Access

You can also use Virtual Private Database (or secure application role) to limit access to the database from particular network nodes. Note that you would not want to make IP address a primary way of authenticating or authorizing users, since IP addresses can be faked. However, you can use IP address as an additional means of limiting data access for otherwise authorized users. For example, user Jane may have access to the `emp` table, but company policy may dictate that she is not allowed to access employee data unless she is inside the corporate intranet—perhaps even from a particular subnet for the HR department.

See Also: "Virtual Private Database in Oracle9i" on page 9-14

"Database-Enforced VPD Network Access" on page 9-58

Encrypting Data for Network Transmission

Sensitive information that travels over an intranet or the Internet can be protected by encryption. Encryption is the mutation of information into a form readable only with a decryption key. Encryption is a powerful security mechanism because it can make decryption mathematically infeasible if you do not possess the decryption key.

Consider, for example, an Internet buyer who wishes to purchase a company's product by using a credit card in a secure fashion. The buyer's credit card number is encrypted with an encryption key. The encrypted credit card number is sent across the network to the database. Encryption scrambles the message, rendering it unreadable to anyone but the recipient. The server decrypts the message with a decryption key and reads the credit card number.

Note that the secrecy of encrypted data depends on the existence of a secret key shared between the communicating parties. Providing and maintaining such secret keys is known as key management. In a multiuser environment, secure key distribution may be difficult; public key cryptography was invented to solve this problem.

Encryption must address all communications with the database, including transmissions from clients and transmissions from middle tiers. It must also secure all protocols into the database.

Encryption Algorithms

Table 3–1 lists encryption algorithms that have become industry standard for the encryption and decryption of data.

Table 3–1 Encryption Algorithms

Algorithm	Characteristics
RSA Data Security RC4	Allows high-speed encryption for data privacy. By using a secret, randomly generated key unique to each session, all network traffic is fully safeguarded—including all data values, SQL statements, and stored procedure calls and results. The client, server, or both, can request or require the use of the encryption module to guarantee that data is protected.
Data Encryption Standard (DES)	Uses symmetric key cryptography to safeguard network communications. DES is required for financial institutions and many other institutions.
Triple DES (3DES)	Encrypts message data with three passes of the DES algorithm. 3DES provides a high degree of message security. However, it entails a performance penalty, the magnitude of which is dependant upon on the speed of the processor performing the encryption. 3DES typically takes three times as long to encrypt a data block as compared with the standard DES algorithm.

See Also: "Selective Encryption of Stored Data" on page 2-10

"Network Security Services of Oracle Advanced Security" on page 9-27

Data Integrity Checking

In addition to encryption, there are integrity algorithms which can ensure that data has not been tampered with or packets replayed. A database can use these algorithms to detect corruption in data blocks. Table 3–2 lists industry standard integrity algorithms.

Table 3–2 Integrity Algorithms

Algorithm	Characteristics
MD5 Checksum	Provides data integrity through hashing and sequencing to assure that data is not altered or stolen as it is transmitted over a network
Secure Hash Algorithm (SHA)	Similar to MD5, but produces a larger message digest, for greater security

Secure Sockets Layer (SSL) Protocol

The Secure Sockets Layer (SSL) protocol, developed by Netscape Corporation, is an industry-accepted standard for network transport layer security. SSL is supported by all currently available Web servers and Web browsers. It is also gaining acceptance for other protocols, including LDAP and IMAP. The SSL protocol provides authentication, data encryption, and data integrity, in a public key infrastructure (PKI).

SSL addresses the problem of protecting user data exchanged between tiers in a three-tier system. By providing strong, standards-based encryption and integrity algorithms, SSL provides system developers and users with confidence that data will not be compromised in the Internet. 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 an application Web server before they provide the server with sensitive information, such as credit card numbers.

See Also: "Secure Sockets Layer Authentication and X.509v3 Digital Certificates" on page 8-7

"Secure Sockets Layer (SSL) Encryption Capabilities" on page 9-29

"Secure Sockets Layer (SSL) Authentication in Oracle Advanced Security" on page 9-35

Firewalls

To eliminate potential weak points in the network infrastructure, you may opt to pass data from protocol to protocol without the complexity of decryption and re-encryption. To do so securely, you must have some way to securely transfer data across network protocol boundaries.

The Internet enables you to connect your corporate intranet to a broad public network. Although this capability provides enormous business advantages, it also entails risk to your data and your computer system. One way of protecting the privacy and integrity of your system is to place a firewall between the public network and your intranet.

A firewall is a single point of control on a network, used to prevent unauthorized clients from reaching the server. It acts as a filter, screening out unauthorized network users from using the intranet. It does this by enforcing access controls based on the contents of the packets of data being transmitted, and can thus protect against attacks on individual protocols or applications. Firewalls are rule-based. They have a list of rules that define which clients can connect, and which cannot. They can compare the client's host name or IP name with the rules, and either grant the client access, or not.

See Also: "Firewall Support with Oracle Net Services" on page 9-56

Ensuring Security in Three-Tier Systems

The following sections discuss security issues in multitier systems:

- Proxy Authentication to Ensure Three-Tier Security
- Java Database Connectivity (JDBC)

Proxy Authentication to Ensure Three-Tier Security

An important security feature for three-tier systems is the ability to proxy authenticated user identity from a middle tier to the database. This feature (also known as *n*-tier authentication) enables you to identify the real user who is accessing the database through a middle tier. It authenticates users and machines by way of a database password or other credential, without the overhead of a separate database connection. It protects data on the server by ensuring that unauthorized users cannot access data on the server over the Internet or through a middle tier. It ensures accountability by keeping track of what users have logged on to an application through an intermediate tier, so that it is possible to trace who has done what in an application. Scalability is further improved through the introduction of support for enterprise users.

See Also: "Proxy Authentication and Authorization" on page 4-8

"Proxy Authentication in Oracle9i" on page 9-9

Java Database Connectivity (JDBC)

You can use Java to transmit data securely in a three-tier environment. Java is the language of the Internet, and also the language of OLAP applications. Application developers use Java to build applications and applets. As an object-oriented, platform-independent, network-based, and secure language, Java is fast superseding C++ and Visual Basic as the language of choice for application developers.

JDBC (Java Database Connectivity) is an industry-standard API (Applications Programming Interface) which allows Java programs to send SQL statements to an object-relational database such as Oracle. JDBC enables a middle tier server to access a database on behalf of a client user by establishing a lightweight session for the user.

Java applets can thus transmit data over secure channels. You can have secure connections from middle tier servers with Java Server Pages (JSPs) to the database. This enhances security because:

- Every protocol can be secured.
- JDBC-Oracle Call Interface and thin clients can be supported.
- Two-tier and three-tier architectures can be supported.

There are two ways to implement Java security and negotiate algorithms:

- Hard code it into your JDBC client application
- Configure it like native network cryptography

JDBC-Oracle Call Interface Driver

The JDBC-Oracle Call Interface (JDBC-OCI) driver can be used for client side use with an Oracle client installation.

JDBC Thin Driver

The JDBC Thin driver is a Type 4 (100% pure Java) driver that uses Java sockets to connect directly to a database server. It has a lightweight Java implementation of Oracle Net called Java Net.

The Thin driver does not require Oracle software on the client side. It does need a TCP/IP listener on the server side. Use this driver in regular Oracle Net listener Java applets that are downloaded into a Web browser. The Thin driver is self-contained, but it opens a Java socket, and thus can only run in a browser that supports sockets.

See Also: Java Encryption Features of Oracle Advanced Security on page 9-30

4

Authenticating Users to the Database

Authentication of user identity is imperative in distributed environments. Without it, there can be little confidence in network or database security. This chapter contains these sections:

- Introduction to User Authentication
- Passwords for Authentication
- Strong Authentication
- Proxy Authentication and Authorization
- Single Signon

Introduction to User Authentication

A basic security requirement is that you know your users: you must first identify users before you can determine their privileges and access rights. You must know who a user is so that you can audit his or her actions upon the data.

Users can be authenticated in a number of different ways before they are allowed to create a database session. In database authentication, you can define users such that the database performs both identification and authentication of users. In external authentication, you can define users such that authentication is performed by the operating system or network service. Alternatively, you can define users such that they are authenticated by the Secure Sockets Layer (SSL). For enterprise users, an enterprise directory can be used to authorize their access to the database through enterprise roles. Finally, you can specify users who are allowed to connect through a middle-tier server. The middle-tier server authenticates and assumes the identity of the user and is allowed to enable specific roles for the user. This is called proxy authentication.

Passwords for Authentication

Passwords are one of the basic forms of authentication. A user must provide the correct password when establishing a connection to prevent unauthorized use of the database. In this way, users attempting to connect to a database can be authenticated by using information stored in that database. Passwords are assigned when users are created. A database can store a user's password in the data dictionary in an encrypted format. Users can change their passwords at any time.

Database security systems that are dependent on passwords require that passwords be kept secret at all times. But, passwords are vulnerable to theft, forgery, and misuse. A number of steps can strengthen the basic password feature and provide greater control over database security:

- Password management policy can be controlled by DBAs and security officers through user profiles.
- The DBA can establish standards for password complexity, such as minimum password length.
- Passwords should be words that cannot be found in the dictionary. They should not consist of people's names or birthdates.
- Passwords can be timed out, expiring after a certain amount of time. This requires users to change the value periodically.
- Reuse of passwords can be prohibited for a certain amount of time.

- When a particular user exceeds a designated number of failed login attempts, the server can automatically lock that user's account.

Strong Authentication

Having a central facility authenticate all members of the network (clients to servers, servers to servers, users to both clients and servers) is one effective way to address the threat of nodes on a network falsifying their identities. Strong authentication can also be established by using two-factor authentication: the combination of something a user knows (such as a PIN), and something the user has (such as a token card).

Strong authentication has important advantages:

- More choices of authentication mechanism are available, such as smart cards, Kerberos, or the operating system.
- Many network authentication services, such as Kerberos and DCE, support single signon. This means that users have fewer passwords to remember.
- If you already use some external mechanism for authentication, then there may be less administrative overhead to use that mechanism with the database as well.

This section describes the following strong authentication methods that can be used in a distributed environment:

- Kerberos and CyberSafe
- RADIUS
- Token Cards
- Smart Cards
- Distributed Computing Environment (DCE)
- Biometrics
- PKI and Certificate-Based Authentication

See Also: "Authentication and Access Controls in Oracle9i" on page 9-4

Kerberos and CyberSafe

Kerberos is a trusted third-party authentication system that was created by the Massachusetts Institute of Technology. It is provided free of charge on the Internet.

Kerberos relies on shared secrets. It presumes that the third party is secure, and provides single signon capabilities, centralized password storage, database link authentication, and enhanced PC security. It does this through a Kerberos authentication server, or through CyberSafe ActiveTrust, a commercial Kerberos-based authentication server.

Kerberos single signon provides a number of benefits. With only one centralized password store, it reduces the administrative overhead and requires users to remember only one password. It enables network access time to be controlled, and by means of DES encryption and CRC-32 integrity, it secures against unauthorized access and packet replay. Further, it enables current user database links. Kerberos-enabled databases can propagate a client's identity to the next database for Kerberos users connecting with single signon through Kerberos.

CyberSafe is a commercial version of Kerberos, which adds certain extra features and support, including support for the CyberSafe ActiveTrust server. CyberSafe centralizes security and provides single signon. Like Kerberos, it is based on passwords, but provides a much stronger authentication mechanism.

See Also: "Kerberos and CyberSafe with Oracle Advanced Security" on page 9-36

RADIUS

The RADIUS protocol (Remote Authentication Dial-In User Service) is an industry standard protocol adopted by authentication vendors as a common communication method. RADIUS provides user authentication, authorization and accounting between a client and an authentication server. It has been implemented by almost all organizations enabling users to access the network remotely. Enterprises have standardized on RADIUS because of its widespread acceptance in the industry, its flexibility, and its ability to centralize all user information in order to ease and reduce the cost of user administration. From the user's perspective, the entire authentication process takes place seamlessly and transparently.

See Also: "RADIUS with Oracle Advanced Security" on page 9-36

Token Cards

Token cards provide a two-factor method of authenticating users to the database. To gain access, a user must possess the physical card, and must know the password.

Token cards (SecurID or other RADIUS-compliant cards) can improve ease of use through several different mechanisms. Some token cards dynamically display one-time passwords that are synchronized with an authentication service. The server can verify the password provided by the token card at any given time by contacting the authentication service. Other token cards have a keypad and operate on a challenge-response basis. In this case, the server offers a challenge (a number) that the user enters into a token card. The token card provides a response (another number cryptographically derived from the challenge) that the user enters and sends to the server.

Token cards provide the benefits described in Table 4–1:

Table 4–1 Benefits of Token Cards

Benefit	Description
Strong authentication	To masquerade as a user, a malefactor must have the token card as well as the personal identification number (PIN) required to operate it. This is called two-factor authentication.
Ease of use	Users need only remember a PIN, instead of multiple passwords.
Ease of password management	Password management is easy because there is one token card rather than multiple passwords.
Enhanced accountability	Token cards provide a stronger authentication mechanism; users are thus more accountable for their actions.

See Also: "Token Cards with Oracle Advanced Security" on page 9-37

Smart Cards

A RADIUS-compliant smart card is a credit card-like hardware device. It has memory and a processor and is read by a smart card reader located at the client workstation. Smart cards provide the benefits described in Table 4–2:

Table 4–2 Benefits of Smart Cards

Benefit	Description
Increased security	Smart cards rely on two-factor authentication. The smart card can be locked, and only the user who possesses the card and knows the correct personal identification number (PIN) can unlock it.
Improved performance	Some sophisticated smart cards contain hardware-based encryption chips that can provide better throughput than software-based implementations. A smart card can also store a username.
Accessibility from any workstation	Users log in by inserting the smart card in a hardware device that reads the card and prompts the user for whatever authentication information the card requires, such as a PIN. Once the user enters the correct authentication information, the smart card generates and enters whatever other authentication information is required.
Memory	Because smart cards have memory, they can potentially store things like encryption keys, a user's private key, and even digital certificates.

See Also: "Smart Cards with Oracle Advanced Security" on page 9-37

Distributed Computing Environment (DCE)

The Distributed Computing Environment (DCE) from the Open Software Foundation (OSF) is a set of integrated network services that work across multiple systems to provide a distributed environment. The network services include remote procedure calls (RPCs), directory service, security service, threads, distributed file service, diskless support, and distributed time service.

DCE is the middleware between distributed applications and the operating system and network services, and is based on a client/server model of computing. By using the services and tools that DCE provides, users can create, use, and maintain distributed applications that run across a heterogeneous environment.

See Also: "Distributed Computing Environment (DCE) with Oracle Advanced Security" on page 9-38

Biometrics

Biometric solutions are another means of achieving strong authorization. In this approach, a physical characteristic such as a fingerprint or voice is used to identify and authenticate an individual.

See Also: "Biometric Authentication with Oracle Advanced Security" on page 9-38

PKI and Certificate-Based Authentication

Public key infrastructure (PKI) is an industry-standard set of procedures and policies which can be used to guarantee secure information exchange. It provides encryption methods and access controls, as well as secure credentials in the form of digital certificates which can be used to authenticate users.

See Also: "Secure Credentials: Certificate-Based Authentication in PKI" on page 8-5

"PKI Implementation in Oracle Advanced Security" on page 9-42

Proxy Authentication and Authorization

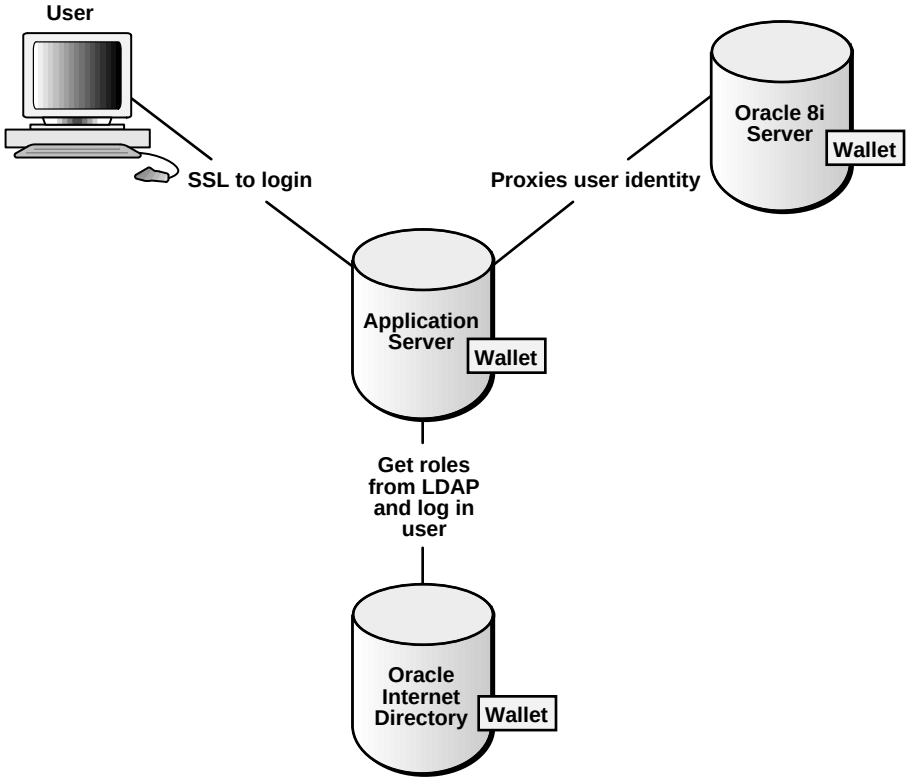
In multitier environments, such as a transaction processing monitor, it is necessary to control the security of middle-tier applications by preserving client identities and privileges through all tiers, and auditing actions taken on behalf of clients. Proxy authentication permits you to do this. For example, this feature enables the identity of someone using a Web application (also known as a *proxy*) to be passed through the application to the database server. This has several ramifications:

- It permits the application to validate the credentials of a user by passing them to the database server, in some cases.
- It enables the database administrator to regulate which users are allowed to access the database server through a given application.
- It enables the administrator to audit actions of the application acting on behalf of a given user.

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, proxy authentication supports a limited trust model for the middle-tier server, and avoids the problem of an all-privileged middle tier. It is also possible to give more privilege to a trusted middle tier (for example, one that is within the corporate firewall) than to a less-trusted middle tier (for example, 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 a lightweight user session, this feature makes it easier to audit the actions of users in a three-tier system, and thus improves accountability.

Figure 4–1 illustrates authentication in a multitier environment.

Figure 4–1 Proxy Authentication in a Multitier Environment



Proxy authentication can potentially support:

- Database users
- Enterprise users
- Application users not known to the database

One advantage of a middle tier is connection pooling, which enables multiple users to access a data server without each of them needing a separate connection. In such environments, you need to be able to set up and break down connections very quickly. For these environments, lightweight sessions are important. These lightweight sessions enable each user to be authenticated by a database password without the overhead of a separate database connection, and they preserve the identity of the real user through the middle tier.

See Also: "Proxy Authentication to Ensure Three-Tier Security" on page 3-8

"Proxy Authentication in Oracle9i" on page 9-9

Single Signon

Intranet users are commonly required to use a separate password to authenticate themselves to each server they need to access in the course of their work. Multiple passwords, however, present several problems. Users have difficulty keeping track of different passwords, tend to choose poor ones, and tend to record them in obvious places. Administrators must keep track of a separate password database on each server and must address potential security problems arising from the fact that passwords are routinely and frequently sent over the network.

Single signon (SSO) does away with these problems. It enables a user to log in to different servers using a single password to obtain authenticated access to all servers she is authorized to access. It eliminates the need for multiple passwords. In addition, it simplifies management of user accounts and passwords for system administrators.

You can implement SSO in different ways, as described in the following sections:

- Server-Based Single Signon
- Middle Tier Single Signon

Server-Based Single Signon

A centralized directory server can be used to store user, administration, and security information. This enables the administrator to modify information in only one location, namely, the directory. This centralization lowers the cost of administration and makes the enterprise more secure.

A directory server can be used to provide centralization of user account, user role, and password information. A database server authenticates a user with the information stored in the directory. Once authenticated, a user can access the databases configured to use enterprise user security.

Middle Tier Single Signon

Oracle9*i*AS Single Sign-On server is part of the Oracle9*i* infrastructure and provides Web-based single signon and integration with legacy applications. With single sign-on, users need maintain only a single strong user name / password account for accessing all Web applications throughout the enterprise.

Applications integrated with Oracle9*i*AS Single Sign-On securely delegate the user authentication process. Among other things, this enforces password rules, account lockouts based on repeated login failures, and auditing. The Single Sign-On server also has the ability to authenticate a user by means of an external repository such as LDAP or a database user repository, or by local authentication.

The first time that a user tries to access an application using Single Sign-On, the server:

- Authenticates the user by means of user name and password
- Passes the client's identity to the Single Sign-On enabled applications
- Marks the client being authenticated with an encrypted login cookie

In subsequent user logins, the login cookie provides the Single Sign-On Server with the user's identity and indicates that authentication has already been performed.

See Also: "Single Sign-On Implementations in Oracle Advanced Security" on page 9-38

"Single Sign-On in Oracle9i Application Server" on page 9-61

5

Using and Deploying a Secure Directory

Many security advantages can be had by centralizing in a directory the storage and management of user information such as identity, credentials, and other attributes. This chapter describes how to protect a directory, and how access can be controlled using a directory.

- Introduction
- Centralizing Shared Information with LDAP
- Securing the Directory
- Directory-Based Application Security

See Also: Chapter 6, "Administering Enterprise User Security"

"Oracle Internet Directory" on page 9-48

Introduction

Administrators today must manage complex user information, keeping it current and secure. These tasks become all the more challenging with increased use of technology and a high user turnover in enterprises. For example, in a typical enterprise, each user can have multiple accounts on different databases. This means too many passwords for users to remember, and too many accounts for administrators to manage. Consequently, users write down their passwords, make them easy to remember (and easy for someone else to guess), or choose the same password for all accounts.

Administrators must manage multiple accounts for every user. As a result, they devote significant resources to user administration. Common information used by multiple applications—such as username, user's office location and phone number, and system privileges—is often fragmented across the enterprise, leading to data that is redundant, inconsistent, and expensive to manage.

There are security problems as well. For example, any time a user leaves a company or changes jobs, his privileges should change the same day in order to guard against misuse of his old or unused accounts and privileges. However, in a large enterprise, with user accounts and passwords distributed over multiple databases, an administrator may not be able make all the changes as expeditiously as good security requires.

Enterprise user security management must address these user, administration, and security challenges. The best way is to centralize storage and management of user-related information in an LDAP-compliant directory service such as Oracle Internet Directory. Then, when an employee changes jobs, the administrator needs to modify information in only one location—the directory. This centralization lowers the cost of administration and makes the enterprise more secure.

Centralizing Shared Information with LDAP

Today, network information is stored in multiple systems and in multiple directory formats. With new requirements for Internet computing and new e-business technologies, there is a growing need for a common repository infrastructure to serve as a foundation for management and configuration of all data and resources. Such a common infrastructure reduces the cost of managing and configuring resources in heterogeneous networks.

Lightweight Directory Access Protocol (LDAP) technology was initially developed at the University of Michigan. It is currently an industry-accepted standard and is available in a variety of implementations.

Support of LDAP-compliant directory servers provides a centralized vehicle for managing and configuring a distributed network. The directory can act as a central repository for all data on database network components, user and corporate policies, and user authentication and security, thus replacing client-side and server-side localized tnsnames.ora files.

An LDAP-compliant directory can provide many powerful features to protect information:

Table 5–1 Security Features of an LDAP-Compliant Directory

Feature	Purpose
Data Integrity	Ensures that data—including passwords—is not modified, deleted, or replayed during transmission
Data Privacy	Ensures that data is not inappropriately detected during transmission by using public-key encryption. In public-key encryption, the sender of a message encrypts the message with the public key of the recipient. Upon delivery, the recipient decrypts the message by using the recipient's private key
Password Protection	Stores passwords as hashed values, to ensure that intruders can neither read nor decrypt them
Password Policy Management	Enables directory to centralize policy management for its users and accounts.
Authentication	Ensures that the identities of users, hosts, and clients are correctly validated
Authorization	Ensures that a user reads or updates only the information for which that user has privileges

To gain all these advantages of security directory integration, you must first ensure that the directory itself is secure. This involves:

- Secure connections to the directory on the part of the user and the administrator
- Access controls on the directory itself

Once your directory has been secured, other applications in an enterprise or hosted environment can take advantage of all these features. They can use the directory for administrative delegation, and control access to application metadata.

See Also: "Secure Sockets Layer (SSL) Protocol" on page 3-6

Securing the Directory

This section describes how access can be controlled within the directory.

- Directory Access Controls and Authorization
- Password Protection in a Directory
- Directory Access Controls and Authorization

Directory Authentication of Users

Authentication is the process by which the directory server establishes the true identity of the user connecting to the directory. To verify the identities of users, hosts, and clients, the directory can provide various authentication options:

Table 5–2 Directory Authentication Options

Option	Description
Anonymous Authentication	Users simply leave blank the user name and password fields when they log in. Each anonymous user then exercises whatever privileges are specified for anonymous users.
Simple Authentication	The client identifies itself to the directory by means of a distinguished name and a password that are not encrypted when sent over the network.
Authentication Using Secure Sockets Layer (SSL)	Authentication can occur through the exchange of certificates issued by trusted certificate authorities.
Authentication Through a Middle Tier	Authentication can occur through a middle tier, such as a RADIUS server or an LDAP self-service servlet. This involves a proxy user that performs directory operations on the end user's behalf.

See Also: Chapter 4, "Authenticating Users to the Database"

Password Protection in a Directory

Oracle Internet Directory can protect passwords by storing them as one-way hashed values. This approach secures passwords better than the approach of storing them as clear text or encrypted values, because a malicious user can neither read nor decrypt them if they are hashed.

During authentication to a directory server, a user enters a password in clear text. The directory server hashes this user password by using the specified hashing algorithm, then verifies it against the hashed password that has been stored. If the hashed password values match, then the server authenticates the user.

You can specify one of the following hashing schemes:

Table 5–3 Hashing Algorithms

Hashing Scheme	Description
MD4	The default hashing scheme. MD4 is a one-way hash function that produces a 128-bit hash, or message digest.
MD5	An improved, and more complex, version of MD4
SHA	Secure Hash Algorithm, which produces a 160-bit hash, longer than MD5. The algorithm is slightly slower than MD5, but the larger message digest makes it more secure against brute-force collision and inversion attacks.
UNIX Crypt	The UNIX hashing algorithm
No Hashing	The password is not hashed.

See Also: "Encrypting Data for Network Transmission" on page 3-4

Directory Access Controls and Authorization

Authorization is the process of ensuring that a user reads or updates only the information for which that user has privileges. When directory operations are attempted within a directory session, the directory server ensures that the user has the requisite permissions to perform those operations. If the user does not have the requisite permissions, then the directory server disallows the operation. Through this mechanism, the directory server protects directory data from unauthorized operations by directory users.

The following features of directory access control can be used by applications running in the hosted environment.

Table 5–4 Directory Access Control Features

Feature	Description
Prescriptive Access Control	Enables the service provider to specify access control lists (ACLs) for a collection of directory objects, instead of having to state the policies for each individual object. This feature simplifies the administration of access control, especially in large directories where many objects are governed by identical or similar policies.
Hierarchical Access Control Administration Model	Enables the service provider to delegate directory administration to subscribers. The subscriber could in turn delegate further if necessary.
Administrative Override Control for Delegated Domains	Enables the service provider to perform diagnosis and recovery from unintentional account lockout or accidental security exposure.
Dynamic Evaluation of Access Control Entities	Enables subtree administrators to identify both subjects and objects in terms of their namespace and their association with other objects in the directory. For example, the administrator of one subscriber subtree can allow only a user's manager to update that user's salary attribute. The administrator of another subscriber subtree can establish and enforce a different policy regarding salary attributes.

Directory-Based Application Security

In an enterprise or hosted environment, you can use the features of an LDAP-compliant directory to control access to application metadata—the information governing how applications behave and who can access them.

Because directory access control policies are stored as LDAP attributes, you can set metapolicies controlling who can modify them. This enables a global administrator to assign privileges to administrators of specific subtrees—for example, to administrators of applications in a hosted environment. Similarly, a global administrator can delegate to department administrators access to the metadata of applications in their departments. Department administrators can then control access to their department applications. In this way, you can implement access control on two levels: users and administrators.

This section includes:

- Authorization of Users
- Authorization of Administrators
- Administrative Roles in the Directory

Authorization of Users

In this case, the directory stores access control policies that external applications then read and enforce. When a user tries to perform an operation by using an application, the application verifies that the user has the correct authorization to perform the operation.

Authorization of Administrators

In this case, the directory serves as the trusted point of administration for all application-specific access control polices. To govern who can administer the access control policies of specific applications, you set access control policies at the directory level for these applications. Then, when a user attempts to change an application-specific access control policy, the directory verifies that the user has the correct authorization to make that change.

Figure 5–1 shows the relationship between directory access control and the application-specific access control mechanisms in a hosted environment.

Figure 5–1 Delegating Administration for Directory-Based Application Security

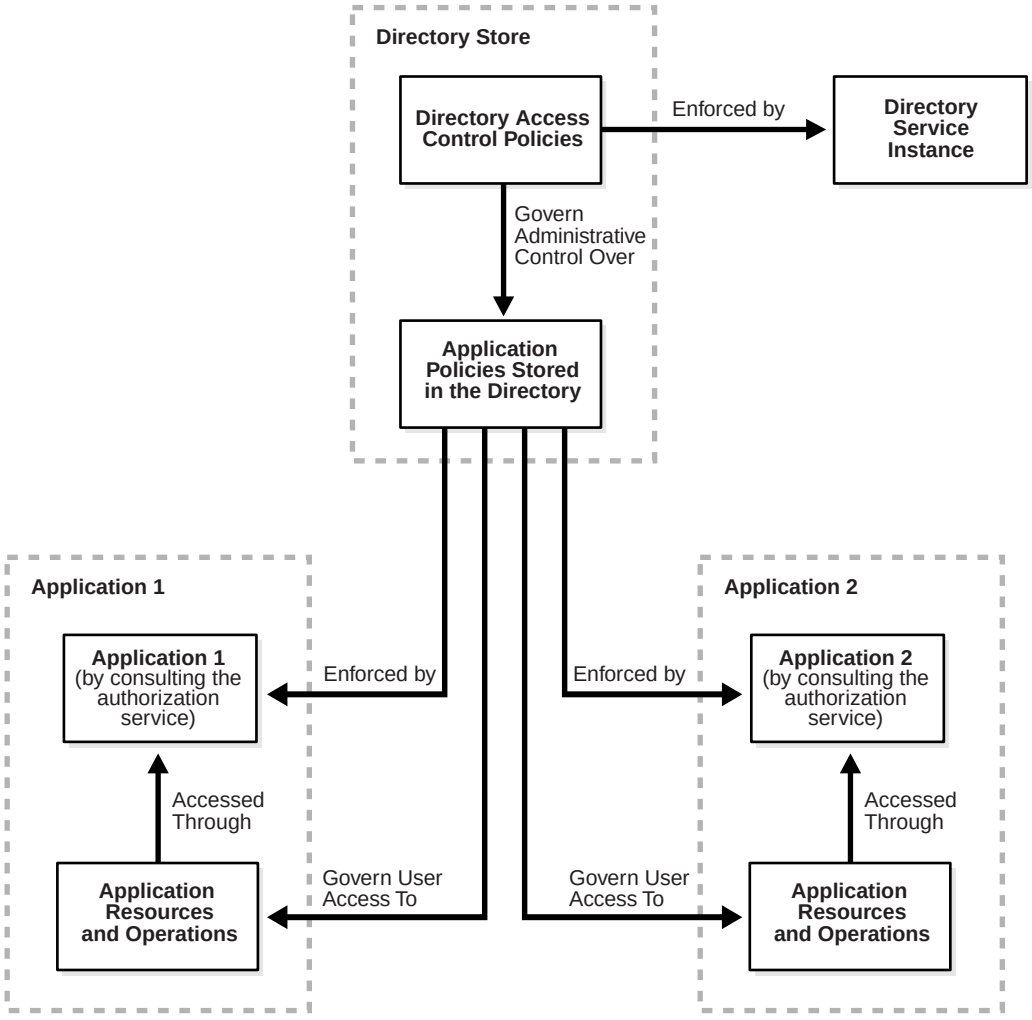
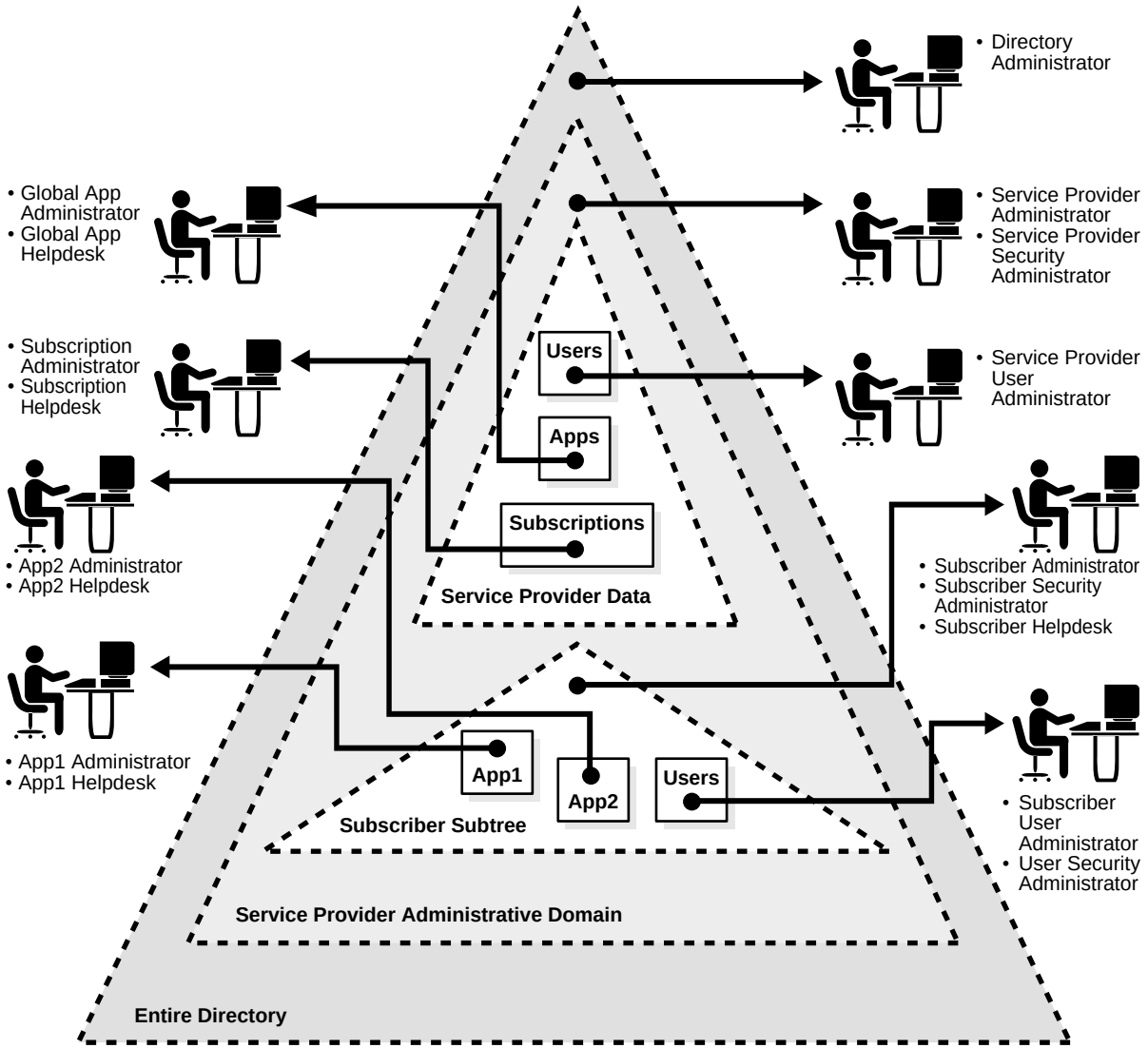


Figure 5–2 illustrates the various domains and the roles associated with them in the directory.

Figure 5–2 Directory Domains and Roles in a Hosted Environment



In Figure 5–2, each triangle represents a portion of a directory information tree (DIT).

- The outermost triangle represents the entire directory. The directory administrator has privileges extending across the entire directory.
- Immediately inside the outermost triangle lies another triangle, the service provider administrative domain. In this portion, privileges to add new entries are delegated to the service provider's administrators.
- Inside the service provide administrative domain, privileges can be further delegated based on the ownership of directory information. For example, the delegation can depend on whether the information is private to a specific subscriber or global to the service provider.

Figure 5–2 shows only a single subscriber represented in the directory. In reality there are multiple subscribers, each with its own domain requiring protection from the others. Some of the protection domains in this model are:

- Entire directory
- Service provider administrative domain
- Service provider-specific directory information tree
- Subscriber-specific subtree
- Application-specific footprint in the directory
- User-specific information

Administrative Roles in the Directory

Three types of role support the protection domains listed in the previous section. These roles enable the service provider or subscriber to customize access control if required.

Table 5–5 Administrative Roles in the Directory

Role	Function
Global Administrative Roles	These roles have rights to perform activities that span the entire directory.
Subscriber-Specific Roles	These roles are limited to the directory trees specific to the subscribers.
Application-Specific Roles	When hosting directory-enabled applications, it is not necessary to represent all application-specific roles in the directory. However, it is better that applications, when representing roles that directly affect their directory footprint, follow the delegation model recommendations described earlier. This enables applications to leverage the directory-based delegation model when granting directory-specific privileges to users.

6

Administering Enterprise User Security

This chapter describes the elements which make up a strong enterprise user management facility.

- Introduction
- Enterprise Privilege Administration
- Shared Schemas
- Password-Authenticated Enterprise Users
- Enterprise Roles
- Multitier Authentication and Authorization
- Single Sign-On

See Also: Chapter 5, "Using and Deploying a Secure Directory"

"Oracle Internet Directory" on page 9-48

Introduction

Most organizations, whether eBusinesses or not, face daunting obstacles in user management. Users within an organization often have far too many user accounts, a problem exacerbated by the growth in web-based self-service applications. Every other week, users have a new user account and password to remember. Organizations that want data access and accountability by user do not want the administrative nightmare of managing users in each database a user accesses.

This problem is compounded for web-facing, eBusiness applications. An organization opening its mission-critical systems to partners and customers does not want to create an account for each partner in each database the partner accesses, yet privilege and accountability for each partner is highly desired. Powerful, enterprise user management tools are necessary to meet these needs.

Enterprise Privilege Administration

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 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.

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.

See Also: How Oracle Internet Directory Organizes Enterprise User Management on page 9-53

"Enterprise User Security Features of Oracle Advanced Security" on page 9-39

Shared Schemas

A shared schema (also known as a schema-independent user) is a database user whose identity is maintained in a central LDAP repository. When a schema-independent 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, who require access to data on several database servers in the enterprise. Instead of maintaining 500 different user accounts on each database, Oracle9*i* allows the system administrator to create a single shared schema (such as HRAPPUSER for the HR application), with appropriate privileges, on each database, and then create 500 enterprise users in an Oracle Internet Directory. When they connect to any specific database, these users are mapped to the appropriate schema on the database (such as HRAPPUSER), and inherit the privileges associated with the schema, as well as any additional privileges that are associated with the roles granted to them in the directory. Although these users share a common schema, individual schema-independent users' identities are associated with their sessions by the database, and are used for access control or auditing purposes. Once created, these user accounts in LDAP can be used within multiple applications, as well.

The shared schema user feature has a number of benefits. It reduces the administrative burden associated with managing users in an enterprise, and allows effective management of much larger communities of users than was previously possible. Moreover, it can provide a mechanism for integrating user account and privilege management across tiers in a multitier 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 can be registered once in a directory, and this gives them appropriate access to the middle tier as well as any databases in the enterprise that they need to access. In the future, it should be possible to build three-tier systems (such as 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.

See Also: "Shared Schemas in Oracle Advanced Security" on page 9-41

"Shared Schemas with Oracle Internet Directory" on page 9-53

Password-Authenticated Enterprise Users

To reduce processing overhead and the administrative burden, it is desirable to have password-based authentication for enterprise users. Further, enterprise users should be able to use a single enterprise username and password to connect to multiple databases, if desired. This feature is particularly useful for large user communities accessing multiple applications.

See Also: "Enterprise User Security Features of Oracle Advanced Security" on page 9-39

Enterprise Roles

If you have centralized management of user-related information in an LDAP-compliant directory service, you can store and manage enterprise roles to determine enterprise users' access privileges on databases. An enterprise role is a directory structure which contains global roles on multiple databases, and which can be granted to enterprise users.

Multitier Authentication and Authorization

In applications that use a heavy middle tier, such as a transaction processing monitor, it is important to be able to preserve the identity of the client connecting to the middle tier. Client identities and privileges must be preserved and audited through all tiers.

Single Sign-On

A typical user on a corporate intranet has access to a multitude of client applications. Such a user must remember a username and password for each application being accessed. From the user's perspective, keeping in mind a myriad of username and password combinations, and having to re-enter them (for authentication) each time a different application is accessed, is not efficient. This may lead to the user maintaining just one username and password for all the applications to which access has been granted. This is generally not recommended as sound security practice, because a hacker can choose a number of attack points within such a framework.

From the application's perspective, such a framework requires the maintenance of each user's username and password store. This leads to password store redundancy and non-communication between applications due to the lack of transferable information regarding the user's roles and privileges (which may be granted on an enterprise-wide basis).

With single sign-on technology, a user can enter a unique username and password once. These are subsequently used to automatically authenticate the user to a number of different client applications without the user having to re-enter a username, password, or both. The user's roles and privileges are propagated from one application to another such that he or she is appropriately privileged in the application being accessed.

See Also: "Single Sign-On Implementations in Oracle Advanced Security" on page 9-38

7

Auditing to Monitor System Security

Auditing, a means of monitoring the effectiveness of your security policies, is a critical aspect of system security. This chapter includes:

- Introduction
- Fundamental Auditing Requirements
- Fine Grained, Extensible Auditing
- Auditing in Multitier Application Environments

Introduction

Any security policy must maintain a record of system activity to ensure that users are held accountable for their actions. Auditing helps deter unauthorized user behavior which may not otherwise be prevented. It is particularly useful to ensure that authorized system users do not abuse their privileges.

Fine-grained auditing can serve as an "early warning system" of users misusing data access privileges, as well as an intrusion detection system for the database itself.

Fundamental Auditing Requirements

This section describes the basic requirements for security monitoring

- Robust, Comprehensive Auditing
- Efficient Auditing
- Customizable Auditing

Robust, Comprehensive Auditing

A strong audit facility allows businesses to audit database activity by statement, by use of system privilege, by object, or by user. You can audit activity as general as all user connections to the database, and as specific as a particular user creating a table. You can also audit only successful operations, or unsuccessful operations. For example, auditing unsuccessful SELECT statements may catch users on "fishing expeditions" for data they are not privileged to see. Audit trail records can be stored in an Oracle9*i* table or combined with operating system audit trails on selected operating systems, for ease of management. Audit trail records stored in an Oracle9*i* table can be viewed through ad hoc queries or any appropriate application or tool. Storing certain audit trails separately enables an enterprise to audit the actions of even the most privileged users.

> **See Also:** The `audit_sys_operations` parameter in Oracle9i Database Administrator's Guide

Efficient Auditing

Auditing should be implemented efficiently: statements are parsed once for both execution and auditing, not separately. Also, auditing is implemented within the server itself, not in a separate, add-on server which may be remotely situated from the statements which are being executed (thereby incurring network overhead). The granularity and scope of these audit options allow businesses to record and monitor specific database activity without incurring the performance overhead that more general auditing entails. And, by setting just the options of interest, you should be able to avoid catch-all, and throw-away audit methods which intercept and log all statements, and then filter them to retrieve the ones of interest.

Customizable Auditing

To record customized information that is not automatically included in audit records, it is possible to use triggers to further customize auditing conditions and audit record contents. Database triggers are user-defined sets of PL/SQL or Java statements, stored in compiled form. While users explicitly execute stored procedures, database triggers are automatically executed (or "fired") within the data server based on pre-specified events. A trigger is defined to execute either before or after an INSERT, UPDATE or DELETE, so that when that operation is performed on that table, the trigger automatically fires. For example, you could define a trigger on the EMP table to generate an audit record whenever an employee's salary is increased by more than 10 percent and include selected information, such as before and after values of SALARY.

See Also: "Auditing" on page 9-5

Fine Grained, Extensible Auditing

Fine-grained auditing enables organizations to define specific audit policies that can alert administrators to misuse of legitimate data access rights.

Standard auditing can monitor privileges and objects, and provides triggers to monitor DML operations such as INSERT, UPDATE, and DELETE. By contrast, monitoring SELECT statements is facilitated by fine-grained auditing, which allows the monitoring of data access based on content. In this way, you can specify auditing conditions, and obtain more specific information about the environment and query result. This additional information helps you reconstruct audited events, and determine whether access rights have been violated. It prevents users from bypassing database auditing.

See Also: "Fine-Grained Auditing" on page 9-21

Auditing in Multitier Application Environments

Many three-tier applications authenticate users to the middle tier, and then the transaction processing monitor or application server connects as super-privileged user, and performs all activity on behalf of all users. But it is important to be able to preserve the identity of the real client over the middle tier and enforce "least privilege" through a middle tier. In addition, you may want to audit actions taken on behalf of the user by the middle tier. Auditing user activity, whether users are connected through a middle tier or directly to the data server, enhances user accountability, and thus the overall security of multitier systems.

See Also: "Oracle Auditing for Three-Tier Applications" on page 9-23

8

The Public Key Infrastructure Approach to Security

Public Key Infrastructure (PKI) is a set of policies and procedures to establish a secure information exchange. This chapter describes the elements which make up PKI, and explains why it has become an industry standard approach to security implementation.

- Introduction
- Public Key Cryptography and the Public Key / Private Key Pair
- Secure Credentials: Certificate-Based Authentication in PKI
- Storing Secure Credentials with PKI
- Single Sign-On Using PKI
- Network Security Using PKI

Introduction

This section presents basic concepts of a Public Key Infrastructure (PKI):

- Security Features of PKI
- Components of PKI
- Advantages of the PKI Approach

Security Features of PKI

PKI is emerging as the foundation for secure electronic commerce and Internet security by providing the cornerstones of security:

Authentication	The importance of authentication, verifying the identity of users and machines, becomes crucial when an organization opens its doors to the Internet. Strong authentication mechanisms ensure that persons and machines are the entities they claim to be.
Encryption	Encryption algorithms are used to secure communications and ensure the privacy of data sent from one computer to another.
Non-repudiation	PKI can be used to provide non-repudiation through digital signatures. This proves that a specific user performed certain operations at a given time.

Together, these elements combine to provide a secure, non-breakable environment for deploying e-commerce and a reliable environment for building virtually any type of electronic transactions, from corporate intranets to Internet-based eBusiness applications.

Components of PKI

The main components of a public key infrastructure are:

Digital certificates	Digital "identities" issued by trusted third parties, that identify users and machines. They may be securely stored in wallets or in directories.
Public and private keys	Form the basis of a PKI for secure communications, based on a secret private key and a mathematically related public key
Secure sockets layer (SSL)	An Internet-standard secure protocol
Certificate Authority (CA)	Acts as a trusted, independent provider of digital certificates

Other important factors which enable the deployment of PKI include: secure storage of certificates and keys; management tools to request certificates, access wallets and administer users; and a directory service acting as a centralized repository for certificates.

Advantages of the PKI Approach

The PKI approach to security does not take the place of all other security technologies; rather, it is an alternative means of achieving security. The following advantages of PKI have led to its emergence as an industry standard for securing Internet and e-commerce applications.

- PKI is a standards-based technology.
- It allows the choice of trust provider.
- It is highly scaleable. Users maintain their own certificates, and certificate authentication involves exchange of data between client and server only. This means that no third party authentication server needs to be online. There is thus no limit to the number of users who can be supported using PKI.
- PKI allows delegated trust. That is, a user who has obtained a certificate from a recognized and trusted certificate authority can authenticate himself to a server the very first time he connects to that server, without having previously been registered with the system.
- Although PKI is not notably a single sign-on service, it can be implemented in such a way as to enable single sign-on.

Public Key Cryptography and the Public Key/Private Key Pair

Public-key cryptography requires that entities which want to communicate in a secure manner, possess certain security credentials. This collection of security credentials is stored in a wallet. Security credentials consist of a public/private key pair, a "user" certificate, a certificate chain, and "trusted" certificates.

The secrecy of encrypted data generally depends on the existence of a secret key shared between the communicating parties. Providing and distributing such secret keys is one aspect of key management. In a multiuser environment, secure key distribution may be difficult; public key cryptography was invented to solve this problem.

Public key cryptography is based on a secure secret key pair. Each key (one half of the pair) can only decrypt information encrypted by its corresponding key (the other half of the pair). A key pair includes:

The private key	Known only to its owner
The public key	Distributed widely, but still associated with its owner

Use of the cryptographic key pair to set up a secure, encrypted channel ensures the privacy of a message and validates the authenticity of the sender of the message. It also provides an important benefit: the ability to widely distribute the public key on a server, or in a central directory, without jeopardizing the integrity of the private key component of the key pair. This eliminates the need to transmit the public key to every correspondent in the system.

Each entity that participates in a public key system must have a public/private key pair. The public key for an entity is published by a certificate authority (CA) in a user certificate. Then, other entities that want to send it secure information can encrypt the information with the recipient entity's public key. Another use for a public key is for an entity that receives a communication to validate the sender's organizational affiliation.

Secure Credentials: Certificate-Based Authentication in PKI

Establishing user identity is of primary concern in distributed environments; otherwise, there can be little confidence in limiting privileges by user. Passwords are the most common authentication method in use, but for particularly sensitive data, you need to employ stronger authentication services. This section describes:

- Certificates and Certificate Authorities
- Authentication Methods Used with PKI

Certificates and Certificate Authorities

Having a central facility authenticate all members of the network (clients to servers, servers to servers, users to both clients and servers) is one effective way to address the threat of nodes on a network falsifying their identities. This method involves certificates and certificate authorities.

Certificate Authorities

A certificate authority (CA) is a trusted third party which certifies that other entities--users, databases, administrators, clients, servers--are who they say they are. When it certifies a user, the certificate authority verifies the user's identity and grants a certificate, signing it with the certificate authority's private key. The certificate authority has its own certificate and public key, which it publishes, as well as a private key, which is securely maintained. Servers and clients use the CA’s root certificate to verify signatures which the certificate authority has made. A certificate authority might be an external company that offers certificate services, or an internal organization such as a corporate MIS department

Certificates

A certificate is like an electronic passport which proves the identity of a user or device that seeks to access the network. The certificate ensures that the entity's information is correct and that the public key actually belongs to that entity. A certificate is created when an entity's public key is signed by a trusted identity (a certificate authority). It contains information such as the following:

- the certificate user's name
- an expiration date
- a unique serial number assigned to the certificate by the CA
- the user's public key
- information about the rights and uses associated with the certificate
- the name of the certificate authority that issued the certificate
- the CA's signature
- an algorithm identifier that identifies which algorithm was used to sign the certificate

A trusted certificate, sometimes known as a root key certificate, typically belongs to a third party entity that is trusted to issue certificates. It is obtained in a secure manner and, operationally, does not need to be validated for its authenticity each time it is accessed because it is self-signed. A client or a server can validate that an entity is who it claims to be by verifying that the entity's certificate was issued by a known and trusted certificate authority.

Typically, certificate authorities whom you trust issue the user certificates. Oracle provides several default trusted certificates, so users do not have to install their own. These trusted certificates also enable servers to perform SSL authentication to clients who have wallets containing only trusted certificates.

Clients and servers use these credentials to access secure services, such as SSL, using public key cryptography. A wallet also represents a storage facility that is location- and type-transparent once it is opened.

Authentication Methods Used with PKI

Popular authentication methods used with PKI include:

- Secure Sockets Layer Authentication and X.509v3 Digital Certificates
- Entrust/PKI Authentication

Secure Sockets Layer Authentication and X.509v3 Digital Certificates

The Secure Sockets Layer (SSL) is an industry standard protocol that provides authentication, data encryption, and data integrity, in a public-key infrastructure. SSL is widely employed over the Internet to give users established digital identities and to prevent eavesdropping, tampering with, or forging messages.

SSL provides authentication through the exchange of certificates that are verified by trusted certificate authorities. SSL uses digital certificates (X.509 v3), and a public/private key pair to authenticate users and systems.

The most widely used public key certificates comply with the X.509 format, and the X.509 Version 3 certificate is the current industry standard format. A public key infrastructure relies on X.509 certificates, also called digital certificates, or public-key certificates, for public-key authentication.

X.509v3 digital certificates contain the following:

- The certificate owner's Distinguished Name (DN), which uniquely identifies the owner
- The Distinguished Name of the certificate issuer, which uniquely identifies the certificate authority
- The certificate owner's public key
- The issuer's signature
- The dates for which the certificate is valid
- The serial number of the certificate

The SSL protocol has gained the confidence of users, and it is perhaps the most widely-deployed and well-understood encryption protocol in use today.

See Also: "Secure Sockets Layer (SSL) Authentication in Oracle Advanced Security" on page 9-35

Entrust/PKI Authentication

Entrust Technologies, Inc. is a market-leading provider of Public Key Infrastructure solutions, through their Entrust/PKI software. Entrust/PKI includes many products, such as Entrust Profile, which secures users' PKI credentials, and Entrust Authority, Entrust's certificate authority product. Oracle Corporation has modified its SSL implementation to integrate with Entrust/PKI.

Note that Entrust/PKI is not fully compliant with all relevant PKI standards.

> **See Also:** "Entrust/PKI Support in Oracle Advanced Security" on page 9-35

Storing Secure Credentials with PKI

Many organizations manage users and authorizations separately in an LDAP-compliant directory. Now they can also store credentials securely in the directory, enhancing their ability to manage users.

With PKI, secure credentials such as digital certificates can be stored in containers called "wallets". A wallet is a transparent database used to manage authentication data such as keys, certificates, and trusted certificates needed by SSL. Wallets can be stored in an LDAP-compliant directory. This implementation enables you to centrally manage users.

Security administrators use a tool such as Oracle Wallet Manager to manage security credentials on the server. Wallet owners use it to manage security credentials on clients.

Public Key Certificate Standard #12 (PKCS#12) is the standard for secure credential storage.

> **See Also:** "Centralizing Shared Information with LDAP" on page 5-3
>
> "Components of Oracle Public Key Infrastructure-Based Authentication" on page 9-42

Single Sign-On Using PKI

Single sign-on enables users to access multiple accounts and applications with a single password. This feature eliminates the need for multiple passwords for users and simplifies management of user accounts and passwords for system administrators. Single sign-on enhances ease-of-use for users, and provides centralized management to security administrators.

Because all clients, application servers, and data servers can authenticate themselves to one another, PKI provides an important security infrastructure to a network.

See Also: "Single Sign-On Implementations in Oracle Advanced Security" on page 9-38

Network Security Using PKI

In addition to centralized network authentication, a PKI implementation can provide encryption of network traffic as well as integrity checking. The Secure Sockets Layer provides strong, standards-based encryption and data integrity algorithms.

See Also: "Encrypting Data for Network Transmission" on page 3-4

"PKI Implementation in Oracle Advanced Security" on page 9-42

Part III

Oracle9*i* Security Products

Part III presents the suite of Oracle security products which can meet your data security requirements.

- Chapter 9, "Oracle9i Security Products and Features"

9

Oracle9*i* Security Products and Features

This chapter introduces the Oracle products and special features which can protect your data using the latest security technology.

- Oracle9i Standard Edition
- Oracle9i Enterprise Edition
- Oracle Advanced Security
- Oracle Label Security
- Oracle Internet Directory
- Oracle Net Services
- Oracle9i Application Server

Oracle9*i* Standard Edition

Database security entails permitting or denying user actions on the database and the objects within it. Oracle uses schemas and security domains to control access to data and to restrict the use of various database resources. This section describes the many intrinsic security mechanisms of the Oracle9*i* database.

- Integrity
- Authentication and Access Controls in Oracle9i
- Privileges
- Roles
- Auditing
- Views, Stored Program Units, Triggers
- Data Encryption
- High Availability
- Proxy Authentication in Oracle9i

For a thorough discussion of these features, see the Oracle9*i* documentation set.

See Also: *Oracle9i Database Concepts*

Oracle9i Database Administrator's Guide

Integrity

Oracle9*i* contains many mechanisms to ensure the integrity of the database, and to provide concurrency, serializability of transactions, and to prevent data corruption. The access control mechanisms that enforce mandatory access control are also used to prevent unauthorized modification and deletion of data by users.

Data Integrity

Oracle9*i* provides data integrity through the use of declarative entity and referential integrity constraints as defined in the ISO / ANSI SQL standards. Integrity rules are specified declaratively as part of the table definition, and are checked by the database server whenever transactions update, insert, or delete rows in the table. Defining and enforcing these rules in the server ensures that all applications consistently and reliably apply the same rules, which can be maintained centrally. Enforcement in the server also provides performance benefits over programmatic enforcement in the application.

More complex business rules can be enforced through the use of stored procedures and triggers. However, these mechanisms are not normally used to enforce entity, referential, or transaction integrity.

Database integrity mechanisms also guarantee that all steps in a transaction are committed as a complete unit, so that either all parts are committed or all parts are rolled back (transaction integrity).

Entity Integrity Enforcement

Entity integrity enforcement guarantees that each row in a table is uniquely identified by non-null values contained in its primary key columns. An example of entity integrity would be ensuring that every employee number in the EMP table is unique.

Referential Integrity

Referential integrity constraints are used to enforce dependencies and relationships between rows in tables. An example of this occurs when an employee's department number in the EMP table (foreign key) must be a valid department as specified in the DEPT table (primary key). Primary key / foreign key relationships are defined as part of table creation.

See Also: "Database Integrity Mechanisms" on page 2-11

Authentication and Access Controls in Oracle9*i*

Oracle9*i* provides user authentication to ensure that the identity of a user, host or client is correctly known. To access a database, a user must supply a valid username and associated password of the database. These prevent unauthorized use. Oracle9*i* also provides authorization, to ensure that a user, program, or process receives the appropriate privileges to access an object or set of objects

To prevent unauthorized use of a database username, Oracle provides user validation by several different methods for normal database users. You can perform authentication by:

- The operating system
- The associated Oracle database

Further, Oracle Enterprise Edition supports additional modes of authentication:

- The Oracle database of a middle-tier application that performs transactions on behalf of the user
- The Secure Socket Layer (SSL) protocol
- A network service (through Oracle Advanced Security)

For simplicity, one method is usually used to authenticate all users of a database. However, Oracle permits use of all methods within the same database instance.

See Also: Chapter 4, "Authenticating Users to the Database"

Privileges

Oracle9*i* regulates all user access to data through privileges. It supports the concept of *least privilege*, which states that users should be granted the least number of privileges necessary to perform their jobs. Oracle9*i* enforces this concept by not automatically granting users any direct privileges when they are created. It supports both column-level and row-level privileges. Column-level privileges can be granted directly, and row-level privileges can be granted programmatically or through Oracle Label Security. The highly granular system and object privileges of Oracle9*i* enable you to grant users only the specific privileges they need, rather than having to grant them more encompassing privileges.

See Also: "System and Object Privileges" on page 2-2

Roles

Oracle9*i* has extensive support of roles, to enable administrators to optimally manage users' privileges. Oracle9*i* Standard Edition supports

- Database roles
- Global roles

Note that Oracle Enterprise Edition supports additional roles:

- Enterprise roles
- Secure application roles

See Also: "Using Roles to Manage Privileges" on page 2-4

"Secure Application Role" on page 9-21

Auditing

Oracle9*i* permits selective auditing of user actions to provide accountability. Audit records can also be a useful tool in identification of suspicious user activity. Auditing can be performed at different levels: by user, by statement, by privilege (such as SELECT), and by schema object (such as SELECT FROM EMP).

See Also: Chapter 7, "Auditing to Monitor System Security"

Views, Stored Program Units, Triggers

Oracle9*i* views and stored program units can add an additional level of security to your system. Views can restrict user access to a predetermined set of rows and columns of a table. Stored program units (such as stored procedures, packages, and triggers) can be used for such purposes as performing a set of related tasks, enforcing complex security authorizations, or restricting certain DML operations.

See Also: "Using Stored Procedures to Manage Privileges" on page 2-6

"Using Views to Manage Privileges" on page 2-7

Data Encryption

Among other security technologies, Oracle protects data in eBusiness systems through strong, standards-based encryption. Oracle has supported encryption of network data through Oracle Advanced Security (formerly known as "Secure Network Services", and then "Advanced Networking Option") since Oracle7. Oracle9*i* also supports protection of selected data by means of encryption within the database.

To address the need for selective data encryption, Oracle9*i* provides a PL/SQL package to encrypt and decrypt stored data. The package, DBMS_OBFUSCATION_TOOLKIT, supports bulk data encryption using the Data Encryption Standard (DES) algorithm, and includes procedures to encrypt and decrypt using DES. In addition to single DES, Oracle's DBMS_OBFUSCATION_TOOLKIT supports triple DES (3DES) encryption, in both two and three key modes, for those who demand the strongest commercial available level of encryption. The toolkit also supports the MD5 secure cryptographic hash to ensure data integrity, and a random number generator for generating secure encryption keys.

See Also: "Encrypting Data on the Server" on page 2-10

High Availability

Multiple Oracle9*i* mechanisms - including resource limits and user profiles, online backup and recovery, and advanced replication - help provide uninterrupted database processing and minimize denial of service in order to support today's on-line transaction processing and decision support environments.

User Profiles

Resource limitation and user profile mechanisms prevent "run-away" queries, or more deliberate and malicious manipulation of system resources by a particular user. A user profile is a set of administrator-defined resource limits assigned to a username; through the use of user profiles, Oracle9*i* enables the database administrator to define and limit the amount of certain system resource available to a user. System resources that can be limited include:

- Total Connect and Idle Time
- Total Amount Of Logical Input Or Output
- Number of Concurrent, Multiple Sessions for Each Username
- Amount of Memory Used

- Composite System Usage, Based On a Site-Defined Weighting of the Above.

Through user profiles, Oracle9*i* prevents resource hogs from denying service to other users, either inadvertently or maliciously.

Online Backup and Recovery

Oracle9*i* also ensures high availability by providing robust online backup and recovery, so that mission-critical applications are not inhibited by these necessary activities. Oracle9*i* provides an integrated method for creating, managing, and restoring backups of a database, providing greater ease of management and administration of the backup and recovery operations, while maintaining superior performance and increased availability of the database. Oracle9*i* databases can be backed up on-line, even during periods of peak transaction processing activity. Server-managed backup and recovery improves database administrator productivity as well as simplifying the backup and recovery process. Oracle9*i* backup and recovery permits backing up of the entire database, or a subset of the database, in one operation, and minimizes time needed for backup and restore operations by performing automatic parallelization of backups and restores. Oracle9*i* backup and recovery also supports sequential I/O devices for output during backup and for input during restore operations. Tape backups are supported in conjunction with vendor-provided tape management systems.

Advanced Replication

The advanced replication facilities of Oracle9*i* can be used to increase the availability of systems by off-loading large scale queries from transaction processing databases. For example, large tables of customer purchasing data may be replicated to customer service databases, so that data-intensive queries do not contend with transactions against the same tables. Advanced replication facilities can also be useful in protecting the availability of a mission-critical database. For example, symmetric replication can replicate an entire database to a failover site should the primary site be unavailable do to a system or network outage. Advanced replication for both read and write access ensures data consistency; refresh groups preserve referential integrity and transaction consistency and the table snapshots of related master tables. For example, customers, orders, order lines are all related, so could be refreshed as a group.

Data Partitioning

Data partitioning in Oracle9*i* is a powerful tool for dramatic improvements in the manageability, performance, and scale of applications deployed using the Oracle9*i* data server. Oracle9*i* permits range partitioning of tables and multiple partitioning

strategies for indexes, providing very large database support, and improves administrative operations. In the real world, media failure, access balancing for performance, and table de-fragmentation are just a few of the areas where partitioning can reduce the impact of a outage or increase availability under high loads.

Oracle9*i* with the Partitioning option supports all DML operations in parallel today. In addition, scans of indexes, export and import of table data, and estimating and calculating statistics can also be performed in parallel on individual partitions. Partitions can be loaded individually and in parallel, with or without index pre-creation. Loading, backup, recovery, computing statistics, and import and export are all supported for each partition. These can be performed individually without interfering with operations underway on other partitions. With every operation available on a partition basis, it is possible to have truly dramatic performance improvements.

Very High Availability with Real Application Clusters

Real Application Clusters provide very high levels of availability for mission critical applications. In a Real Application Clusters environment, Oracle runs on two or more systems in a cluster, while concurrently accessing a single shared database. In the event of a failure of one of the systems, the surviving systems perform recovery of the failed Oracle instance. This provides some tremendous availability and scalability benefits over simple cold cluster failover.

- Failures of one of the systems only affects a subset of its users. The failure does not affect users connected to other systems in the cluster.
- Users of the failed system can failover to the surviving systems in the cluster. This failover is faster, because the Oracle server is already running on the surviving systems. Performance after failover benefits from data already loaded in the cache of the surviving systems.
- Users of the database can pre-connect to a backup server used in case of failure, mitigating the delay reconnecting users and further speeding recovery.
- Because all servers are active during normal operation, system resources are better utilized, and larger loads can be sustained. In addition, the solution is scaleable. If additional capacity is required, additional servers can be added to the cluster.

See Also: "System Availability Factors" on page 2-12

Proxy Authentication in Oracle9*i*

This section describes Oracle9*i* support for proxy authentication.

- Introduction
- Support for Additional Protocols
- Expanded Credential Proxy
- Application User Proxy Authentication

Introduction

The OCI proxy authentication feature was initially released in Oracle8*i*, and enabled 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.

In Oracle9*i* proxy authentication, authentication of the client is supported in the following ways:

- Through a database password that is given when the user accesses Oracle9*i* proxy authentication
- Through a distinguished name or X.509 certificate

In Oracle9*i* this 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, Oracle9*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.

See Also: "Proxy Authentication to Ensure Three-Tier Security" on page 3-8

"Proxy Authentication and Authorization" on page 4-8

Support for Additional Protocols

In Oracle8*i* the proxy authentication feature was limited to communications to the database which used the Oracle Call Interface (OCI), but in Oracle9*i* the feature has been extended to Java Database Connectivity (JDBC) access to the database. A middle tier server can now access the Oracle9*i* database on behalf of a client user by establishing a lightweight session for that user through JDBC-OCI.

Expanded Credential Proxy

Oracle8*i* supported proxy authentication for database users authenticated by password only; the password could be passed as an attribute to be verified by the database, or not, depending on an organization's security preferences.

Oracle9*i* extends proxy authentication to include additional credential proxy of either the Distinguished Name (DN) or full X.509 certificate to the database. This provides strong, three-tier security by enabling an SSL credential—an X.509 certificate or DN—to be passed to the database for purpose of identifying (but not authenticating) the user. (SSL cannot be used to authenticate a user through multiple tiers, since it is a point-to-point protocol rather than an end-to-end protocol.) For example, a user can authenticate to a middle tier using SSL, the middle tier can extract the DN from the certificate and pass it (or the full certificate) to the database. As an additional benefit, the DN or certificate is available in the lightweight session and the elements contained therein can be used with Virtual Private Database to limit access. For example, an organization could restrict data access based on the Organizational Unit (OU) element in a user certificate presented to the database.

The database can use the DN or certificate to look up a user in Oracle Internet Directory or other LDAP-based directory certified for enterprise user security (an Oracle Advanced Security feature). Integration of proxy authentication with enterprise user security enables the user identity to be maintained throughout all tiers of an application, yet the user need only be created once, in the directory. This also enables enterprise user security to be used in three-tier applications, instead of merely client/server, as was the case with Oracle8*i*.

See Also: Password-Authenticated Enterprise Users on page 9-40

Application User Proxy Authentication

Many applications use session pooling to set up a number of sessions which are reused by multiple users. In this context, "application users" are users who are authenticated to the middle tier of an application, but are not known to the database. Oracle9*i* introduces application user proxy authentication for these types of applications.

In this model, the middle tier passes a *client identifier* to the database upon session establishment. (The client identifier could be anything that represents the client connecting to the middle tier; a cookie, for example, or an IP address.) The client identifier, representing the application user, is available in user session information and can also be accessed within an application context (using the USERENV naming context), thus enabling applications to use Virtual Private Database to limit user access, even if the application users are not known to the database. Applications can set up and reuse sessions, while still being able to keep track of the "application user" in the session.

Application user proxy authentication, available in JDBC-OCI, provides the benefits of connection pooling without the overhead of setting up and managing separate user sessions (even "lightweight" ones), and enables even those applications whose users are unknown to the database to utilize Virtual Private Database. Application user proxy authentication is thus particularly valuable in eBusiness applications with thousands of users, as it supports data access control by user while meeting user scalability requirements.

Oracle9*i* Enterprise Edition

By providing deep data protection, Internet-scale security, and security mechanisms specifically targeted for hosting applications and exchanges, Oracle9*i* Enterprise Edition is an ideal platform on which to build and deploy eBusiness applications. It contains all of the powerful features of Oracle9*i* Standard Edition, and more. This section includes:

- Internet Scale Security Features
- Application Security
- Virtual Private Database in Oracle9i
- Secure Application Role
- Fine-Grained Auditing
- Oracle Auditing for Three-Tier Applications
- Java Security Implementation in the Database

For a thorough discussion of these features, see the Oracle9*i* documentation set.

See Also: *Oracle9i Database Concepts*

Oracle9i Application Developer's Guide - Fundamentals

Internet Scale Security Features

EBusiness depends on providing customers, partners, and employees with access to information, in a way that is controlled and secure. Oracle9*i* addresses eBusiness security challenges through deep data protection, internet-scale security, and secure hosting and data exchange.

Deep Data Protection

Deep data protection, ensuring well-formed, comprehensive security from client to application server to data server, as well as throughout the layers of an application.

Deploying eBusiness systems on the Internet increases risk. Among the best ways to mitigate security risk is to provide multiple layers of security mechanisms, so that failure of a single mechanism does not result in compromise of critical information. We refer to this concept as *deep data protection*; Oracle9*i* provides it through Virtual Private Database (VPD), Oracle Label Security, selective data encryption, and extensive auditing.

Internet-Scale Security

Internet-scale security enables user and privilege management to scale to hundreds of thousands of users accessing data. Oracle9*i* Enterprise Edition is the foundation for the Oracle Advanced Security features of user management, PKI integration, and directory-based privilege management.

Security mechanisms must scale to Internet size—support many thousands or millions of users—and still be practical to administer. Oracle9*i* provides a number of security features tailored to building Internet-scale applications, including proxy authentication, support for Internet standards such as Secure Sockets Layer (SSL) and relevant public key infrastructure (PKI) standards, Java security, and enterprise user security.

Secure Hosting and Data Exchange

Secure hosting and data exchange enable economical, secure partitioning of data access by customer or by user, while supporting secure data sharing among communities of interest. Oracle9*i* Enterprise Edition is the foundation for Virtual Private Database technology, for the Oracle Advanced Security features of public key infrastructure (PKI) and enterprise user security, and for Oracle Label Security.

Application Security

Each database application can have its own security policies. It can have its own privileges, and one or more database roles that provide different levels of security when executing the application. The database roles can be granted to user roles, or directly to specific usernames.

Applications that potentially permit unrestricted SQL statement execution (through tools such as SQL*Plus) also can have security policies that prevent malicious access to confidential or important schema objects. In this way you can ensure that users do not misuse their roles and privileges when they are not actually using the application.

Virtual Private Database in Oracle9*i*

Oracle9*i* Enterprise Edition provides row-level access control through its Virtual Private Database (VPD) technology, which is available only from Oracle Corporation. In addition, it supports the Oracle Label Security product, built on the Virtual Private Database toolkit, which adds label based access control.

This section describes:

- Virtual Private Database in Oracle8i and Oracle9i
- How Virtual Private Database Works
- Application Context in Oracle9i
- How Application Context Facilitates VPD
- How Partitioned Fine-Grained Access Control Facilitates VPD
- User Models and Virtual Private Database
- Oracle Policy Manager

> **See Also:** For a complete discussion of application context, fine-grained access control, and VPD, see *Oracle9i Application Developer's Guide - Fundamentals*

Virtual Private Database in Oracle8*i* and Oracle9*i*

Oracle8*i* set a new standard in database security with the introduction of Virtual Private Database (VPD): 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. Within a single database, the Virtual Private Database enables data access control by user or by customer 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 Oracle9*i* database, while permitting each company to see only its own data.

Within the enterprise, the Virtual Private Database results in lower cost of ownership in deploying applications. Security can be built once, in the data server, rather than in each application that accesses data. Security is stronger, because it is enforced by the database, no matter how a user accesses data. Security is no longer bypassed when a user accesses an ad hoc query tool or new report writer. Virtual Private Database is key enabling technology for organizations building hosted, web-based applications, as well as for Oracle itself. Multiple Oracle applications,

including Oracle SalesOnline.com and Oracle Portal, use VPD to enforce data separation for hosting.

In Oracle8*i* the Virtual Private Database feature provided fine-grained access control and application context. It secured data in the database by providing security at the row level, across all applications, by attaching a security policy directly to a table or view.

Oracle9*i* expands the Virtual Private Database by adding several new enhancements:

- Oracle Policy Manager, a tool to facilitate security policy administration
- Partitioned fine-grained access control, to ease VPD deployment in multi-application and hosted environments
- Global application context, to support application user models
- VPD support for synonyms

> **See Also:** "Application Query Rewrite: Virtual Private Database" on page 2-9
>
> "Database Enforced Network Access" on page 3-4
>
> "Oracle Policy Manager" on page 9-20

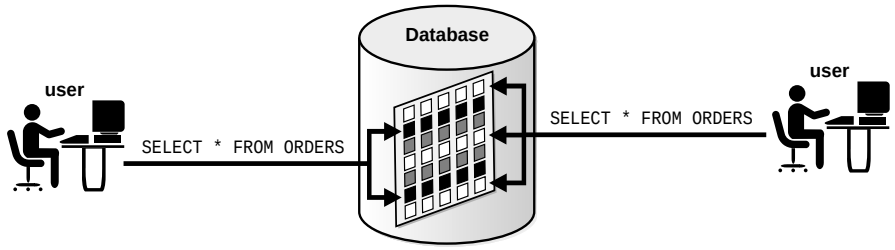
How Virtual Private Database Works

The Virtual Private Database is enabled by associating one or more security policies with tables or views. A security policy is a restriction on the type of access or view that a user can aquire. 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 statement, thus dynamically modifying the user's data access.

You can implement VPD by writing a stored procedure to append a SQL predicate to each SQL statement to control row level access for that statement. Your security policy then links the function to the desired schema and table. For example, if John Doe (who belongs to Department 10) inputs the statement SELECT * FROM emp, you can use VPD to tack on the clause WHERE DEPT = 10. In this way query modification is used to restrict data access to certain rows.

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, an Web order entry system can enforce access based on customer number, and whether the user is a customer or a sales representative. In this way, customers can view their order status online (but only for their own orders), while sales representatives can view multiple orders, but only for the their own customers.

Figure 9–1 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. In this way, VPD 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.

These fine-grained access control capabilities also apply when a synonym is used for the database name. Policy functions applied to a synonym can create the same constraints formerly imposed by creating views, without the costs in resources and processing that otherwise grow proportionately with the number of users.

Application Context in Oracle9*i*

Application context facilitates the implementation of fine-grained access control. It enables you to implement security policies with functions and then associate those security policies with applications. Each application can have its own application-specific context. Users are not permitted to arbitrarily change their context (for example, through SQL*Plus).

Application contexts permit flexible, parameter-based access control, based on attributes of interest to an application. For example, context attributes for a human

resources application could include "position", "organizational unit", and "country" while attributes for an order-entry control might be "customer number" and "sales region".

Note that enterprise user security requires Oracle Advanced Security. This feature also supports Oracle Label Security labels and privileges.

How Application Context Facilitates VPD

Most applications contain information about the basis on which access is to be limited. In an order entry application, for example, customers should be limited to access their own orders (`ORDER_NUMBER`) and customer number (`CUSTOMER_NUMBER`). Application context is an underlying database feature that enables you to define, set, and access attributes that an application can use to enforce access control. You can securely store such user attributes as a user name, employee number, the set of books she is authorized to access, and her position in the management hierarchy. You can then retrieve that information later in the session and use it for fine-grained access control.

Application contexts can be initialized in four different ways:

- Application Context Accessed Locally
- Application Context Initialized Externally
- Application Context Initialized Globally
- Application Context Accessed Globally

Application Context Accessed Locally

The application context feature was introduced in Oracle8*i*. Within a local database environment, attribute values can be initialized from a user's session information. Each application can have its own context with its own attributes.

Application Context Initialized Externally

This feature lets you specify a special type of namespace that accepts initialization of attribute values from external resources. This enhances performance and enables the automatic propagation of attributes from one session to the other. Many applications store attributes used for fine-grained access control within a database metadata table that they use for access control. For example, an EMPLOYEES table could include cost center, title, signing authority, and other information useful for fine-grained access control. However, many organizations centralize user information and user management in an LDAP-based directory such as Oracle

Internet Directory. These organizations also wish to centralize the information about users that is used for access control. Application context attributes can be stored in the directory and assigned to one or more enterprise users. They can be retrieved automatically upon login for an enterprise user, and used to initialize an application context.

Application Context Initialized Globally

This feature provides a centralized location to store the user's application context, enabling applications to set up the user's contexts during initialization based upon the user's identity. In particular, it supports Oracle Label Security labels and privileges. This feature makes it much easier for the administrator to manage contexts for large numbers of users and databases.

Application context initialized globally utilizes the Lightweight Directory Access Protocol (LDAP), which stores a list of users to which this application is assigned. Oracle9*i* can use Oracle Internet Directory as the directory service for authentication and authorization of enterprise users.

Application Context Accessed Globally

Global application context can be shared among trusted sessions. In addition to driving the enforcement of the fine-grained access control policies, applications (especially middle-tier products) can use this support to manage application attributes securely and globally.

Many web-based applications use connection pooling to achieve high scalability and thereby support hundreds of thousands of users. These applications set up and reuse connections instead of having different sessions for each user. For example, web user Jane and Ajit connect to a middle tier application, which establishes a session in the database used by the application on behalf of both users. The application is responsible for switching the username on the connection, so that, at any given time, it is either Jane or Ajit using the session.

Oracle9*i* VPD capabilities facilitate connection pooling by enabling multiple connections to access one or more *global* application contexts, instead of setting up an application context for each user session. Global application contexts provide additional flexibility for web-based applications to use Virtual Private Database, as well as enhanced performance through reuse of common application contexts among multiple sessions instead of setting up application contexts for each session.

Application user proxy authentication can be used with global application context for additional flexibility and high performance in building eBusiness applications. For example, suppose a web-based application that provides information to

business partners has three types of users: Gold, Silver, and Bronze, representing different levels of information available. Instead of each user having his own session—with individual application contexts—set up, the application could set up global application contexts for Gold, Silver or Bronze and use the client identifier to point the session at the correct context, in order to retrieve the appropriate type of data. The application need only initialize the three global contexts once, and use the client identifier to access the correct application context to limit data access.

How Partitioned Fine-Grained Access Control Facilitates VPD

Fine-grained access control enables you to build applications that enforce security policies at a low level of granularity. You can use it, for example, to restrict a customer who is accessing an Oracle server to see only his own account, a physician to see only the records of her own patients, or a manager to see only the records of employees who work for him.

The ability to partition security policy enforcement by application facilitates VPD deployment. For example, suppose both an Order Entry and Inventory application access the Orders table. The Order Entry application limits access based on customer number, while the Inventory application limits access based on part number. It is very useful to be able to partition fine-grained access control so that *different* security policies apply, depending upon which application is accessing the data. Otherwise, application developers of the respective Order Entry and Inventory applications have to agree upon a mutual policy, which may not be feasible or possible. Applications can thus have different security policies based upon their individual application needs.

Oracle9*i* enables partitioning of Virtual Private Database through policy groups and a driving application context. A driving application context securely determines which application is accessing data, and policy groups facilitate managing the policies which apply by application. Oracle9*i* also supports default policy groups, which always apply to data access. For example, an application "striped" for application hosting using a subscriber ID could have a default policy, "Subscriber," that always enforces data separation by subscriber, and additional policy groups for Inventory and Order Entry-based access, which apply depending on the particular application accessing data.

User Models and Virtual Private Database

Applications may have differing user models, but still want to use VPD to limit access by user. Oracle9*i* provides a number of ways in which applications can enforce fine-grained access control by user, regardless of whether the user is a database user, or an application user unknown to the database.

For applications in which the application users are also database users, VPD enforcement is relatively simple; users connect to the database, and the application can set up application contexts for each session. Each session is initiated under a different username, so that it is simple to enforce different fine-grained access control conditions for "Jane" and "John". This is also possible with use of proxy authentication, since each "lightweight" session in JDBC-OCI is still a distinct database session, and can have its own application context. Since proxy authentication can be integrated with Enterprise User Security, user roles can be retrieved from Oracle Internet Directory, as well as other attributes that can be used for VPD enforcement.

For applications in which a single user (such as OneBigApplicationUser) connects to the database on behalf of all users, fine-grained access control by user is still possible. An application developer can create a context attribute to represent the application user (such as "realuser"). While all database sessions (and thus all audit records) are initiated as OneBigApplicationUser, each session can nonetheless have attributes that vary, depending on who the "real user" is. This model works best for applications with a limited number of users where there is no requirement for session reuse. Of course, each session—from the database standpoint—is created as the same database user, so that the ability to use roles, database auditing, and so on, is greatly diminished for reasons previously enumerated.

Oracle Policy Manager

Oracle9*i* offers improved management of VPD policies through Oracle Policy Manager, an easy-to-use graphical user interface (GUI) accessed through Oracle Enterprise Manager. Developers can use Oracle Policy Manager to apply security policies to schema objects, such as tables and views, as well as creating application contexts, thus making VPD much easier to develop and manage. Oracle Policy Manager is also the administration tool for Oracle Label Security, a VPD-based product that provides label-based access to data. Oracle Label Security is thus a generic solution to the problem of fine grained data access control.

See Also: "Oracle Label Security" on page 9-47

Secure Application Role

This feature, unique to Oracle9*i*, enables you to base use of roles on user-defined criteria. A secure application role is a role which is implemented by a package. For example, you could write a package permitting use of a role by a user connecting only from a particular IP address, or accessing the database only through a particular middle tier.

In three-tier systems using proxy authentication, the package can validate that the user session was created by a middle tier, and thus that the user is accessing the database through the correct application. The secure application role can also ensure that a user connecting directly to the database is not able to access any data. A secure application role can enforce other security conditions, as well; for example, the user may not be permitted to access especially sensitive human resources data from the Internet.

A secure application role enhances the native strong authentication and fine-grained access control of the database to prevent users from assuming any privileges unless the correct access conditions are met. Secure application role solves a very difficult security issue and supports secure web-based application data access.

See Also: "Secure Application Roles" on page 2-6

Fine-Grained Auditing

Oracle9*i* expands upon the existing robust, granular auditing capabilities of the database by introducing extensible, fine-grained auditing. Fine-grained auditing enables organizations to hone their auditing capabilities to capture and identify particular, specific data access of concern. In addition to providing more granular, targeted audit information, such as detecting misuse of legitimate access, fine-grained auditing can also serve as an intrusion detection system for the Oracle9*i* database itself.

Fine-grained auditing enables organizations to define audit policies, which specify the data access conditions that trigger the audit event, and use a flexible event handler to notify administrators that the triggering event has occurred. For example, an organization may permit HR clerks to access employee salary information, but audits access when salaries greater than $500K are accessed. The audit policy "where SALARY > 500000" is applied to the EMPLOYEES table through an audit policy interface (a PL/SQL package named DBMS_FGA).

For additional flexibility in implementation, organizations can employ a user-defined function to determine the policy condition, and identify a relevant column for auditing. For example, the function could permit unaudited access to *any* salary as long as the user is accessing data within the intranet, but audit access to executive-level salaries when they are accessed from the Internet. An audit column helps reduce the instances of false or unnecessary audit records, because the audit need only be triggered when a particular column is referenced in the query. For example, an organization may only wish to audit executive salary access when an employee name is accessed, because accessing salary information alone is not meaningful unless an HR clerk also selects the corresponding employee name.

Upon a triggering audit event, Oracle9*i* captures the exact SQL text of the statement the user executed in audit tables, along with additional information such as the user executing the query, a timestamp, and so on. In conjunction with other database features such as LogMiner, fine-grained auditing can be used to re-create the exact records returned to a user. This may be especially important to organizations which have especially sensitive information they wish to share, for which they require strict accountability. For example, many law enforcement organizations at the international, federal, state and local level are increasingly becoming "eBusinesses" by sharing information among themselves, yet it is more important than ever that they audit access to sensitive information, such as informant data, to know who accessed what *exact* data.

The event handler provides organizations with flexibility in determining how to handle a triggering audit event. A triggering audit event could be written into a special audit table for further analysis, or could activate a pager for the security administrator. The event handler enables organizations to fine-tune their audit response to appropriate levels of escalation.

See Also: "Fine Grained, Extensible Auditing" on page 7-3

Oracle Auditing for Three-Tier Applications

With Oracle9*i*, Oracle customers are not only able to preserve the identity of the real client over the middle tier and enforce "least privilege" through a middle tier, but can also audit actions taken on behalf of the user by the middle tier. Oracle9*i* audit records capture both the logged-in user (that is, the middle tier) who initiated the connection, and the user on whose behalf an action is taken.

See Also: "Auditing in Multitier Application Environments" on page 7-4

Java Security Implementation in the Database

Oracle9*i* contains a Java security implementation in the server. The Java virtual machine (JVM) is the Java interpreter that converts the compiled Java bytecode into the machine language of the platform and runs it. JVMs can run on a client, in a browser, in a middle tier, on a Web, on an application server such as Oracle9*i* Application Server, or in a database server such as Oracle9*i*.

Class Execution

In the Oracle9*i* JVM implementation, the right to execute code in classes is controlled by execute privileges on the classes themselves. This is the same database privilege as execute privilege on a PL/SQL package, and is managed in the same way.

SecurityManager Class

The Oracle9*i* JVM starts with the class java.lang.SecurityManager installed. The Oracle9*i* database is based on the Java Developer's Kit 1.2 release from Sun Microsystems, and implements the security features of that release. In this implementation, permissions are controlled by the contents of a database table. The table is normally managed by PL/SQL procedures (and Java methods). The table can be used to grant permissions to either users or roles, and the "code source" of a class is identified with the user in whose schema the class has been loaded. Specific Oracle permissions control the right to update the table and perform other security sensitive operations.

See Also: *Oracle9i Java Developer's Guide*

Oracle Advanced Security

Oracle Advanced Security is the value-added Internet security bundle for Oracle9*i*. Its functionality falls into three categories: network security services, enterprise user security, and public key infrastructure (PKI). The features of this product are described in the following sections:

- Introduction to Oracle Advanced Security
- Network Security Services of Oracle Advanced Security
- Enterprise User Security Features of Oracle Advanced Security
- PKI Implementation in Oracle Advanced Security

Note: Oracle Advanced Security Release 8.1.6 has been validated under U.S. Federal Information Processing Standard 140-1 (FIPS) at the Level 2 security level. This is an important U.S. federal government security assessment, and Level 2 is the highest possible level for software. This provides independent confirmation that Oracle Advanced Security conforms to stringent federal government standards for encryption.

Introduction to Oracle Advanced Security

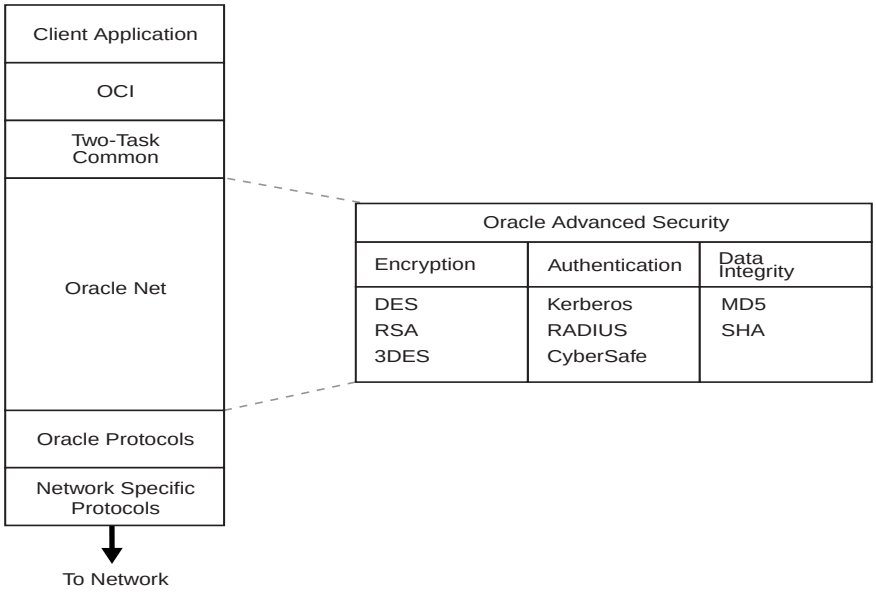
Oracle Advanced Security bundles security services for Oracle9*i*. It is provided as a separately priced option which may be purchased along with Oracle9*i* Enterprise Edition.

- Oracle Advanced Security provides data encryption and integrity for all network protocols into the Oracle database, including Oracle Net with native encryption, Oracle Net / SSL, IIOP / SSL, and Java-based encryption for Thin JDBC clients.
- It integrates with third-party authentication, authorization, and single sign-on services.
- It integrates a Public Key Infrastructure (PKI) and supports public-key solutions including Secure Sockets Layer (SSL) and X.509 Version 3 certificates (provided by third-party certificate servers). It bundles related tools, including Oracle Wallet Manager and Oracle Enterprise Login Assistant. It enables SSL-based single sign-on as well as certificate-based server-to-server authentication and database links. It also integrates with Entrust PKI.
- Oracle Advanced Security utilizes LDAP Version 3 compliant directory servers to centralize user management, and it bundles Oracle Enterprise Security Manager to manage enterprise users and enterprise roles. It provides restricted use of Oracle Internet Directory for storage of users and authorizations, and also integrates with Microsoft Active Directory for role management.

Although installed by default, Oracle Advanced Security is a separately priced option to Oracle9*i* Enterprise Edition, and must be purchased when used. This licensing requirement also affects customers wishing to use security features in combination with Java Beans (EJB over IIOP / SSL) or database enterprise users with Oracle Net / SSL. The exclusive exception is an HTTPS (HTTP / SSL) connection to the RDBMS, which does not require an Oracle Advanced Security license.

Figure 9–2 shows the Oracle Advanced Security architecture within an Oracle networking environment.

Figure 9–2 Oracle Advanced Security in an Oracle Networking Environment



Oracle Advanced Security supports authentication through adapters that are very much like the existing Oracle protocol adapters.

> **See Also:** for more information about stack communications in an Oracle networking environment, see *Oracle9i Net Services Administrator's Guide*

Network Security Services of Oracle Advanced Security

Oracle Advanced Security provides several methods of protecting the privacy of data transmissions.

- Oracle Net Services Native Encryption
- Data Integrity Features of Oracle Advanced Security
- Secure Sockets Layer (SSL) Encryption Capabilities
- Java Encryption Features of Oracle Advanced Security
- Strong Authentication Methods Supported by Oracle Advanced Security
- Single Sign-On Implementations in Oracle Advanced Security

Oracle Net Services Native Encryption

Oracle Advanced Security ensures data privacy by encrypting network traffic in order to prevent anyone from reading the data during transmission.

Oracle Advanced Security provides several industry-standard encryption and checksumming algorithms which can be selected based on the particular requirements of your system. Selection of the network encryption method offers varying levels of security and performance for different types of data transfers.

Note that the strength of cryptosystems depends on key management. Oracle Advanced Security uses the public-key based Diffie-Hellman key negotiation algorithm to perform secure key distribution for both encryption and data integrity. When encryption is used to protect the security of encrypted data, keys should be changed frequently to minimize the effects of a compromised key. For this reason, the Oracle Advanced Security key management facility changes the session key with every session. With Oracle Advanced Security, Diffie-Hellman Key Exchange is automatic, eliminating administration issues associated with encryption systems.

Prior versions of Oracle Advanced Security provided three editions: Domestic, Upgrade, and Export—each with different encryption key lengths. Release 9.0.1 now contains a full complement of the available encryption algorithms and key lengths for Oracle customers worldwide, previously only available in the U.S. Domestic edition. Users deploying prior versions of the product can obtain the U.S. Domestic edition for a specific product release.

Note: The U.S. government has relaxed its export guidelines for encryption products. Accordingly, Oracle can now ship Oracle Advanced Security with its strongest encryption features—to virtually all of its customers around the world.

Oracle Advanced Security comes out of the box with industry-standard algorithms and a FIPS-compliant implementation of cryptography, which help to simplify the often difficult task of implementing encryption. The following industry-standard encryption algorithms are supported:

Table 9–1 Encryption Algorithms

Name	Description
RSA Encryption	The RSA encryption module uses the RSA Security, Inc. RC4 encryption algorithm. Oracle's optimized implementation provides a high degree of security for a minimal performance penalty. For the RC4 algorithm, Oracle provides encryption key lengths of 40-bits, 56-bits, 128-bits, and 256-bits.
DES Encryption	Oracle Advanced Security implements DES with a standard, optimized 56-bit key encryption algorithm, and also provides DES40, a 40-bit version, for backwards compatibility.
Triple-DES Encryption	Oracle Advanced Security also supports Triple-DES encryption (3DES), which is available in two-key and three-key versions, with effective key lengths of 112-bits and 168-bits, respectively. Both versions operate in outer Cipher Block Chaining (CBC) mode.

Oracle Advanced Security hides the complexity of key management and encryption from the administrator and the users. Users need only perform a few simple steps to configure Oracle Advanced Security encryption. You can either use the Oracle Net Manager graphical user interface tool to select encryption algorithms, or else manually set six `sqlnet.ora` parameters. Once configured, the encryption is transparent to users.

Very little overhead is associated with Oracle Advanced Security encryption. Performance varies (depending on the operating system, the encryption algorithm

chosen, and other factors); however, performance tests show a degradation of approximately one-tenth of a second.

Data Integrity Features of Oracle Advanced Security

Oracle Advanced Security ensures data integrity with sequenced, cryptographic checksums. To ensure that data has not been modified, deleted, or replayed during transmission, Oracle Advanced Security optionally generates a cryptographically secure message digest—through cryptographic checksums using the MD5 algorithm—and includes it with each packet sent across the network. Alternatively, Oracle Advanced Security can use SHA-1 (with SSL). Data integrity algorithms add little overhead, and protect against data modification attacks, deleted packets, and replay attacks.

Secure Sockets Layer (SSL) Encryption Capabilities

Oracle Advanced Security provides SSL encryption capabilities, as described in this section.

Oracle Advanced Security Support for SSL

The Oracle Advanced Security SSL feature can be used to secure communications between any client and any server. This includes data in Oracle Net Services, LDAP, JDBC-OCI, and IIOP format. SSL encryption provides users with an alternative to the native Oracle Net Services encryption protocol which is supported in Oracle Advanced Security. A benefit of SSL is that it is a de facto Internet standard, and can be used with clients which use protocols other than Oracle Net Services.

SSL support in Oracle Advanced Security encrypts network traffic and provides integrity checking, authenticates Oracle clients and servers, and brings public key-based single sign-on to the Oracle environment. SSL provides encryption and data integrity through the use of cipher suites, which are sets of authentication, encryption, and data integrity types. The client and server each have a list of cipher suites they support (such as RSA for authentication, with 3DES for encryption and SHA-1 for data integrity). They negotiate which one is to be used during connection.

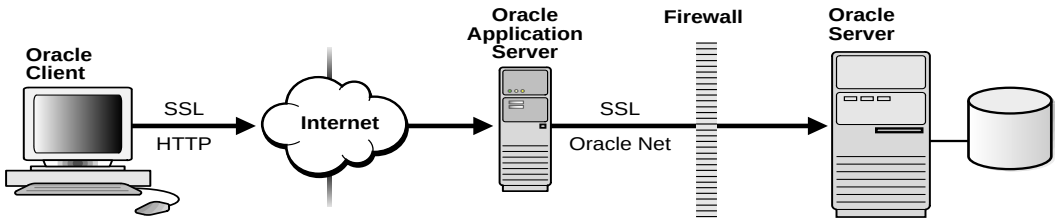
Checksumming in Oracle Advanced Security SSL

The SSL feature of Oracle Advanced Security permits the use of the Secure Hash Algorithm (SHA) as well as MD5. SHA is slightly slower than MD5, but produces a larger message digest to make it more secure against brute-force collision and inversion attacks.

Oracle9*i* Application Server Support for SSL

Oracle9*i* Application Server supports SSL encryption between thin clients and the Oracle9*i* Application Server, as well as between Oracle9*i* Application Server and Oracle9*i* Data Server.

Figure 9–3 SSL Secures Internet and Oracle Communications



See Also: "Secure Sockets Layer (SSL) Protocol" on page 3-6

"Secure Sockets Layer Authentication and X.509v3 Digital Certificates" on page 8-7

Java Encryption Features of Oracle Advanced Security

Sun Microsystems defined the Java Database Connectivity (JDBC) standard, and Oracle Corporation, as an individual provider, implements and extends the standard with its own JDBC drivers. Oracle offers 4 kinds of JDBC driver:

- JDBC-OCI built on top of OCI libraries and Oracle Net Services client
- JDBC thin for both applets and applications (that do not have client installation and want maximum portability)
- JDBC server-side thin for connecting to remote Oracle databases from inside the target database
- Server-side internal driver for code such as stored procedures and functions executing inside the database.

JDBC-OCI Driver Since the JDBC-OCI driver uses the full Oracle Net Services communications stack on both client and server, it can take advantage of existing Oracle Advanced Security encryption and authentication mechanisms. In Oracle9*i*, proxy authentication has been extended to Java Database Connectivity (JDBC-OCI),

which enables a middle tier server to access the Oracle9*i* database on behalf of a client user by establishing a lightweight session for the user.

Thin JDBC

Because the thin JDBC driver is designed to be used with downloadable applets used over the Internet, Oracle9*i* includes a 100% Java implementation of Oracle Advanced Security encryption and integrity algorithms for use with thin clients. Several benefits enable eBusinesses deploying Oracle and other components to securely transmit a variety of information over a variety of channels:

- Data encryption for privacy of communications
- Data integrity checking to safeguard against data modification, replay, and eavesdropping
- Secure connections from thin JDBC clients to the Oracle9*i* database
- Ability for developers to build applets that transmit data over a secure communication channel
- Secure connections from Oracle9*i* databases to older versions of Oracle Advanced Security-enabled databases
- Secure connections from middle tier servers with Java Server Pages (JSP) to the Oracle RDBMS

The Oracle JDBC Thin driver implements the Oracle password protocol for authentication. It does not support Oracle Advanced Security SSL implementation, nor does it support third party authentication features such as RADIUS or Kerberos. The Oracle JDBC-OCI driver supports all Oracle Advanced Security features.

Oracle Advanced Security continues to encrypt and provide integrity checking of Oracle Net Services traffic between Oracle Net Services clients and Oracle servers using algorithms written in C. The Oracle Advanced Security Java implementation for Thin JDBC provides Java versions of the following encryption algorithms:

- RC4_256
- RC4_128
- RC4_56
- RC4_40
- DES56
- DES40

Secure Connections for Virtually Any Client

On the server, the negotiation of algorithms and the generation of keys function exactly the same as Oracle Advanced Security Oracle Net Services encryption, thus enabling backward and forward compatibility of clients and servers. On the clients, the algorithm negotiation and key generation occur in exactly the same manner as C-based Oracle Advanced Security encryption. The client and server negotiate encryption algorithms, generate random numbers, use Diffie-Hellman to exchange session keys, and use the Oracle Password Protocol, in the same manner as traditional Oracle Net Services clients. Thin JDBC contains a complete implementation of a Oracle Net Services client in pure Java. Consistent with other encryption implementations, the Java implementation of Oracle Advanced Security prevents access to the cryptographic algorithms, makes it impossible to double encrypt data, and encrypts data as it passes through the network. Users cannot alter the keyspace nor alter the encryption algorithms themselves.

See Also: Java Database Connectivity (JDBC) on page 3-8

Oracle Java SSL

Oracle Java SSL is a commercial-grade implementation of Java Secure Socket Extension (JSSE). In order to create a secure, fast implementation of SSL, Oracle Java SSL uses native code to improve the performance of critical components. In addition to the functionality included in the JSSE specifications, Oracle Java SSL supports the following:

- Multiple cryptographic algorithms
- Certificate and key management using Oracle Wallet Manager
- SSL-specific session capabilities, including authentication, that can be used by applications built on top of Oracle Java SSL

See Also: *Oracle Advanced Security Administrator's Guide*

Strong Authentication Methods Supported by Oracle Advanced Security

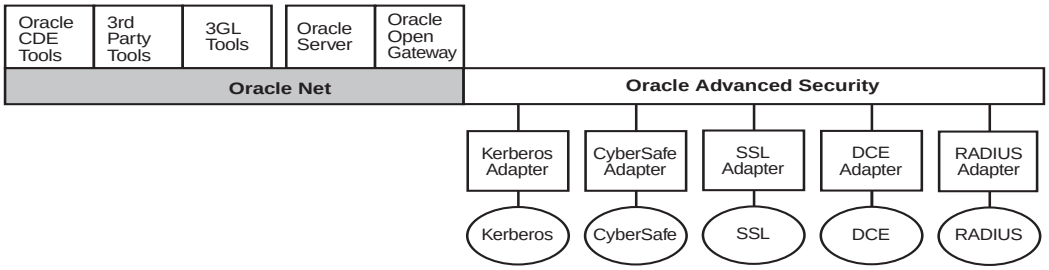
Oracle Advanced Security provides enhanced user authentication through several third-party authentication services, and through the use of SSL with digital certificates. Many of these options use centralized authentication, which can give you high confidence in the identity of users, clients, and servers in distributed environments. It also provides for enhanced authentication by integrating technologies such as token cards to prove users' identities. User authentication, a function of Oracle9*i*, is significantly enhanced by using the authentication methods supported by Oracle Advanced Security.

Supported authentication methods include:

- Oracle Public Key Infrastructure-Based Authentication
  - Secure Sockets Layer (SSL) Authentication in Oracle Advanced Security
  - Entrust/PKI Support in Oracle Advanced Security
  - Standard PKI Support in Oracle Advanced Security
- RADIUS with Oracle Advanced Security
- Kerberos and CyberSafe with Oracle Advanced Security
- Smart Cards with Oracle Advanced Security
- Token Cards with Oracle Advanced Security
- Biometric Authentication with Oracle Advanced Security
- Distributed Computing Environment (DCE) with Oracle Advanced Security

Figure 9–4 shows some of the strong authentication components of Oracle Advanced Security. The authentication adapters integrate below the Oracle Net Services interface and permit existing applications to take advantage of new authentication systems transparently, without any changes to the application.

Figure 9–4 Oracle Net Services with Authentication Adapters



See Also: "Strong Authentication" on page 4-3

Oracle Public Key Infrastructure-Based Authentication

Oracle provides a public key infrastructure (PKI) for using public keys and certificates. This section summarizes the Oracle PKI authentication capabilities.

- Standard PKI Support in Oracle Advanced Security
- Secure Sockets Layer (SSL) Authentication in Oracle Advanced Security
- Entrust/PKI Support in Oracle Advanced Security

Standard PKI Support in Oracle Advanced Security

Oracle9*i* supports standard X.509 version 3 certificates and relevant Public Key Certificate Standards (PKCS) for certificate request and installation. This enables users to request certificates from any certificate authority (CA) which also supports these standards. It also enables users to install trusted root certificates from their choice of CAs, enabling the server to recognize and validate certificates issued by those CAs. Oracle works with leading PKI service and product vendors, including VeriSign and Baltimore Technologies, to ensure that their CA trusted roots are pre-installed in Oracle9*i*, enabling customers to use certificates from those vendors to authenticate to Oracle9*i* out-of-the-box.

See Also: "PKI Implementation in Oracle Advanced Security" on page 9-42

Secure Sockets Layer (SSL) Authentication in Oracle Advanced Security

Oracle Advanced Security SSL can be used to authenticate:

- Any client or server to one or more Oracle servers
- An Oracle server to any client

As in Oracle9*i*, anonymous, server-only, and client/server authentication by X.509 certificates are supported.

SSL features can be used by themselves or in combination with other authentication methods supported by Oracle Advanced Security. For example, SSL can be used with Kerberos, using the encryption provided by SSL in combination with the Kerberos authentication method.

Users and administrators use Oracle Wallet Manager to manage digital certificates for use with SSL. Oracle Wallet Manager gives users and database administrators control over the contents of their wallets. The administrator can centrally manage wallets on an LDAP-compliant directory. Oracle Enterprise Login Assistant, an easy-to-use tool, is provided for end users to open the wallet and perform the login over SSL. This tool enables users to achieve single sign-on, simply and transparently, using certificates for authentication. The wallet and management tools are used together to securely store and manage certificates, private keys, and requests to certificate server.

See Also: "Secure Sockets Layer Authentication and X.509v3 Digital Certificates" on page 8-7

"Oracle Wallet Manager" on page 9-43

Entrust/PKI Support in Oracle Advanced Security

Oracle Advanced Security enables customers of both Oracle Corporation and Entrust Technologies, Inc. to incorporate Entrust-based single sign-on into their Oracle applications. By integrating with Entrust/PKI, Oracle enhances its ability to provide X.509-based single sign-on to large customers who require the extensive key management, certificate revocation, and other features that Entrust provides.

Oracle Advanced Security supports Entrust Profile, which is the Entrust mechanism for storage of certificates and private keys and for secure credential management. Instead of accessing user credentials (private key and certificate) from an Oracle wallet, Oracle Advanced Security can access a user's Entrust Profile for authentication and single sign-on. Entrust integration requires Entrust Authority 5.

See Also: "Entrust/PKI Authentication" on page 8-8

Kerberos and CyberSafe with Oracle Advanced Security

Oracle Advanced Security support for Kerberos and CyberSafe provides the benefits of single sign-on and centralized authentication of Oracle users.

Note: Oracle authentication for Kerberos provides database link authentication (also called proxy authentication). CyberSafe does not support proxy authentication.

See Also: "Kerberos and CyberSafe" on page 4-4

RADIUS with Oracle Advanced Security

RADIUS (the Remote Authentication Dial-In User Service) support provides two major benefits for Oracle customers. First, it enables support for authentication technologies including token cards, smart cards, and challenge-response. Second, it readily integrates into existing systems by making the Oracle9*i* data server a RADIUS client, thus capitalizing on the infrastructure and investment that organizations have already made.

With RADIUS you can choose virtually any mechanism available to authenticate network users. Many token and smart card manufacturers support RADIUS, and any RADIUS-compliant device can integrate with Oracle Advanced Security to authenticate Oracle users with little modification required by the authentication provider. Since many organizations have implemented RADIUS for remote access to their networks, Oracle easily integrates into existing systems and takes advantage of the investments that an organization has already made.

Any third party authentication vendor can implement the client graphical user interface by customizing the Java interface class that ships with Oracle Advanced Security. Products from the following vendors integrate with Oracle Advanced Security by means of the RADIUS interface:

- ActivCard Inc. ActivRadius, ActivCard Gold Tokens and Smart Cards
- Lucent Technologies Radius Server

- RSA ACE/Server. This is a RADIUS server, and SecurID tokens authenticate Oracle users through the RADIUS interface.
- Secure Computing SafeWord Products, including the SafeWord RADIUS server and SafeWord tokens
- Funk Software Steel Belted Radius: They provide an API for third party authentication vendors to write the authentication backend.

See Also: "RADIUS" on page 4-4

Token Cards with Oracle Advanced Security

Token card technologies enhance user authentication. Oracle Advanced Security supports SecurID tokens from RSA, which strengthen security through two-factor authentication: the user must know the PIN and have the SecurID electronic token card. In addition, RADIUS support in Oracle Advanced Security permits integration with a variety of token cards. Organizations can choose which token(s) they would like to use to protect networks from unauthorized use.

See Also: "Token Cards" on page 4-5

Smart Cards with Oracle Advanced Security

Oracle Advanced Security integrates with RADIUS-compliant smart cards, in order to authenticate Oracle users. Smart cards are becoming popular as strong security devices. Since they contain a processor, they can generate dynamic passwords. Because they have memory, they are useful for storing data such as a username, a certificate, or a medical record. Smart cards are being widely deployed, and organizations relying on them for proof of user identities can do so when users connect to Oracle.

See Also: "Smart Cards" on page 4-6

Biometric Authentication with Oracle Advanced Security

A biometric device vendor who supports RADIUS can integrate with Oracle Advanced Security. The biometric device, deployed on clients and/or servers requiring strong authentication, provides user authentication based on a physical characteristic of an individual.

Distributed Computing Environment (DCE) with Oracle Advanced Security

Distributed Computing Environment (DCE) integration enables users to transparently use Oracle tools and applications to access Oracle9*i* databases in a DCE environment. Oracle Advanced Security supports DCE 1.0 from OSF, on certain platforms, such as Solaris, Windows, HP, AIX.

You can integrate your Oracle network with any or all of the DCE services, which include security services, authentication and single sign-on, and mapping of Oracle roles to DCE groups for central authorization management.

See Also: "Distributed Computing Environment (DCE)" on page 4-7

Single Sign-On Implementations in Oracle Advanced Security

Oracle Advanced Security minimizes maintenance of multiple passwords by supporting secure, single sign-on capabilities in a distributed environment. A user only needs to log on once a day, and can automatically connect to other services without having to give a user name and password again. This eliminates both the need for the user to remember and administer multiple passwords, and the time spent logging into multiple services. Single sign-on also simplifies management of user accounts and passwords for system administrators.

Centralized authentication makes single sign-on possible. Different configurations are supported:

- Single Sign-On Configuration with Third-Party Products
- PKI-Based Single Sign-On Configuration

See Also: "Single Signon" on page 4-10

"Single Sign-On Using PKI" on page 8-9

"Single Sign-On in Oracle9i Application Server" on page 9-61

Single Sign-On Configuration with Third-Party Products

Oracle Advanced Security is integrated with several different technologies to support single sign-on functionality. These include Kerberos, CyberSafe, and DCE.

PKI-Based Single Sign-On Configuration

Oracle Advanced Security provides SSL-based single sign-on and Entrust-based single sign-on for Oracle users by integrating with LDAP v3-compliant directory services. The combination of integrated directory services and the Oracle PKI implementation enable SSL-based single sign-on to Oracle9*i* databases. Single sign-on lets users be authenticated once, with subsequent connections relying on the user's digital certificate.

Enterprise User Security Features of Oracle Advanced Security

Enterprise User Security addresses user, administrative, and security challenges by centralizing storage and management of user-related information in an LDAP-compliant directory service. When an employee changes jobs in such an environment, the administrator need only modify information in one location—the directory—to make effective changes in multiple databases and systems. This centralization can substantially lower administrative costs while materially improving enterprise security.

This release extends Enterprise User Security support into three-tier environments. Oracle9*i* proxy authentication features enable:

- Proxy of user names and passwords through multiple tiers
- Credential proxy of X.509 certificates and distinguished names through multiple tiers

Note that this combination applies to both SSL-authenticated and password-authenticated enterprise users.

This section describes:

- Password-Authenticated Enterprise Users
- Shared Schemas in Oracle Advanced Security
- Current User Database Links
- Directory Integration
- Tools for Enterprise User Security

See Also: *Oracle9i Application Developer's Guide - Fundamentals*

Password-Authenticated Enterprise Users

Oracle Advanced Security enables two types of enterprise users: those authenticated by SSL, and those authenticated with passwords.

SSL-authenticated users benefit from single sign-on to Oracle9*i* using industry-standard, interoperable X.509 v3 certificates over Secure Sockets Layer v3.

Oracle Advanced Security also implements password-based authentication for enterprise users, eliminating the requirement for client-side wallets and most Secure Socket Layer (SSL) processing. (SSL is still required to secure connections between the database and Oracle Internet Directory.) Password-authenticated enterprise users can use the same password, securely stored in the LDAP-compliant directory, to authenticate to multiple databases. Administrators can manage both types of user within one directory.

With its reduced processing overhead, improved ease-of-use, and simplified setup and administration, password-authenticated enterprise users are particularly useful for large user communities accessing multiple applications. Oracle Advanced Security supports enterprise user logins with password-based authentication for all prior Oracle client versions. Furthermore, enterprise users can use a single enterprise username and password to connect to multiple databases, if desired.

Note that with single sign-on, the user needs to be authenticated only once. With single password functionality, by contrast, the user can employ the same password for many different databases, but she may need to enter the password multiple times.

See Also: "Password-Authenticated Enterprise Users" on page 6-5

Tools for Enterprise User Security

Three graphical user interfaces are provided with Oracle Advanced Security. You can use these tools to administer large user communities accessing multiple applications, and to ease logins for users.

- Oracle Enterprise Security Manager
- Oracle Wallet Manager
- Oracle Enterprise Login Assistant

See Also: "Components of Oracle Public Key Infrastructure-Based Authentication" on page 9-42, where these tools are described

Shared Schemas in Oracle Advanced Security

With shared schemas (formerly known as schema-independent users), multiple enterprise users can share a database schema. In this way, there is no need to create the same users in each database. The payoff for deploying the directory in this way is fewer user accounts. It enables application developers to integrate user accounts, and scales user management to the Internet.

See Also: "Shared Schemas" on page 6-4

Current User Database Links

Oracle Advanced Security provides a new type of database link for SSL-enabled databases and enterprise users. Current user database links enable the user to connect as the procedure owner (or connected user) to the next database. The current user can access the procedure owner's tables in the next database.

They provide SSL mutual authentication between servers. Databases can trust one another for access control and other things. To use database links, databases must have implemented enterprise users and SSL. Database links used with SSL implemented with the Distributed_Trust_Admin package permits finer-grained control by the DBA.

Directory Integration

The Oracle Advanced Security licence includes use of Oracle Internet Directory for storage of enterprise users, their passwords, their Oracle wallets, and their enterprise roles. In conjunction with other components of Oracle Advanced Security, Oracle Internet Directory enables you to accomplish centralized user management and authorization. Oracle Enterprise Security Manager is provided to create user entries in the directory, and manage authorizations for those users.

Oracle Advanced Security also supports the Microsoft Active Directory.

See Also: Chapter 6, "Administering Enterprise User Security"

"Oracle Internet Directory" on page 9-48

PKI Implementation in Oracle Advanced Security

Oracle offers a variety of choices for implementing security, and the Public Key Infrastructure (PKI) approach is just one of those choices. PKI is an emerging means of achieving security and single sign-on, adding extra value to the Oracle Advanced Security option. For example, to use RSA RC4 encryption, you can use Oracle Advanced Security for its native encryption component, or for SSL-based encryption. You can choose between a variety of authentication methods, including passwords, smart cards, and X.509 certificates. This section describes:

- Components of Oracle Public Key Infrastructure-Based Authentication
- PKI Integration and Interoperability
- Oracle PKI Implementation Summary

Components of Oracle Public Key Infrastructure-Based Authentication

The Oracle9*i* PKI implementation establishes a secure information exchange, in compliance with the industry-standard specifications for PKI-related components. It incorporates a whole suite of products and features, as described in this section.

Secure Sockets Layer

This protocol uses public key cryptography to provide authentication, secure session key management, encryption, and data integrity.

Oracle Call Interface

Oracle Call Interface (OCI) and PL/SQL functions are used to sign user-specified data using a private key and certificate, and to verify the signature on data using a trusted certificate.

Trusted Certificates

A trusted certificate is a third-party identity that is trusted. The trust is used when an identity is being validated as the entity it claims to be. Typically, the certificate authorities you trust issue user certificates. If there are several levels of trusted certificates, a trusted certificate at a lower level in the certificate chain does not need to have all its higher level certificates re-verified. Oracle Advanced Security automatically installs trusted certificates from VeriSign, RSA, Entrust, and GTE CyberTrust.

X.509 Version 3 Certificates

Such a certificate is created when an entity's public key is signed by a trusted entity (a trusted Certificate Authority outside of Oracle). The certificate contains the identity of the user or service, a public key, and other information used to enable authentication. It ensures that the entity's information is correct and that the public key belongs to the entity. The certificate is loaded into an Oracle wallet to enable authentication.

Oracle Wallets

An Oracle wallet is a container in which certificates and trusted certificates are stored and managed, such that there is no need for real time checking with the certificate authority. These data structures securely store a user private key, a user certificate, and a set of trusted certificates (the list of root certificates which the user trusts).

Oracle Wallet Manager

This is a Java-based application that security administrators use to manage public-key security credentials on both Oracle clients and database servers. It creates an Oracle wallet that can be opened using the Oracle Enterprise Login Assistant.

Oracle Wallet Manager creates a public-private key pair and manages credentials for a user. It issues PKCS#10 certificate requests to the certificate authority, and installs the certificate in the wallet. It ships with trusted certificates from VeriSign, RSA, and GTE CyberTrust, and can use a site's own in-house certificate authority.

Oracle Enterprise Login Assistant

Oracle Enterprise Login Assistant is a Java-based tool for opening and closing a user wallet in order to enable or disable secure SSL-based communications for an application. It provides single sign-on capability for SSL-authenticated users. Further, it provides the benefit of strong authentication as well as single sign-on in either a client/server or three-tier environment. Enterprise users authenticating with passwords also use it to change their passwords in a database or a directory.

Oracle Internet Directory

This LDAP v3-compliant directory, built on the Oracle9*i* database, helps to enable PKI-based single sign-on. It enables you to securely manage the user and system configuration environment, including security attributes and privileges, for users authenticated using X.509 certificates. Oracle Internet Directory enforces

attribute-level access control, enabling the directory to restrict read, write, or update privileges on specific attributes to specific named users (for example, an enterprise security administrator). It also supports protection and authentication of directory queries and responses through SSL encryption.

Oracle Enterprise Security Manager

Oracle Enterprise Security Manager is the graphical user interface used to centrally administer enterprise users and enterprise roles, in an LDAP directory. Database administrators can use this tool to perform a variety of tasks, including the following:

- Create new enterprise domains
- Assign enrolled users and published databases in an enterprise domain
- Authorize enterprise roles on databases in the enterprise domain for the identities within the enterprise domain. It lets you store and retrieve roles from Oracle Internet Directory if the roles support LDAP. It may also enable you to store roles in other LDAP v3-compliant directory servers if they can support the installation of the Oracle schema and related Access Control Lists.

Oracle Enterprise Security Manager launches out of Oracle Enterprise Manager. It scales to tens of thousands of users, and enables you to manage thousands of databases in various domains, as well as the users who connect to the databases.

PKI Integration and Interoperability

Oracle9*i* expands PKI integration and interoperability through:

- PKCS #12 Support
- Wallets Stored in Oracle Internet Directory
- Multiple Certificate Support
- Strong Wallet Encryption

PKCS #12 Support

Oracle Advanced Security supports X.509 certificates stored in PKCS #12 containers, making the Oracle wallet interoperable with third party applications like Netscape Communicator 4.x and Microsoft Internet Explorer 5.x, and providing wallet portability across operating systems. Users who have existing PKI credentials may export them in PKCS#12 format and reuse them in Oracle Wallet Manager, and vice versa. PKCS#12 thus increases interoperability and reduces the cost of PKI deployment for organizations.

Wallets Stored in Oracle Internet Directory

Oracle Enterprise Security Manager creates user wallets as part of the user enrollment process. The wallet is stored in Oracle Internet Directory, or other LDAP-compliant directory. Oracle Wallet Manager can upload wallets to—and retrieve them from—the LDAP directory.

Storing the wallet in a centralized LDAP-compliant directory supports user roaming, enabling users to access their credentials from multiple locations or devices, ensuring consistent and reliable user authentication, while providing centralized wallet management throughout the wallet life cycle.

Multiple Certificate Support

In Oracle9*i*, Oracle Wallet Manager and Oracle Enterprise Login Assistant support multiple certificates for each wallet, including:

- S/MIME signing certificate
- S/MIME encryption certificate
- Code-signing certificate

Oracle Wallet Manager supports multiple certificates for a single digital entity in a persona—with multiple private key pairs in a persona (each private key can match only one certificate). This enables consolidation of and more secure management of users' PKI credentials.

Strong Wallet Encryption

The private keys associated with X.509 certificates require strong encryption, over secure channels. Oracle9*i* replaces DES encryption with 3-key triple DES (3DES), which is a substantially stronger encryption algorithm and provides superior security for Oracle wallets.

Oracle PKI Implementation Summary

As the public key infrastructure is deployed more frequently to secure such applications as email and electronic commerce, PKI is one of the most important investments companies are making. Because all clients, application servers and data servers can authenticate themselves to one another, PKI provides an important security infrastructure to a network.

SSL secures not only Oracle Net, but also other protocols such as IIOP (Internet Inter-ORB Protocol). By capitalizing on Java support, Oracle Advanced Security secures IIOP connections, giving Oracle the ability to work with thin clients and Enterprise JavaBeans (EJB).

Support for SSL in Oracle Advanced Security closes the loop for secure end-to-end communications between any client, a web server or application server, and any Oracle9*i* database. For example, w hen a user wants to connect to her financial institution to transfer funds, she must be able to verify beyond a doubt that she is providing sensitive information such as passwords and account numbers to the proper server. With SSL and public-key authentication, the server can verify its identity to her browser, and the client can identify itself to the server.

Now that organizations are implementing application servers and firewalls to protect their networks, the connection process expands. Using the same example, the financial information can be stored in an Oracle9*i* data server secured behind a firewall. The user connects to the database using SSL to connect over the Internet and to the application server, which passes the connect request over Oracle Net (still protected with SSL) through a firewall and to the secured Oracle9*i* server with her financial account information.

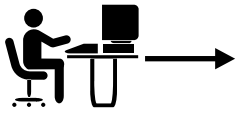
Certificates not only authenticate clients to servers, but they also authenticate servers to other servers. This expands the security of the entire system with secure database links for mutual authentication of servers. With SSL deployment, all clients and all servers, including database servers and application servers, have credentials that identify them to all other machines and services with which they communicate.

The complete package that Oracle delivers provides standards-based methods to prevent eavesdropping, tampering with, or forging messages sent over the network, while providing single sign-on and strong authentication of clients and servers in the network and over the Internet. A public key infrastructure paves the way for secure electronic commerce in the Information Age.

Oracle Label Security

Oracle Label Security, an add-on security option for the Oracle9*i* Enterprise Edition, enables you to customize your own label-based access control policies. Oracle Policy Manager is a convenient graphical user interface provided with this product.

Figure 9–5 Oracle Label Security



User session label
is UNCLASSIFIED

GRADE	RATE	ROW LABEL
Manager	600	UNCLASSIFIED
Senior	400	UNCLASSIFIED
Director	750	HIGHLY_SENSITIVE
Principal	600	SENSITIVE
Senior	450	SENSITIVE

This product enables an administrator to add label based access control to the access mediation process when standard access controls are insufficient. Oracle Label Security is built on the Virtual Private Database toolkit and requires no programming whatsoever. It mediates access to rows in database tables based on a label contained in the row, a label associated with each database session, and Oracle Label Security privileges assigned to the session. Oracle Label Security delivers a data dictionary and administrative tools you can use to construct valid labels, set user label authorizations and privileges, and apply the resulting Oracle Label Security policy to tables and schemes.

The Oracle Virtual Private Database toolkit and Oracle Label Security provide very useful mechanisms for hosting and exchanges. Virtual Private Database provides fine grained access control within the database. It can be configured to keep data from different organizations separate within a single database instance, so that organizations can share database tables but only see data which pertains to them. This makes it ideal for hosting, since a system administrator for a hosting company can set up and configure a single version of each application for which they provide hosted services, but use Virtual Private Database on the underlying application tables to provide separate virtual applications instances for each hosted customer. This can substantially reduce the costs associated with hosting. Because hardware, database, and applications instances can be shared, the costs associated with hardware, as well as installation and configuration of software, are lower than if physically separate instances were required for each hosted customer.

Oracle Label Security is particularly useful for hosting environments in which access to information can be formalized by means of sensitivity levels, access categories, or user groups. For these environments, Oracle Label Security makes it easy for hosting companies to define and administer label-based security policies. Oracle Label Security provides particular advantages for exchanges, because the label-based access policies include automatic, easy-to-administer "group" access embedded within a data label that can support communities of interest.

The label-based access policies of Oracle Label Security are also ideal for enforcing privacy concerns of users accessing eBusiness applications. Many consumers are reluctant to purchase goods and services over the Internet because of privacy concerns. With Oracle Label Security, data can be labeled with an "opt out" provision for users who do not wish their data to be used for targeted marketing campaigns, or who do not wish their purchasing data to be sold. Data labels—and therefore users' privacy policies—remain with the data, making it easy to secure and enforce user privacy preferences across multiple applications.

See Also: "Oracle Policy Manager" on page 9-20

Oracle Label Security Administrator's Guide

Oracle Internet Directory

Oracle Internet Directory is a directory service implemented as an application on the Oracle9*i* database. It enables retrieval of information about dispersed users and network resources. It combines Lightweight Directory Access Protocol (LDAP) Version 3, the open Internet standard directory access protocol, with the high performance, scalability, robustness, and availability of the Oracle9*i* Server.

Oracle Internet Directory is not itself a security product, but rather a technology for managing enterprise data very efficiently. It contributes to data security by supporting LDAP directory enterprise user security.

The Oracle platform has been designed to be LDAP-aware in numerous ways. The stringent security requirements of Oracle customers limits the choice of LDAP servers which are adequate to the task. In most cases, Oracle Internet Directory is the only LDAP server supported.

This section includes:

- Introduction to Oracle Internet Directory
- LDAP Compliance

- How Oracle Internet Directory Organizes Enterprise User Management

See Also: Chapter 5, "Using and Deploying a Secure Directory"

Chapter 6, "Administering Enterprise User Security"

Oracle Internet Directory Administrator's Guide

Introduction to Oracle Internet Directory

Oracle Internet Directory offers comprehensive and flexible support for directory access control. This includes entry level, attribute level, and prescriptive access control to provide varying levels of security to meet the specific needs of enterprise and service providers. An administrator can grant or control access to a specific directory object or to an entire directory subtree. Oracle Internet Directory implements three levels of user authentication: anonymous, password-based, and certificate-based using Secure Sockets Layer (SSL) Version 3 for authenticated access and data privacy.

In addition, Oracle Internet Directory provides many powerful features you can use in an enterprise or hosted environment to control access to application metadata—the information governing how applications behave and who can access them. To do this, you deploy the directory for administrative delegation. This deployment enables, for example, a global administrator to delegate to department administrators access to the metadata of applications in their departments. These department administrators can then control access to their department applications.

Oracle Internet Directory offers important benefits:

Table 9–2 Benefits of Oracle Internet Directory

Benefit	Explanation
Data Integrity	Oracle Internet Directory uses the Secure Sockets Layer (SSL) to ensure that data has not been modified, deleted, or replayed during transmission. SSL can generate a cryptographically secure message digest—through cryptographic checksums using either the MD5 algorithm or the Secure Hash Algorithm (SHA)—and include it with each packet sent across the network.
Data Privacy	Oracle Internet Directory ensures that data is not detected during transmission by using public-key encryption available with SSL

Table 9–2 Benefits of Oracle Internet Directory

Benefit	Explanation
Password Protection	To protect passwords, Oracle Internet Directory uses the MD4 algorithm as the default. MD4 is a one-way hash function that produces a 128-bit hash, or message digest.

The components of Oracle Internet Directory include:

Table 9–3 Components of Oracle Internet Directory

Benefit	Explanation
Oracle Directory Server	Responds to client requests for information about people and resources, and to updates of that information, using a multitiered architecture directly over TCP/IP
Oracle Directory Replication Server	Replicates LDAP data between Oracle Directory Servers
Oracle Directory Manager	A graphical user interface administration tool
Tools	A variety of command line administration and data management tools

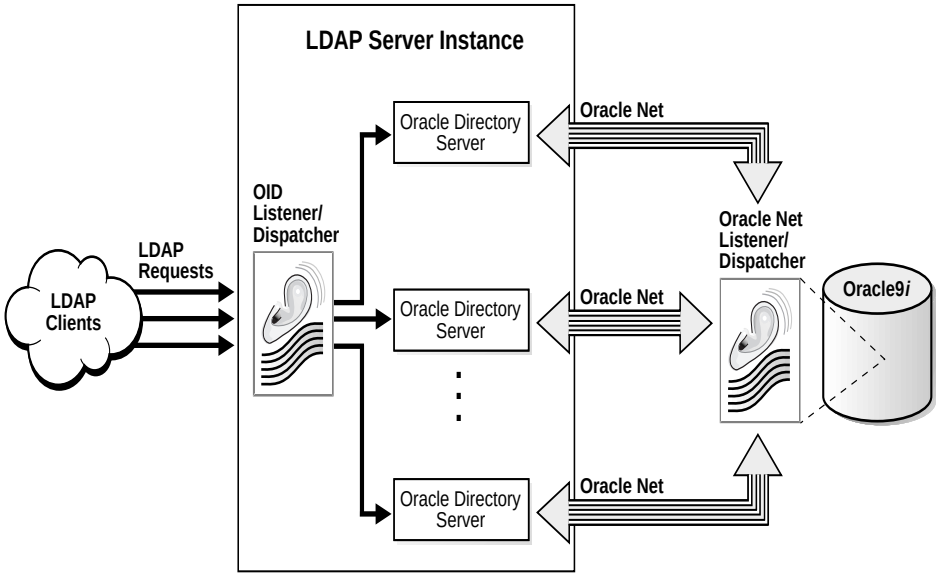
LDAP Compliance

The Lightweight Directory Access Protocol (LDAP) has been incorporated into the Oracle Internet Directory. Oracle Internet Directory is probably the most scalable LDAP directory. It leverages the intrinsic scalability of the Oracle9*i* database, simplifying the management of hundreds of thousands of users. LDAP Naming, along with support for the Oracle Internet Directory centralized directory service offers clients a new, unified naming mechanism in addition to the above technology.

Oracle Internet Directory implements Version 3 of the Lightweight Directory Access Protocol (LDAP). This is the emerging Internet standard for directory services. It is based on the earlier ISO X.500 Directory Access Protocol (DAP) standard, but simplifies that standard considerably, enabling LDAP to be more efficient, straightforward, and easier to implement. LDAP is especially suited for deployment with Internet-centric, "thin-client" applications.

Each LDAP directory server instance looks like the configuration in Figure 9–6.

Figure 9–6 LDAP Server Instance Architecture

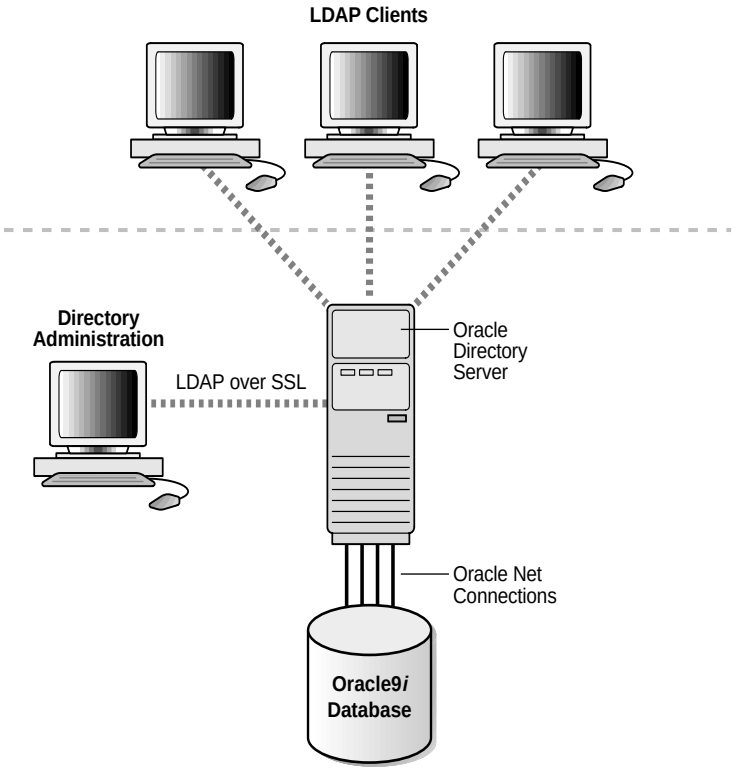


Oracle Advanced Security can integrate with LDAP version 3-compliant directories. Your Oracle Advanced Security license entitles you to deploy Oracle Internet Directory for user management as well as authorization storage and retrieval.

How Oracle Internet Directory is Implemented

An Oracle Internet Directory node is implemented as an application running on the Oracle9*i* server. To communicate with the database, which may be on the same platform or on a different one, the Oracle Internet Directory uses Oracle Net Services, the Oracle platform-independent database connectivity solution. This relationship is illustrated in Figure 9–7.

Figure 9–7 Oracle Internet Directory Architecture



How Oracle Internet Directory Organizes Enterprise User Management

This section describes enterprise user administration and shared schemas with Oracle Internet Directory.

Enterprise User Administration with Oracle Internet 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 permitted to update directory information about user privileges or access.

Enterprise roles are centrally-administered privilege sets, maintained in Oracle Internet Directory, or in directories from selected partners which meet Oracle 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.

See Also: "Secure Sockets Layer (SSL) Protocol" on page 3-6

Shared Schemas with Oracle Internet Directory

Oracle Internet Directory supports shared schemas, which extend the benefits of directory integration by enabling the database to delegate administration of user identity, as well as privilege, to the directory.

See Also: "Shared Schemas" on page 6-4

Oracle Net Services

Oracle Net Services is a software layer that resides on the client and the Oracle database server. It is responsible for establishing and maintaining the connection between the client application and server, as well as exchanging messages between them using industry-standard protocols. This section includes:

- Components of Oracle Net Services
- Firewall Support with Oracle Net Services
- Valid Node Checking in Oracle Net Services
- Database-Enforced VPD Network Access

See Also: *Oracle9i Net Services Administrator's Guide*

Components of Oracle Net Services

Oracle Net Services is composed of Oracle Net, the listener, and Oracle Connection Manager. This section includes:

Oracle Net on the Client

On the client side, applications communicate with Oracle Net Client to establish and maintain connections. Oracle Net Client, in turn, uses Oracle protocol support that is able to communicate with an industry-standard network protocol, such as TCP/IP, to communicate with the Oracle database server.

Oracle Net on the Database Server

The Oracle database server side is similar to the client side. A network protocol sends client request information to an Oracle protocol support layer, which then sends information to Oracle Net. Oracle Net then communicates with the Oracle database server to process the client request. The one operation unique to the Oracle database server side is the act of receiving the initial connection through a process called the listener. The listener brokers a client request, handing off the request to the server.

Oracle Protocol Support

Oracle Net uses Oracle protocol support to communicate with the following industry-standard network protocols:

- TCP/IP
- TCP/IP with SSL
- Named Pipes
- LU6.2
- VI

Oracle Connection Manager

Oracle Connection Manager is a software component that resides on its own computer, separate from a client or an Oracle database server. It proxies requests destined for the database server. You can also configure Oracle Connection Manager to multiplex sessions, control access, or convert protocols.

Protocol Conversion

As a protocol converter, Oracle Connection Manager enables a client and an Oracle database server that have different networking protocols to communicate with each other. Oracle Advanced Security is fully supported by Oracle Connection Manager, making secure data transfer a reality across network protocol boundaries. Clients and a database server configured with different network protocols can securely share data with one another. To eliminate potential weak points in the network infrastructure and to maximize performance, Connection Manager passes encrypted data from protocol to protocol without the cost and exposure of decryption and re-encryption.

Access Control

As an access control filter, Oracle Connection Manager controls access to Oracle databases. It can be configured to grant or deny clients access to a particular database service or a computer. By specifying filtering rules on source, destination, and database service name, you can permit or restrict specific clients access to a server.

Session Multiplexing

In its session multiplexing role, Oracle Connection Manager funnels multiple sessions through a single transport protocol connection to a particular destination. This reduces the demand on resources needed to maintain multiple sessions between two processes by enabling the Oracle database server to use fewer connection end points for incoming requests. You can thus increase the total number of network sessions that a server can handle. To increase the number of concurrent users, multiple instances of Oracle Connection Manager can be installed.

When Oracle Connection Manager is run on the same computer as an application Web server, the application Web server can route multiple client sessions through Oracle Connection Manager to ensure that those sessions have continuous access to an Oracle database server. This functionality is especially useful for Web applications where session availability and response time are major concerns.

Firewall Support with Oracle Net Services

Firewalls can be implemented in two ways:

- Firewalls Using Oracle Connection Manager in an Intranet Environment
- Firewalls Using Oracle Net Firewall Proxy in an Internet Environment

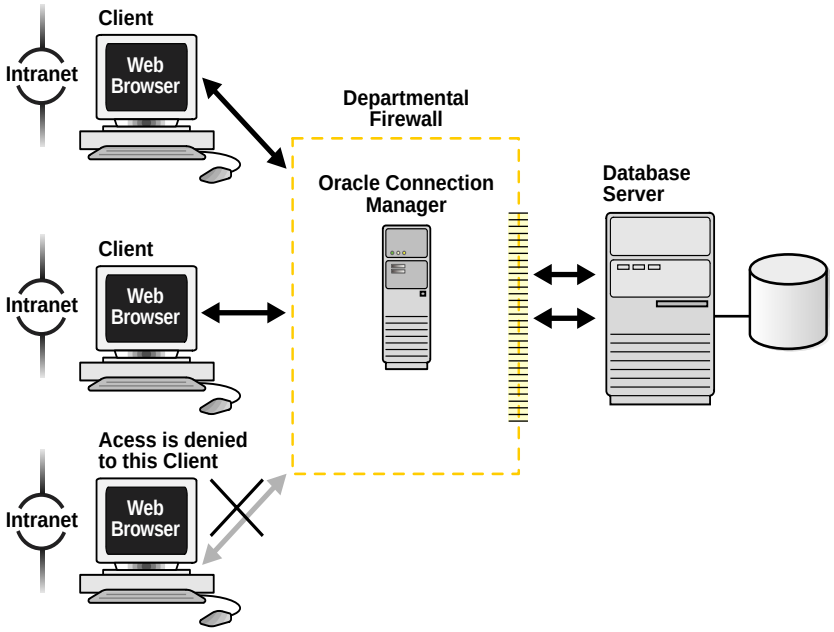
Firewalls Using Oracle Connection Manager in an Intranet Environment

Oracle Connection Manager can be deployed as a firewall within an intranet. It can be configured to grant or deny client access to a particular database service or a computer. By specifying filtering rules, you can permit or restrict specific client access to a server, based on the following criteria:

- Source host names or IP addresses for clients
- Destination host names or IP addresses for servers
- Destination database service names
- Client use of Oracle Advanced Security

Figure 9–8 shows an Oracle Connection Manager positioned between three Web clients and an Oracle database server. Oracle Connection Manager is configured to permit access to the first two Web clients and to deny access to the third. For this configuration to work, clients require the JDBC Thin driver.

Figure 9–8 Intranet Network Access Control with Oracle Connection Manager

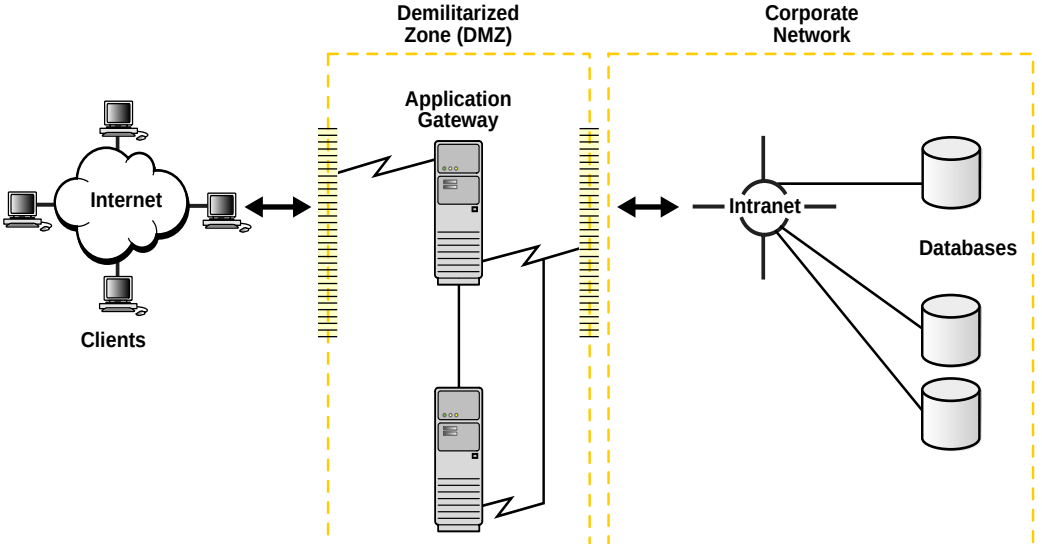


Firewalls Using Oracle Net Firewall Proxy in an Internet Environment

Oracle Corporation works with firewall vendors to incorporate key firewall technologies in its server products and thereby provide support for distributed database network traffic. Oracle Connection Manager functionality is offered by some firewall vendors through a software component called Oracle Net Firewall Proxy. A host computer, called an application gateway, runs the Oracle Connection Manager software.

Figure 9–9 shows an application gateway controlling traffic between internal and external networks and providing a single checkpoint for access control and auditing. As a result, unauthorized Internet hosts cannot directly access the database inside a corporation, but authorized users can still use Internet services outside the corporate network. This capability is critical in Internet environments to restrict remote access to sensitive data.

Figure 9–9 Internet Network Access Control with an Application Gateway



See Also: "Firewalls" on page 3-7

Valid Node Checking in Oracle Net Services

Besides using Oracle Connection Manager to check for valid nodes, two Oracle Net Services protocol-specific parameters (TCP.EXCLUDED_NODES and TCP.INVITED_NODES) enable you to configure client access control to the database. In addition, you can use the parameter TCP.VALIDNODE_CHECKING to check for the TCP.INVITED_NODES and TCP.EXCLUDED_NODES to determine which clients to permit or deny access.

See Also: "Native Network Capabilities (Valid Node Checking)" on page 3-3

Database-Enforced VPD Network Access

You can also use Virtual Private Database (or secure application roles) to limit access to the database from particular network nodes. Note that you would not want to make IP address a primary way of authenticating or authorizing users,

since IP addresses can be spoofed. However, you can use IP address as an additional qualifier to limit data access for users whose authentication was separately established. For example, user Jane may have access to the EMP table, but company policy may dictate that she is not permitted to access EMP data unless she is inside the corporate intranet—perhaps even from a particular subnet for the HR department.

Both VPD and a secure application role can be used to limit access to data based on IP address. In the case of VPD, a policy function can access the IP address of the client connection using the USERENV naming context as follows:

```
SYS_CONTEXT('userenv', 'ip_address')
```

Further, the policy function can be defined to permit access to data only if the IP address is within the range of acceptable values (such as inside the corporate intranet, or within the range of addresses reserved for the HR department).

In the case of proxy authentication, in which the IP address of the client connection is the IP address of the application server that initiated the lightweight session, you can effectively force users to access the database only through the application server. Specifically, the VPD policy function can use the USERENV naming context

```
SYS_CONTEXT('userenv', 'ip_address')
```

to enforce that no records are returned unless the IP address matches that of the application server.

Note that this does not prohibit users from connecting directly to the database, as long as they are properly authenticated. Rather, it merely restricts the records returned to them. Using valid_node checking is actually a better general way to control user access, since it directly restricts connection to the database to particular IP addresses.

The secure application role can also use the USERENV naming context (that is, `SYS_CONTEXT('userenv', 'ip_address')`) to permit the enabling of roles only when connecting from a particular IP address.

Policies can be used on views, achieving a far lower cost in dictionary processing. Policies can also be used on synonyms, enabling applications that rely on synonymsto achieve better security by using VPD.

Oracle9*i* Application Server

Oracle9*i* Application Server is a reliable, scalable, secure, middle-tier application server designed to support your evolution into an eBusiness. With this product, the

technological complexity of assembling a complete middle-tier Internet infrastructure is managed for you. Oracle9*i* Application Server provides an infrastructure that can grow with your business. It can start small and support growing numbers of users and sophisticated functionality on your web sites.

This section introduces security features of Oracle9*i* Application Server, which includes components that provide a general framework for development and deployment of applications, such as the HTTP Server and Portal, and components which provide specific application services or functionality. This section focuses on the security services provided by the HTTP Server and Portal, since that is where the general security functionality in Oracle9*i* Application Server is implemented.

- Oracle HTTP Server
- Oracle Portal
- Single Sign-On in Oracle9i Application Server

See Also: Oracle documentation and collateral on the Oracle9*i* Application Server

Oracle HTTP Server

The Oracle HTTP Server is a web server based on Apache, an open source web server which is among the most widely adopted web server products. The Oracle HTTP Server extends Apache with a variety of standard and Oracle-unique enhancements (or "mods," as they are referred to in the Apache community). It provides a basic HTTP listener capability as well as the ability to support both static and dynamic web pages. In addition, it provides security services such as Secure Sockets Layer (SSL) encryption, and integration with the other Oracle9*i* Application Server components and products such as the Oracle database.

The Oracle9*i* Application Server provides a comprehensive set of security services. These include the ability to restrict or permit access to files and services based on the identity of users established by means of basic challenge/response operations, by means of client-supplied X.509 certificates, and by means of IP or hostname addresses. Confidentiality is provided by the SSL protocol, which is also used to present X.509 certificates to the HTTP Server. In addition, the HTTP Server provides logging and other facilities needed to detect and resolve intrusion attempts.

Oracle Portal

Oracle Portal is a key component of Oracle Corporation's product offering in the "Enterprise Portal" category. This is an emerging class of products which provide a gateway to business-related information on corporate intranets, in the same way that Internet portals are the gateway to content on the Internet.

Enterprise portals, as a both a consolidation and extension of existing market spaces, is a logical market for Oracle Corporation, which has a strong technology base, a wide range of applications which manage critical business data (ERP / CRM / BI), and a framework which leverages the technology to bring the applications together with other datastores on the Intranet (Oracle Portal). The functionality built into Oracle Portal provides a common framework across multiple Oracle products and applications. A customer who has purchased "portal enabled" Oracle products can easily extend them to other uses in an incremental fashion, as dictated by business needs and priorities.

Single Sign-On in Oracle9*i* Application Server

An important security feature of Oracle9*i* Application Server is support for single sign-on (SSO) to web-based applications. There are a number of reasons why businesses are considering SSO. These include the increasing use of web-based eBusiness applications which companies are deploying for use by employees, customers and partners. Without SSO, each user must maintain a separate identity and password for each application they access. Maintaining multiple accounts and passwords for each user is insecure and expensive. This section describes:

- Web SSO Technology
- Login Server
- LDAP Integration
- PKI Support
- Multitier Integration
- Oracle Single Sign-On Summary

Web SSO Technology

The Oracle Web SSO technology provides single sign-on for web users. It is designed to work in a portal environment such as that provided by Oracle9*i* Application Server, where multiple web-based applications are accessible through the portal.

Two types of applications are supported by Web SSO. Partner applications are those which work within the SSO framework and rely on the SSO service for authentication of users. External applications continue to use their own usernames and passwords. The Oracle Web SSO approach is based on cookies, which are created both by partner applications and a centralized server called the Login Server.

Login Server

The core of Oracle Corporation's SSO technology is the Login Server. This product provides web-based single sign-on and integration with legacy applications. With single sign-on, the Login Server authenticates users, and passes their identity securely to partner applications. The Login Server prompts users for a username and password when they access the system for the first time in a given time period (usually a day), and verifies the password presented by the user. Login Server SSO uses cookies, which are formatted pieces of information stored on a browser client by a web server. Cookies permit web servers to store and retrieve information about the client user, effectively maintaining client state information in the otherwise stateless web environment. Cookies are supported by all current browsers, although they can be disabled by the user (in which case Login Server will not provide SSO).

LDAP Integration

Oracle Login Server permits SSO usernames and passwords to be verified using Oracle Internet Directory. When a user submits an SSO username and password to the Login Server as part of the initial authentication, the Login Server performs an LDAP-bind against the Oracle Internet Directory using this username and password. If the LDAP-bind succeeds, the SSO username and password is considered to be verified.

PKI Support

PKI authentication is beginning to replace passwords in many applications. In web-based applications, PKI authentication is typically performed through an exchange of X.509 certificates, as part of a Secure Sockets Layer (SSL) session establishment. PKI by itself can be used to provide SSO, since a user with a certificate can authenticate to multiple applications without entering a password. In the future, users will be able to authenticate to the Login Server by means of PKI. This will provide SSO both to web-based applications supported by Login Server, and other PKI-enabled applications.

Multitier Integration

The Login Server provides SSO for web client access to web servers. Web servers are increasingly being deployed as the middle tier in a three tier architecture, where they provide access to a back-end tier database. It is desirable that users who access web applications that require access to the database not have to supply a database username and password for access to data stored there. Although the Login Server does not support non-web based applications, the Oracle database includes features specifically designed to support secure access to databases through three-tier architectures.

Oracle Single Sign-On Summary

The Oracle strategy for SSO encompasses a variety of technologies. For the growing field of web-based applications, Oracle has developed an SSO framework and Login Server which is specifically designed to provide web SSO. The Oracle Web SSO approach has a number of benefits. It provides a framework for secure SSO from browser clients to web-based applications, including Oracle Applications and tools, through standard protocols. It supports both partner applications, which take full advantage of the SSO framework, as well external applications for support of legacy and third party products. It is well-integrated with the Oracle middle tier web portal product, Oracle Portal, and permits management of user information in an external directory, permitting integration with SSO technologies for other, non-Oracle applications. It will soon support PKI client authentication, which will enable PKI authentication to a wide range of web applications.

> **See Also:** Oracle documentation and collateral on the Oracle9*i* Application Server

Index

A

access
  unauthorized, 1-14
access control
  described, 1-6
  directory, 5-7
  least privilege, 9-4
  Oracle Connection Manager, 9-55
access control lists (ACLs), 6-3
administration
  delegation of, 5-8, 9-49
  enterprise user, 9-53
application context
  accessed globally, 9-18
  accessed locally, 9-17
  initialized externally, 9-17
  initialized globally, 9-18
  overview, 9-16
  secure, 9-15
  virtual private database (VPD), 9-17
application security
  directory-based, 5-8
  policies, 9-13
  requirements, 1-15
  secure application role, 9-21
auditing
  customizable, 7-3, 9-5
  fine-grained, 7-3, 9-21
  in multitier systems, 7-4
  introduction, 7-2
  multitier applications, 9-23
  security requirements, 7-2
authentication, 9-26
  application user proxy authentication, 9-18
  biometric, 9-37
  CyberSafe, 4-4, 9-36
  DCE, 4-7, 9-38
  described, 1-6, 4-2, 9-4
  directory, 5-5
  Entrust/PKI, 8-8, 9-36
  Kerberos, 4-4
  methods, 8-7, 9-4, 9-33
  multitier, 6-5
  password-authenticated users, 6-5
  password-based, 4-2
  PKI certificate-based, 4-7, 8-5
  PKI methods, 8-7, 9-34
  proxy, 3-8, 4-8, 9-9
  RADIUS protocol, 4-4, 9-36
  SecurID, 9-37
  smart cards, 4-6, 9-37
  SSL, 8-7, 9-35
  strong, 4-3, 9-33
  token cards, 4-5, 9-37
authorization
  biometrics, 4-7
  described, 1-6
  directory, 5-7, 5-8
  multitier, 6-5
  proxy, 4-8
availability
  Real Application Clusters, 9-8
  security factors, 1-7, 2-12, 9-6

B

backup and recovery, 9-7

Baltimore Technologies, 9-34
biometric authorization, 4-7, 9-37

C

certificate authorities, 9-34
  introduction, 8-5
certificates
  contents, 8-6
  introduction, 8-5
  support for multiple, 9-45
  trusted, 8-6, 9-42
  X.509 Version 3, 8-7
checksums, 9-29, 9-49
  algorithms, 3-6
  SSL, 9-29
confidentiality, 1-5
connection
  management, 9-55
  multitier, 3-3
connection pooling, 4-9, 9-18
credentials
  secure storage, 8-8
CyberSafe ActiveTrust, 4-4
CyberSafe authentication, 4-4, 9-36

D

data
  deep data protection, 9-12
  encryption of stored, 2-10
Data Encryption Standard (DES), 2-11, 3-5, 9-6, 9-28
database links
  current user, 9-41
DBMS_OBFUSCATION_TOOLKIT, 9-6
directory security
  administrative roles, 5-12
  application security, 5-8, 9-41
  domains and roles, 5-10
discretionary access control (DAC)
  least privilege, 9-4
Distributed Computing Environment (DCE)
  authentication, 4-7, 9-38

E

encryption
  algorithms, 2-11, 3-5
  for network transmission, 3-4, 9-27
  stored data, 2-10, 9-6
enterprise roles, 2-5, 9-53
enterprise user security
  features, 9-39
  global roles, 2-5
  graphical user interfaces, 9-40
  introduction, 6-1, 6-2
  privilege administration, 6-3
enterprise users
  password authenticated, 6-5, 9-40
Entrust certificates, 9-42
Entrust Profile, 9-35
Entrust/PKI authentication, 8-8, 9-35

F

failover, 9-8
Federal Information Processing Standard 140-1 (FIPS), 9-24
fine-grained access control
  facilitating VPD, 9-19
  per-user, 9-20
fine-grained auditing, 7-3, 9-21
firewalls, 3-7, 9-56, 9-57

G

GTE CyberTrust certificates, 9-42, 9-43

H

hashing, password, 5-6

I

integrity
  checking, 3-6
  database mechanisms, 2-11, 9-3
  described, 1-6
  directory, 9-49
  entity integrity enforcement, 9-3

Oracle Advanced Security features, 9-29
referential, 2-11, 9-3
Internet
access control, 9-57
data access increased, 1-9
hosted system security, 1-11, 9-13
increased data availability, 1-9
large user communities, 1-10
scalability of security, 1-10, 9-13
security challenges, 1-8
security features, 9-12
security requirements, 1-8

J

Java
class execution, 9-23
security implementation, 9-23
Java Database Connectivity (JDBC)
application user proxy authentication, 9-11
encryption, 9-31
JDBC-OCI driver, 3-9, 9-10, 9-30
network security, 3-8
supported drivers, 9-30
Thin driver, 3-9, 9-31
Java Secure Socket Extension (JSSE), 9-32
Java virtual machine (JVM), 9-23
java.lang.SecurityManager, 9-23

K

Kerberos authentication, 4-4, 9-36
Kerberos Single Sign-On, 4-4

L

label based access control
introduction, 2-9
Oracle Label Security, 9-47
LDAP
application security, 5-8
compliance, 9-51
delegation of administration, 5-8
directory access controls, 5-7
introduction, 5-3
Oracle Internet Directory, 9-43
security features, 5-4
server instance architecture, 9-51
single sign-on, 9-39
lightweight sessions, 4-9
Login Server, 4-11

M

MD4 hashing scheme, 5-6, 9-50
MD5 Checksum, 3-6, 5-6, 9-6, 9-29, 9-49
Microsoft Active Directory, 9-41
multitier systems
auditing, 7-4, 9-23
authentication, 6-5
proxy authentication, 4-8, 9-10
security, 3-8
single sign-on, 4-11

N

network security
database enforced, 3-4
encryption, 3-4
firewalls, 3-7
Java Database Connectivity (JDBC), 3-8
managing privileges, 2-7
multitier connection management, 3-3
Oracle Advanced Security features, 9-27
PKI, 8-9
Secure Sockets Layer, 3-6
valid node checking, 3-3
VPD database enforced access, 9-58

O

Oracle Advanced Security, 9-24, 9-26
authentication, 9-33
PKI implementation, 9-42
Oracle Call Interface (OCI)
JDBC driver, 9-10
JDBC-OCI driver, 3-9
PKI, 9-42
Oracle Connection Manager, 3-3
firewall support, 9-57

firewalls, 9-56
security features, 9-55
Oracle Enterprise Login Assistant, 9-35, 9-43
Oracle Enterprise Security Manager, 9-41, 9-44, 9-45
Oracle Internet Directory, 9-43
architecture, 9-52
components, 9-50
enterprise user administration, 9-53
security benefits, 9-49
security features, 9-48
Oracle Java SSL, 9-32
Oracle Label Security, 9-20, 9-47
Oracle Net Firewall Proxy, 9-57
Oracle Net Services, 9-27
protocol support, 9-55
security features, 9-54
Oracle Password Protocol, 9-32
Oracle Policy Manager, 9-20
Oracle Wallet Manager, 8-8, 9-32, 9-35, 9-43, 9-45
Oracle wallets, 9-43
Oracle9i Application Server
SSL encryption, 9-30

P

partitioning, 9-19
virtual private database (VPD), 9-19
passwords
authentication, 4-2
authentication of enterprise users, 6-5, 9-40
protection in directory, 5-6, 9-50
security risks, 1-13
PKCS #12 containers, 9-45
PKCS#10 certificates, 9-43
policy function, 9-59
privacy of communications, 1-5
privileges
enterprise administration, 6-3
least, 9-4
managing, 2-3
network facilities, 2-7
roles to manage, 2-4
schema object, 2-2, 2-3
stored procedures to manage, 2-6
system, 2-2
views to manage, 2-7
profiles
user, 9-6
protocol conversion, 9-55
proxy authentication, 3-8, 4-8, 9-9
application user, 9-11, 9-18
directory, 9-10
expanded credential, 9-10
Kerberos and CyberSafe, 9-36
proxy authorization, 4-8
Public Key Certificate Standard #12 (PKCS#12), 8-8
Public Key Certificate Standards (PKCS), 9-34
public key infrastructure (PKI)
advantages, 8-3
authentication, 4-7, 9-34
authentication methods, 8-7
certificate-based authentication, 8-5
components, 8-3, 9-42
cryptography, 8-4
interoperability, 9-44
introduction, 8-1
network security, 8-9
Oracle Advanced Security, 9-42
Oracle implementation, 9-46
security features, 8-2
single sign-on, 8-9
supported vendors, 9-34

R

RADIUS protocol
authentication, 4-4, 9-36
smart cards, 9-37
supported vendors, 9-36
RADIUS-compliant smart cards, 4-6
RADIUS-compliant token cards, 4-5
RC4 encryption algorithm, 2-11, 3-5, 9-28
Real Application Clusters
availability, 9-8
referential integrity, 9-3
replication, advanced, 9-7
resource limitation, 9-6
roles
database, 2-4

directory administration, 5-12
enterprise, 2-5, 6-5
global, 2-5
managing privileges, 2-4
secure application, 2-6
secure application role, 9-21
types of, 9-5
row level security
introduction, 2-8
RSA certificates, 9-42, 9-43
RSA Data Security RC4, 3-5, 9-28
RSA SecurID tokens, 9-37

S

scalability
security, 1-15, 9-18
schema objects
privileges on, 2-3
secure application roles, 2-6, 9-21, 9-59
Secure Hash Algorithm (SHA), 3-6, 5-6, 9-29, 9-49
Secure Sockets Layer (SSL), 9-42
authentication, 8-7, 9-35
checksums, 9-29
encryption, 9-29
network security, 3-6
Oracle Internet Directory, 9-49
single sign-on, 9-43
SecurID token cards, 9-37
security
administration team, 1-18
application, 9-13
application context, 9-16
application user proxy authentication, 9-18
auditing, 7-2
availability, 1-7, 2-12
credentials, storage, 8-8
database, 2-2
database integrity mechanisms, 2-11
deep data protection, 9-12
directory authentication, 5-5
directory-based, 5-8, 9-41
enterprise user, 6-2
firewalls, 3-7
good practices, 2-13
hosted systems, 1-11
integrity, 1-6
Internet, 1-8, 1-10, 9-12, 9-13
Java Beans, 9-25
Java implementation, 9-23
label based access control, 2-9
LDAP features, 5-4
multitier systems, 1-15, 3-8
myths, 1-2
network, 9-27
Oracle Advanced Security, 9-24
Oracle Internet Directory, 9-48
Oracle Label Security, 9-47
Oracle Net Services, 9-54
Oracle9i Enterprise Edition, 9-12
Oracle9i Standard Edition, 9-2
password protection, 1-13, 5-6
personnel dimension, 1-4
physical dimension, 1-4
PKI, 8-1
privileges, 2-2
procedural dimension, 1-4
requirements, 1-15
row level, 2-8
scalability, 1-15, 9-18
scope of issues, 1-3
secure application role, 9-21
security directory integrity, 5-2
shared schemas, 6-4
single sign-on, 4-10, 6-6
strong authentication, 4-3
technical dimension, 1-4
threats and countermeasures, 1-12, 1-16
virtual private database (VPD), 2-9
SecurityManager class, 9-23
sessions
lightweight, 4-9
multiplexing, 9-56
shared schemas
Oracle Internet Directory, 9-53
security features, 6-4, 9-41
single sign-on
Entrust-based, 9-35, 9-39
implementations, 4-10, 9-38
introduction, 6-6

multitier, 4-11
Oracle Enterprise Login Assistant, 9-43
PKI, 8-9, 9-38, 9-39
server-based, 4-10
Single Sockets Layer (SSL)
current user database links, 9-41
smart cards, 4-6, 9-37
storage
secure credentials, 8-8
secure data, 1-5
stored data encryption, 1-5
stored program units
managing privileges, 2-6, 9-5

T

tables
privileges on, 2-3
TCP.EXCLUDED_NODES parameter, 9-58
TCP.INVITED_NODES parameter, 9-58
TCP.VALIDNODE_CHECKING parameter, 9-58
token cards, 9-37
benefits, 4-5
Triple DES (3DES), 2-11, 3-5, 9-6, 9-28, 9-45

U

UNIX hashing scheme, 5-6
user models, 9-20
users
authentication of, 9-4

V

valid node checking, 3-3, 9-58
VeriSign, 9-34, 9-42, 9-43
views
complex and dynamic, 2-9
managing privileges, 2-7, 9-5
virtual private database (VPD), 9-19
application context, 9-17
database enforced network access, 9-58
how it works, 9-15
introduction, 2-9
network security, 3-4
Oracle Label Security, 9-20, 9-47
Oracle Policy Manager, 9-20
overview, 9-14
user models, 9-20

W

wallets, 9-43
encryption, 9-45

X

X.509 Version 3 certificates, 8-7, 9-9, 9-10, 9-34, 9-35, 9-43, 9-45