Using Oracle Database Vault with Oracle Database 12*c*

Student Guide

D86597GC10

Edition 1.0

August 2014

Authors

Maria Billings

Dominique Jeunot

Technical Contributors and Reviewers

Chi Ching Chui

Pat Huey

Yaping Li

Paul Needham

James Spiller

Sailaja Pasupuleti

Sravanti Tatiraju

Editor

Daniel Milne

Graphic Designer

Maheshwari Krishnamurthy

Publisher

Jayanthy Keshavamurthy

Contents

1 Introduction
Lesson Objectives 1-2
Course Objectives 1-3
Curriculum Context 1-4
Suggested Schedule 1-5
Oracle Database Security 1-6
Enterprise Cloud Computing 1-8
Your Learning Aids 1-9
Basic Workshop Architecture 1-10
Quiz 1-11
Summary 1-12
Practice 1-13

2 Database Vault Overview
Objectives 2-2
Access Control Components 2-3
What Is a Secure Application Role? 2-7
Integration with Other Oracle Features and Options 2-16
Quiz 2-17
Summary 2-20
Practices 2-21

3 Configuring Database Vault
Configuring Database Vault 3-4
What to Expect After You Enable Database Vault 3-9
Revoked Privileges 3-11
Prevented Privileges 3-12
Securing Data in Multitenant Environments 3-13
Configuring Database Vault Users in Cloud Control 12c 3-14
Quiz 3-15
Summary 3-17
Practices 3-18

4 Analyzing Privileges
Objectives 4-2
Privilege Analysis Overview 4-3
Privilege Analysis Features 4-4
What Is Privilege Analysis? How Does It Work? 4-5
Types of Privilege Analysis 4-6
What Are Your Tools and Prerequisites? 4-7
Managing Privilege Analysis Policies 4-8
Analyzing ANY Privilege Use 4-9
Analyzing Privilege Use by a User Who Has the DBA Role 4-10
Determining Least Privilege Access Using Privilege Analysis 4-11
Quiz 4-12
Summary 4-13
Practices 4-14

5 Configuring Realms
Objectives 5-2
Quiz 5-3
Benefits of Using Realms 5-6
Protecting Roles 5-10
Mandatory Realms and Object Privileges 5-11
Protecting with a Mandatory Realm 5-12
Characteristics of Mandatory Realms 5-13
Benefits of Mandatory Realms 5-14
Protecting Sensitive Data During Patching 5-15
Protecting Sensitive Data During Run Time 5-16
Tasks Involving Realms 5-17
Realm Attributes 5-18
Predefined Reports 5-22
Quiz 5-23
Summary 5-25
Practices 5-26

6 Defining Rule Sets
Objectives 6-2
Quiz 6-3
Oracle-Defined Rule Sets 6-8
Creating and Maintaining Rules 6-10
Rule Set Tasks 6-11
Auditing Rule Sets 6-12
Setting a Custom Event Handler 6-13

Quiz 6-19
Summary 6-21
Practice 6-22

7 Configuring Command Rules
Objectives 7-2
Quiz 7-3
Command Rules 7-4
Use Case 7-5
Scope of Command Rules 7-7
Disallowing ALTER TABLE in a Schema 7-8
Delivered Command Rules 7-9
Reports and Views 7-10
Command Rule API 7-11
Quiz 7-13
Summary 7-15
Practices 7-16

8 Extending Rule Sets
Objectives 8-2
Quiz 8-3
Using Factors 8-6
Predefined Factors 8-7
Factors and Contexts 8-9
Factor Scenarios 8-10
Factor Types 8-11
Factor Identification 8-12
Factor Evaluation 8-13
Retrieval Method 8-14
DVSYS.SET_FACTOR 8-15
Assignment Rule Sets for Factors 8-16
Validation Method 8-17
Factor Audit Options 8-18
Error Options 8-19
Quiz 8-20
Purpose of Identities 8-21
Identity Example 8-22
Trust Levels 8-23
Map Conditions 8-25
Factor and Identities Views 8-26
Reports Related to Factors and Their Identities 8-27

Maintaining Factors and Identities 8-28
Quiz 8-29
Summary 8-32
Practices 8-33

9 Configuring Secure Application Roles
Objectives 9-2
Quiz 9-3
Secure Application Roles 9-4
Secure Application Role 9-5
Using a Secure Application Role 9-6
Secure Application Role Changes in Database Vault 9-7
Tasks with Secure Application Roles 9-8
Examples 9-9
Reports and Views 9-11
Maintaining Secure Application Roles 9-12
Quiz 9-13
Summary 9-15
Practice 9-16

10 Auditing with Database Vault Reports
Objectives 10-2
Required Privileges 10-3
Using Database Vault Reports 10-4
Security Analysis in Cloud Control 10-5
Database Vault Audit Integration with the Unified Audit 10-6
Database Vault Audit Views 10-7
Paying Attention to Configuration Changes 10-8
Mandatory Auditing of Configuration Changes 10-9
Audit Views: Configuration Changes 10-10
Audit Views: Violated Events 10-11
Oracle Database Vault Audit Policy 10-12
Checking for Configuration Issues 10-13
Reviewing Database Vault Configuration 10-14
Database Vault Enforcement Audit Reports 10-15
Reviewing Database Vault Audit Reports 10-17
General Security Reports 10-18
Database Account Password Reports 10-19
Privileged Database Accounts and Roles Reports 10-20
Initialization Parameter and Operating System Directory Permission Reports 10-21
General Database Privilege and Resource Profile Reports 10-22

Database Audit and Privilege Reports 10-23
Object Privilege Reports 10-24
Sensitive Objects Reports 10-25
Unified Audit Trail 10-27
Other Security Vulnerability Reports 10-28
Quiz 10-30
Summary 10-32
Practices 10-33

11 Implementing Best Practices
Objectives 11-2
Identifying Your Security Requirements 11-3
Separation of Duty Best Practices 11-5
Separation of Duty Matrix 11-6
Application Protection Matrix 11-7
Building and Documenting Your Implementation 11-8
Trusted People for Trusted Positions 11-10
Recommended Naming Conventions 11-12
Transition to Production 11-13
Implementing Separation of Duties 11-14
Application DBA: Attributes 11-15
Application DBA: Implementation 11-16
Application DBA: Workflow 11-17
Application DBA: Result 11-18
Dual Key Security 11-19
Dual Key Security: Use Case 1 11-20
Dual Key Security: Use Case 2 11-21
Database Account Considerations 11-22
Dynamic Auditing 11-23
Protecting from Accidental Object Loss 11-24
Quiz 11-25
Connection Pooling Considerations 11-27
Enforcing Connections from an Application Server 11-28
Fast Response to Policy Changes 11-29
Performance Considerations: Auditing 11-30
Performance Considerations: Unused Factors 11-31
Diagnosing Database Vault Using Trace Files 11-32
Password Policy Considerations 11-33
Guidelines for Procedures and Packages 11-34
Other Guidelines 11-36
Miscellaneous Recommendations 11-38

Oracle Database Vault Application Certification 11-39
Quiz 11-40
Summary 11-42
Practices 11-43

Appendix A: Your Learning
Overview A-2
Enterprise Manager Database Express Menus A-5
Request Handling in EM Express A-6
Oracle SQL Developer: Connections A-7
Oracle SQL Developer: DBA Actions A-8
Continuing Your Learning A-9
Further Information A-10

Appendix B: Using Enterprise Manager Cloud Control
Objectives B-2
Key Challenges for Administrators B-3
Enterprise Manager Cloud Control B-4
Cloud Control Components B-6
Components and Communication Flow B-7
Oracle Management Repository B-8
Controlling the Enterprise Manager Cloud Control Framework B-9
Starting the Enterprise Manager Cloud Control Framework B-10
Stopping the Enterprise Manager Cloud Control Framework B-11
Different Target Types B-12
Target Discovery B-13
Enterprise Manager Cloud Control B-14
User Interface B-15
Security: Overview B-16
Managing Securely with Credentials B-17
Distinguishing Credentials B-18
Quiz B-20
Practices B-21

1

Introduction

Lesson Objectives

After completing this lesson, you should be able to:

- Explain the course objectives
- Describe the curriculum context and course schedule
- Identify learning aids during and after the course
- Log in to the technical course setup

Course Objectives

After completing this course, you should be able to:

- Enable and configure Database Vault and describe the impact of this action
- Reduce the attack surface for potential security breeches through limiting privileges
- Implement separation of duties
- Use and maintain realms, rules sets, command rules, factors, identities, and secure application roles
- Perform security analysis and report security vulnerabilities
- Implement best practices

ORACLE

Curriculum Context

Before this course:

- Oracle Database 12*c*: Security course
- Oracle Database Administration experience
- Working knowledge of SQL and PL/SQL

In this course, you learn to:

- Enable and configure Database Vault
- Configure realms, rule sets, rules, SQL command rules, and secure application rules
- Define factors to extend rule sets
- Audit with Database Vault reports

After this course, consider deepening your knowledge with other security courses

http://education.oracle.com

ORACLE

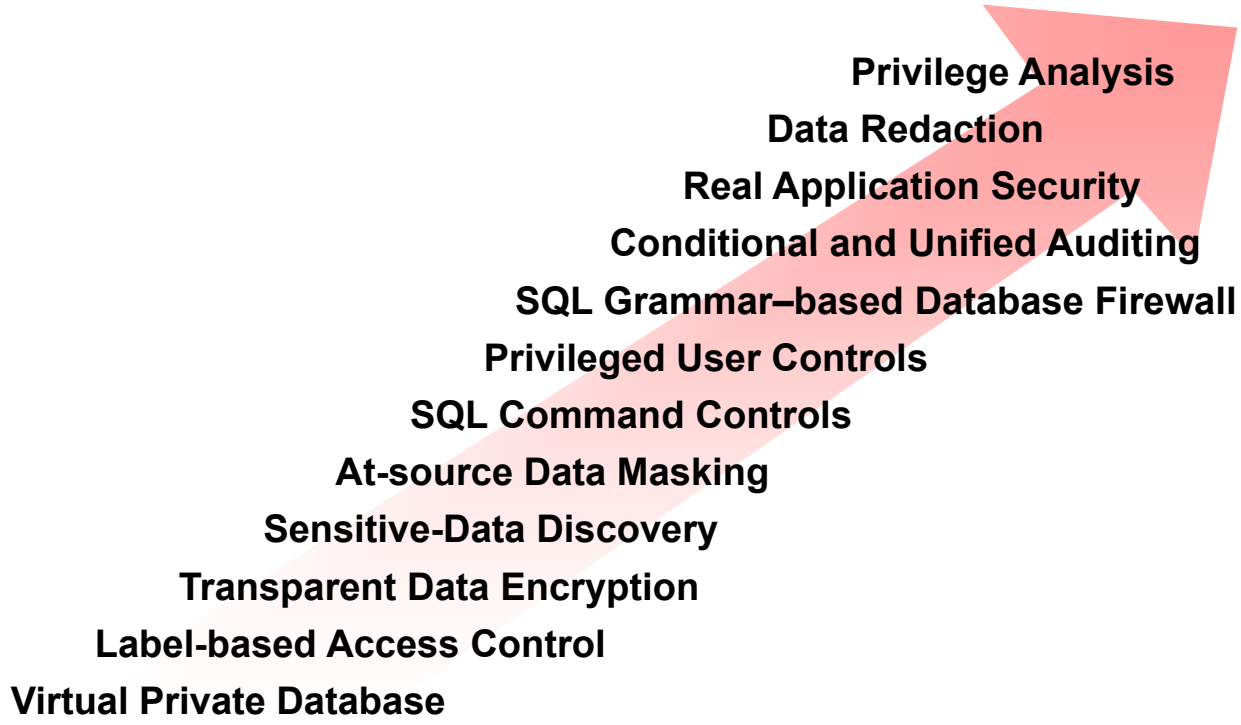
- **Before** taking this course, you should ensure that you fulfill the prerequisites, which include knowledge of Oracle Database 12*c*, SQL, and PL/SQL (for DBA use). It is recommended that you are also familiar with Oracle Enterprise Manager Cloud Control 12*c*.
- With **this course**, you achieve the objectives listed in the slide.
- **After** this course, you can continue your education by taking courses about other security options.
- Query http://education.oracle.com for up-to-date course offerings.

Suggested Schedule

Day	Lessons
1	1. Introduction 2. Database Vault Overview 3. Configuring Database Vault 4. Analyzing Privileges 5. Configuring Realms 6. Defining Rule Sets
2	7. Configuring Command Rules 8. Extending Rule Sets 9. Configuring Secure Application Roles 10. Auditing with Database Vault Reports 11. Implementing Best Practices (a workshop approach)

This schedule is just a very general outline. Your instructor determines the actual class schedule.

Oracle Database Security



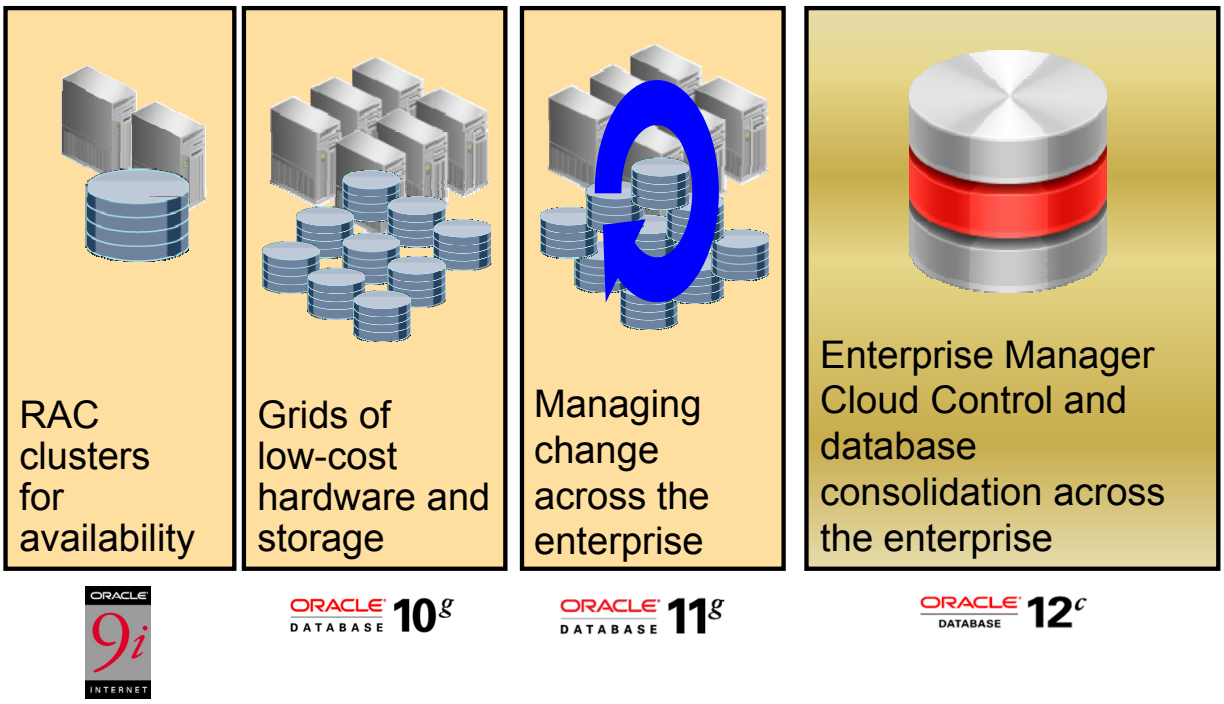
As a result of its early focus on security, Oracle has maintained the lead in the industry with a large number of security products.

Some of the marquee security areas in the Oracle Database 12*c* are the following:

- **Virtual Private Database** (VPD) is used when the standard object privileges and associated database roles or views are insufficient to meet application security requirements providing views that are more complex. VPD policies can be simple or complex depending on your security requirements.
- **Oracle Label Security** (OLS) provides multi-level security capabilities within the Oracle Database by using label-based access control. Oracle Label Security has the ability to control access based on data classification, adding a powerful dimension to the access control decision process and enforce traditional multi-level security policies.
- **Transparent Data Encryption** (TDE) makes encryption of sensitive data simple with no changes to the existing application code. TDE is one of the two components of the Oracle Advanced Security option for Oracle Database 12*c* Release 1 Enterprise Edition. It provides transparent encryption of stored data to support your compliance efforts.
- Enterprise Manager's **Data Discovery** and Modeling (DDM) capability enables operations such as sensitive-data discovery. Discovering what sensitive data you have is important for masking data. Data masking works on top of an Application Data Modeling (ADM) to scan your database for columns that are known to be sensitive. Sensitive data from your production system can be replaced with fictitious data in a development system during testing using Oracle Data Masking.

- **Data masking** is the act of *anonymizing* customer, financial, or company confidential data retaining the original data's properties, such as width, type, and format, and replacing sensitive information copied from production databases to non-production databases with realistic, but scrubbed, data based on masking rules. Data masking is ideal for virtually any situation when confidential data needs to be shared with non-production users. These users may include internal users such as application developers or external business partners such as offshore testing companies, suppliers, and customers.
- Authorization includes primarily two processes:
  - Permit only certain users to access, process, or alter data (object **privileges**).
  - Permit only certain users to execute certain commands (system **privileges**).
- **Database Firewall** provides two features for databases. DF allows logs, alerts, substitutes, and blocks on SQL statements on the network, and uses a SQL grammar analysis engine for high performance and accuracy, an approach that is superior to first-generation database firewalls that relied on regular expressions.
- The **unified auditing** facility groups audit options into simple audit policies, merges all audit trails into a single unified audit trail table, relying on a read-only audit trail table, and audits any `SYS` user auditable action
- **Data Redaction** enables you to redact (mask) column data. Data Redaction performs the redaction at run time—that is, the moment that the user tries to view the data.
- **Privilege Analysis** is able to find privileged users, and therefore privileges that are unnecessarily granted to the `PUBLIC` role and to application database users.

Enterprise Cloud Computing



Oracle Database 10*g* was the first database management system designed for grid computing.

Oracle Database 11*g* consolidates and extends Oracle's unique ability to deliver the benefits of grid computing, transforming data centers from silos of isolated system resources to shared pools of servers and storage.

Oracle Database 12*c* and Enterprise Manager Cloud Control are designed for cloud computing. Cloud computing creates a complete, pre-integrated, off-the-shelf private cloud solution that allows you to quickly transform the enterprise data center into a private cloud.

The key benefits are the following:

- Reduce server sprawl and improve CPU utilization by consolidating on fewer servers.
- Reduce the amount of time a DBA spends installing and configuring databases, by automating deployment of standard database configurations.
- A single console manages the entire cloud life cycle—plan, set up, deliver, and operate.
- Prevent resource hogging by setting quotas for individual users.
- Forecast future resource needs by analyzing trending reports.
- Compute chargeback based on performance and configuration metrics.

Your Learning Aids

During the course:

- Your instructor
- Videos and demonstrations
- Product documentation and online help
- Appendixes for your learning aids during and after the course

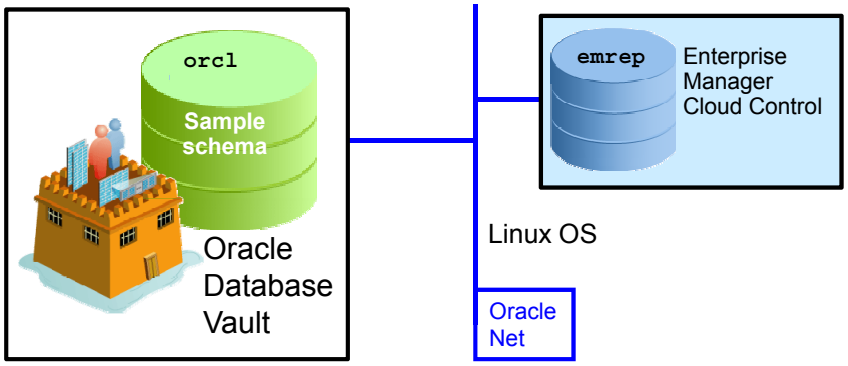
Oracle Database Vault Administrator's Guide

During this course, the instructor introduces many learning aids that you can explore on your own.

Basic Workshop Architecture

`orcl` database instance with:

- Sample schema
- Unified auditing
- Database Vault installed, but not enabled



For the hands-on practice, the instances displayed in this slide are pre-installed on a Linux OS: `orcl` and `emrep` are instances of the Oracle Database 12*c*. ORCL database has the sample schema configured and unified is not auditing enabled. Database Vault is installed by default with the Oracle Database 12*c* installation, but it is not enabled or configured.

During this course, you enable Database Vault and use its components to secure the ORCL database.

Quiz

Database Vault is installed by default.

a. True
b. False

Answer: a

Summary

In this lesson, you should have learned how to:

- Explain the course objectives
- Describe the curriculum context and course schedule
- Identify learning aids during and after the course
- Log in to the technical course setup

Practice

- 1-1: Your Course Setup
- 1-2: Enabling Unified Auditing
- 1-3: Adding a Cloud Control Target

Familiarize yourself with your course environment and perform setup tasks:

- In Practice 1-1, you confirm that Cloud Control is up-and-running and list your hostname and IP address for ease of access.
- In Practice 1-2 , you shutdown the database and middle-tier, enable Unified Auditing, and restart the processes.
- In Practice 1-3, you log in to Enterprise Manager Cloud Control and add the `orcl` database instance as a new target.

2

Database Vault Overview

Objectives

After completing this lesson, you should be able to:

- Define and use Database Vault terminology.
- List the components of Database Vault and their relationship to one another and describe how they prevent common security threats inside the Oracle database.
- Describe how to prevent privileged users (DBA) from accessing sensitive application data.
- List the tasks that can be performed by using the Database Vault API.
- Describe how Database Vault can be accessed by Database Vault administrators.

ORACLE

Access Control Components

Database Vault provides the following components for securing a database:

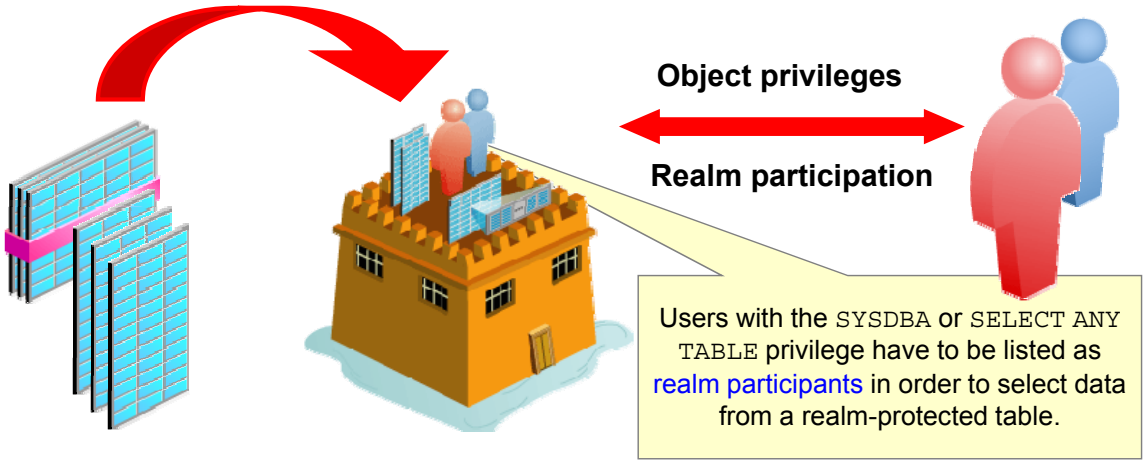
- Realms
- Rule sets
- Command rules
- Secure application roles
- Factors

The following components of Database Vault provide highly configurable access control:

- **Realms:** A boundary defined around a set of objects in a schema, a whole schema, multiple schemas, or database roles. Specific conditions must be met to gain access.
- **Rule sets:** A collection of rules that are evaluated for the purpose of granting access
- **Command rules:** A set of specific conditions that must be in effect for a given SQL command to be executed on a given object or set of objects. This provides very granular control over what can be done to certain objects and by whom.
- **Secure application roles:** A role that can be enabeled by a session only under the condition of passing a rule set.
- **Factors:** Attributes of a user or the system at any given time. Factors contribute to the decision process of granting access, and combinations of several factors may be considered at once. This is multifactored authorization. Identities are specific values that the factors may take on. Of the possible set of values for a given factor, some or all may have been assigned a name, which is an identity.

What Is a Realm?

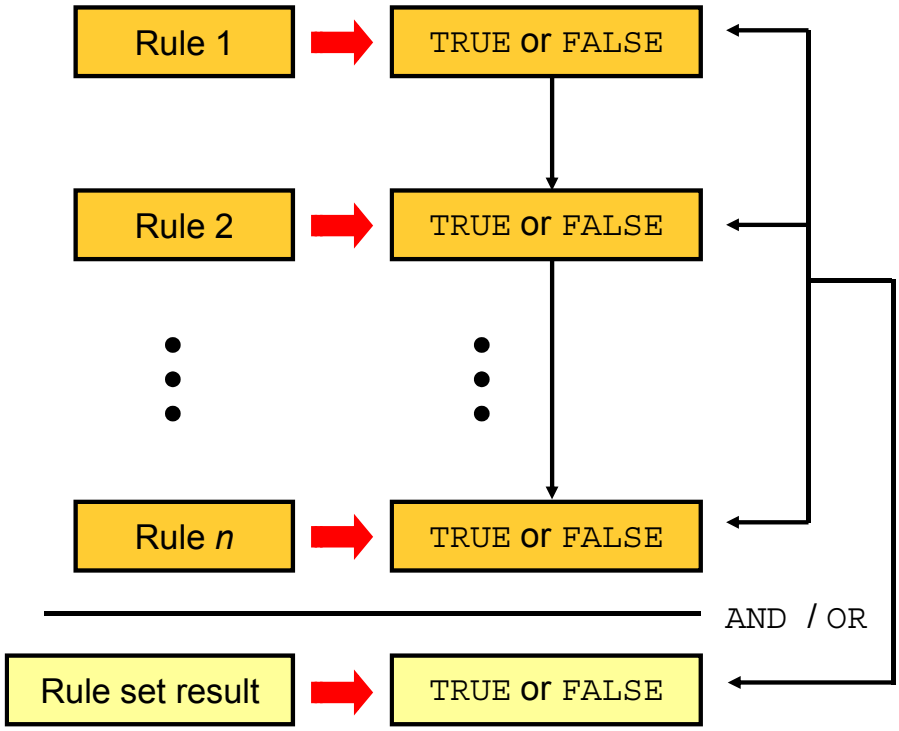
- A secure grouping of database schemas and roles
- A zone of protection for your database objects
- Reference users who are allowed to access the objects



- A realm is a functional grouping of database schemas and roles that must be secured for a given application. Think of a realm as a zone of protection for your database objects. Realms contain objects such as tables, roles, and packages.
- A realm may also have authorizations given to users or roles as participants or owners. A realm protects the objects in it from users exercising privileges, such as `SELECT ANY TABLE` and `SYSDBA`. Therefore, any such privileged user must be defined as a realm participant (or have a realm-participating role that is granted to him or her) to access the protected objects.
- If a user or role already has direct object privileges granted to access the realm-protected objects, the realm has no effect. These users or roles with direct object privileges can access the objects in the same manner as before the realm existed.

Realms are covered in detail in the lesson titled “Configuring Realms.”

What Is a Rule Set?

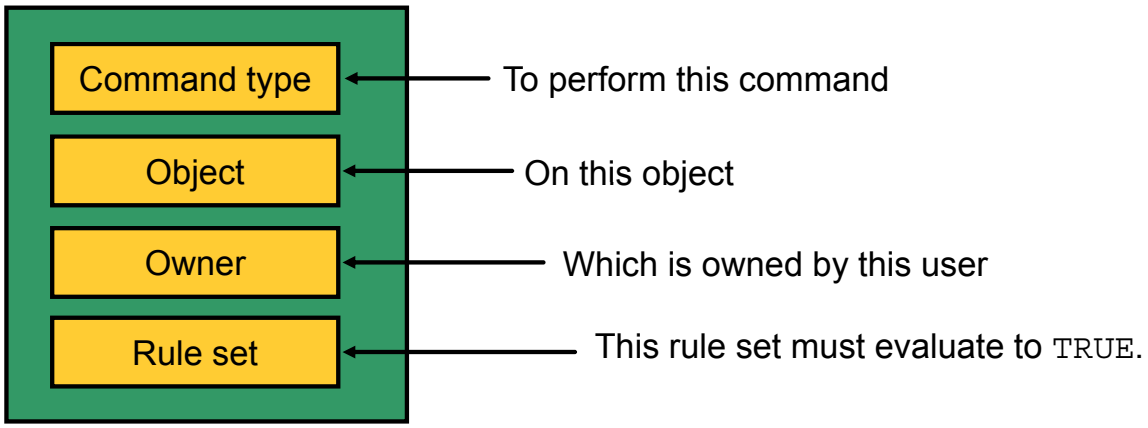


ORACLE

- A rule set is a collection of rules that are evaluated together to produce a result. Each rule is defined as a `WHERE` clause expression. The rule set specifies whether the results of the rules are to be `AND`ed or `OR`ed together. After each rule is evaluated, the results are `AND`ed or `OR`ed together, and the end result is a single value of `TRUE` or `FALSE`. Database Vault provides a rule engine to process rule sets.
- You can use a rule set to provide dual key security, whereby another separate action must take place for the requested access to be granted. An example is requiring a specific user to be logged in from a specific client machine in order for another user to have access to certain data or packages.

Rule sets are covered in detail in the lesson titled “Defining Rule Sets.”

What Is a Command Rule?

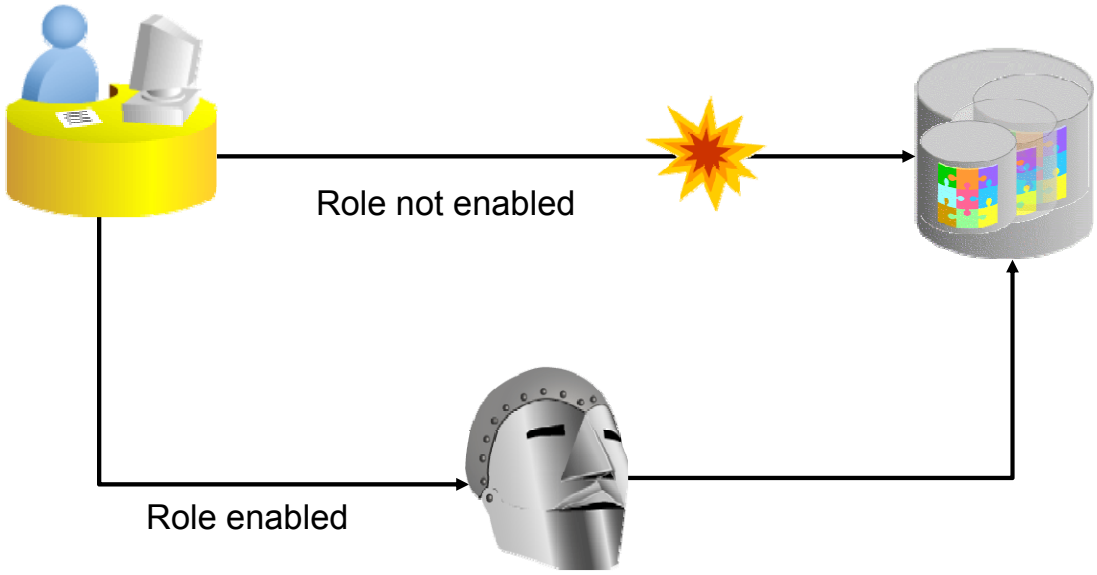


Command rule

A command rule defines the rules that must be followed to perform a SQL command on an object. These commands include most data definition language (DDL) commands, and also `SELECT`, `UPDATE`, and `DELETE`. You can implement a command rule to restrict specific commands from executing on specific objects or groups of objects. The restriction is based on the evaluation to `TRUE` of a rule set.

Command rules are covered in detail in the lesson titled “Configuring Command Rules.”

What Is a Secure Application Role?



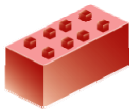
- Secure application roles are created to require specific conditions to be true before taking advantage of the permissions or privileges that are granted to the role. The conditions are represented as a single rule set. A secure application role consists entirely of a secure application role name and an associated rule set. Permissions granted to this role can be exercised only if the rule set evaluates to `TRUE`.
- Secure application roles in Database Vault are different from those provided by the Oracle database. The secure application role is still enabled by a procedure in a secure package, but the package is provided by Database Vault. Instead of writing a procedure, the security administrator simply uses a rule set to determine whether an application role should be enabled.

Secure application roles are covered in detail in the lesson titled “Configuring Secure Application Roles.”

What Is a Factor?

A factor:

- Is a named variable or attribute
- Has a value or an identity
- Can be easily referenced in other Database Vault components to discern access
- Can be combined with other factors to provide multifactored authorization



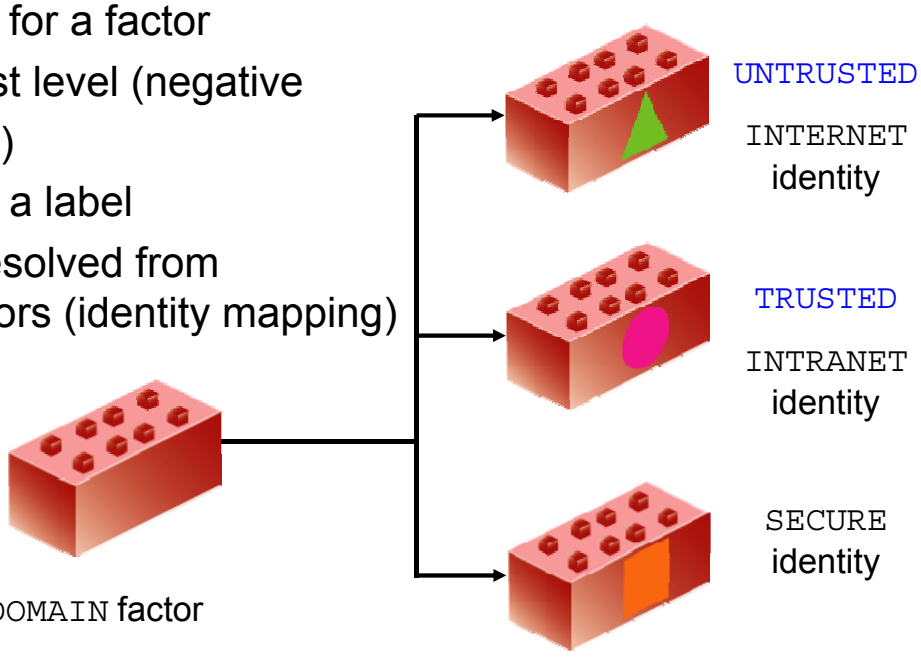
- A factor is a named variable or attribute, such as a user location, database IP address, or session user, that Database Vault can recognize.
- Each factor has a value assigned to it. This value is called an identity.
- Factors are combined in logical ways with other factors in rules and rule sets to provide the basis for access control policies. You can use factors for activities such as authorizing database accounts to connect to the database or creating filtering logic to restrict the visibility and manageability of data.
- You implement multifactored authorization by combining factors. The same concept is used when a customer representative you call on the phone asks for your name, billing address, and the last four digits of your social security number. Those are three factors that contribute to the authorization process.

Factors are covered in detail in the lesson titled “Extending Rule Sets.”

Factors and Identities

An identity:

- Is a value for a factor
- Has a trust level (negative by default)
- Can have a label
- Can be resolved from other factors (identity mapping)

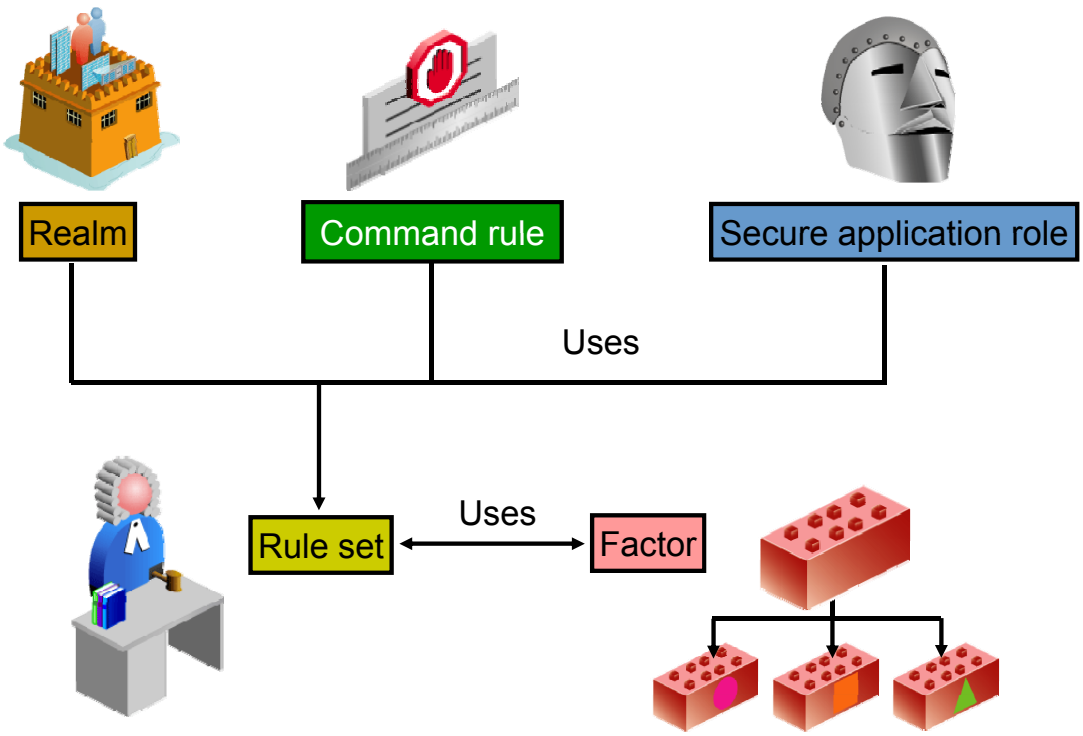


ORACLE

- An identity is a value for a factor.
  The identity of a factor can vary according to the session, application, user, or even the time of the day. Trust levels enable you to assign a numeric value to indicate the measure of trust allowed. A trust value of 1 signifies some trust. A higher value indicates a higher level of trust. A negative value or zero indicates distrust. When the factor identity returned from a factor retrieval method is not defined in the identity, Database Vault automatically assigns the identity a negative trust level.
- For example, when a user connects through a machine in the office, the `DOMAIN` factor identity is set to `INTRANET` with a `TRUSTED` trust level. When the same user connects from home, the `DOMAIN` factor identity is set to `INTERNET` with an `UNTRUSTED` trust level. In this example, the `DOMAIN` factor can be used to determine what data the user is allowed to access and what commands the user is allowed to perform.
- You can assign Oracle Label Security (OLS) labels to factor identities. A label acts as an identifier for a database table row to assign privileges to the row.
- A factor can have an identity assigned by a combination of multiple factors. The identity of these factors can in turn be resolved by other factors. Using factors to resolve an identity is called identity mapping.

Identities are covered in detail in the lesson titled “Extending Rule Sets.”

Component Relationships

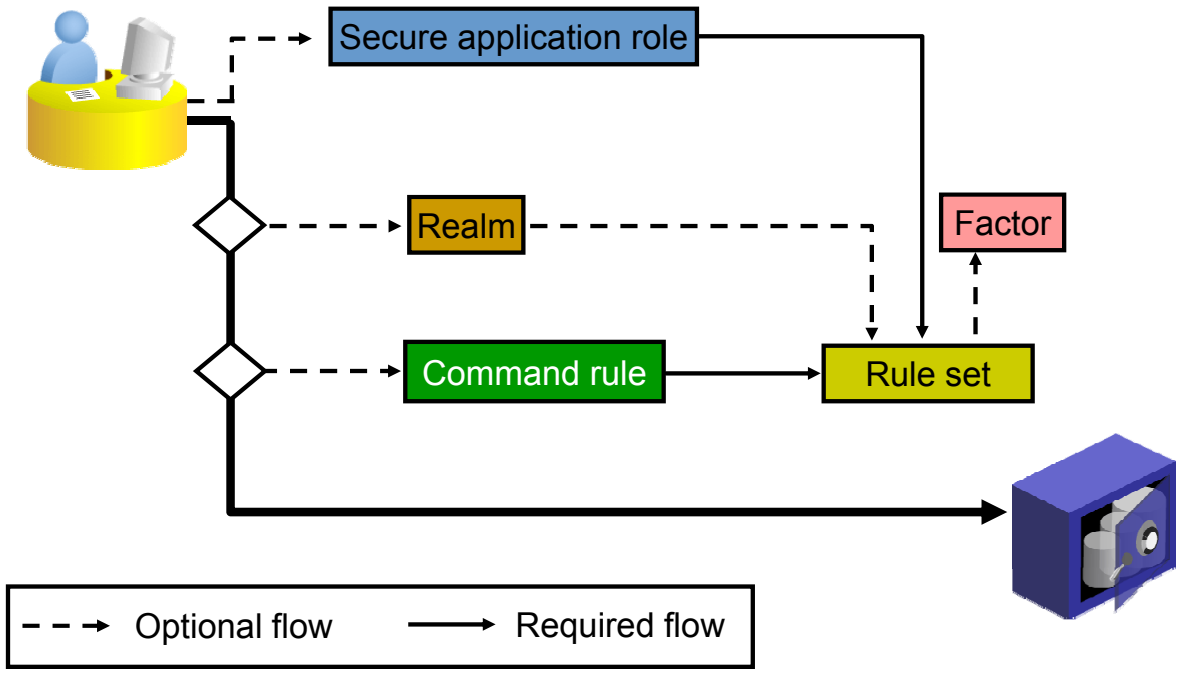


The relationship between the Database Vault components is shown in the slide. Based on the direction of the arrow, you can say that a given component type uses another component type. Realms, command rules, and secure application roles use rule sets to define their behavior.

Factors use rule sets and rule sets can refer to factors in their definition. Factors use identities and identities can refer to other factors. Rather than creating “factors” or “identities,” you can simply reference `USERENV` variables of the `SYS_CONTEXT` function in the rule set associated to a command rule.

In the graphic in the slide, the three components at the top are the ones that actually cause security to be enforced. The two components at the bottom of the slide support the enforcement. That is, you can create rule sets and factors with identities, but until you create a realm, a command rule, or a secure application role that refers to them, nothing changes about data access in your database.

Evaluation Sequence



When a session accesses data in Database Vault, it may or may not have to pass through security checks involving the Database Vault components. As shown in the slide, it may or may not require evaluation of any combination of realms or command rules. And if a realm is involved, there may or may not be a rule set defined that further modifies the access requirements. Both secure application roles and command rules must have a rule set defined for them.

The diagram illustrates the dynamic data access paths. Before a command is executed, any secure application role checks have already been done, along with the associated rule set, when the role is requested to be enabled. Then when a statement is submitted, all objects are checked to see whether they are in a realm. Next, the command rules are checked to ensure that the particular command being issued is allowed. It is possible that at least one component of each type of component requires evaluation during the execution of a single statement. It is also possible that only a single realm or a single command rule is evaluated. Any permutation of these is possible.

Note: When reading the flow chart in the slide, consider that at the end of each branch, the flow returns to the main path to continue. The secure application role path goes to its end, and then the flow returns to check realms, including its applicable flow path. Then the flow returns to check command rules, and so on.

Database Vault Example: Limiting Access to Table Content

1. The DBA can view the ORDERS table data.

```
SQL> SELECT order_total FROM oe.orders
  2  WHERE customer_id = 101;

ORDER_TOTAL
-----------
    78279.6
```

2. The security manager protects the OE.ORDERS table with a realm.
3. The DBA can no longer view the ORDERS table data.

```
SQL> SELECT order_total FROM oe.orders
  2  WHERE customer_id = 101;
SELECT order_total FROM oe.orders
                         *
ERROR at line 1:
ORA-01031: insufficient privileges
```

ORACLE

This example shows how to prevent privileged users (DBA) from accessing sensitive application data. DBA users typically have the DBA role granted to them. This means that they have the SELECT ANY TABLE system privilege. In this example, a DBA is prevented from accessing a table, even though the DBA continues to retain the SELECT ANY TABLE privilege.

1. The user with the DBA role is able to select data from the OE.ORDERS table.
2. The security manager protects the OE.ORDERS table with a Database Vault realm.
3. The DBA can no longer access the table.

You learn how to set up a realm to provide this protection in the lesson titled “Configuring Realms.”

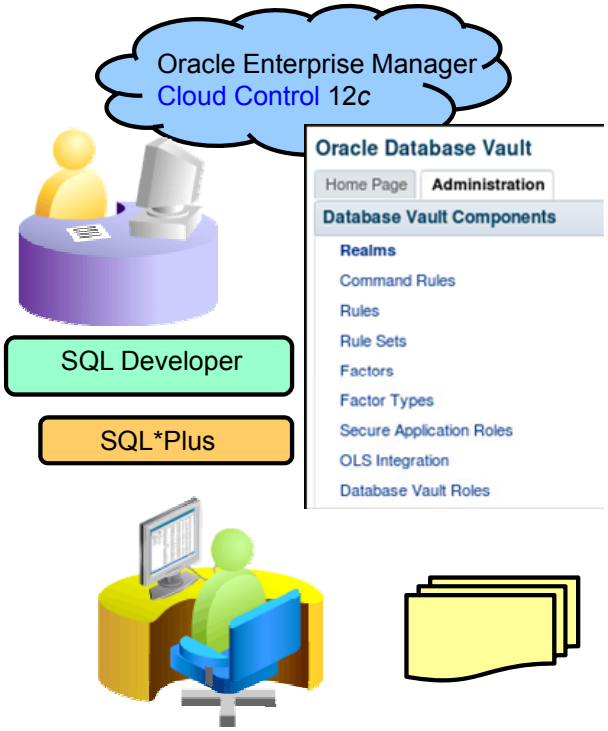
Database Vault: Effects

The configuration of the Database Vault option:

- Is transparent to applications
- Does not affect access paths for queries
- May affect what data is accessible for a given session under certain circumstances
- Requires near zero performance overhead

- Just having the Database Vault option configured does not affect the functionality of the database in any way.
- Applications do not have to be changed and all queries work the same way as before the installation. Even when the Database Vault components are configured, there is no need to modify the applications. Database Vault provides the extra security without changes to applications or how SQL is submitted to the database.
- However, if the new security configurations are there to prevent access to data by an application, that application will be affected accordingly.
- Even if more complex security configurations are created, the performance will not be affected. This is addressed in greater detail in the lesson titled “Implementing Best Practices.”

Software Overview



PL/SQL API:
- **DVF.<*functions*>**
- **DVSYS.DBMS_MACADM**
- **DVSYS.DBMS_MACSEC_ROLES**
- **DVSYS.DBMS_MACUTL**
- **DVSYS.CONFIGURE_DV**

DVSYS Data Dictionary views:
- DBA_DV_REALM
- DBA_DV_USER_PRIVS
- DV$CONFIGURATION_AUDIT
- DV$ENFORCEMENT_AUDIT

and many more
SYS audit views:
- DV$CONFIGURATION_AUDIT
- DV$ENFORCEMENT_AUDIT

ORACLE

The API is composed of PL/SQL procedures and functions that implement the Database Vault functionality. Administrators can use it to enable and refine the default protection of Database Vault (that is to configure the required access control policies). The PL/SQL procedures and functions allow the general database accounts to operate within the boundaries of access control policy in the context of a given database session.

Database Vault has the following schemas:

- DVSYS schema: Owns the Database Vault schema and related objects
- DVF schema: Owns the Database Vault functions that are created to retrieve factor identities

Database Vault administrators can perform their tasks through multiple interfaces (graphic and command-line).

- Oracle Enterprise Manager Cloud Control 12*c*
- SQL Developer
- SQL*Plus

You can generate reports on the various activities that Database Vault monitors. In addition, you can monitor policy changes, security violation attempts, and database configuration and structural changes.

Cloud Control provides the ability to show warnings and alerts of all target databases with Database Vault enabled in one central place. These alerts and warnings have their own metrics and thresholds, which can be edited and modified.

Over 30 data dictionary views provide access to the various underlying Database Vault tables in the DVSYS, SYS, and LBACSYS schemas without exposing the primary and foreign key columns that may be present. These views are intended for the database administrative user to report on the state of the Database Vault configuration.

Database Vault API

The Database Vault application program interface (API) provides the functionalities to perform the following:

- Create, modify, and delete Database Vault components
- Define the security environment for sessions
- Query the state and values of components
- Administer and configure systemwide Database Vault parameters

- **DVF.<*functions*>**
- **DVSYS.DBMS_MACADM**
- **DVSYS.DBMS_MACSEC_ROLES**
- **DVSYS.DBMS_MACUTL**
- **DVSYS.CONFIGURE_DV**

ORACLE

Database Vault provides a schema, `DVSYS`, which stores the database objects needed to process Oracle data for Database Vault. This schema contains the roles, views, accounts, functions, and other database objects that Database Vault uses.

The `DVF` schema contains public functions to retrieve (at run time) the factor values set in the Database Vault access control configuration.

Database Vault provides a collection of PL/SQL interfaces that enable security managers or application developers to configure the access control policy as required. The PL/SQL procedures and functions allow the general database account to operate within the boundaries of the access control policy in the context of a given database session.

The packages that are provided include:

- Functions owned by **`DVF`** for returning information about factors
- **`DVSYS.DBMS_MACADM`:** Procedures for creating and managing Database Vault components
- **`DVSYS.DBMS_MACSEC_ROLES`:** Procedures for activating secure application roles
- **`DVSYS.DBMS_MACUTL`:** Utility procedures for interrogating various security-related characteristics of the database environment
- **`DVSYS.CONFIGURE_DV`:** Procedure to configure the initial two Oracle Database user accounts, which are granted the `DV_OWNER` and `DV_ACCTMGR` roles.

Note: These APIs are covered where appropriate in this course.

Integration with Other Oracle Features and Options

- Integrating Database Vault with Transparent Data Encryption
- Attaching Factors to an Oracle Virtual Private Database
- Integrating Database Vault with Oracle Label Security
- Integrating Database Vault with Oracle Data Guard

This slide lists other Oracle features and options that can be integrated with Database Vault. For details, see the *Oracle Database Vault Administrator's Guide*.

Quiz

Secure application roles are checked before command rules when accessing an object.

a. True
b. False

Answer: a

Quiz

Rule sets are required to build a realm.

a. True
b. False

Answer: b

Quiz

Database Vault is a stand-alone product that cannot be integrated with other security products and functionality.

a. True
b. False

Answer: b

Summary

In this lesson, you should have learned how to:

- Define and use Database Vault terminology.
- List the components of Database Vault and their relationship to one another and describe how they prevent common security threats inside the Oracle database.
- Describe how to prevent privileged users (DBA) from accessing sensitive application data.
- List the tasks that can be performed by using the Database Vault API.
- Describe how Database Vault can be accessed by Database Vault administrators.

ORACLE

Practices

- 2-1: Testing Your Knowledge
- 2-2: Viewing the “Quick Start Tutorial: Securing a Schema from DBA Access” video

In Practice 2-1, you answer questions that test your understanding of the topics covered in this lesson.

In Practice 2-2, you view the “Quick Start Tutorial: Securing a Schema from DBA Access” video. The steps covered in the video are available as step-by-step instructions in the product documentation. The tutorial has the following steps:

1. Log On as `SYSTEM` to Access the `HR` Schema
2. Create a Realm
3. Create the `SEBASTIAN` User Account
4. Create an Authorization for the Realm
5. Test the Realm
6. If Unified Auditing Is Not Enabled, Run a Report
7. Remove the Components for This Tutorial

3

Configuring Database Vault

Objectives

After completing this lesson, you should be able to:

- Enable and register Database Vault
- List the accounts and schemas that are created during configuration and describe how they assist with the separation of duties
- Explain the configuration, privilege, and role changes that occur when Database Vault is enabled

Getting Started with Database Vault

Database Vault:

- Is installed by default
- Must be registered in the installed database
- Optionally, can be installed via DBCA for custom databases

- Registration:
  - Includes configuration and enabling of Database Vault
  - Configures top-level Database Vault administrative accounts
  - Enables Oracle Label Security
  - Applies to PDBs, single-instance, and Oracle RAC database

Used in this training

- After you install Oracle Database 12*c*, you must register (that is, configure and enable) Database Vault with the Oracle database in which it was installed. Oracle Database includes the Database Vault option when you include a default database in the installation process, but you must register Database Vault before you can use it.
- If you create a custom database, you can use DBCA to install and enable Database Vault for it.
- Registration is the process of configuring and enabling Database Vault.
  - As part of the registration process, you configure the Database Vault administrative accounts.
  - The registration process enables Oracle Label Security if it is not already enabled.
  - This procedure applies to the current pluggable database (PDB), as well as to both single-instance and Oracle Real Application Cluster (RAC) installations.
  - In this course, you work with a single instance.

Configuring Database Vault

1. Configure top-level accounts with SYSDBA privilege:

```
SQL> exec DVSYS.CONFIGURE_DV                <<< one time only
      (dvowner_uname    =>'dbv_owner',      <<< mandatory
      dvacctmgr_uname =>'dbv_acctmgr')      <<< recommended
```

2. Enable Database Vault as the dbv_owner user:

```
SQL> exec DVSYS.DBMS_MACADM.ENABLE_DV<<< can be repeated
```

3. You can enable and disable Database Vault. Restart the database after this.

4. Confirm that Database Vault is enabled:

```
SQL> select * from v$option
     where parameter = 'Oracle Database Vault';
```

ORACLE

As SYSDBA, create two top-level Oracle user accounts for Database Vault. One account is for the Database Vault owner and the second (optional) account for Database Vault account manager. For better separation of duty, Oracle recommends that you create both accounts. This course uses DBV_OWNER and DBV_ACCTMGR for the top-level accounts. During the registration process, which is a one-time only process, these accounts are registered for the DV_OWNER and DV_ACCTMGR roles.

```
connect / as sysdba
create user dbv_owner    identified by oracle_4U;
create user dbv_acctmgr  identified by oracle_4U;
grant create session to dbv_owner, dbv_acctmgr;
exec DVSYS.CONFIGURE_DV(dvowner_uname =>'dbv_owner', -
  dvacctmgr_uname =>'dbv_acctmgr')
connect dbv_owner/oracle_4U
exec DVSYS.DBMS_MACADM.ENABLE_DV
select * from v$option where parameter = 'Oracle Database Vault';
```

Restart the database after configuration changes.

DV_OWNER Versus DV_ACCTMGR

DV_OWNER can:

- Modify Database Vault security configurations
- Disable security components and/or auditing

DV_ACCTMGR can:

- Create new users
- Modify the passwords of existing users
- NOT modify the DV_OWNER password

leo_dvowner
DV_OWNER

bea_dvacctmgr
DV_ACCTMGR

Note: SYS and SYSTEM can no longer create users.

Oracle recommends that you do not use these top-level accounts for regular administration tasks; instead, you should give the DV_OWNER and DV_ACCTMGR role to individual administrators who secure them with their own password. If the top-level passwords were forgotten, you could no longer enable or disable Database Vault. This course uses LEO_DVOWNER and BEA_ACCTMGR as sample users.

These two roles are very different. The DV_OWNER role is able to change the Database Vault configuration, thus affecting which users can access what data. The DV_ACCTMGR role is able to create new users and modify the passwords of existing users, including those with the DV_MONITOR, DV_SECANALYST, and DV_AUDIT_CLEANUP roles. Having a separate user with the DV_ACCTMGR role—which means that this user is the only user who can create new users—provides a separation of duties.

- Because of the delivered configuration of Database Vault, the SYS user can no longer create users, but it can grant system privileges.
- To ensure proper security, users with the DV_ACCTMGR role are not allowed to change the password of security administrators who have the DV_OWNER role.

All this works together to provide strong controls inside the database over who can do what and controls over when and how applications, data, and databases can be accessed.

Database Vault Roles

Category	Role	Description
Security administration	DV_OWNER	The Database Vault owner has all other security administrative roles.
	DV_ADMIN	Configuration Administrator role
	DV_MONITOR	Enables Cloud Control to monitor Database Vault
	DV_SECANALYST	Allows access to Database Vault reports
	DV_STREAMS_ADMIN	For configuring Streams in a Database Vault environment
	DV_XSTREAM_ADMIN	For configuring Oracle XStream with Database Vault
	DV_GOLDENGATE_ADMIN	For configuring Oracle GoldenGate with Database Vault
	DV_GOLDENGATE_REDO_ACCESS	To access redo logs in a Database Vault environment
	DV_AUDIT_CLEANUP	For purging the Database Vault audit trail in a non-unified auditing environment
	DV_PATCH_ADMIN	For the DBA doing patches
Account management	DV_ACCTMGR	For the Database Vault account manager (for example, bea_dvacctmgr)
Resource management	DV_REALM_OWNER	For the DBA of the Database Vault realm
	DV_REALM_RESOURCE	For the Database Vault application resource owner
All responsibilities	DV_PUBLIC	A public role that has EXECUTE permissions for benign procedures

Database Vault creates ten new roles to support its functionality.

Users with the following security administrative roles can perform various tasks in Database Vault Administrator:

- **DV_OWNER:** This role is for the Database Vault owner. It represents a version of the DV_ADMIN role that is able to grant itself and the DV_ADMIN role to others. It has all the security administrative roles (DV_ADMIN, DV_MONITOR, DV_SECANALYST, DV_STREAMS_ADMIN, and DV_PATCH_ADMIN) and has the most privileges on the DVSYS schema.
- **DV_ADMIN:** This role allows the execution of administrative tasks, including monitoring and viewing of reports. It has the DV_SECANALYST role.
- **DV_MONITOR:** This role enables the Oracle Enterprise Manager Cloud Control agent to monitor Database Vault for attempted violations and configuration issues with realm or command rule definitions.
- **DV_SECANALYST:** This role allows monitoring and running of reports, also in Cloud Control if set up as an EM_USER. It does not allow execution of any other tasks.
- **DV_STREAMS_ADMIN:** This role should be granted to any user who is responsible for configuring Oracle Streams in the Database Vault environment.

Database Vault Roles (continued)

- **DV_XSTREAM_ADMIN:** Grant this role to any user who is responsible for configuring Oracle XStream in the Database Vault environment.
- **DV_GOLDENGATE_ADMIN**: Grant this role to any user who is responsible for configuring Oracle GoldenGate in a Database Vault environment.
- **DV_GOLDENGATE_REDO_ACCESS:** Grant this role to any user who is responsible for using the Oracle GoldenGate `TRANLOGOPTIONS DBLOGREADER` method to access redo logs in a Database Vault environment.
- **DV_AUDIT_CLEANUP:** Grant this role to any user who is responsible for purging the Database Vault audit trail in a non-unified auditing environment.
- **DV_PATCH_ADMIN:** Grant this role temporarily to any database administrator who performs database patching. After the patch is complete, you should immediately revoke this role.

For managing accounts, the following role is created:

- **DV_ACCTMGR:** This is the only role that allows creating and altering users and profiles. This role is granted to the user that you specify as the Database Vault Account Manager during installation. If you do not specify a user (*not recommended*), this role is granted to the Database Vault Owner user, effectively making that user the one that manages the accounts. In addition to maintaining users and profiles, this role can grant `CONNECT` and itself (the `DV_ACCTMGR` role).

For establishing users who manage realms, you can use the following roles:

- **DV_REALM_OWNER:** This role is used for managing database objects in multiple schemas that define an application or a realm. The role should be granted to the database account owner who would manage several schema database accounts within a realm and the roles associated with the realm. A user who is granted this role can use powerful privileges, such as `CREATE ANY`, `ALTER ANY`, and `DROP ANY` system privileges within the realm. Note that although this role has system privileges that the `SYS` user controls, it does not have the `DV_OWNER` or `DV_ADMIN` privileges, which means that it cannot change the Database Vault configuration.
- **DV_REALM_RESOURCE:** This role provides system privileges similar to those that exist in the Oracle `RESOURCE` role (which allows users to create objects in their own schema). The role can be granted to a database account that owns the database tables, objects, triggers, views, procedures, and so on that support a database application. The role is geared toward a schema type database account. As with the `DV_REALM_OWNER` role, the role does not have the `DV_OWNER` or `DV_ADMIN` privilege. The `SYS` user is the only one who can grant this role.

To support public permissions, use the following role:

- **DV_PUBLIC:** The only permissions granted to this role upon installation are several `EXECUTE` permissions on the Database Vault procedures. These procedures are benign in that they do nothing to reconfigure the system without the appropriate checks being done. Most of them are simple interrogation routines to retrieve information about the components or the environment. For more information about the procedures that this role can execute, refer to the *Oracle Database Vault Administration Guide*.

Database Vault Schemas

The following database accounts are set up in a Database Vault installation:

- DVSYS owns:
  - Database Vault views
  - The procedures that make up the Database Vault API
  - Realm definitions, command rule definitions, and other Database Vault definitions that have been configured
  - DV audit records
- DVF owns:
  - Factor API functions

Two other accounts are created in a Database Vault installation that are locked and not intended to be connected in a session. They hold the schemas for Database Vault–specific objects.

The DVSYS schema contains:

- The views that display information about all the Database Vault objects
- The packages that provide the Database Vault API functionality
- Realm definitions, command rule definitions, and other Database Vault definitions that have been configured
- The Database Vault audit information
  Note that in a unified auditing enabled database, the Database Vault audit information is stored in SYS schema tables.

The DVF schema contains all the Database Vault functions used to retrieve factor identities.

What to Expect After You Enable Database Vault

- How Database Vault restricts user authorizations
- Using new database roles to enforce clear separation of duties between:
  - Account management responsibility
  - Data security responsibility
  - Database resource management
- Modified AUDIT statement settings for a non-unified audit environment

More details follow for:

- Initialization and password parameter settings that change
- Privileges that are revoked from existing users and roles
- Privileges that are prevented for existing users and roles

ORACLE

- One common audit problem that has affected several large organizations is the unauthorized creation of new database accounts by a database administrator within a production instance. Upon installation, Database Vault prevents anyone other than the Database Vault account manager or a user granted the Database Vault account manager role from creating users in the database.
- To meet regulatory, privacy and other compliance requirements, Database Vault implements the concept of separation of duty. Database Vault makes clear separation between the account management responsibility, data security responsibility, and database resource management responsibility inside the database. This means that the concept of a super privileged user (for example, `DBA`) is divided among several new database roles to ensure that no one user has full control over both the data and configuration of the system. Database Vault prevents the `SYS` user and other accounts with the `DBA` role and other system privileges from accessing designated protected areas of the database called realms.
- When you configure Database Vault and if you decide not to use unified auditing, Database Vault configures several `AUDIT` statements in the database.

Database Parameters Altered During Configuration

Parameter	Default Value	New Value
AUDIT_SYS_OPERATIONS	FALSE	TRUE
OS_ROLES	Not Configured	FALSE
RECYCLEBIN	ON	OFF
REMOTE_LOGIN_PASSWORDFILE	EXCLUSIVE	EXCLUSIVE
SQL92_SECURITY	FALSE	TRUE

Confirm that the new values are OK for your environment.

The Database Vault configuration modifies several database initialization parameter settings to better secure your database configuration. If these changes adversely affect your organizational processes or database maintenance procedures, you can revert to the original settings. The parameters are changed regardless of whether the parameter was a default or custom value before the Database Vault configuration:

- **AUDIT_SYS_OPERATIONS = TRUE:** This enables the auditing of operations that are issued by the SYS user and users connecting with the SYSDBA or SYSOPER privileges.
- **OS_ROLES = FALSE:** This disables the operating system to completely manage the granting and revoking of roles to users.
- **RECYCLEBIN=OFF:** This turns off RECYCLEBIN so that dropped tables do not go into the recycle bin. If the recycle bin is enabled, the realm-protected objects that are dropped would go into the recycle bin, and then the object would no longer be protected by the realm.
- **REMOTE_LOGIN_PASSWORDFILE = EXCLUSIVE:** This enforces the use of a password file to authenticate users.
- **SQL92_SECURITY = TRUE:** This specifies that users must have been granted the SELECT object privilege on a table to execute the UPDATE and DELETE statements where that table is referenced in a WHERE or SET clause.

Revoked Privileges

Privileges that are revoked from existing users and roles

DBA role:
```
SELECT ANY TRANSACTION
CREATE ANY JOB
CREATE EXTERNAL JOB
EXECUTE ANY PROGRAM
EXECUTE ANY CLASS
MANAGE SCHEDULER
DEQUEUE ANY QUEUE
ENQUEUE ANY QUEUE
MANAGE ANY QUEUE
```

IMP_FULL_DATABASE role:
```
BECOME USER
MANAGE ANY QUEUE
```
PUBLIC role:
```
EXECUTE ON UTL_FILE
```

EXECUTE_CATALOG_ROLE role:
```
EXECUTE ON DBMS_LOGMNR
EXECUTE ON DBMS_LOGMNR_D
EXECUTE ON DBMS_LOGMNR_LOGREP_DICT
EXECUTE ON DBMS_LOGMNR_SESSION
EXECUTE ON DBMS_FILE_TRANSFER
```

SCHEDULER_ADMIN role:
```
CREATE ANY JOB
CREATE EXTERNAL JOB
EXECUTE ANY PROGRAM
EXECUTE ANY CLASS
MANAGE SCHEDULER
```

ORACLE

When you install Database Vault, it revokes a set of privileges from several Oracle Database-supplied users and roles, as part of the separation of duty and to help achieve a least-privilege model that makes the databases and applications more secure.

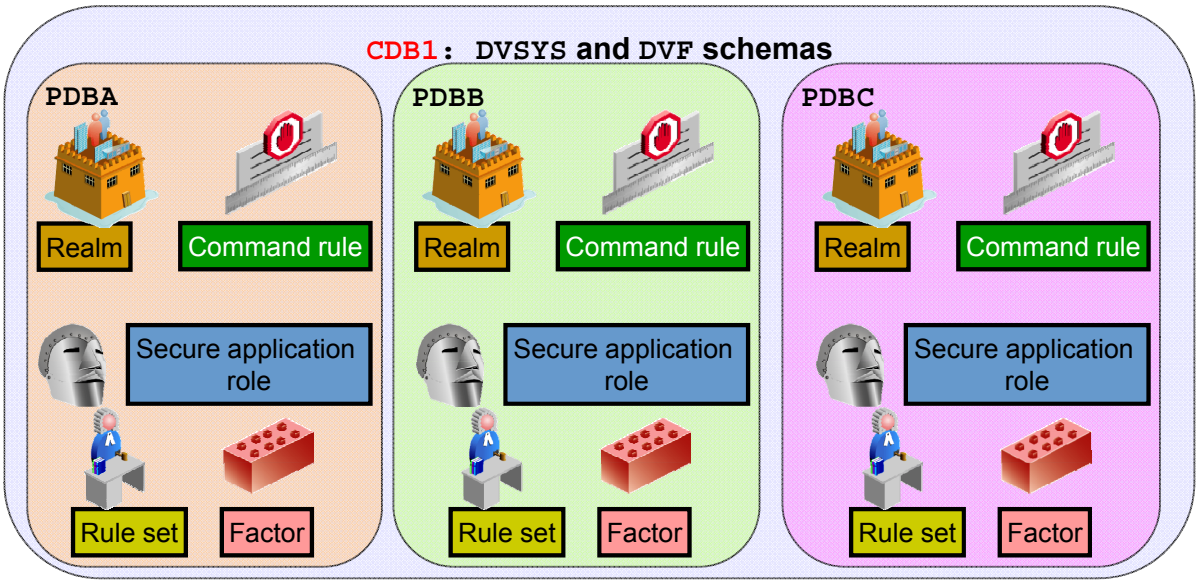
Prevented Privileges

- CREATE USER
- ALTER USER
- DROP USER
- CREATE PROFILE
- ALTER PROFILE
- DROP PROFILE

The privileges listed in the slide are prevented for all users and roles that have been granted these privileges, including users SYS and SYSTEM. This reduces the attack surface that a potential intruder can misuse.

Securing Data in Multitenant Environments

- The DVSYS and DVF schemas are common schemas.
- Each PDB has its own Database Vault metadata.

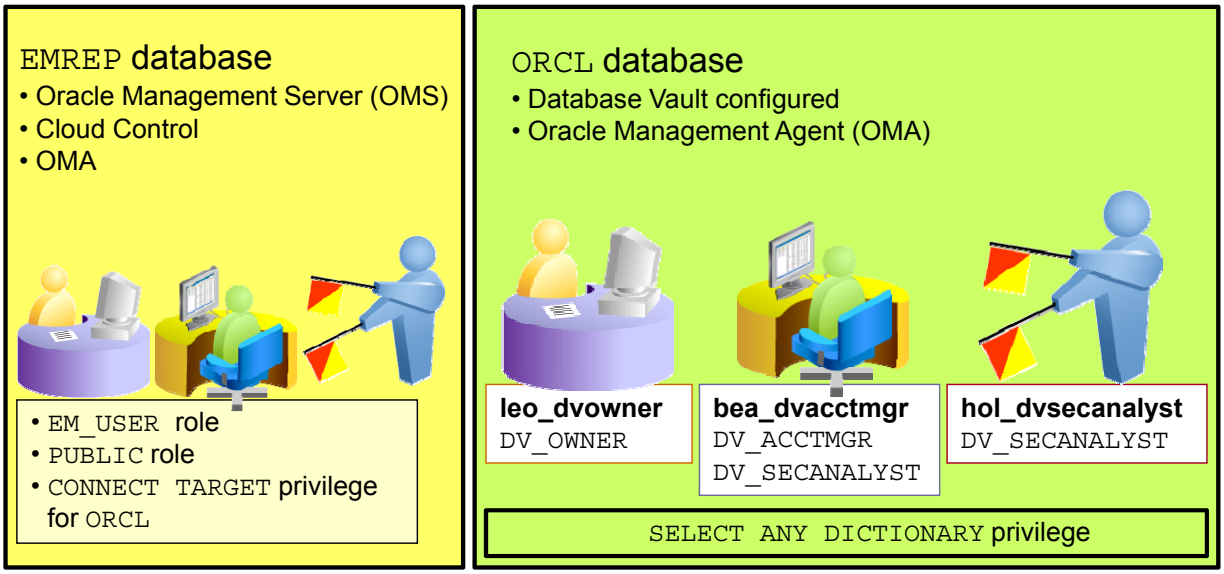


In a Container Database, the DVSYS and DVF schemas are common users stored in the root. Configure and enable Database Vault at the pluggable database (PDB) level. Database Vault policies are scoped to individual pluggable databases, including realms, rules, rule sets, command rules, and secure application roles. This means that each PDB has its own database vault metadata.

1. In the PDB, connect to the PDB as SYS to create the users representing the Database Vault owner and the optional Database Vault account manager as local users.
2. As LEO_DVOWNER in the PDB, connect to configure and enable Database Vault.
3. As SYS in the PDB, connect to restart the PDB.
4. Check that Database Vault is enabled in the PDB:

```
 SQL> select parameter, value from v$option
  2  where parameter like '%Vault';
PARAMETER                      VALUE
------------------------------ --------
Oracle Database Vault          TRUE
```

Configuring Database Vault Users in Cloud Control 12*c*



This course uses two database instances: `EMREP` for Cloud Control and `ORCL` for Database Vault. The Oracle Management Agent (OMA) has discovered `ORCL` and it is promoted to be a Cloud Control target.

The following is an example procedure to set up Oracle users so that they have access to Database Vault via Cloud Control:

1. As `SYSMAN`, log on to Cloud Control via https://<hostname>:7802/em.
2. Navigate to Setup > Security > Administrators > Create.
3. Define the user with `EM_USER` and `PUBLIC` role and the `CONNECT` to the ORCL target privilege.

Tip: It is helpful to give the Oracle user accounts on both instances the same name: `leo_dvowner` on `ORCL` is `leo_dvowner` on `EMREP`.

The “Configuring Database Vault Users in Cloud Control 12*c*” video shows you these steps.

Quiz

The user with the `DV_ACCTMGR` role can modify the passwords of existing users, including that of the user with the `DV_OWNER` role.

a. True
b. False

Answer: b

Quiz

Select all statements that are correct for when Database Vault is enabled.

a. There are no changes for the `SYS` user.
b. The `DBA` role has several privileges removed.
c. `CREATE`, `ALTER`, and `DROP USER` are prevented for all users including `SYS` and `SYSTEM`.
d. Security administrators with the `DV_ACCTMGR` role can create user accounts.

Answer: b, c, d

Summary

In this lesson, you should have learned how to:

- Enable and register Database Vault
- List the accounts and schemas that are created during configuration and describe how they enhance security
- Explain the configuration, privilege, and role changes that occur when Database Vault is enabled

Practices

- 3-1: Configuring Database Vault
- 3-2: Setting Up Practice Accounts
- 3-3: (Optional) Viewing Configuration Videos
  - "Configuring Database Vault in Oracle Database 12*c*"
  - "Configuring Database Vault Users in Cloud Control 12*c*"
- 3-4: Configuring a Database Vault User in Cloud Control 12*c*

ORACLE

Practices 3-1, 3-2, and 3-4 are mandatory. All following practices in this course depend on their correct completion.

4

Analyzing Privileges

Objectives

After completing this lesson, you should be able to:

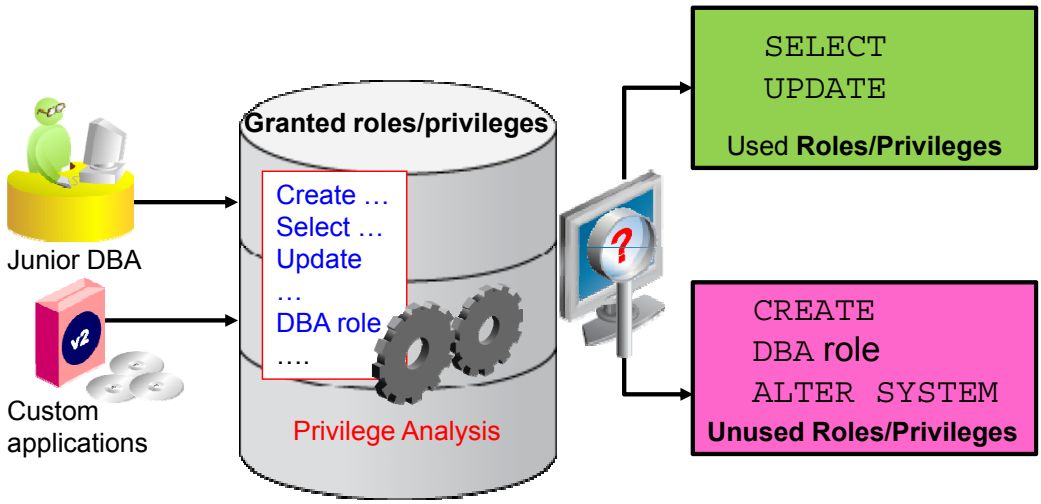
- Reduce the attack surface for potential security breeches
- Find the actual privilege usage in your database
- Use graphical and command-line tools
- Manage privilege analysis policies
  - Describe and use different policy types
  - Set up a privilege analysis policy
  - Generate and view privilege reports

ORACLE

To achieve the objectives listed in the slide, you first gain an overview of concepts and functionality, and then you view videos or demonstrations and learn from real-world examples.

Privilege Analysis Overview

- Identify privileged users and applications
- Report on actual privileges and roles used in the database
- Reduce attack surface by reducing privileges and roles to what is needed



ORACLE

To enhance your database protection (that is, to reduce potential attack surfaces), you need to know the actual privileges and roles used in the database. This allows you to identify privileged users and applications and to reduce the privileges and roles to what is actually needed. The graphic in the slide illustrates the process of distinguishing between used and unused roles and privileges.

Privilege Analysis Features

Privilege analysis:

- Can be performed with or without the Database Vault configuration
- Applies across all database sessions and remains active even after database restarts
- May have a scope of: user, role, context, or database
- Has minimal performance overhead
- Captures direct and indirect role grants

Features of privilege analysis include that it:

- Can be performed with or without the Database Vault configuration
- Applies across all database sessions and remains active even after database restarts
- May have a scope of user, role, context (for example, IP address) or database
- Has minimal performance overhead—less than 5% on tested production systems
- Captures direct and indirect role grants

What Is Privilege Analysis? How Does It Work?

Finding actual privilege usage including system and object privileges to determine least privilege access.

1. Create and enable a privilege analysis policy.
2. Generate a report that describes the used privileges
3. View this report by querying the “Privilege Analysis Policy and Report Data Dictionary” views

ORACLE

To determine least privilege access, you need to find information about the actual privilege usage in a database. For example, you might need to know the privileges required to run an application module or the privileges used in a given user session. The privilege analysis includes both system privileges and object privileges.

1. When a user performs an action and you want to monitor the privileges that are used for this action, you can create and enable a privilege analysis policy (one analysis policy at a time). It captures direct and indirect role grants.
2. Afterwards, you can generate a report that describes the used privileges.
3. To view this report, query the data dictionary views under “Privilege Analysis Policy and Report Data Dictionary Views.”

Types of Privilege Analysis

Analyzing:

- Role
- Context
- Role and context
- Database

Within the pluggable database (PDB) where the privilege analysis resides

Restrictions:

- One enabled policy at a time
  - Except for a database-wide privilege analysis policy at the same time as a non-database-wide privilege analysis policy, such as a role analysis
- Not for the privileges of the `SYS` user

ORACLE

- **Role**: You must provide a list of roles. If a used privilege is from one of the provided roles, Oracle Database analyzes the privilege use.
- **Context**: You must specify a Boolean expression only with the `SYS_CONTEXT` function. The used privileges are analyzed if the condition evaluates to `TRUE`.
- **Role and context**: You must provide both a list of roles to analyze and a `SYS_CONTEXT` Boolean expression for the condition. When a used privilege is from one of the analyzed roles and the given context condition is satisfied, the privilege is analyzed.
- **Database**: If you do not specify conditions in your privilege analysis policy, all privilege use in the database is analyzed, except for privileges that user SYS exercises. (This is also referred to as unconditional analysis, because it is turned on without any conditions.)

  Note: If you are using a multitenant environment, each privilege analysis policy analyzes and reports only privileges exercised within the pluggable database (PDB) where the privilege analysis resides.

Restrictions:

- You can enable only one privilege analysis policy at a time. The only exception is that you can enable a database-wide privilege analysis policy at the same time as a non-database-wide privilege analysis policy, such as an analysis policy driven by a role or context.
- You cannot analyze the privileges of the `SYS` user.

What Are Your Tools and Prerequisites?

Tools:

- Enterprise Manager Cloud Control (GUI)
- The `DBMS_PRIVILEGE_CAPTURE` package (command line)

Prerequisites:

- With or without Database Vault configuration
- The `CAPTURE_ADMIN` role:
  - The `EXECUTE` privilege for the `DBMS_PRIVILEGE_CAPTURE` package
  - The `SELECT` privilege on the `DBA_*` report views

- To analyze privileges, use Enterprise Manager Cloud Control or the `DBMS_PRIVILEGE_CAPTURE` package.
- You can perform privilege analysis with or without having Database Vault configured and enabled. But you must be granted the `CAPTURE_ADMIN` role, which provides the `EXECUTE` privilege for the `DBMS_PRIVILEGE_CAPTURE` package and the `SELECT` privilege on the `DBA_*` report views to view the generated report.

Managing Privilege Analysis Policies

General steps :

1. Define the privilege analysis policy.
2. Enable the privilege analysis policy.
3. Disable the privilege analysis policy's recording of privilege use.
4. Generate privilege analysis results.
5. Optionally, disable and then drop the privilege analysis policy.

General Steps for Managing Privilege Analysis

1. Define the privilege analysis policy.
2. Enable the privilege analysis policy. This step begins recording the privilege use that the policy defined.
3. Disable the privilege analysis policy's recording of privilege use. This step enables you to define a snapshot of the privilege based on an ending time.
4. Generate privilege analysis results. This step writes the results to the data dictionary views.
5. Optionally, disable and then drop the privilege analysis policy. Dropping a privilege analysis policy deletes the analyzed privilege reports associated with the policy.

Analyzing ANY Privilege Use

Use the DBMS_PRIVILEGE_CAPTURE package:

1. Define the analysis with the CREATE_CAPTURE procedure.
2. Enable the analysis with ENABLE_CAPTURE procedure.
3. Perform the action to be tested.
4. Disable the privilege analysis policy's recording by using the DISABLE_CAPTURE procedure.
5. Generate privilege analysis report by using the GENERATE_RESULT procedure, and use the DBA_USED_PRIVS view to analyze the results.
6. Optionally, disable and then drop the privilege analysis policy with the DROP_CAPTURE procedure.

ORACLE

View the video and/or the product documentation tutorial to see this example. The step-by-step tutorial has the following steps:

1. Create User Accounts
2. Create and Enable a Privilege Analysis Policy: You define the privilege analysis policy by using the CREATE_CAPTURE procedure of the DBMS_PRIVILEGE_CAPTURE package.
3. Use the READ ANY TABLE System Privilege
4. Disable the Privilege Analysis Policy
5. Generate and View a Privilege Analysis Report
6. Remove the Components for This Tutorial

Report example:

```
SELECT USERNAME, SYS_PRIV, OBJECT_OWNER, OBJECT_NAME
FROM DBA_USED_PRIVS
WHERE USERNAME = 'APP_USER';
```

Analyzing Privilege Use by a User Who Has the DBA Role

Use the DBMS_PRIVILEGE_CAPTURE package:

1. Define the analysis with the CREATE_CAPTURE procedure.
2. Enable the analysis with ENABLE_CAPTURE procedure.
3. Perform a database tuning operation.
4. Disable the privilege analysis policy's recording by using the DISABLE_CAPTURE procedure.
5. Generate privilege analysis report by using the GENERATE_RESULT procedure, and analyze the result in the DBA_USED_SYSPRIVS_PATH and DBA_USED_OBJPRIVS views.
6. Optionally, disable and then drop the privilege analysis policy with the DROP_CAPTURE procedure.

ORACLE

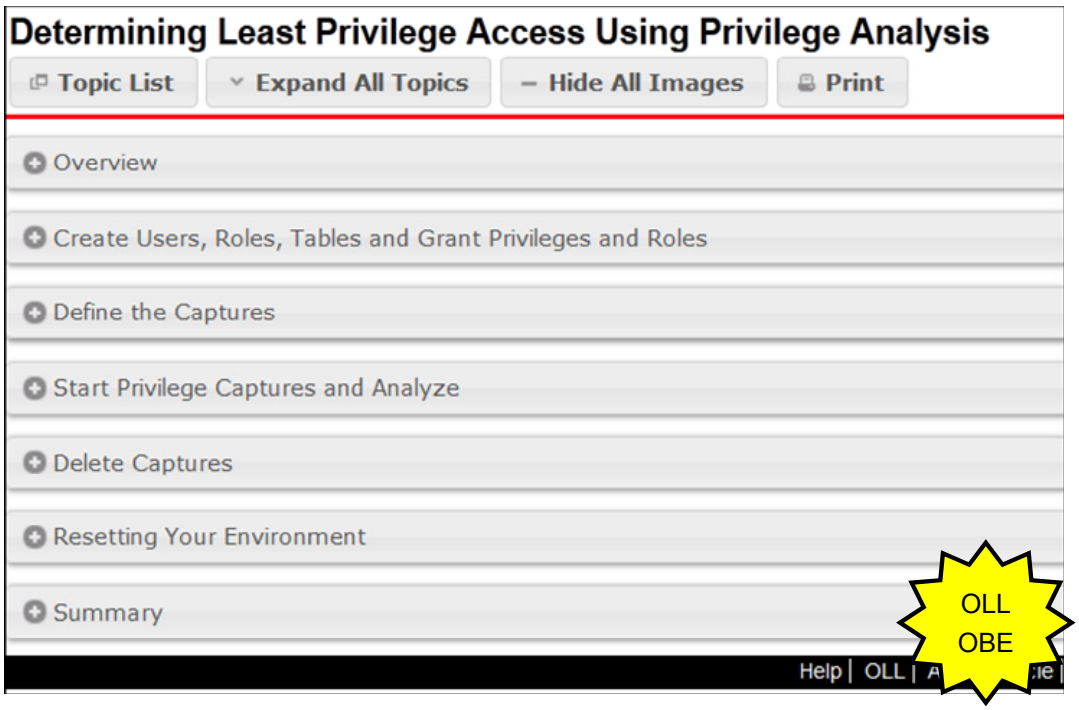
View the video and/or the product documentation tutorial to see this example. The tutorial has the following steps:

1. Create User Accounts
2. Create and Enable a Privilege Analysis Policy
3. Perform the Database Tuning Operations
4. Disable the Privilege Analysis Policy
5. Generate and View a Privilege Analysis Report
6. Remove the Components for This Tutorial

Report example:

```
SELECT USERNAME, SYS_PRIV,  USED_ROLE,  PATH
FROM DBA_USED_SYSPRIVS_PATH
WHERE USERNAME = 'TJONES'
ORDER BY USERNAME, SYS_PRIV, USED_ROLE;
```

Determining Least Privilege Access Using Privilege Analysis



The Oracle Learning Library (OLL) contains many useful items to assist you with your Oracle tasks; for example, the Oracle By Example (OBE)s show tasks with screenshots, step-by-step, and deliver the setup files that you may need for this type of learning.

View and do the “Determining Least Privilege Access Using Privilege Analysis” OBE.

Quiz

Select the statements that are true about Privilege Analysis:

a. It applies across all database sessions and remains active even after database restarts.
b. You must have Database Vault configured to perform Privilege Analysis.
c. It captures direct and indirect role grants.
d. It has no performance impact at all.
e. You need to disable the analysis before you can generate the resulting report.

Answer: a, c, e

Summary

In this lesson, you should have learned how to:

- Reduce the attack surface for potential security breeches
- Find the actual privilege usage in your database
- Use graphical and command-line tools
- Manage privilege analysis policies
  - Describe and use different policy types
  - Set up a privilege analysis policy
  - Generate and view privilege reports

Practices

- 4-1: Analyzing Privileges Used by Any User
- 4-2: Analyzing `ANY` Privilege Use in Context
- 4-3: Analyzing Role-Based Privileges

Three different practices are available to determine least-privilege access. You can choose the practices that are the most valuable for your organization. To list all existing captures, you could use:

```
COL name      FORMAT A12
COL type      FORMAT A16
COL enabled   FORMAT A2
COL roles     FORMAT A24
COL context   FORMAT A43

select name, type, enabled, roles, context
from   dba_priv_captures;
```

5

Configuring Realms

Objectives

After completing this lesson, you should be able to do the following:

- Describe how realms work
- Identify the uses of a realm
- Protect schema objects from DBAs
- Create and update realms that prevent unauthorized granting of privileges (protect a role with a realm)
- Protect a schema with a mandatory realm
- Maintain realms with Cloud Control or by using the realm API
- Describe the use of default realms and reports

ORACLE

Quiz

Self-assessment: Select all statements that are true about Database Vault.

a. Users with the `SELECT ANY TABLE` privilege have to be listed as realm participants in order to select data from a realm-protected table.
b. Realms do not use rule sets.
c. Rule sets may help define the behavior of realms.
d. Realm access is evaluated when a SQL statement is requested to be executed.

Answer: a, c, d

This self-assessment is based on the lesson titled “Database Vault Overview”.

How Realms Work

If YES:	Evaluation step	If NO:
Go to 2	1. Referenced objects in realm?	Go to 7
Go to 3	2. Mandatory realm?	Go to 4
Go to 4	3. Using system privilege?	Go to 6
Go to 5	4. Realm owner or realm participant?	Realm violation
Go to 6	5. Conditionally based on a rule set?	Go to 7
Go to 7	6. Rule set `TRUE`?	Realm violation
Command rule violation	7. Command rule restriction?	Successful command

When a SQL statement is processed by Database Vault, it is checked for realm violations. The steps are as follows:

1. Does the SQL statement affect objects secured by a realm? If yes, then go to Step 2. If no, then realms do not affect the SQL statement. Go to Step 7.
2. Is the realm a mandatory realm? If yes, then go to Step 4. If it is a regular realm, then go to Step 3.
3. Is the database account using a system privilege to execute the SQL statement? If yes, then go to Step 4. If no, then go to Step 6. If the session has object privileges on the object in question for `SELECT`, `EXECUTE`, and DML statements only, then the realm protection is not enforced. Realms protect against the use of any system privilege on objects or roles protected by the realm. Remember that if the `O7_DICTIONARY_ACCESSIBILITY` initialization parameter has been set to TRUE, then non-SYS users have access to `SYS` schema objects. For better security, ensure that `O7_DICTIONARY_ACCESSIBILITY` is set to `FALSE`.
4. Is the database account a realm owner or realm participant? If yes, then go to Step 5.

How Realms Work (continued)

4. Otherwise, a realm violation occurs and the statement is not allowed to succeed. If the command is a `GRANT` or `REVOKE` of a role that is protected by the realm, or the `GRANT` or `REVOKE` of an object privilege on an object protected by the realm, then the session must be authorized as the realm owner directly or indirectly through roles.
5. Is the realm authorization for the database account conditionally based on a rule set? If yes, then go to Step 6. If no, then go to Step 7.
6. Does the rule set evaluate to `TRUE`? If yes, then go to Step 7. If no, then there is a realm violation and the SQL statement is not allowed to succeed.
7. Does a command rule prevent the command from executing? If yes, then there is a command rule violation and the SQL statement fails. If no, then there is no realm or command rule violation and the command succeeds.

Benefits of Using Realms

A realm protects areas of your database from otherwise privileged users. A realm:

- Protects against the insider threat of the all-powerful DBA
- Defines which users or roles can exercise their system privileges against which objects
- Limits which users can grant privileges on protected objects
- Helps with compliance requirements found in PCI, SOX, HIPAA, and others
- Protects against cyber threats that target privileged users
- Provides additional protection for privacy related data

Use realms to protect a set of database objects or roles from otherwise privileged users.

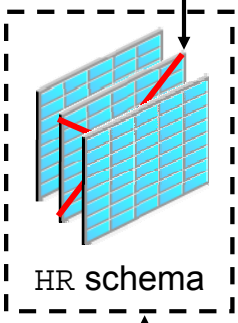
- For instance, DBAs have sweeping privileges, such as `SELECT ANY TABLE` or `DROP ANY TABLE`. These users can read or destroy data that they may not necessarily require access to. It would be better to limit access to that data to application users and application DBAs.
- A realm can be defined to limit access. You can define the realm, add the objects to it that are protected, and add the users who are allowed to access the objects that are under the realm.
- Realms can also limit the set of users who can grant privileges on the objects inside the realm. This would apply to, for example, granting `EXECUTE` on packages, `SELECT` on views, and `GRANT` on roles.
- Realms also help with compliance requirements found in PCI, SOX, HIPAA, and others that require granting access to sensitive data based on a need-to-know basis.
- Mandatory realms can be used in response to a cyber threat, preventing all access until the threat has been analyzed.

Protecting Objects from DBAs

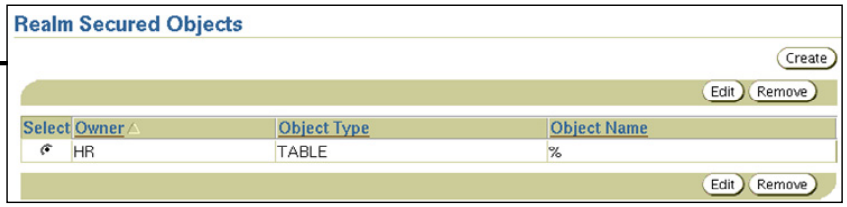
Use Case

1 SALES_DBA can drop an HR table because this user has the DBA role:

```
SQL> CONNECT sales_dba/password
SQL> DROP TABLE hr.bonus_it;
Table dropped.
```



2 leo_dvowner creates a realm to secure the HR table:



3 SALES_DBA attempts to drop the restored HR table:

```
SQL> DROP TABLE hr.bonus_it;
ORA-47401: Realm Violation for drop table
    on HR.BONUS_IT
```

ORACLE

In this example, the SALES_DBA user has been set up as the database administrator for the Sales application. Accordingly, this user has been assigned the DBA role, which means that the role has many powerful privileges, including DROP ANY TABLE. Normally, SALES_DBA would be able to drop any table in the database, including those in other schemas, such as HR. Step 1 shows how this application DBA is able to drop a table belonging to another application. In step 2, the leo_dvowner user uses Database Vault Administrator to create a realm and secure all the HR tables in that realm. Then, in step 3, the same SALES_DBA user attempts to drop the same (now restored) HR table, but is unable to. Instead, a realm violation error is returned, and the DROP TABLE command fails.

Effect of Realms on Nonmembers

- Realms prevent access to database objects, except when object privileges are granted to provide access, such as:
  - `SELECT ON HR.EMPLOYEES`
  - `EXECUTE ON HR.GIVE_RAISE`
- Therefore, system privileges such as the following do not provide access to realm-secured objects:
  - `SELECT ANY TABLE`
  - `EXECUTE ANY PROCEDURE`
  - `CREATE TABLE`
- Schema owners in regular realms:
  - Are allowed DML operations, such as `SELECT`, `INSERT`, `UPDATE`, and `DELETE`
  - Need to be authorized to the realm to perform DDL operations, such as `DROP`, `CREATE`, and `ALTER`

ORACLE

If a user is not a member of a realm authorization, the only way that user can access the realm-secured objects is to have the relevant object privileges granted to him or her. A user cannot rely on system privileges, such as `SELECT ANY TABLE` to access objects, if those objects are secured in a realm.

Also, users **cannot** rely on the fact that they own the schema being accessed. Schema owners cannot drop, alter, or create objects in their own schemas if that schema is protected in a realm, and the schema owners are not members of that realm. To provide this access, the schema owner can be authorized to the realm. The other users can be granted either the `DV_REALM_OWNER` role or direct object-level privileges on the schema's objects. The grant would have to be done by a user who is already a member (specifically an owner) in the realm.

Protecting Roles

Use Case

1. The HR user creates the BENNIES standard database role:

```
SQL> CREATE ROLE bennies;
Role created.
```

2. leo_dvowner secures the BENNIES role in a realm:

Select	Owner	Object Type	Object Name
◉	ANONYMOUS	ROLE	BENNIES

3. leo_dvowner adds the HR user as a participant in the realm:

Select	Grantee	Authorization Options	Authorization Rule Set Name
◉	HR	Participant	

4. HR is not able to grant the role:

```
SQL> grant bennies to sh;
ORA-47401: Realm Violation for grant role privilege on NULL.NULL
```

ORACLE

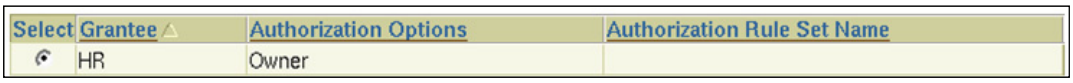
The following steps illustrate how to protect a role by using a realm:

1. A user creates the BENNIES role.
2. The leo_dvowner user uses Database Vault Administrator (DVA) to secure the BENNIES role in a realm.
3. The leo_dvowner user uses DVA to add HR as a participant in that realm.
4. The HR user attempts to grant the role, but fails with a realm violation. This is because the HR user is a participant and not an owner in the realm.

Protecting Roles

Use Case

5 `leo_dvowner` changes the `HR` user to be an owner in the realm:



Select	Grantee	Authorization Options	Authorization Rule Set Name
◉	HR	Owner	

6 `HR` is able to grant the role:

```
SQL> GRANT bennies TO sh;
Grant succeeded.
```

7 `HR` is also able to revoke the role:

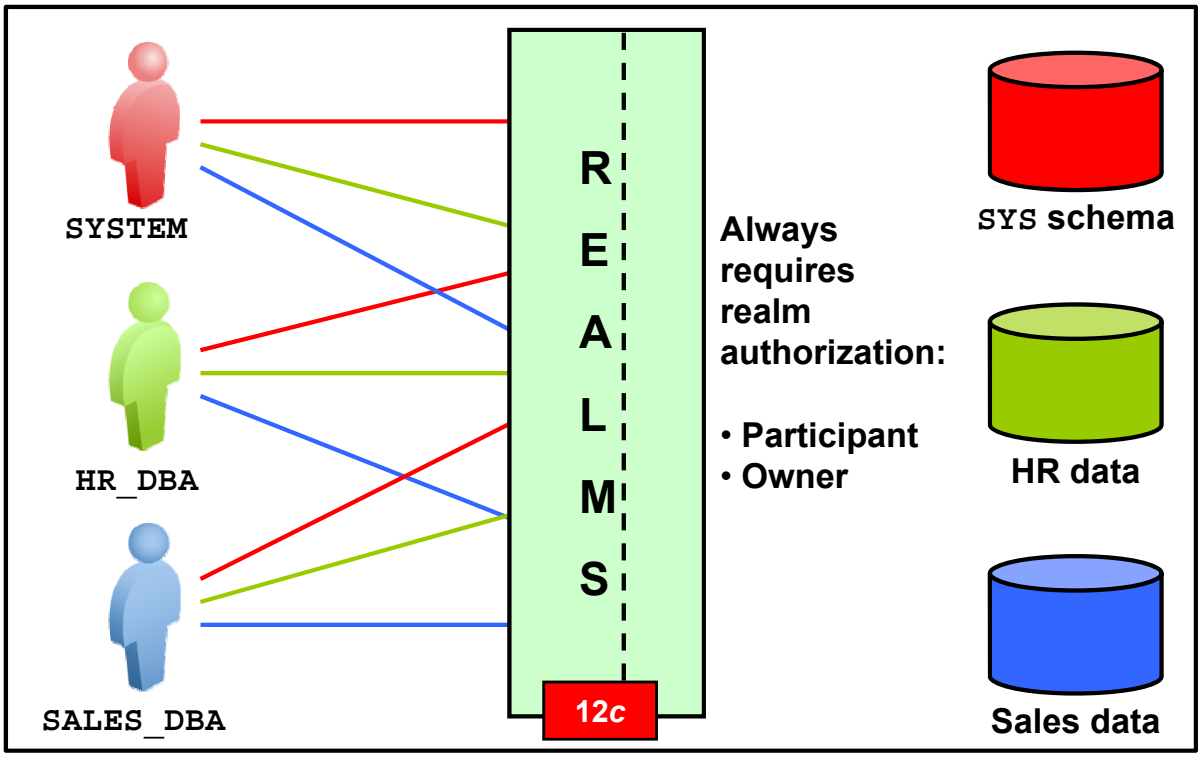
```
SQL> REVOKE bennies FROM sh;
Revoke succeeded.
```

ORACLE

Protecting Roles (continued)

5. The `leo_dvowner` user uses DVA to change the `HR` user to be an owner in the realm rather than a participant.
6. The `HR` user is now able to grant the role.
7. The `HR` user is also able to revoke the role.

Mandatory Realms and Object Privileges

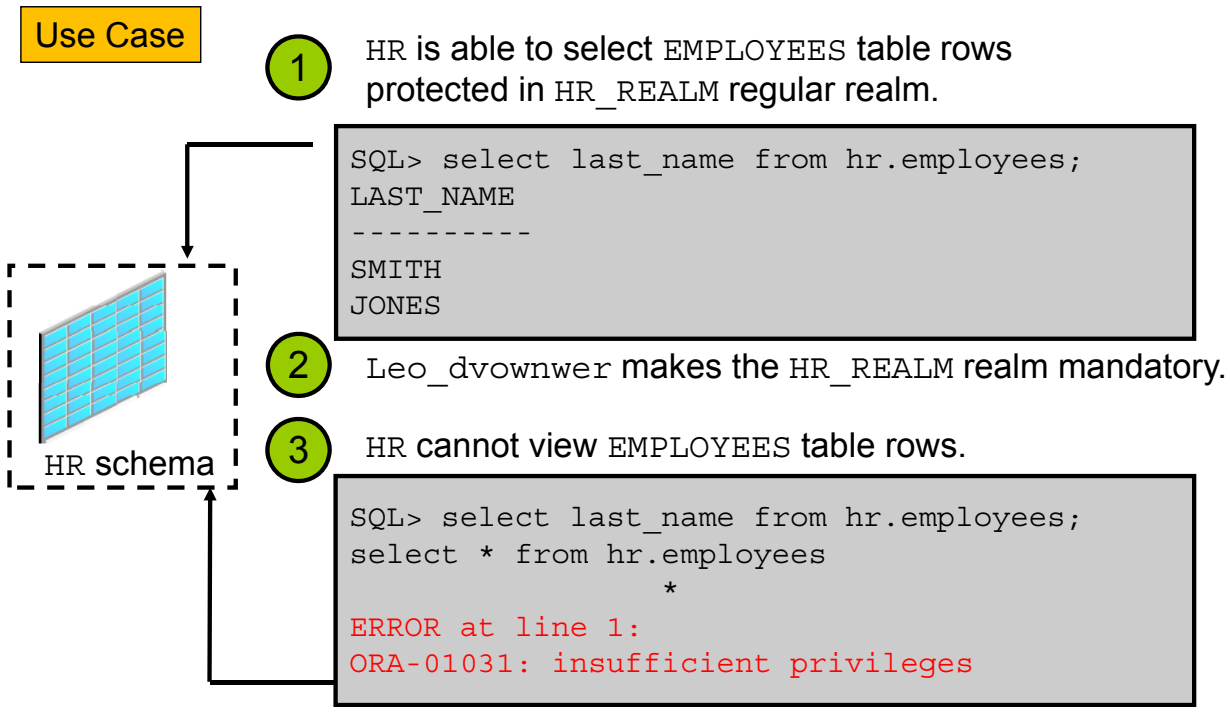


By default, users who own or have object privileges are allowed to access realm-protected objects without explicit realm authorization. However, you can configure the realm to prevent these users from accessing objects, by configuring the realm to be a mandatory realm.

In Oracle Database 12*c*, if you need to prevent users from accessing realm-secured objects by using object privileges, you create a mandatory realm. For this realm, users can access the realm-secured objects only if they are members of the realm authorization. Consequently, the users granted object privileges on the realm-secured objects and the schema owner of realm-secured objects need to be members of the realm to be able to access the realm-secured objects.

You can also use mandatory realms in response to a cyber threat, preventing all access until the threat has been analyzed.

Protecting with a Mandatory Realm



A realm protects the HR schema. The HR user is able to select rows of the protected tables in the HR schema because the owner is always granted OBJECT privileges on his objects. Because this privilege is not always a desirable situation, you decide to disallow the HR user from selecting data from its own tables. You can either update the HR_REALM as mandatory or you can create a mandatory realm and protect the sensitive objects.

Characteristics of Mandatory Realms

- Object in regular AND mandatory realm: More secure rules apply
- Objects in MULTIPLE mandatory realms: User or role must be authorized in all of them
- Protected role: Only realm owner can grant or revoke privileges

- When an object is protected by a regular and a mandatory realm, the more secure rules apply.
- If there are multiple mandatory realms on the same object, you must authorize the user or role on all the mandatory realms before the user can access the protected object.
- If a role is protected by a mandatory realm, then no privileges can be granted to or revoked from the protected role except by the realm owner.

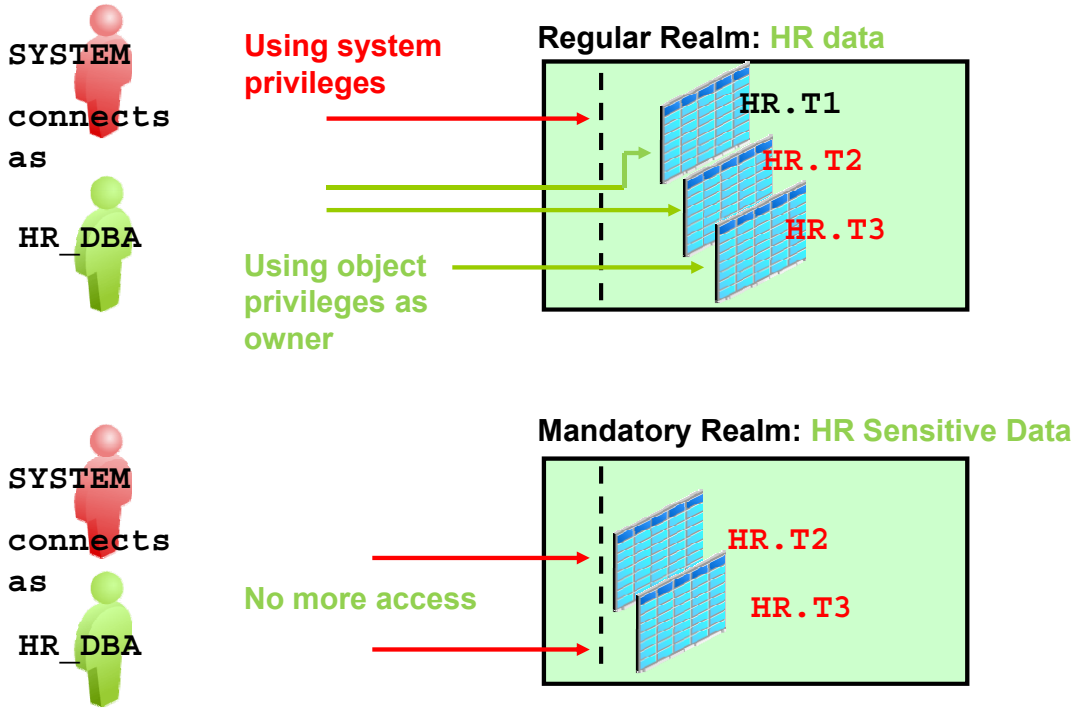
Benefits of Mandatory Realms

- Can block object owners and object privileged users
- Provide more flexible configurations for access control
- Add a layer of protection during patch upgrades
- Secure tables during run time
- Allow freezing of security settings by preventing changes to configured roles
- Add an additional multifactor authorization check even for connections coming through the application account

The benefits of mandatory realms are listed in the slide.

Protecting Sensitive Data During Patching

Use Case

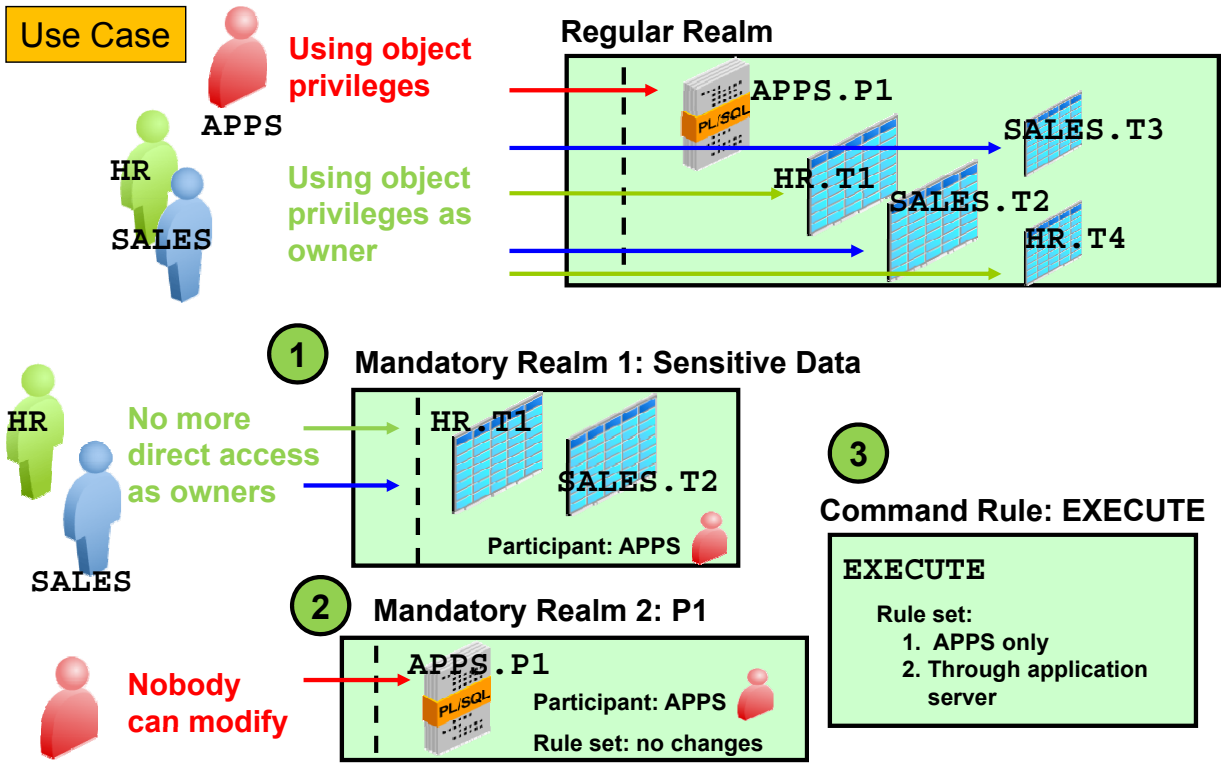


ORACLE

During a patch upgrade, a database administrator may need direct access to a realm-protected object in order to perform a patch on the object. If tables in the same realm contain sensitive data, such as Social Security numbers, you can use mandatory realms to protect these tables from the administrator's access during the patch upgrade.

When the administrator completes the patching and no longer needs access to the objects, you can disable the mandatory realm protection so that the applications can work normally. In this way, mandatory realms can provide protection against DBAs during patching time, even if the DBAs can log in as application schema owners.

Protecting Sensitive Data During Run Time



During run time, application data is stored in many tables of different schemas. It is recommended that you have a single-user `APPS`, such as a runtime schema, to access these tables so that data integrity and accuracy are maintained. When application data scatter in many different schemas, schema owners, as well as users with object privileges, can also change the data if they log in to the database directly.

To make sure that no users can update the tables without running the runtime schemas procedures, use realms to protect the tables. This way, only the authorized user's procedures can access them. Because a regular realm does not block object owners and object privileged users, use mandatory realms to block them, as in steps 1 and 2 in the slide. Only authorized users can access these tables.

Tasks Involving Realms

Graphically or via the command line (the DBMS_MACADM package):

- Create a realm (CREATE_REALM procedure).
- Edit a realm (RENAME_REALM, UPDATE_REALM).
- Maintain objects in a realm (ADD_OBJECT_TO_REALM, DELETE_OBJECT_FROM_REALM).
- Maintain authorizations (ADD_AUTH_TO_REALM, UPDATE_REALM_AUTH, DELETE_AUTH_FROM_REALM).
- Specify an authorization rule set.
- Delete a realm (DELETE_REALM, DELETE_REALM_CASCADE).

- You have to create the realm before adding the objects. This can be done in different tools. All use the CREATE_REALM procedure of the DBMS_MACADM package in the DVSYS schema.
- Editing a realm enables you to change anything that is defined for the realm. This is also the method for securing objects under the realm.
- When you create a realm-secured object, you put the object under the protection of the realm. You specify the owner of the object or objects to be added. Objects belonging to different users, of different types, can be under the same realm. You can use “%” for all object types, or a specific type, such as TABLE or CLUSTER. Similarly, you can use “%”for object names.
- To add authorizations to a realm, define the grantee (the user or role name that is being authorized) and the type: participant or owner (which is the equivalent of the WITH ADMIN option).
- When you add an authorization to a realm, you can also specify an authorization rule set. This rule set must be satisfied to allow access to the realm-protected objects.
- Database Vault deletes the configuration for the realm, including realm authorizations. It does not delete the rule sets used for realm authorizations.

Realm Attributes

When you create a realm, you define:

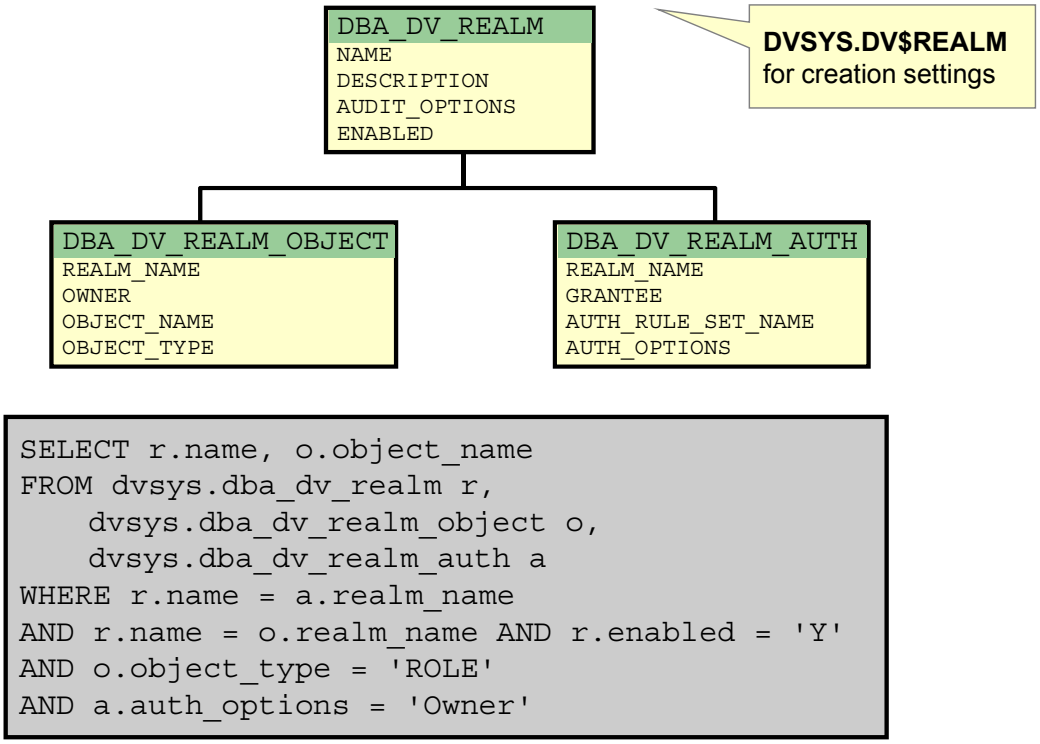
- The realm name and description
- Its status: Enabled or Disabled
- Audit options
- A list of secured objects, qualified by the object:
  - Owner
  - Type, which can be a wildcard, meaning all types
  - Name, which can be a pattern string
- A list of authorizations, including:
  - Authorized user
  - Authorization type indicating participant or owner
  - Authorization rule set (optional)

The following are the attributes of a realm:

- **Name:** The name of the realm. This is used to refer to it later. It is case-sensitive.
- **Description:** A description of the realm
- **Status:** Either Enabled or Disabled. If it is Disabled, it has no effect. The Status is Enabled by default.
- **Audit Options:** Audit options for a realm can be set to one of the following values:
  - Audit Disabled
  - Audit on Failure (default)
  - Audit on Success or Failure

  In a non-unified auditing environment, Database Vault writes the audit trail to the `DVSYS.AUDIT_TRAIL$` table. If you have enabled unified auditing, this setting does not capture audit records. Instead, you must create and enable audit policies to capture this information.
- **Realm Secured Objects:** The list of schema objects and roles that are protected by the schema
- **Realm Authorizations:** The list of authorized users or roles. This defines which users are able to access the objects that are secured by the realm.

Realm Views




```
SELECT r.name, o.object_name
FROM dvsys.dba_dv_realm r,
    dvsys.dba_dv_realm_object o,
    dvsys.dba_dv_realm_auth a
WHERE r.name = a.realm_name
AND r.name = o.realm_name AND r.enabled = 'Y'
AND o.object_type = 'ROLE'
AND a.auth_options = 'Owner'
```

ORACLE

The views that contain realm information are:

- **DBA_DV_REALM:** Each realm is represented here with one row.
  - NAME: The name of the realm
  - DESCRIPTION: The description of the realm
  - AUDIT_OPTIONS: A number indicating when auditing is done:
    - 0: Never audit
    - 1: Audit on failure
    - 3: Audit on success or failure
  - ENABLED: Whether the realm is enabled or not. The value can be Y or N.
- **DBA_DV_REALM_OBJECT:** This view contains the list of realm-secured objects.
  - REALM_NAME: Name of the realm
  - OWNER: Owner of the realm-secured object
  - OBJECT_NAME: Name of the secured object
  - OBJECT_TYPE: Type of the secured object

Realm Views (continued)

- **DBA_DV_REALM_AUTH:** The realm authorizations are represented in this view.
  - REALM_NAME: Name of the realm
  - GRANTEE: User or role authorized to access the realm-secured objects
  - AUTH_RULE_SET_NAME: Rule set that must evaluate to TRUE in order for the grantee to access the realm-secured objects
  - AUTH_OPTIONS: Indicates whether the grantee is able to grant any roles that are secured in this realm:

    Participant: Not able to grant

    Owner: Able to grant
- The **DVSYS.DV$REALM** view describes settings that were used to create Database Vault realms, such as which audit options have been assigned, whether the realm is a mandatory realm, and so on. For more details, see the *Oracle Database Vault Administrator's Guide*.

Oracle-Defined Realms

- Oracle Database Vault
- Database Vault Account Management
- Oracle Enterprise Manager
- Oracle Default Schema Protection Realm
- Oracle System Privilege and Role Management Realm
- Oracle Default Component Protection Realm

- Enabled
- Audit on failure

Default realms are enabled and audit on failure.

- **Oracle Database Vault:** Protects configuration and role information in the Database Vault `DVSYS`, `DVF`, and `LBACSYS` schemas.
- **Database Vault Account Management:** Defines the realm for the administrators who manage and create database accounts and database profiles. This realm protects the `DV_ACCTMGR` and `CONNECT` roles. The owner of this realm can grant or revoke the `CREATE SESSION` privilege to or from a user.
- **Oracle Enterprise Manager:** Protects Oracle Enterprise Manager accounts that are used for monitoring and management (`DBSNMP` user and the `OEM_MONITOR` role).
- **Oracle Default Schema Protection Realm:** Protects roles and schemas that are used with Oracle features such as Oracle OLAP, Oracle Spatial, and Oracle Text.
- **Oracle System Privilege and Role Management Realm:** Protects all sensitive roles that are used for exporting and importing data to and from an Oracle database. This realm also contains authorizations for users who must grant system privileges.

  User `SYS` is the only default owner of this realm. Only owners of this realm can grant the protected roles to other users.
- **Oracle Default Component Protection Realm:** Protects the `SYSTEM` and `OUTLN` schemas. The authorized users of this realm are users `SYS` and `SYSTEM`.

Predefined Reports

To analyze realms:

- Realm Audit Report
- Realm Authorization Configuration
- Rule Set Configuration Issues Report
- Object Privilege Reports
- Privilege Management – Summary Reports
- Sensitive Objects Reports

- **Realm Audit Report Audits**: Records generated by the realm protection and realm authorization operations. This is helpful in troubleshooting rule sets and monitoring failed authorization attempts. Realm violations are also displayed in this report. This report would show when a database account attempts to perform an action on a realm object on which it is not authorized to perform that action. When you configure a realm, you set the audit options for the realm operations.
- **Realm Authorization Configuration**: Lists authorization configuration information, such as incomplete or disabled rule sets, or nonexistent grantees or owners that may affect the realm
- **Rule Set Configuration Issues Report**: Lists rule sets that do not have rules defined or enabled, which may affect the realms that use them
- **Object Privilege Reports:** Lists object privileges that the realm affects
- **Privilege Management – Summary Reports**: Provides information about grantees and owners for a realm
- **Sensitive Objects Reports:** Lists objects that the command rule affects

Quiz

Schema owners must be authorized to access their own objects when they are protected by a mandatory realm.

a. True
b. False

Answer: a

Quiz

Objects belonging to different users or of different types cannot be under the same realm.

a. True
b. False

Answer: b

Summary

In this lesson, you should have learned how to:

- Describe how realms work
- Identify the uses of a realm
- Protect schema objects from DBAs
- Create and update realms that prevent unauthorized granting of privileges (protect a role with a realm)
- Protect a schema with a mandatory realm
- Maintain realms with Cloud Control or by using the realm API
- Describe the use of default realms and reports

Practices

- 5-1: Using Realms to Protect a Schema
- 5-2: Using Realms to Protect Roles (optional)
- 5-3: Using Regular and Mandatory Realms (optional)

Learners should complete at a minimum Practice 5-1, and then they can choose between 5-2 and 5-3 according to which one is most relevant for their organization. If time allows, all three can be completed.

6

Defining Rule Sets

Objectives

After completing this lesson, you should be able to do the following:

- Identify the uses of a rule set
- Manage a rule set
- Create rules and add them to a rule set
- Associate a rule set with a realm to limit access
- View the rule set violation reports

Quiz

Self-assessment: Select all statements that are true about Database Vault.

a. Each rule in a rule set is evaluated by itself and cannot be combined with a logical `AND` or `OR`.
b. Each rule is defined as a `WHERE` clause expression.
c. Database Vault does not have a rule engine to process rule sets. You must configure a third-party product.
d. You can use a rule set to provide dual key security, whereby another separate action must be taking place for the requested access to be granted.

ORACLE

Answer: b, d

This self-assessment is based on the lesson titled “Database Vault Overview.”

Quiz

Self-assessment: Select all statements that are true about Database Vault.

a. Realms, command rules, secure application roles, and factors use rule sets to define their behavior.
b. Rule sets can not refer to factors in their definition.
c. The end result of Database Vault's rule-set evaluation is a single value of `TRUE` or `FALSE`.

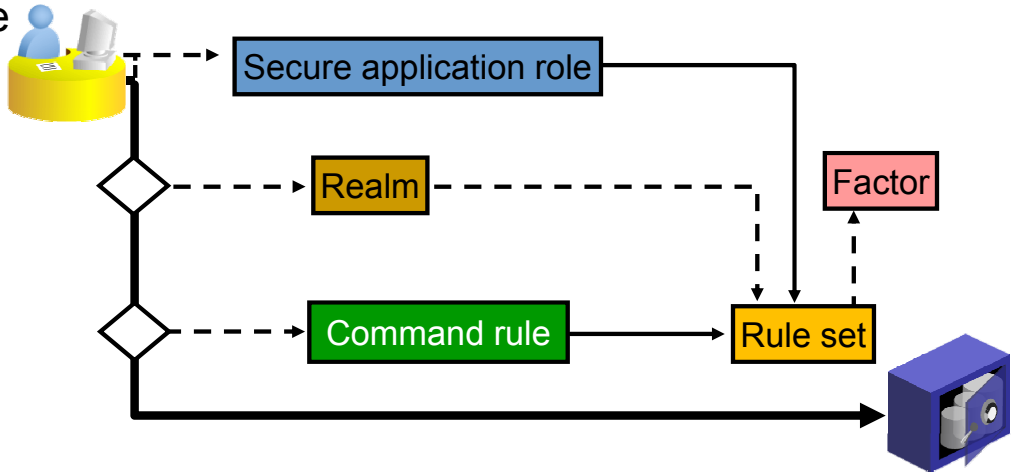
Answer: a, c

This self-assessment is based on the lesson titled "Database Vault Overview."

Quiz

Self-assessment: Realm access is evaluated when a SQL statement is requested to be executed. It may or may not involve rule-set evaluation.

a. True
b. False



Answer: a

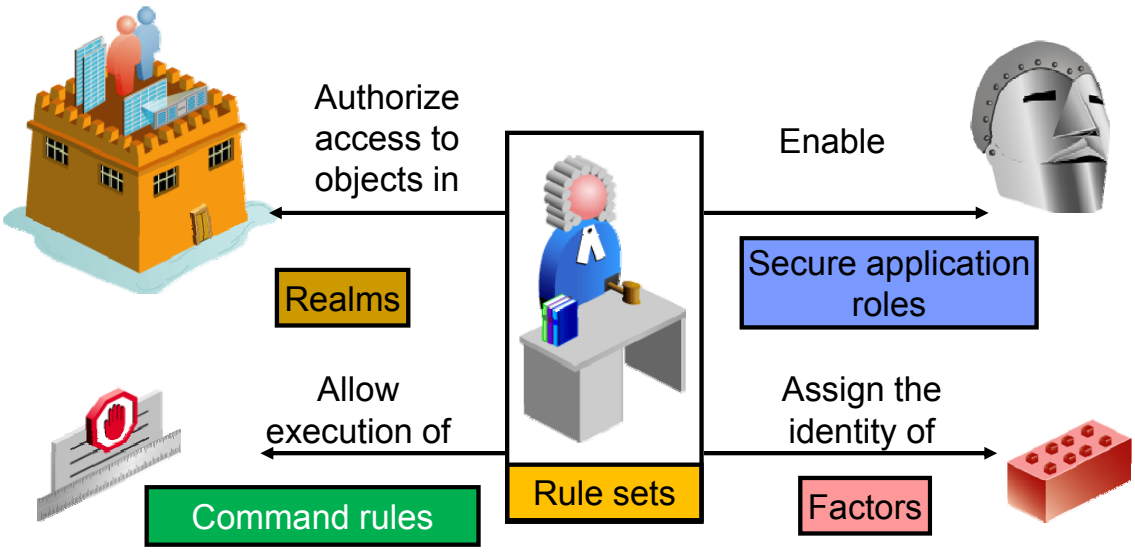
This self-assessment is based on the lesson titled “Database Vault Overview.”

Evaluation of Rule Sets

ORACLE

In this case, where the rule set is defined such that all rules must be TRUE, the rule set evaluates to FALSE. Because the machine is local and the APP.STATUS column is greater than zero, the second rule, which checks whether it is currently a weekend, resolves to FALSE. This result ANDed with the other rule results evaluates to a FALSE value for the entire rule set. If this were defined as an OR situation, the result would have been TRUE, because at least one of the rules evaluates to TRUE.

Using Rule Sets



After you create a rule set, Database Vault makes it available for selection for realms, factors, command rules, and secure application roles. You can also use predefined rule sets. For example, use rule sets to:

- Define the conditions under which a realm authorization is active (as a further restriction to realm authorization)
- Define when to execute a command rule
- Enable a secure application role
- Define when to assign the identity of a factor

Oracle-Defined Rule Sets

- `Allow System Parameters`: Set initialization parameters
- `Can Grant VPD Administration`: `EXECUTE` the `DBMS_RLS` package
- `Allow Sessions`: Limit `SYSDBA` access via the `CONNECT` command rule, by default not populated
- `Can Maintain Accounts/Profiles`: Controls Oracle user management
- `Can Maintain Own Account`: `TRUE` for user with the `DV_ACCTMGR` role or self-maintenance
- `Disabled`: Always `FALSE`, 1=0
- `Enabled`: Always `TRUE`, 1=1

It is recommended that you use, but not modify, the default rule sets.

- `Allow System Parameters`: Controls the ability to set system initialization parameters
- `Can Grant VPD Administration`: Controls the ability to grant the `GRANT EXECUTE` or `REVOKE EXECUTE` privileges on the Oracle Virtual Private Database `DBMS_RLS` package
- `Allow Sessions`: Controls the ability to create a session in the database. This rule set enables you to add rules to control database logins using the `CONNECT` command rule. The `CONNECT` command rule is useful to control or limit `SYSDBA` access to programs that require its use. This rule set is not populated.
- `Can Maintain Accounts/Profiles`: Controls the roles that manage user accounts and profiles, through the `CREATE USER`, `DROP USER`, `CREATE PROFILE`, `ALTER PROFILE`, or `DROP PROFILE` statements
- `Can Maintain Own Account`: Allows the accounts with the `DV_ACCTMGR` role to manage user accounts and profiles with the `ALTER USER` statement. It also allows individual accounts to change their own password using the `ALTER USER` statement.

Oracle-Defined Rule Sets (continued)

Two other delivered rule sets are very simple yet very useful. They provide a quick way to achieve an always-true or always-false rule set result.

- The always-true rule set is called **Enabled**. Its single rule has the expression "`1 = 1`," which is always true.
- The always-false rule set is called **Disabled.** Its single rule has the expression "`1 = 0`," which is always false.

These rule sets can be used to react quickly in an emergency security situation where something has to be protected or opened immediately.

Creating and Maintaining Rules

- A rule:
  - Can be any expression in a `WHERE` clause
  - Evaluates to `TRUE` or `FALSE`
  - Can be nested within a rule set with `AND` or `OR` logic
- Quick test:
  - SELECT 1 FROM DUAL WHERE <*expression*>
  - 1: `TRUE`
  - No rows returned: `FALSE`

You can create and maintain rules in Cloud Control or via the command line.

To create a rule, you can enter the name of the rule and the expression that should be evaluated. This expression can be any expression that appears in a `WHERE` clause. It evaluates to either `TRUE` or `FALSE`. Rules can be nested within a rule set by including multiple expressions involving `AND` or `OR` logic in the Rule Expression. For example:

```
TO_CHAR(SYSDATE,'HH24') between '22' AND '23' AND SYS_CONTEXT('USERENV',
'SESSION_USER'='SUPERADMIN_USER'
```

Note: An effective way to test an expression is to put it in the `WHERE` clause of the following `SELECT` statement:

```
SELECT 1 FROM DUAL WHERE <expression>
```

If the value `1` is returned, the expression is `TRUE`. If no value is returned, the expression is `FALSE`. If you receive an error, the expression has an error in it.

Rule Set Tasks

- Creating a rule set
- Adding existing rules to one or more rule sets
- Auditing rule sets
- Setting custom event handlers
- Using rule sets with realms

You can perform the rule set tasks listed in the slide in Cloud Control or via the command line.

- When you create a rule set, you specify a name, (optionally) a description, the status of "enabled" or "disabled," and evaluation options.
  - For Evaluation Options, a value of `All True` means that all the rules listed in the rule set must be true in order for the rule set to evaluate to `TRUE`.
  - A value of `Any True` means that as long as any one of the rules evaluates to `TRUE`, the entire rule set is considered to be `TRUE`.
- A rule set consists of one or more rules. In the process of creating or editing a rule set, you can add new or existing rules to it.
- The following pages have more details about the additional bullet points.

Auditing Rule Sets

Issue a command that violates Database Vault security:

```
SQL> CREATE TABLE hr.x2 (a INT);
```

- In a non-unified auditing environment, query the DVSYS.AUDIT_TRAIL$ audit table:

```
SQL> SELECT username, action_command, rule_set_name, rule_name
     FROM   dvsys.audit_trail$;

USERNA ACTION_COMMAND              RULE_SET_NAME      RULE_NAME
------ --------------------------- ------------------ ----------------
BERNST create table hr.x2 (a int)  Work Hours APPS    Weekday Daytime
```

- In a unified auditing environment, query the SYS.DV$ENFORCEMENT_AUDIT table or the unified audit trail:

```
SQL> SELECT userid, dv_action_name, sql_text , dv_rule_set_name
     FROM   sys.dv$enforcement_audit;

USERID DV_ACTION_NAME        SQL_TEXT                      DV_RULE_SET_NAME
------ --------------------- ----------------------------- ----------------
BERNST Realm Violation Audit create table hr.x2 (a int)    Work Hours APPS
```

ORACLE

Database Vault Audit Views in a Non-Unified Audit Environment

In a non-unified auditing environment, Database Vault writes the audit trail to the DVSYS.AUDIT_TRAIL$ table.

The DVSYS.DV$ENFORCEMENT_AUDIT data dictionary view captures DVSYS.AUDIT_TRAIL$ audit trail records for violations of realms, rules, rule sets, factors, and other Oracle Database Vault policy enforced objects.

Database Vault Audit Views in a Unified Audit Environment

If you have enabled unified auditing, this setting does not capture audit records. Instead, you must create and enable audit policies to capture this information. The SYS.DV$ENFORCEMENT_AUDIT data dictionary view provides information about enforcement-related audits from the UNIFIED_AUDIT_TRAIL view.

```
SQL> CREATE AUDIT POLICY audpol1 ACTIONS COMPONENT = dv realm violation on "HR Schema";
SQL> audit policy audpol1;
SQL> select DV_ACTION_NAME,DV_RETURN_CODE, DV_ACTION_OBJECT_NAME, DV_RULE_SET_NAME
     from   unified_audit_trail where DV_RETURN_CODE<>0;
DV_ACTION_NAME          DV_RETURN_CODE  DV_ACTION_OBJECT_NAME  DV_RULE_SET_NAME
----------------------- --------------- ---------------------- ----------------------
Realm Violation Audit             47401  HR Schema              Non Work Hours
```

Specific columns are defined for the rule sets in the audit trail: the RULE_SET_NAME and RULE_NAME columns in DVSYS.DV$ENFORCEMENT_AUDIT or the DV_RULE_SET_NAME column in SYS.DV$ENFORCEMENT_AUDIT. Consider Audit on Failure, for example: If a rule set evaluates to FALSE, an audit record is written, which includes the rule set that failed along with the specific name of the rule that caused the failure.

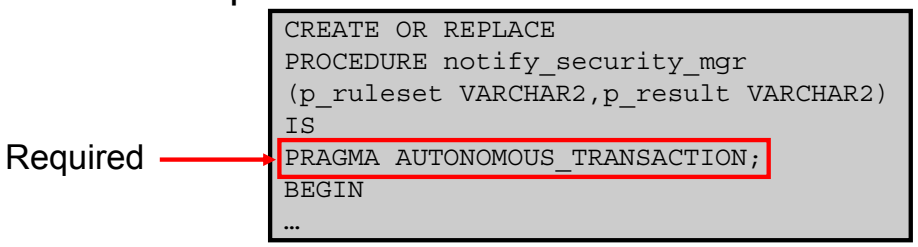
Setting a Custom Event Handler

Custom event-handler options:

- Handler disabled
- Execute on failure
- Execute on success

Custom event-handler logic:

- PL/SQL expression up to 255 characters in mixed-case
- Any package procedure or stand-alone procedure
- Your own expression or Rule Set APIs.



You can set a procedure to be called when a rule set is evaluated. It can be set to either execute on failure or execute every time the rule set is evaluated (on success or failure).

Custom event-handler options:

- **Handler Disabled:** Does not run any custom event method
- **Execute On Failure:** Runs the custom event method when the rule set evaluates to false or one of the associated rules contains an invalid PL/SQL expression
- **Execute On Success:** Runs the custom event method when the rule set evaluates to true

Custom event-handler logic: Enter a PL/SQL expression up to 255 characters in mixed-case capitalization. An expression may include any package procedure or stand-alone procedure. You can create your own expression or use the PL/SQL interfaces, which are part of the Database Vault Rule Set APIs.

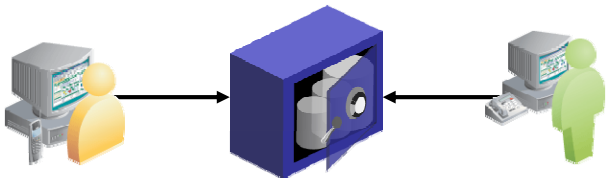
In this example, when the rule set fails, the `APPS.NOTIFY_SECURITY_MGR` procedure is called.

Note: The procedure that is called must be an autonomous procedure.

Using Rule Sets with Realms

A realm's authorization rule set requires the rules to be satisfied for access to the realm-protected objects.

- Use case 1: `HR` access during work week only
- Use case 2:
  - Some users need to query the SH schema tables to perform large and complex queries.
  - The username can vary, because different people do this, but the queries must be run during nonwork hours.
  - These users are granted the `NIGHT_REPORTS` database role.



ORACLE

A rule set can be used to provide an extra check on whether a user can access realm-protected objects. When you add an authorization to a realm, you can specify an authorization rule set. This rule set must be satisfied in order to allow access to the realm-protected objects.

Use case 1: The `HR` user is a realm participant and can access protected objects. However, the addition of a rule means that it must also resolve to a true condition for access to be granted. For example, the "Work Week" rule requires that the current day on the database server be a weekday.

Use case 2: Consider a situation in which several users need to run large and complex queries against the tables in the SH schema. So as not to overload the system during office hours, it is decided that these queries should run during nonwork hours. The users that run these queries have all been granted the `NIGHT_REPORTS` database role, which gives them the access they need to the database objects involved in their queries. You need to create a rule set that ensures that any users with the `NIGHT_REPORTS` role are only allowed to run `SELECT` statements on the SH schema tables and views if the time is not between the hours of 6:00 AM and 5:00 PM.

Rule Set Reports

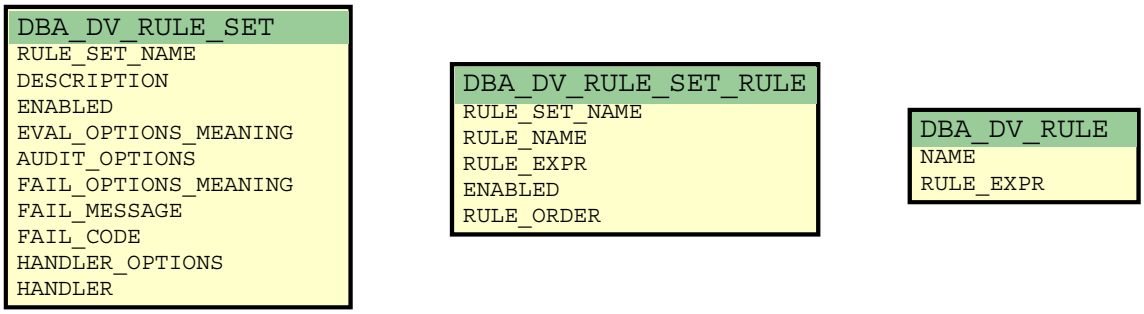
Are the rule sets incomplete or disabled?

- Rule Set Configuration Issues report
- Secure Application Configuration Issues report
- Command Rule Configuration Issues report

Reports that analyze rule sets include:

- The Rule Set Configuration Issues report displays the Database Vault rule set information where no rules are defined or enabled for a rule set.
- The Secure Application Configuration Issues report lists secure application roles that have incomplete or disabled rule sets.
- The Command Rule Configuration Issues report lists rule sets that are incomplete or disabled.

Rule Set Views



Example of a query:

```
SELECT rs.rule_set_name, rs.enabled, rsr.rule_name, rsr.rule_expr,
       rsr.rule_order
FROM   dvsys.dba_dv_rule_set rs, dvsys.dba_dv_rule_set_rule rsr
WHERE  rs.rule_set_name = rsr.rule_set_name;
```

Output:

```
RULE_SET_NAME              ENABLED RULE_NAME       RULE_EXPR
-------------------------- ------- --------------- ------------------------------
Can Maintain Own Account   Y       Is User Manager DVSYS.DBMS_MACUTL.USER_HAS_...
Disabled                   Y       False           1=0
Enabled                    Y       True            1=1
```


The following views in the DVSYS schema contain information about rule sets and rules:

- **DBA_DV_RULE_SET:** One row per rule set
- **DBA_DV_RULE:** One row per rule
- **DBA_DV_RULE_SET_RULE:** Associates a rule to a rule set

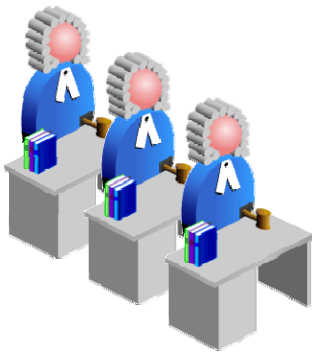
These views are useful for seeing, either programmatically or as an ad hoc query, information about the existing rule sets in the database. All the information viewable in Cloud Control is also available here. DBA_DV_RULE_SET is the parent table and contains information that is specific to the rule set definition, but nothing about the rules contained in it. DBA_DV_RULE contains information about each of the rules. DBA_DV_RULE_SET_RULE is the intersection table; it associates a rule to a rule set. For convenience, this view contains all the information that is in the DBA_DV_RULE view; therefore, there is no need to join it to the DBA_DV_RULE view.

You can run meaningful queries on these views by joining the rule set names, as shown in the example in the slide.

Rule Set API

You can do the following with rule sets:

- Create a rule set.
- Update a rule set.
- Rename a rule set.
- Delete a rule set.



You can do the following with rules:

- Create a rule.
- Update a rule.
- Rename a rule.
- Delete a rule.
- Add a rule to a rule set.
- Delete a rule from a rule set.

ORACLE

The rule set API enables you to do anything that Cloud Control can do with rule sets, including some things that cannot be done. Here are the procedures that you can call to manipulate rule sets and rules:

- **Rule Sets:**
  - `CREATE_RULE_SET (rule_set_name, description, enabled, eval_options, audit_options, fail_options, fail_message, fail_code, handler_options, handler, is_static)`
    The `is_static` parameter determines how often a rule set is evaluated when it is accessed and is not settable in the Cloud Control. If `TRUE`, the rule set is evaluated once during the session; if `FALSE`, it is evaluated every time.
  - `UPDATE_RULE_SET (rule_set_name,description, enabled, eval_options, audit_options, fail_options, fail_message, fail_code, handler_options, handler, is_static)`
    The `is_static` parameter determines how often a rule set is evaluated when it is accessed and is not settable in the Cloud Control. If `TRUE`, the rule set is evaluated once during the session; if `FALSE`, it is evaluated every time.
  - `RENAME_RULE_SET (rule_set_name, new_name)`
  - `DELETE_RULE_SET (rule_set_name)`

Rule Set API (continued)

- **Rules:**
  - `CREATE_RULE (rule_name, rule_expr)`
  - `UPDATE_RULE (rule_name, rule_expr)`
  - `RENAME_RULE( rule_name, new_name)`
  - `DELETE_RULE (rule_name)`
  - `ADD_RULE_TO_RULE_SET (rule_set_name, rule_name)`
  - `DELETE_RULE_FROM_RULE_SET (rule_set_name, rule_name)`

Quiz

After a rule set is created, it is available for selection when you configure the authorization for a realm, factor, command rule, or secure application role.

a. True
b. False

Answer: a

Quiz

When you create a rule to be used in a rule set, the new rule cannot be used in other rule sets.

a. True
b. False

Answer: b

Summary

In this lesson, you should have learned how to:

- Identify the uses of a rule set
- Manage a rule set
- Create rules and add them to a rule set
- Associate a rule set with a realm to limit access
- View the rule set violation reports

Practice

- 6-1: Managing Rule Sets

Create a rule set with two rules and associate it with a realm to ensure access only under certain circumstances. The main steps are:

1. Optionally, log in to SQL*Plus as the `BERNST` user to show the access issue.
2. Use Cloud Control to create a rule set (and review the SQL before executing the commands).
3. Use Cloud Control to create two rules:
   - `Night`: Hours not between 8 AM and 4 PM
   - `Weekend`: Monday – Friday
4. Use the rule set for the HR realm to authorize the `BERNST` DBA.
5. Test the controlled access.

7

Configuring Command Rules

Objectives

After completing this lesson, you should be able to do the following:

- Describe the components of a command rule
- Identify use cases for command rules
- Apply command rules to restrict command execution
- Maintain command rules

Quiz

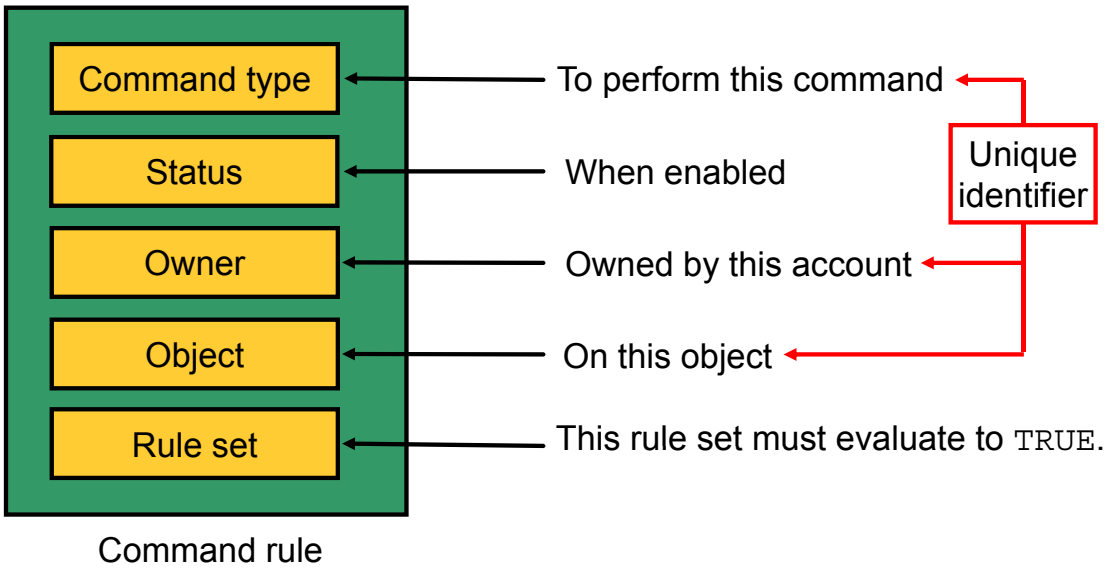
Self-assessment: Select all statements that are true about Database Vault.

a. A command rule defines the rules that must be satisfied before a given command can be performed.
b. These commands include most data definition language (DDL) commands and also `SELECT`, `INSERT`, `UPDATE`, and `DELETE`.
c. The rule set referenced in the command rule is evaluated, which determines whether the statement is allowed to execute or not.
d. If the rule set is `FALSE`, the command is allowed.

Answer: a, b, c

This self-assessment is based on the lesson titled “Database Vault Overview.”

Command Rules



Command rules do not have a name. They are known by the unique combination of a command, an object owner, and an object name. Therefore, you can create multiple `SELECT` command rules—for example, as long as the object owner and object name attributes are different in these command rules.

- **Command:** The command that is being protected against. This includes:
  - Most DDL (`CREATE`, `ALTER`, `DROP`, and `TRUNCATE`)
  - `ALTER SYSTEM`
  - `EXECUTE`
  - `SELECT`, `INSERT`, `UPDATE`, and `DELETE`
- **Status:** Indicates whether the command rule is currently in effect or not. To disable the effects of a command rule, set this to Disabled.
- **Object Owner:** The owner of the object or objects that this rule applies to. This is not applicable in some cases, such as with the `FLASHBACK DATABASE` command.
- **Object Name:** The object within the Object Owner schema that is being protected. It can be a wildcard, which means all objects in the Object Owner schema.
- **Rule Set:** The name of the rule set that protects these objects from this command. If the rule set evaluates to `TRUE` at the time of attempting the command, the command succeeds. Otherwise, a rule set violation error is returned.

Use Case

1 The OE user logs in and alters the OE.ORDERS table:

```
SQL> ALTER TABLE oe.orders ADD (my_code VARCHAR2(10));
Table altered.
```

2 The OE_ORDERS_DBA user notices this new column and wonders why he was not consulted.

3 OE_ORDERS_DBA has a role created and granted only to himself:

```
SQL> CREATE ROLE ORDER_APP_DBA;
SQL> GRANT order_app_dba TO OE_ORDERS_DBA;
```

ORACLE

In this example, there is a user who is normally privileged to alter a table. However, administratively, this user should not do it. There is a user who is specifically assigned these duties. The steps for the example are as follows:

1. The OE user alters the OE.ORDERS table by adding a column. This user is allowed to do this, because she is the owner of the schema. But in this particular organization, duties are assigned such that a different user is in charge of any tables related to orders.
2. The user with database design duties for the ORDERS tables is OE_ORDERS_DBA. He keeps a watch on the table design, making sure that no one else is changing it. He notices that a new column is added and decides that he needs something enforced at the database level.
3. The ORDERS table administrator asks for a role to be created and granted only to himself or herself.

Use Case

4 Create a command rule to prevent altering of OE order-related tables. The user must have the ORDER_APP_DBA role:

```
dvsys.dbms_macutl.user_has_role_varchar('ORDER_APP_DBA')='Y'
```

5 When OE attempts to alter the table, there is a violation:

```
SQL> ALTER TABLE oe.orders ADD (my_code NUMBER);
ORA-47400: Command Rule Violation for alter table on OE.ORDERS
```

6 Even when OE attempts to alter a different orders-related table, there is a violation:

```
SQL> ALTER TABLE oe.order_items ADD (my_code NUMBER);
ORA-47400: Command Rule Violation for alter table on
    OE.ORDER_ITEMS
```

ORACLE

Use Case, Continued

4. The Database Vault Owner creates a command rule that protects the ALTER TABLE command from being executed on any tables in the OE schema that begin with the string ORDER. This includes the ORDERS and ORDER_ITEMS tables. In the process, a rule set called OE_Order_Designer is created. It has a single rule: isOrderAppDBA. This rule's expression ensures that the requesting user has the ORDER_APP_DBA role, which is created in step 3.
5. After OE_ORDERS_DBA removes the new column from the ORDERS table, the OE user again attempts to add the column. But this time, there is a command rule violation for "alter table on OE.ORDERS." The OE user is not able to alter this table.
6. The OE user now attempts to add this new column to the ORDER_ITEMS table. Again, there is a command rule violation. The wildcard caused this command rule to be applied to both the ORDERS and ORDER_ITEMS tables.

Scope of Command Rules

- System-wide scope: for example, `ALTER SYSTEM` and `CONNECT` statements
- Schema specific: `DROP TABLE`
- Object specific: Drop one specific table
- Multitenant environment: In the root container for availability of all pluggable databases

Command rules can be categorized as follows:

- Command rules that have a system-wide scope. With this type, you can only create one command rule for each database instance. Examples are command rules for the `ALTER SYSTEM` and `CONNECT` statements.
- Command rules that are schema-specific. An example is creating a command rule for the `DROP TABLE` statement.
- Command rules that are object-specific. An example is creating a command rule for the `DROP TABLE` statement with a specific table included in the command rule definition.

In a multitenant environment, if you want to create for the `CREATE PLUGGABLE DATABASE`, `ALTER PLUGGABLE DATABASE`, and `DROP PLUGGABLE DATABASE` statements, you can create them in the root so that they can be applied to the entire multitenant environment.

Disallowing `ALTER TABLE` in a Schema

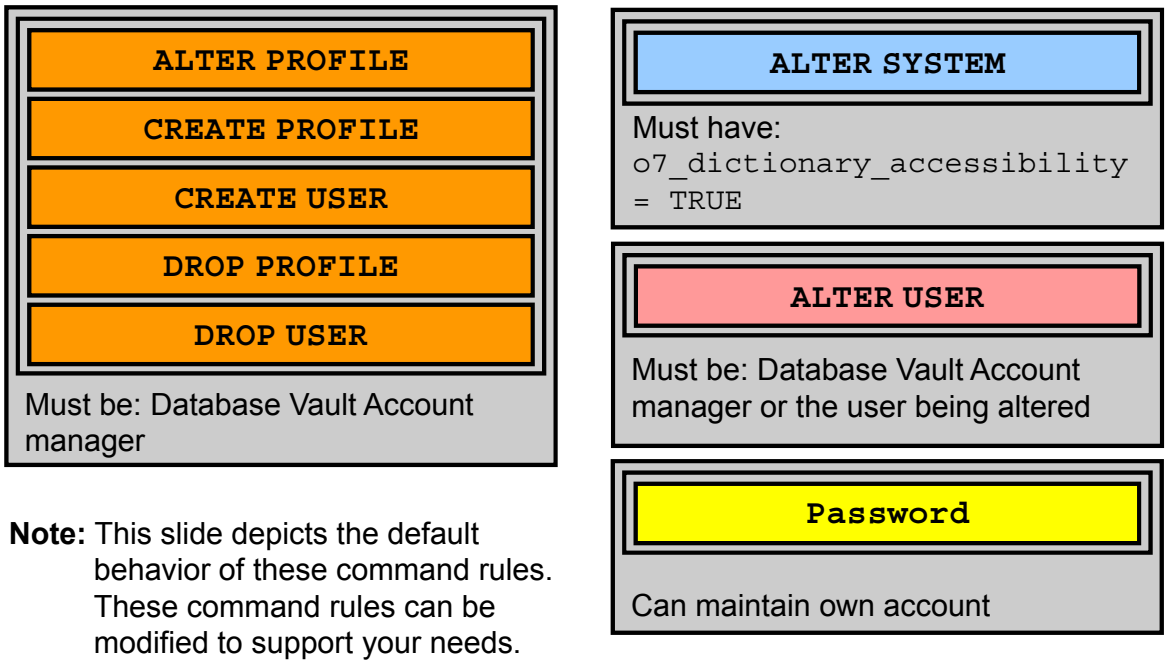
Use case: Prevent users from adding columns to a table

Sample settings:

- Command: `ALTER TABLE`
- Schema: `OE`
- Object Name: `%`
- Rule Set: `Disabled`

A common compliance requirement is to prevent a user from adding columns to a table. The object name is set to the % wildcard, meaning that all tables in `OE` are protected. Rule Set is set to Disabled, indicating that the Disabled rule set must be satisfied for the `ALTER TABLE` statement to proceed. The Disabled rule set, which is delivered with Database Vault, contains a single rule called False, which has the expression `1=0`. But this is never true, so the command rule in the slide always prohibits altering any `OE` tables.

Delivered Command Rules



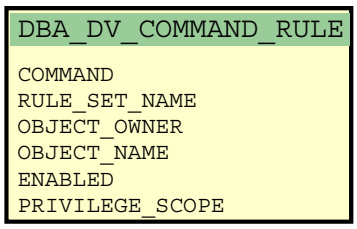
Note: This slide depicts the default behavior of these command rules. These command rules can be modified to support your needs.

The default behavior of the Oracle-provided command rules is as follows:

- **Account management–related command rules:** The Database Vault Account manager is the only user allowed to deal with accounts and profiles:
  - `ALTER PROFILE`
  - `CREATE PROFILE`
  - `CREATE USER`
  - `DROP PROFILE`
  - `DROP USER`
- **`ALTER SYSTEM` command rule:** This command rule is used to restrict the system parameters that can be changed. By default, this command rule uses the Allow System Parameters rule set, which checks to see whether the `o7_dictionary_accessibility` parameter is set to `TRUE`. This rule set can be customized or you can create your own rule set.
- **`ALTER USER` privileges:** This defines the list of users who can do such things as change a user's password, profile, lock status, and default tablespace. A user can alter his or her own account. The Database Account Manager user can alter other users except those with the `DV_OWNER` or `DV_ACCTMGR` roles. Users cannot alter the `DVSYS` user.

Reports and Views

- Command Rule Audit Report
- Command Rule Configuration Issues Report
- Object Privilege Reports
- Sensitive Objects Report
- Rule Set Configuration Issues Report



ORACLE

Reports and a View Related to Command Rules

- **Command Rule Audit Report**: Lists audit records generated by command-rule processing operations
- **Command Rule Configuration Issues Report**: Tracks rule violations, in addition to other configuration issues that the command rule may have
- **Object Privilege Reports**: Lists object privileges that the command rule affects
- **Sensitive Objects Reports**: Lists objects that the command rule affects
- **Rule Set Configuration Issues Report**: Lists rules sets that have no rules defined or enabled, which may affect the command rules that use them
- The `DBA_DV_COMMAND_RULE` view contains information about all the Database Vault command rules in the database.

Command Rule API

The command rule API provides functions that enable you to:

- Create a command rule
- Update a command rule
- Delete a command rule

```
SQL> exec DVSYS.DBMS_MACADM.CREATE_COMMAND_RULE (-
            'DROP TABLE','Am HR User','HR','%','N')
SQL> exec DVSYS.DBMS_MACADM.UPDATE_COMMAND_RULE (-
            'DROP TABLE','Am HR User','HR','%','Y')
SQL> exec DVSYS.DBMS_MACADM.DELETE_COMMAND_RULE (-
            'DROP TABLE','HR','%')
```


The three functions for dealing with command rules are listed as follows. They are in the `DVSYS` schema and are part of the `DBMS_MACADM` package. Therefore, they must be qualified with `DVSYS.DBMS_MACADM`.

To create a command rule, use:

```
CREATE_COMMAND_RULE (command VARCHAR2,
rule_set_name VARCHAR2, object_owner VARCHAR2,
object_name VARCHAR2, enabled VARCHAR2)
```

To update a command rule (identified by the combination of `command`, `object_owner`, and `object_name`), use the following:

```
UPDATE_COMMAND_RULE (command VARCHAR2,
rule_set_name VARCHAR2, object_owner VARCHAR2,
object_name VARCHAR2, enabled VARCHAR2)
```

Note: All parameters are required, even if their values are not to be changed from their current values.

To delete a command rule, use:

```
DELETE_COMMAND_RULE (command VARCHAR2,
object_owner VARCHAR2, object_name VARCHAR2)
```

Note: You must issue a `COMMIT` statement after calling each of these functions for the changes to take effect. You must also make a call to `dvsys.dbms_macadm.SYNC_RULES` for the API changes to be reflected in the DVA:

```
exec  dvsys.dbms_macadm.SYNC_RULES;
```

Quiz

A command rule can allow additional access that is not already allowed for a user.

a. True
b. False

Answer: b

Quiz

A command rule must have a rule set specified for it.

a. True
b. False

Answer: a

Summary

In this lesson, you should have learned how to:

- Describe the components of a command rule
- Identify use cases for command rules
- Apply command rules to restrict command execution
- Maintain command rules

Practices

- 7-1: Using Command Rules
- 7-2: Protecting Application Data

- In Practice 7-1, you restrict the `AHUNOLD` user to creating views only during work hours and days.
- In Practice 7-2, you create command rules so that the `AHUNOLD` user can NOT select his own data, but he can still alter and create objects.

8

Extending Rule Sets

Objectives

After completing this lesson, you should be able to do the following:

- Extend rule sets by using factors
- Create and edit factors
- View existing factors
- Use factor reports
- Describe the concept of identities
- Map identities
- Manage identities

Quiz

Self-Assessment: Select all statements that are true about factors.

a. A factor is a named variable or attribute.
b. A factor has a value or identity.
c. A factor is “stand-alone”—It can NOT be reference in other Database Vault components or combined with other factors.
d. Rule sets can be extended by factors.

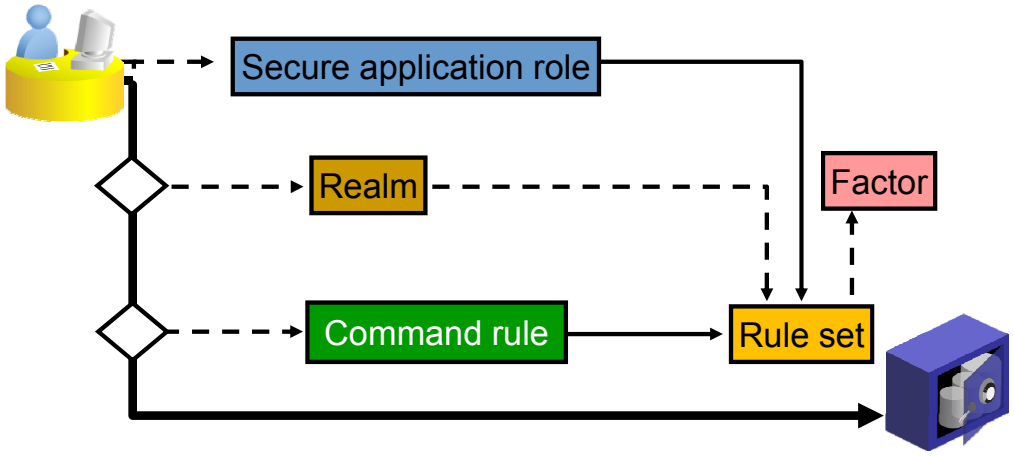
Answer: a, b, d

This self-assessment is based on the lesson titled “Database Vault Overview.”

Quiz

Self-assessment: Factors are referenced by rule sets in their rule expressions.

a. True
b. False



Answer: a

This self-assessment is based on the lesson titled “Database Vault Overview.”

Quiz

Self-Assessment: Select all statements that are true about identities.

a. An identity is a value for a factor.
b. An identity has a trust level.
c. An identity can be resolved from other factors and it can have a label.
d. When the factor identity returned from a factor retrieval method is not defined in the identity, Database Vault automatically assigns the identity a positive trust level

ORACLE

Answer: a, b, c

This self-assessment is based on the lesson titled “Database Vault Overview.”

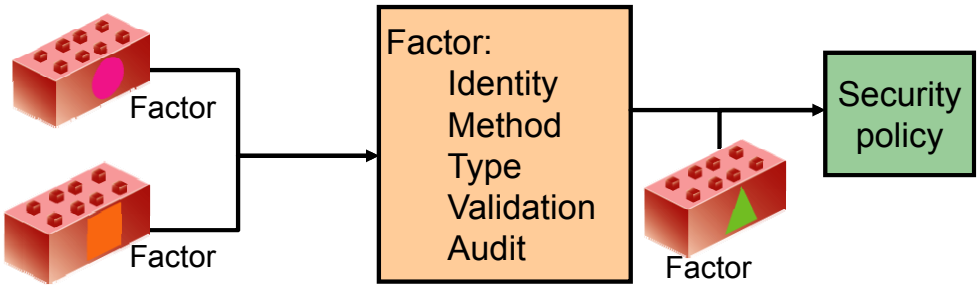
Using Factors

Each factor:

- Is a named piece of data
- Is evaluated at the session level

Factors can be:

- Used as delivered with predefined factors
- Configured, created, and customized
- Used in rule sets as input to conditional statements



Factors are evaluated at the session level. Each session that accesses the factor can have a different identity (value). For example, the `Client_IP` factor evaluates to the IP address of the client machine for each session. You can use factors for activities, such as authorizing database accounts to connect to the database or creating filtering logic to restrict the visibility and manageability of data.

Database Vault provides a selection of factors that enable you to set controls on such components as your site's domain, IP addresses, and databases. You can configure factors by using Cloud Control or the Database Vault APIs. You can create custom factors or customize the existing factors. Custom factors can be created by using your own PL/SQL retrieval methods. The `USERENV` variables of the `SYS_CONTEXT` function have been used for many default factors.

Factors can be used in rule sets as inputs to a conditional statement. A PL/SQL function is created for each factor with the name `F$<factor_name>` in the publicly available `DVF` schema. This function may be used in rules where a PL/SQL expression is required.

Predefined Factors

- Client_IP
  - IP_Address retrieval
- Database_Hostname
  - Host_name retrieval
- Proxy_User
- Authentication_Method
- Database_Domain
- Database_Instance
- Database_IP
- Database_Name
- Domain
- Enterprise_Identity
- Identification_Type
- Lang
- Language
- Machine
- Network_Protocol
- Proxy_Enterprise_Identity
- Session_User

A set of predefined factors is automatically assigned values for each session. These factors can be used to create additional factors, or they can be customized for your site.

- **Client_IP:** Returns the IP address of a client session if the client connects through the listener. Otherwise, the returned value is NULL.
- **Database_Hostname:** Returns the host name of the database as seen in the V$INSTANCE view
- **Proxy_User:** Name of the database user who opened the current session on behalf of SESSION_USER
- **Authentication_Method:** Returns PASSWORD, KERBEROS, SSL, RADIUS, OS, or DCE, depending on the method of authentication. Proxy with certificate, distinguished name (DN), or username without using password returns NONE. You can use Identification_type to distinguish between external and enterprise users when the authentication method is Password, Kerberos, or SSL.
- **Database_Domain:** Domain of the database as specified in the DB_DOMAIN initialization parameter
- **Database_Instance:** Returns the instance identifier as seen in the USERENV context
- **Database_IP:** Returns the IP address of the database server on the basis of the host name of the server

- **Database_Name:** Name of the database as specified in the `DB_NAME` initialization parameter
- **Domain:** Domain is a factor that does not have a predefined identity or method. It can be configured for your site. It is intended to be a named collection of physical factors, or configuration-specific or implementation-specific factors in the run-time environment. A domain can be identified via a number of factors, such as the host name, IP address, and database instance names. Each domain can be uniquely determined by using a combination of factor identifiers that identify the domain. These identifying factors and possibly additional factors can be used to define Maximum Security Label within the domain, restricting data access and commands, depending on the physical factors of the Database Vault session. Examples of domains of interest are Corporate Sensitive, Internal Public, Partners, and Customers.
- **Enterprise_Identity:** The user's enterprise-wide identity. For enterprise users, this returns the Oracle Internet Directory DN. For external users, this returns the external identity (Kerberos principal name, Radius and DCE schema names, OS username, and Certificate DN). For local users and `SYSDBA` and `SYSOPER` logins, this returns `NULL`. The value of the attribute differs by proxy method. For a proxy with DN, this returns the Oracle Internet Directory (OID) DN of the client. For a proxy with certificate, this returns the certificate DN of the client for external users. For global users, this returns the OID DN. For a proxy with username, this returns the OID DN if the client is an enterprise user and `NULL` if the client is a local database user.
- **Identification_Type:** Returns the way the user's schema was created in the database. Specifically, it reflects the `IDENTIFIED` clause in the `CREATE/ALTER USER` syntax. The syntax used during the schema creation is followed by the identification type returned. The `IDENTIFIED BY` password returns `LOCAL`. `IDENTIFIED EXTERNALLY` returns `EXTERNAL`. `IDENTIFIED GLOBALLY` returns `GLOBAL SHARED`. `IDENTIFIED GLOBALLY AS DN` returns `GLOBAL PRIVATE`.
- **Lang:** The International Organization for Standardization (ISO) abbreviation for the language name, a shorter form than the existing `LANGUAGE` parameter
- **Language:** The language and territory currently used by your session, along with the database character set, in this form: `language_territory.characterset`
- **Machine:** Provides the client machine name for the current session
- **Network_Protocol:** Returns the network protocol being used for communication, as specified in the `PROTOCOL=protocol` portion of the connect string
- **Proxy_Enterprise_Identity:** Returns the OID DN when the proxy user is an enterprise user
- **Session_User:** For enterprise users, this returns the schema. This is the database user name by which the current user is authenticated. This value remains the same throughout the duration of the session.

Refer to the "Configuring Factors" chapter of the *Oracle Database Vault Administrator's Guide 12c Release 1 (12.1)* to get the detailed list of the predefined factors.

Factors and Contexts

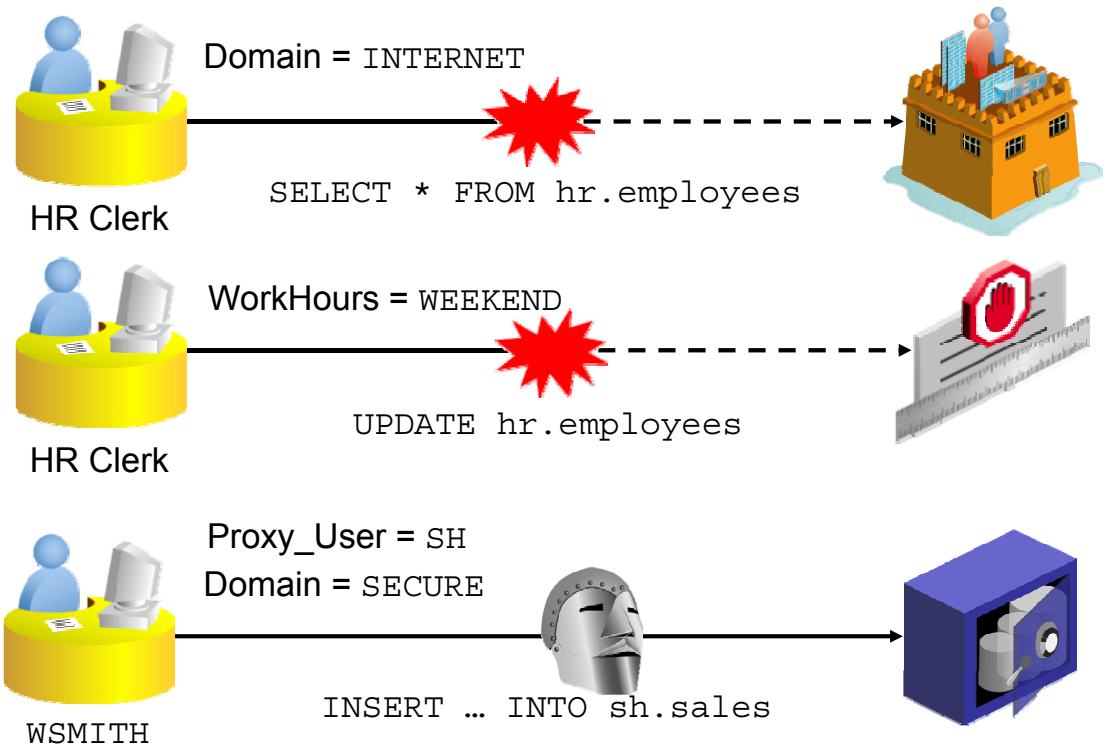
- Differences: Factors can be:
  - Audited
  - Combined logically without programming
  - Assigned trust levels
  - Assigned labels
  - Exposed without coding or creating contexts
- Similarities: Both are:
  - Cached in memory
  - Assigned values at the session level

Database Vault provides several predefined factors whose values are set by calls to the `USERENV` context. Factors use context values because they are memory-based and fast. Factors provide several features that contexts do not, such as the following:

- Factors can be:
  - Created without coding PL/SQL procedures
  - Set and used without changing the application code
  - Audited
  - Logically combined without programming
  - Assigned trust levels
  - Used with Oracle Label Security (OLS) to apply labels to sessions

Factors and contexts are similar because factors use context attributes. When a factor is created, the Database Vault packages create a context attribute for that factor. Factors have the advantages of contexts, without the coding. Factors add security features that are not available with contexts alone.

Factor Scenarios



Factors are useful in building security policies. They are building blocks that reduce the programming necessary to produce a robust security policy.

Suppose that only users connected through the internal network are allowed to access the HR schema. You can place the HR schema objects in a realm. Modify the Domain factor to identify the network source of the session. Create a rule set by using the factor that enforces "Do not allow HR realm access from the internet (Domain=INTERNET)."

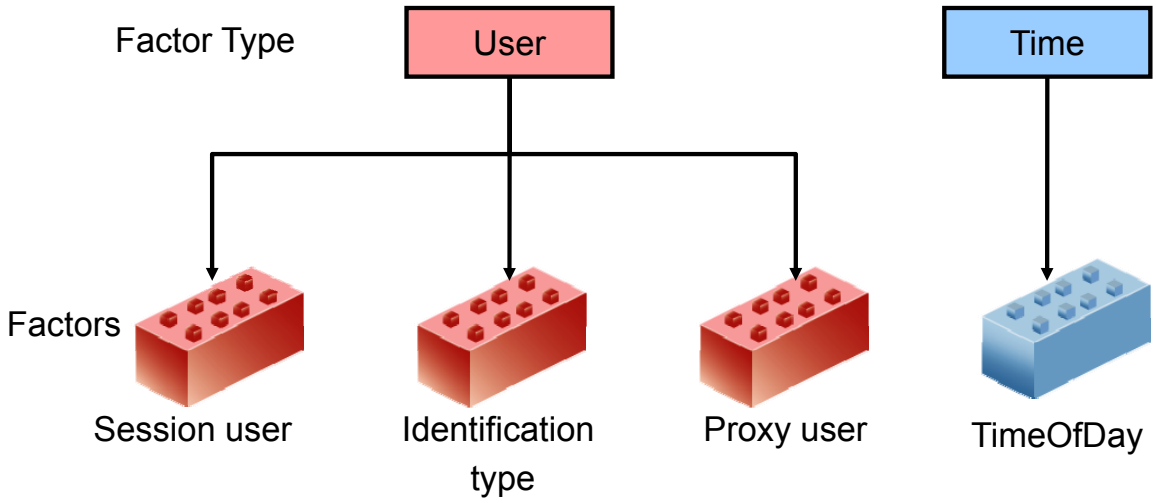
You can define a similar rule set that does not allow any data manipulation language (DML) operations (insert, update, or delete) outside of the standard work hours. (WorkHours = WEEKEND or WorkHours = WEEKNIGHT).

You can ensure that users access the application schema objects only through the application server by using a secure application role. You can define a rule that says "This role can be enabled only when the session originates on the application server machine (Domain = SECURE) and the user is proxied by the application owner (Proxy_User = SH)." The Domain and Proxy User factors contribute to this scenario.

Rules and rule sets are covered in the lesson titled "Defining Rule Sets." Secure application roles are covered in the lesson titled "Configuring Secure Application Roles."

Factor Types

Factor types group factors into categories.



Factor types enable you to group factors into categories. Some predefined factor types include authentication method, host name, and instance. You can define your own factor types, such as application name or certificate information. Factor types are used only for grouping factors and have no effect on the factor or the factor display in Cloud Control. For example, the predefined factors (Session User, OS User, and Proxy User) are grouped in the User factor type.

The `DBMS_MACADM` package provides a function to create a factor type:

```
CREATE_FACTOR_TYPE(name VARCHAR2, description VARCHAR2)
```

The TimeOfDay factor is assigned to the Time factor type.

Factor Identification

- The value of a factor is the identity of that factor.
- Factor identification is the process of assigning a value to the identity.
- Values are assigned by:
  - Method (default)
  - Constant
  - Factors

The value of a factor is the identity of that factor. The identity of a factor is assigned by a method, a constant, or other factors.

On the Edit Factor or Create Factor page, there are three choices for factor identification:

- **By Method:** This is the default option. When this option is chosen, a PL/SQL expression entered in the Retrieval Method field is evaluated to obtain the identity. The method can be any PL/SQL function that returns a `VARCHAR2` data type. Examples can be seen in the predefined factors.
- **By Constant:** When this option is selected, a constant value entered in the Retrieval Method field is assigned to the factor. This option assigns a single value to the factor, and the value is always the same. The constant is a `VARCHAR2` string.
- **By Factors:** When this option is selected, a mapping identity is used to evaluate a set of child factors. The result is assigned to the parent factor. This option is used to easily create a factor by using the results of multiple other factors that are already defined, without having to write a complex PL/SQL function.

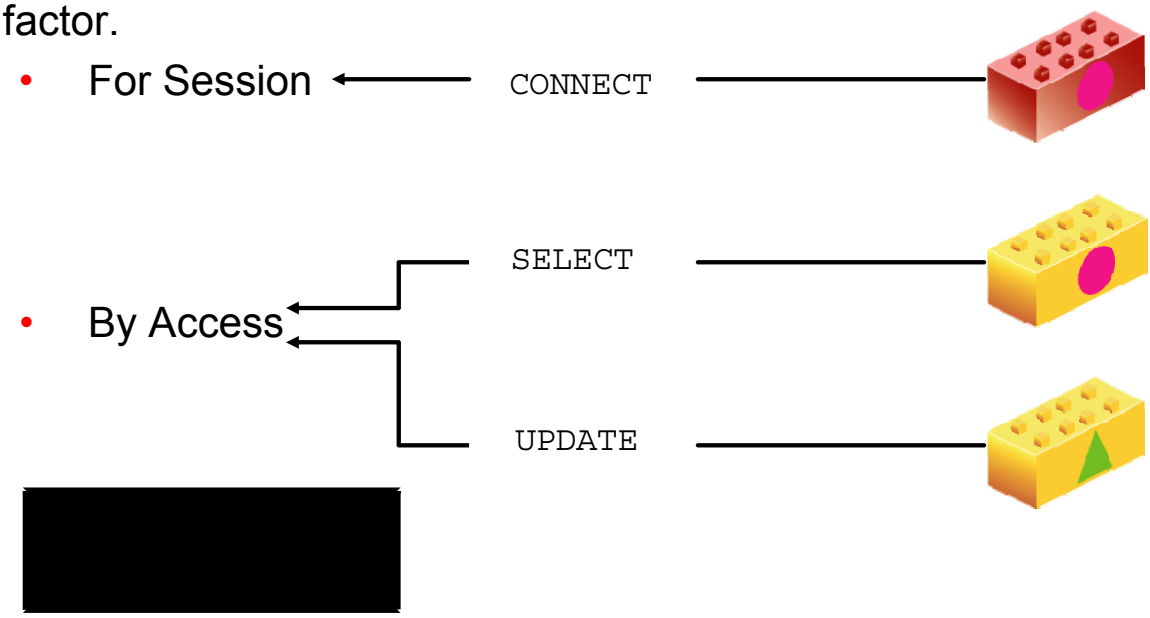
The TimeOfDay factor is identified by factors to allow multiple names for the different times needed for security, such as `Workhours`, `Weeknight`, and `Weekend`.

The value of a factor may also be set by the application by using the `DVSYS.SET_FACTOR` function.

Factor Evaluation

Evaluation determines when an identity is assigned to the factor.

- For Session
- By Access



Factor evaluation determines when the identity is assigned.

- **For Session:** With this option, the factor is evaluated once per session when the session is created. This option incurs less overhead and should be used with factors, such as Client_IP address or Session_User, that remain constant for the entire session.
- **By Access:** With this option, the factor is evaluated when the session is created and each time the factor is accessed. This option forces the factor to be reevaluated. This option has a higher overhead, but ensures that factors that could change during the life of the session are assigned and validated properly. For example, Language-, Module-, and Time-based factors could change.

The TimeOfDay factor should be evaluated by access. This would prevent a session from continuing access or actions at a forbidden time of a day. Factors and rules cannot be avoided just by starting the session during a time when the actions being attempted are allowed or access is allowed.

Retrieval Method

The retrieval method is a PL/SQL expression that evaluates to a VARCHAR2 value.

- A constant: "INTERNET"
- An expression: JLS.MY_FUNCTION(DVF.F$HOST_IP)
  - The function must return VARCHAR2.
  - DVSYS must have EXECUTE granted on the function.

```
UPPER(SYS_CONTEXT('USERENV','CLIENT_IDENTIFIER'))
```


The retrieval method is required if Factor Identification is set to By Constant or By Method. The entry in the Retrieval Method field is a PL/SQL expression. This expression may be a constant VARCHAR2 string or a data type that can be converted to VARCHAR2. This expression may be a packaged or a stand-alone function. The expression cannot be a complete SQL statement.

When you use a function as a retrieval method, it must be completely specified with a schema and a function name, such as JLS.MY_FUNCTION(…). The DVSYS user must have the EXECUTE privilege on the function. As with the JLS.MY_FUNCTION(DVF.F$HOST_IP) function, the value returned by another factor may be accessed by using the DVF-owned function for that factor either as a parameter or in the function body.

DVSYS.SET_FACTOR

Set the factor identity by using the SET_FACTOR function.

- An assignment rule set must be specified.
- A validation method is recommended.

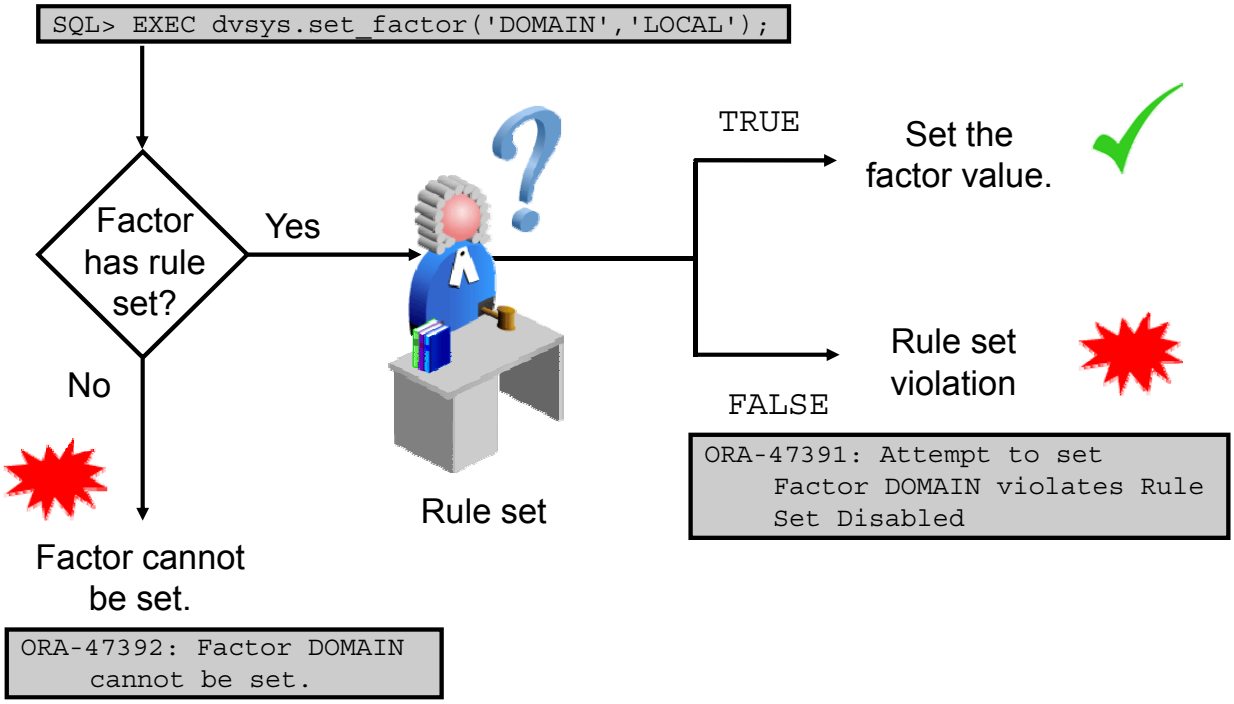
```
BEGIN
    DVSYS.SET_FACTOR('DOMAIN','SECURE');
END;
```


A factor may be set from the session that is using the factor. This function provides a way for applications and application servers to set factor information that may be available only to the application.

A factor cannot be set unless an assignment rule set is specified. The rule set is evaluated when the SET_FACTOR function is called to determine whether the factor is allowed to be set.

Because this function is executable by PUBLIC, it is recommended that a validation method is always specified for any factor that may be set by the SET_FACTOR function.

Assignment Rule Sets for Factors



When you set an assignment rule set for a factor, it allows the factor to be set with a call to the DVSYS.SET_FACTOR procedure. Without an assignment rule set, a factor cannot be set. Therefore, the only way that it has a value is through its evaluation method. If there is no assignment rule set for the factor, you cannot call DVSYS.SET_FACTOR to set its value. If it has an assignment rule set, the rule set is evaluated when DVSYS.SET_FACTOR is called; if it evaluates to TRUE, the factor is set as requested; otherwise, it is not set.

You may want to set assignment rule sets when the evaluation of the factor cannot take into account certain aspects of the connection to the database. For example, an application server may authenticate a user, and then pool connections into a different database account. The application server may need to establish database privileges on behalf of the end user. This can be done by calling DVSYS.SET_FACTOR to force a factor to have a particular identity to provide certain privileges.

Validation Method

The validation method is:

- A PL/SQL expression that returns a `BOOLEAN` value
- An additional check on the identity
- A packaged or stand-alone function
- A condition

```
DVF.F$CLIENT_IP LIKE '192.168.1.%'
```


The validation method is evaluated each time the identity is retrieved, as an additional check on the value. The validation method can be any PL/SQL expression that returns a `boolean` value.

The identity may be set in several ways—through defined method, constant, and other factors, and with the `SET_FACTOR` procedure. A validation method provides another check of the identity at each retrieval. Always use validation methods if the factor is assignable by the `SET_FACTOR` procedure, to verify that invalid entries are not submitted.

You can use a packaged or stand-alone function, or a condition as shown in the slide. There are two signatures available for the function:

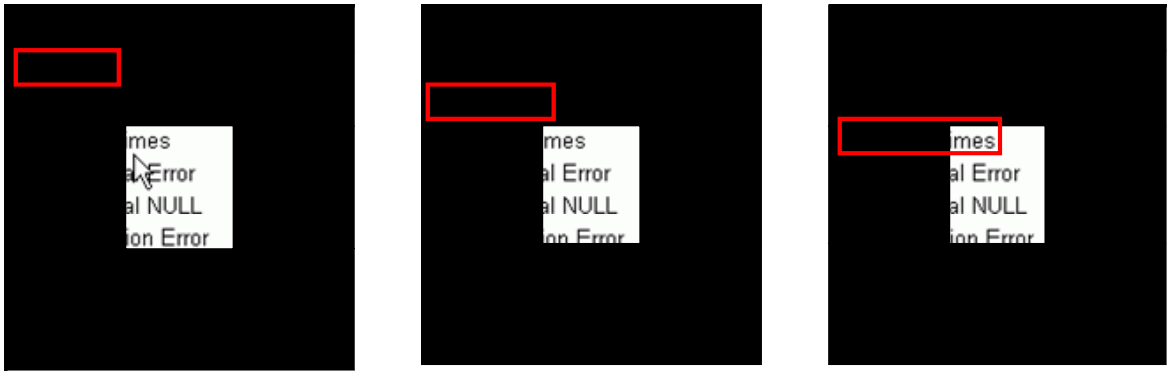
- `FUNCTION is_valid RETURN BOOLEAN`

  This signature is more appropriate for factors that are evaluated by session. The `DVF.F$factor_name` function may be used inside this function.
- `FUNCTION is_valid (p_factor_value VARCHAR2) RETURN BOOLEAN`

Factor Audit Options

Audit options must be set.

- Multiple options may be selected.
- Options are combined to determine behavior.



Audit options control the generation of a custom Database Vault audit record. It is a mandatory attribute, although it can be set to `Never`.
In a non-unified auditing environment, Oracle Database Vault writes the audit trail to the `DVSYS.AUDIT_TRAIL$` table. If you have enabled unified auditing, this setting does not capture audit records. Instead, you can create and enable audit policies to capture this information.

Multiple audit options may be selected at one time. In the slide, the default options are shown for each of the values: `Never, Always`, and `Sometimes`. Each option is converted to a bit mask and added to determine the aggregate behavior. The following audit options are provided:

- **Retrieval Error:** Create an audit record when a factor's identity cannot be resolved and assigned due to an error (such as "No data found" or "Too many rows").
- **Retrieval NULL:** Create an audit record when a factor's identity is resolved to `NULL`.
- **Validation Error:** Create an audit record when the validation method (if provided) returns an error.
- **Validation False:** Create an audit record when the validation method (if provided) returns `FALSE`.
- **Trust Level NULL:** Create an audit record when the factor's resolved identity has an assigned trust level of `NULL`.
- **Trust Level Less Than Zero:** Create an audit record when the factor's resolved identity has an assigned trust level that is less than zero.

Error Options

Error options control the behavior of error messages.

- Show Error Message (default)
- Do Not Show Error Message

The error option is a required attribute that defaults to Show Error Message. Error options control the processing that occurs when the resolution of a factor identity fails. The possible options are:

- **Show Error Message:** Displays an error message to the database session. This option is very useful for debugging.
- **Do Not Show Error Message:** Suppresses the error message. This option does not give any additional information to an unauthorized user, but fails silently.

Quiz

Several audit options are available for auditing factors, but you can select only one audit option for each factor.

a. True
b. False

Answer: b

Purpose of Identities

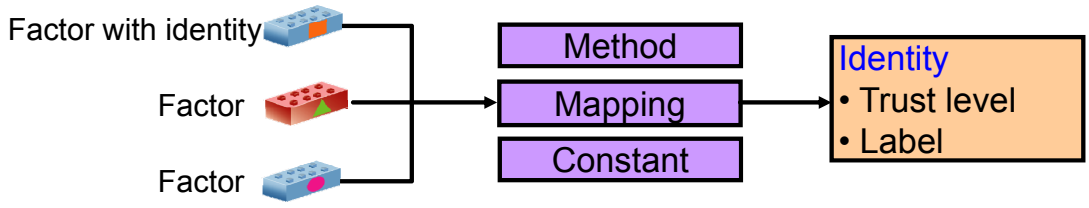
Configure identities to:

- Define the known identities for a factor
- Add a trust level to a factor's identity
- Resolve a factor's identity through child factors (identity mapping)

Example: When the `Client_IP` factor is not an address on the local subnet, the value `INTERNET` is assigned to the `DOMAIN` factor.

- Add an OLS label to a factor's identity

Mapping factors can reduce or eliminate the need to write PL/SQL retrieval methods.



A factor's identity for a given database session is assigned at run time using the Factor Identification and Retrieval Method fields. Configuration of identities is optional and is used to serve the following purposes:

- To define the known identities for a factor
- To add a trust level to a factor's identity
- To add an OLS label to a factor's identity
- To resolve a factor's identity through its child factors (identity mapping)

Identities must be configured to use trust level, OLS label, or child factors.

Example of mapping child factors: The value of an identity can be assigned by evaluating another factor called a child factor. In this case, the value `INTERNET` is assigned to the `DOMAIN` factor when the `Client_IP` factor is not an address on the local subnet. Each identity can have one or more child factors that determine when the identity is set. Mapping factors provides a convenient way to build a modular system. By using factors thus, you can reduce or eliminate the need to write PL/SQL retrieval methods.

Identity Example

The DOMAIN factor is defined with three identities:

- SECURE
- INTRANET
- INTERNET

Adding trust levels:

- SECURE has a trust level of Very Trusted (10).
- INTRANET has a trust level of Trusted (5).
- INTERNET has a trust level of Untrusted (–1).

The DOMAIN factor is defined with three identities: SECURE, INTRANET, and INTERNET. If the assignment rule set is Null or Disabled, these values are the only ones allowed that may be set with the SET_FACTOR function. The conditions that determine which identity is assigned are defined by the settings in the Map Identity region of the Create Identity or Edit Identity pages.

If the retrieval method returns a value that is not one of the defined identities, the identity is set to the returned value and the trust level is set to Untrusted.

Each defined identity must have a trust level assigned, even if it is the Trust Level Not Defined option. The trust level value can be used in rule sets and validation functions. For example, you have a security rule that the HR.EMPLOYEES table cannot be accessed when the user connects from the Internet. You can create a factor named HR_EMP_ACCESS that determines the access to the HR.EMPLOYEES table. Then, attach a validation function and an assignment rule set that prevents the HR_EMP_ACCESS factor from being set if DOMAIN has a trust level of Untrusted or Trust Level Not Defined.

Trust Levels

The trust level is a mandatory attribute and can have one of the following values:

- Very Trusted: Assigns a trust level value of 10
- Trusted: Assigns a trust level value of 5
- Somewhat Trusted: Assigns a trust level value of 1
- Untrusted: Assigns a trust level value of –1
- Trust Level Not Defined: Assigns a trust level value of NULL (default)

The trust level represents the level of confidence in the factor/value pair. It is a mandatory attribute.

A trust value of 1 signifies some trust. A higher value indicates a higher level of trust. A negative value indicates distrust. When the factor identity returned from a factor retrieval method is not defined in the identity table, the identity is automatically assigned a negative trust level.

To determine the trust level of a factor identity at run time, you can use the following functions in the `DVSYS` schema that are publicly available:

- `FUNCTION get_trust_level(p_factor IN VARCHAR2) RETURN NUMBER`
- `FUNCTION get_trust_level_for_identity(p_factor IN VARCHAR2, p_identity IN VARCHAR2) RETURN NUMBER`

For example:

```
SQL> SELECT dvf.f$domain, get_trust_level('DOMAIN')
     FROM dual;
F$DOMAIN    GET_TRUST_LEVEL('DOMAIN')
----------- -------------------------
INTERNET                           -1
```

```
SQL> SELECT get_trust_level_for_identity('DOMAIN','SECURE')
FROM dual;
GET_TRUST_LEVEL_FOR_IDENTITY('DOMAIN','SECURE')
-----------------------------------------------
                                              10
```

In the preceding example, the `INTERNET` identity for the `DOMAIN` factor is untrusted (value equals –1) and the identity for the `SECURE` domain is 10, which implies a greater trust.

Trust-level information can be used in rule sets, validation methods, or directly from the application. The `GET_TRUST_LEVEL` and `GET_TRUST_LEVEL_FOR_IDENTITY` functions are executable by any user. For example, the application can check the trust level before allowing access to certain tables, or a validation method can check that a trust level must be above a certain level before it allows the application to set a factor.

Map Conditions

If there are multiple identities for the same factor:

- Conditions should not overlap
- Identities are evaluated in an ASCII sort order of name

When you have multiple identities for the same factor, the first identity that is evaluated that returns `TRUE` for the map conditions is set. The identities are evaluated in the order of the identity name, in an ASCII sort. If the conditions overlap, the evaluation order can lead to unexpected results.

For example, if the `DOMAIN` factor is assigned the `INTRANET` identity for sessions originating on any machine on the `139.185.35` subnet, except sessions originating on host `139.185.35.114`, `DOMAIN` is assigned `SECURE`.

If `INTRANET` is defined as `CLIENT_IP LIKE 139.185.35.%` and `SECURE` as `CLIENT_IP = 139.185.35.114`, even sessions originating on the secure server would get the `DOMAIN` identity of `INTRANET`, instead of the expected `SECURE`, because the `INTRANET` identity is evaluated first.

Factor and Identities Views

In the DVSYS schema:

- DBA_DV_FACTOR
- DBA_DV_FACTOR_LINK
- DBA_DV_FACTOR_TYPE
- DBA_DV_IDENTITY
- DBA_DV_IDENTITY_MAP

The following views are in the DVSYS schema:

- The DBA_DV_FACTOR view lists the existing factors in the current database instance.
- The DBA_DV_FACTOR_LINK view shows the relationships of each factor whose identity is determined by the association of child factors.
- The DBA_DV_FACTOR_TYPE view lists the names and descriptions of factor types used in the system.
- The DBA_DV_IDENTITY view lists the identities for each factor.
- The DBA_DV_IDENTITY_MAP view lists the mappings for each factor identity.

Reports Related to Factors and Their Identities

Predefined reports:

- Factor Audit Report
- Factor Configuration Issues Report
- Factor Without Identities Report
- Identity Configuration Issues Report
- Rule Set Configuration Issues Report

- **Factor Audit Report**: Audits factors (for example, to find factors that failed to be evaluated)
- **Factor Configuration Issues Report**: Lists configuration issues, such as disabled or incomplete rule sets, or audits issues that may affect the factor
- **Factor Without Identities Report**: Lists factors that have had no identities assigned yet
- **Identity Configuration Issues Report**: Lists factors that have invalid label identities or no map for the identity
- **Rule Set Configuration Issues Report**: Lists rule sets that have no rules defined or enabled, which may affect the factors that use them

Maintaining Factors and Identities

- Using EM Cloud Control or the `DBMS_MACADM` package to maintain factors and identities requires the `DV_OWNER` role.
- Maintain factors, factor links, and factor types.
- Integrate factors with Oracle Label Security (OLS).
- `PUBLIC` is granted the `EXECUTE` privilege on procedures to:
  - Set the factor
  - Get factor attributes
- Maintain identities and identity maps.

- You must have the `DV_OWNER` role to maintain factors and identities
- The `DBMS_MACADM` package in the `DVSYS` schema provides functions and procedures for the maintenance tasks.
- Maintaining factors includes:
  - Creating factors, factor links, and factor types
  - Modifying factors and factor types
  - Deleting factors, factor links, and factor types
- Publicly executable functions for factors include:
  - `SET_FACTOR (p_factor, p_value)`
  - `GET_FACTOR (p_factor)`
  - `GET_TRUST_LEVEL (p_factor)`
  - `GET_TRUST_LEVEL_FOR_IDENTITY (p_factor, p_identity)`
- Maintaining identities includes:
  - Creating identities and identity maps
  - Modifying identities
  - Deleting identities and identity maps

Quiz

Which of the following would you use to assign the identity of a factor? (Choose all that apply.)

a. Constant
b. Method
c. Rule Sets
d. Factors

Answer: a, b, d

Quiz

Each identity must have a trust level assigned.

a. True
b. False

Answer: a

Quiz

An identity cannot be resolved by other factors.

a. True
b. False

Answer: b

Summary

In this lesson, you should have learned how to:

- Extend rule sets by using factors
- Create and edit factors
- View existing factors
- Use factor reports
- Describe the concept of identities
- Map identities
- Manage identities

Practices

- 8-1: Restricting Access by Using the `CLIENT_IP` and `DOMAIN` Factors
- 8-2: Creating a Factor to Determine Job Role
- 8-3: Using Assignment Rule Sets with Factors
- 8-4: Using Rule Sets to Restrict Connection Sources
- 8-5: Using a Factor to Identify a User
- 8-6: Creating Time-Based Factors

All practices are about working with factors and identities.

9

Configuring Secure Application Roles

Objectives

After completing this lesson, you should be able to do the following:

- Create secure application roles
- Edit secure application roles
- Enforce application security by enabling secure application roles

Quiz

Self-assessment: Select all statements that are true about Secure Application Roles in Database Vault?

a. Secure application roles do not use rule sets to define their behavior.
b. Secure application roles are evaluated when a role is requested to be set.
c. Only when the referenced rule set evaluates to `TRUE` is a requested role assigned.

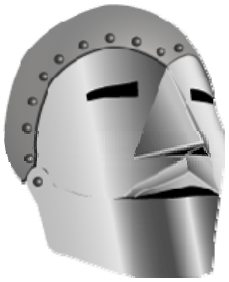
Answer: b, c

This self-assessment is based on the lesson titled “Database Vault Overview.”

Secure Application Roles

Customized application versus Database Vault:

- A secure application role is a database role that can be enabled by using only a specific PL/SQL procedure.
- Database Vault Administrator can create secure application roles that are enabled based on the outcome of a Database Vault rule set.
- Applications can call the Database Vault API to set these roles.
- The role holds the necessary privileges.



A secure application role is a database role that can be enabled by using only a specific, declared PL/SQL procedure. This procedure is usually written by an application developer. In Database Vault, the procedure is implemented by the Database Vault packages.

Database Vault changes the implementation of secure application roles to ease their administration, development, and use. You can create secure application roles that are enabled based on the outcome of a rule set. For example, the application can set the role if the associated rule set evaluates to `TRUE`. If the associated rule set evaluates to `FALSE`, do not allow the role to be set.

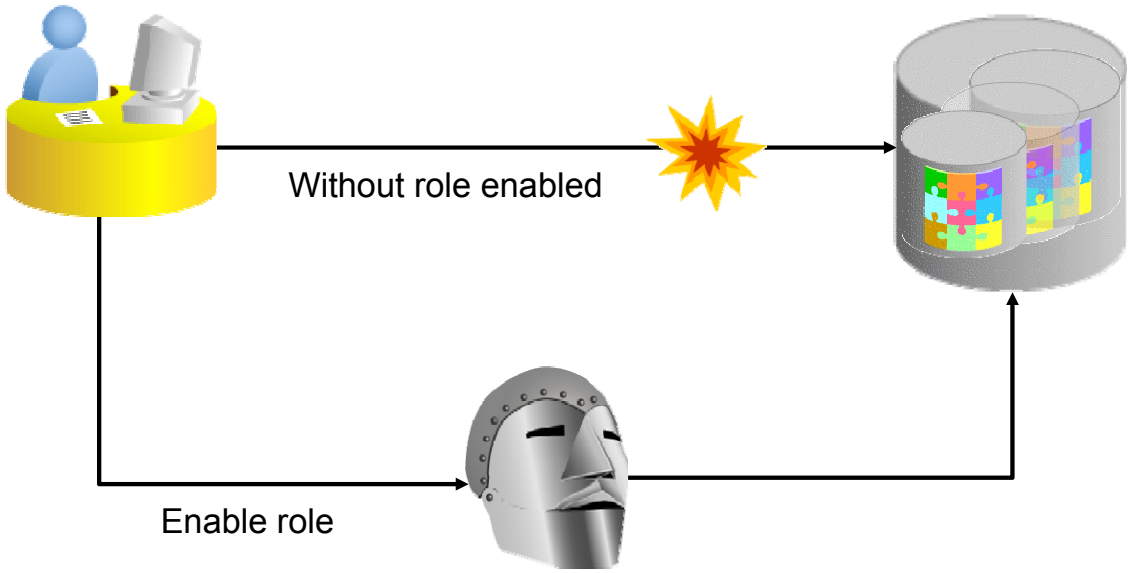
The secure application role is designed to be enabled from an application. In Database Vault, any application or user can execute the `SET_ROLE` function from the API. Database Vault controls the ability to enable the role with an associated rule set.

A secure application role can be assigned system and object privileges just as any other role. In Database Vault, the role is created with separation of duty in mind. System privileges must be assigned by a user with the `DBA` role and object privileges are assigned for realm objects by a user with the `DV_REALM_OWNER` role for the specific realm.

Note: Secure application roles in Database Vault are different than the secure application roles that are written and maintained by developers at the application level (using dbms_session.set_role).

Secure Application Role

Conditional granting of privileges via the `SET_ROLE` function

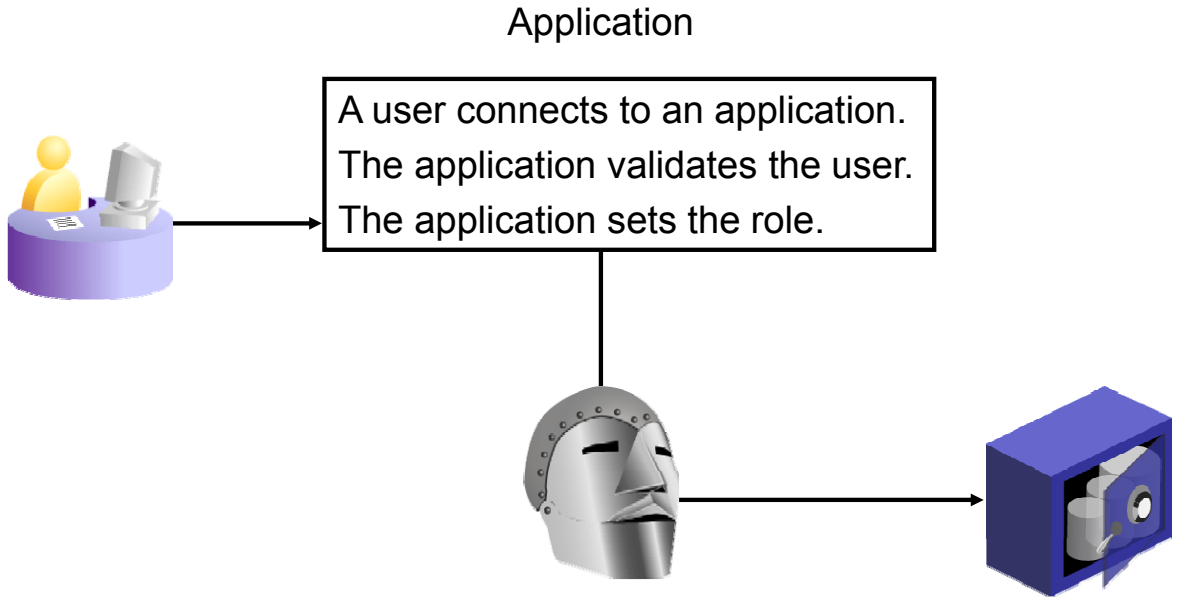


Secure application roles provide a way to grant privileges to users only when certain conditions are met. These conditions can be as simple as when connected through a named application or as complex as privileges that are dependent on the user's job role and the client's IP address.

The user or the application can call the `SET_ROLE` function. If the rule set returns `TRUE`, the role is enabled for the user. After the role is enabled, it remains enabled for the duration of the session.

Do not use factors that are evaluated By Access for secure application roles. If the rule set is `TRUE` when the `SET_ROLE` function is called, the role is enabled. The role remains enabled even if the factor changes so that the rule set is `FALSE`.

Using a Secure Application Role



The secure application role is a secure method for allowing the user to be authorized only when the role is enabled. Before the secure application role was introduced in Oracle9*i* Database, a role could be protected by a password. This role could be enabled from an application with a password, but it required that the password be embedded in the application code. With Oracle9*i* Database, a secure application role could be enabled by using a secure PL/SQL package. Database Vault enables the security administrator to create a secure application role that is validated by a rule set.

In all the versions, the same general procedure is followed in the application.

Secure Application Role Changes in Database Vault

- Do not write an enabling procedure.
- Do not grant the role to users.
- Do not use `DBMS_SESSION.SET_ROLE`.
- Create the role as Database Vault Administrator.
- Use a rule set to control the enabling of the role.
- Use `DVSYS.DBMS_MACSEC_ROLES.SET_ROLE` to set the role.

ORACLE

As in previous releases, a secure application role is enabled by a procedure in a secure package. With Database Vault, the procedure is provided, so you do not have to write it. Instead of writing a procedure, the security administrator uses a rule set to determine whether an application role should be enabled.

The role is automatically created by Cloud Control and can be seen in the `DBA_APPLICATION_ROLES` view. The package that secures this role is listed in this view as `DVSYS.DBMS_MACSEC_ROLES`.

Instead of granting `EXECUTE` on the `DBMS_SESSION.SET_ROLE` procedure to the application account and calling the procedure from the application, call the publicly executable `DVSYS.DBMS_MACSEC_ROLES.SET_ROLE` procedure to set the role.

Tasks with Secure Application Roles

1. Create a rule set to control the enabling of the role.
2. Create one or more rules that enforce the object of the rule set.
3. Create a role that uses a named rule set.

- You can maintain Secure Application Roles with Cloud Control or Database Vault APIs.

1. Create a rule set that returns `TRUE`.
2. For example, the `Local_subnet` rule has the following rule expression:
   `DVF.F$CLIENT_IP LIKE '139.185.35.%'`
3. Give the role a name. Then assign a rule set to control the role. You can choose only from the already created rule sets.

To create or delete Secure Application Roles use Cloud Control or the Database vault APIs.

Examples

The SET_ROLE function fails on a rule set violation.

```
SQL> exec
    DVSYS.DBMS_MACSEC_ROLES.SET_ROLE('HR_EMP_CLERK')
BEGIN DVSYS.DBMS_MACSEC_ROLES.SET_ROLE('HR_EMP_CLERK');
    END;

*
ERROR at line 1:
ORA-47305: Rule Set Violation on SET ROLE
    (ALLOW_LOCAL_SUBNET_ACCESS)
```


The configuration for this example is as follows:
Connect as the bea_dvacctmgr user with the DV_ACCTMGR role and the HR user who is an owner in a realm that protects the HR schema.

```
SQL> connect HR/<password>
Connected.
SQL> grant select on hr.employees to hr_emp_clerk;
Grant succeeded.
```

Attempt to enable the role from a machine that is not on the local subnet:

```
SQL> connect kpartner/q1_w2_e3@otherdb
Connected.
SQL> select * from hr.employees;
select * from hr.employees
                 *
ERROR at line 1:
ORA-01031: insufficient privileges
```

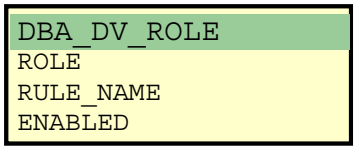
```
SQL> exec DVSYS.DBMS_MACSEC_ROLES.SET_ROLE('HR_EMP_CLERK');
BEGIN DVSYS.DBMS_MACSEC_ROLES.SET_ROLE('HR_EMP_CLERK'); END;
*
ERROR at line 1:
ORA-47305: Rule Set Violation on SET ROLE (ALLOW_LOCAL_SUBNET_ACCESS)
ORA-06512: at "DVSYS.DBMS_MACUTL", line 37
ORA-06512: at "DVSYS.DBMS_MACUTL", line 359
ORA-06512: at "DVSYS.DBMS_MACSEC", line 215
ORA-06512: at "DVSYS.ROLE_IS_ENABLED", line 4
ORA-06512: at "DVSYS.DBMS_MACSEC_ROLES", line 19
ORA-06512: at line 1

SQL> select dvf.F$Client_IP from DUAL;

F$CLIENT_IP
--------------------------------------------
141.144.104.129

SQL> select * from hr.employees
  2  where employee_id = 107;
select * from hr.employees
                 *
ERROR at line 1:
ORA-01031: insufficient privileges
```

Attempt to enable the role from a machine that is on the local subnet:

```
SQL> connect kpartner/q1_w2_e3@orcl
Connected.
SQL> select dvf.f$client_ip from dual;

F$CLIENT_IP
--------------------------------------------------
139.185.35.103

SQL> exec DVSYS.DBMS_MACSEC_ROLES.SET_ROLE('HR_EMP_CLERK');

PL/SQL procedure successfully completed.
SQL> select first_name, Last_name, email, salary
  2  from hr.employees
  3  where employee_id = 107;

FIRST_NAME LAST_NAME  EMAIL         SALARY
---------- ---------- -------- -----------
Diana      Lorentz    DLORENTZ    4,200.00
```

Reports and Views

- Secure Application Role Audit Report
- Secure Application Configuration Issues Report
- Rule Set Configuration Issues Report
- Powerful Database Accounts and Roles Reports



Reports and a View Related to Secure Application Roles

- **Secure Application Role Audit Report**: Lists audit records generated by the Database Vault secure application role-enabling operation. To generate this type of audit record, enable auditing for the rule set associated with the role.
- **Secure Application Configuration Issues Report:** Lists secure application roles that have nonexistent database roles, or incomplete or disabled rule sets
- **Rule Set Configuration Issues Report:** Lists rule sets that have no rules defined or enabled, which may affect the secure application roles that use them
- **Powerful Database Accounts and Roles Reports**: Provides information about powerful database accounts and roles
- You can use the `DBA_DV_ROLE` data dictionary view to find the Database Vault secure application roles used in privilege management.

Maintaining Secure Application Roles

- The administration API `DVSYS.DBMS_MACADM` package contains:
  - `CREATE_ROLE`
  - `DELETE_ROLE`
  - `RENAME_ROLE`
  - `UPDATE_ROLE`
- The run-time API for applications is the `DVSYS.DBMS_MACSEC_ROLES` package, which contains:
  - `CAN_SET_ROLE('<role>')`: Returns `BOOLEAN`
  - `SET_ROLE('<role>')`

You can maintain Secure Application Roles with Cloud Control or with the `DVSYS.DBMS_MACADM` package:

- **`CREATE_ROLE`:** Creates a Database Vault secure application role
- **`DELETE_ROLE`:** Deletes a Database Vault secure application role
- **`RENAME_ROLE`:** Renames a Database Vault secure application role. The name change takes effect wherever the role is used.
- **`UPDATE_ROLE`:** Updates a Database Vault secure application role

The run-time API for use by applications is provided in a separate package to provide better security. The `DVSYS.DBMS_MACSEC_ROLES` package provides the following functions:

- **`CAN_SET_ROLE('<role>')`:** Checks whether the user invoking the method is authorized to use the specified Database Vault secure application role. Returns a `BOOLEAN` value.
- **`SET_ROLE('<role>')`:** Issues the `SET ROLE` command for a Database Vault secure application role

Quiz

After a secure application role has been enabled for a user, it remains enabled for the duration of the session.

a. True
b. False

Answer: a

Quiz

Secure application roles require you to write a procedure to enable the role.

a. True
b. False

Answer: b

Summary

In this lesson, you should have learned how to:

- Create secure application roles
- Edit secure application roles
- Enforce application security by enabling secure application roles

Practice

- 9-1: Managing Secure Application Roles

- In this practice, you give `SMAVRIS` the ability to select from the `HR.EMPLOYEES` table, but not to directly update the table.
- Part of `SMAVRIS`'s responsibilities includes adjusting the salaries of employees, but you do not want to allow `SMAVRIS` to perform any update on `HR.EMPLOYEES`.
- You create a secure application role that provides `SMAVRIS` with the ability to give employees a raise, but only by using the `HR.GIVE_RAISE` procedure.

10

Auditing with Database Vault Reports

Objectives

After completing this lesson, you should be able to do the following:

- Perform security analysis in Cloud Control
- Set up a Database Vault reporting user and view Database Vault reports
- Check for configuration issues and changes
- Report security vulnerabilities

Required Privileges

You must have the DV_OWNER, DV_ADMIN, DV_MONITOR, or DV_SECANALYST role to log in to Cloud Control.

- A security officer (SEC_REP) can view reports but not change the Database Vault configuration.
- The administrator can change the Database Vault configuration and view reports.

To log in to Cloud Control, a user must be granted the DV_OWNER, the DV_ADMIN, or the DV_SECANALYST role. To create a security officer account that has privileges to view reports, but does not have the rights to change the Database Vault configuration, perform the following:

1. As the Database Vault Account Manager, create the security officer user:

```
CONNECT bea_dvacctmgr/<password>
CREATE USER sec_rep IDENTIFIED BY <password>;
GRANT CONNECT TO sec_rep;
```

2. As the Database Vault owner, grant the DV_SECANALYST role to the security user:

```
CONNECT leo_dvowner/<password>
GRANT DV_SECANALYST TO sec_rep;
```

Using Database Vault Reports

- Evaluation of the security measures
- Security-related information from the database
- Custom Database Vault audit event information
- Results of your unified audit policies

Categories:

- **Database Vault Reports**: Potential configuration issues and violations
  - Configuration Issues Reports
  - Enforcement Audit Reports
  - Database Vault Configuration Changes Audit Reports
- **General Security Reports**: Privileges, roles, profiles, accounts, initialization parameters, audit, and so on

ORACLE

In any system that has security requirements, reports are needed to support the evaluation of the security measures that have been implemented. Database Vault provides a selection of reports that display security-related information from the database. These reports also show custom Database Vault audit event information. If you have unified auditing enabled, then the reports capture the results of your unified audit policies.

The reports are in two categories:

- **Database Vault Reports**: These reports allow you to check configuration issues with realms, command rules, factors, factor identities, rule sets, and secure application roles. These reports also reveal realm violations, auditing results, and so on. They are grouped in Cloud Control, as shown in the slide.
- **General Security Reports:** These reports allow you to check the status of object privileges, database account system privileges, sensitive objects, privilege management, powerful database accounts and roles, initialization parameters, profiles, account passwords, security audits, and other security vulnerability reports.

Security Analysis in Cloud Control

Database Vault home page:

- General security status
- Attempted violations and violators (top five) with links to details
- Unified auditing *or* `AUDIT_TRAIL` parameter: `DB` or `DB, EXTENDED`
- Enterprise Manager Cloud Control alerts
- Your choice of aggregation time: last one hour to last 31 days

Available reports:

- Search functionality
- Immediate viewing or export to spreadsheet

ORACLE

To analyze potential security issues and to demonstrate legal compliance, you can use predefined displays in Cloud Control:

- The Database Vault home page gives a general security status, highlights attempted security violations, and lists alerts that administrators can define in Enterprise Manager Cloud Control. You can change the data view by selecting a different time series: Last 1 hour, Last 24 hours, Last 7 days, or Last 31 days.
- Clicking detail links leads you quickly to the most critical information.
  - The Top 5 Attempted Violations displays a pie chart that depicts the top five attempted violations in percentage for the database.
  - The Top 5 Attempted Violators displays a pie chart that depicts the top five violators in percentage—that is, the users who have performed the most attempted violations, for the database.
  - Before you can view these events, if you have not migrated your database to unified auditing, then you must ensure that the `AUDIT_TRAIL` initialization parameter is set to `DB` or `DB, EXTENDED`. If you have migrated your database to use unified auditing, you do not need to configure any additional settings.
- In Cloud Control the available reports are in the left navigation panel. You can use the Search fields to filter the results of the reports and then click Search to view the report. You can also export the report to a spreadsheet at a specified location or you can view it immediately.

Database Vault Audit Integration with the Unified Audit

Only one view for the Oracle Database Vault audit and the Oracle Database audit: `SYS.UNIFIED_AUDIT_TRAIL`

- No more redundant information or duplication of code
- No more data inconsistency between both audit tables
- No missing information
- Easier audit cleanup operation for DBAs, easier audit analysis for auditors
- Easier to read with Database table format (no support of the operating system or XML file formats)
- Better performance
- `DV_*` columns for audit data that is specific to Oracle Database Vault

In Oracle Database 12*c*, the Oracle Database Vault audit is now integrated with the unified audit. The elimination of duplicated information stored in both audit logs results in a more streamlined mechanism for auditing.

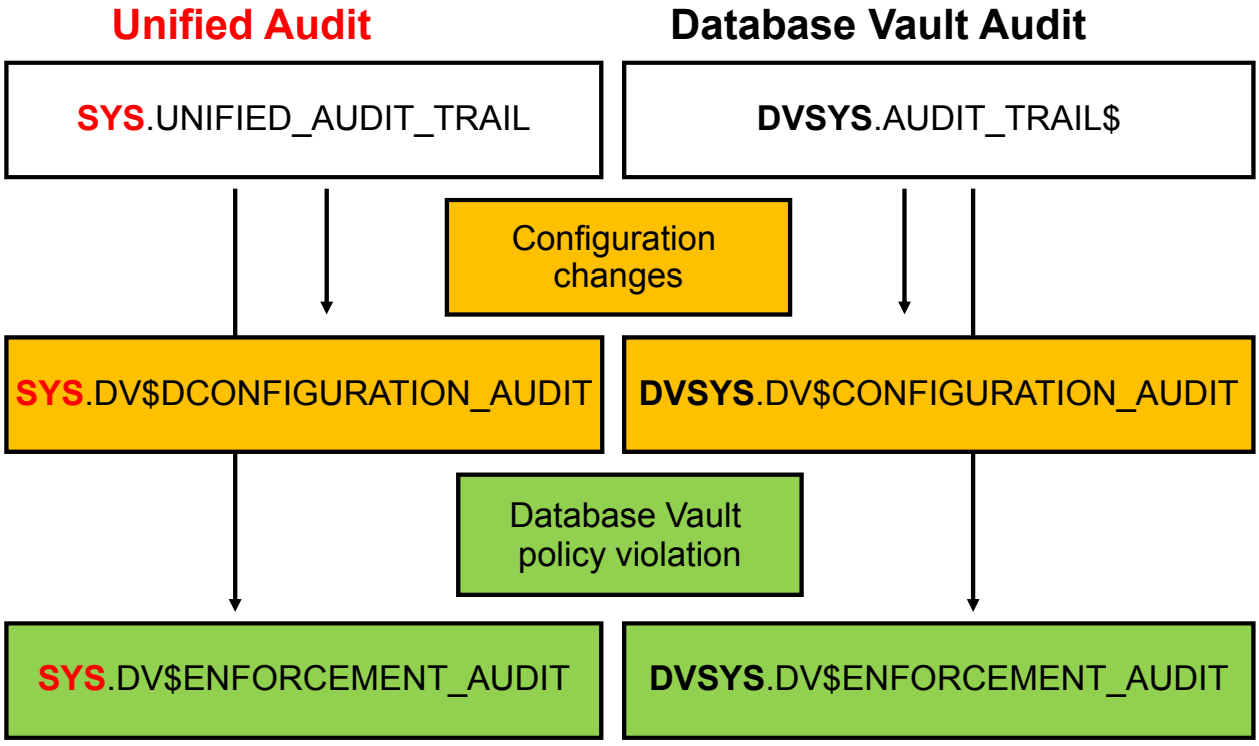
Oracle Database 12*c* has a secure method for cleaning up the Database Vault audit logs without disabling Oracle Database Vault. This secure method makes the audit maintenance much easier for DBAs and the audit information more readable for auditors.

The `SYS.UNIFIED_AUDIT_TRAIL` view contains new DV_* columns:

```
DV_ACTION_CODE, DV_ACTION_NAME, DV_EXTENDED_ACTION_CODE, DV_GRANTEE,
DV_RETURN_CODE, DV_ACTION_OBJECT_NAME, DV_RULE_SET_NAME, DV_COMMENT,
DV_FACTOR_CONTEXT, DV_OBJECT_STATUS
```

The existing implementation of Oracle Database Vault auditing continues to be supported and maintained. You can use the old version by using the `DVSYS.AUDIT_TRAIL$` table.

Database Vault Audit Views



In a unified auditing environment that relies on the `SYS.UNIFIED_AUDIT_TRAIL` view or in a standard Oracle Database Vault auditing that relies on `DVSYS.AUDIT_TRAIL$`, the two following views belong, respectively, to either `SYS` or `DVSYS`:

- The `DV$CONFIGURATION_AUDIT` view captures `Database Vault` audit records that are related to successful and unsuccessful configuration changes made to realms, rules, rule sets, factors, and other Oracle Database Vault policy configuration activities. Unsuccessful changes are not recorded in a unified audit environment.
- The `DV$ENFORCEMENT_AUDIT` data dictionary view provides audit information when users violate Oracle Database Vault policies. In a non-unified auditing environment, violations of Oracle Database Vault components are systematically audited and recorded in `DVSYS.DV$ENFORCEMENT_AUDIT`. In a unified auditing environment, violations are recorded only when audit policies are created against Oracle Database Vault components and enabled.

The column names differ between `SYS.DV$CONFIGURATION_AUDIT` and `DVSYS.DV$CONFIGURATION_AUDIT` and between `SYS.DV$ENFORCEMENT_AUDIT` and `DVSYS.DV$ENFORCEMENT_AUDIT`.

Paying Attention to Configuration Changes

Database Vault Configuration Changes Audit Reports:

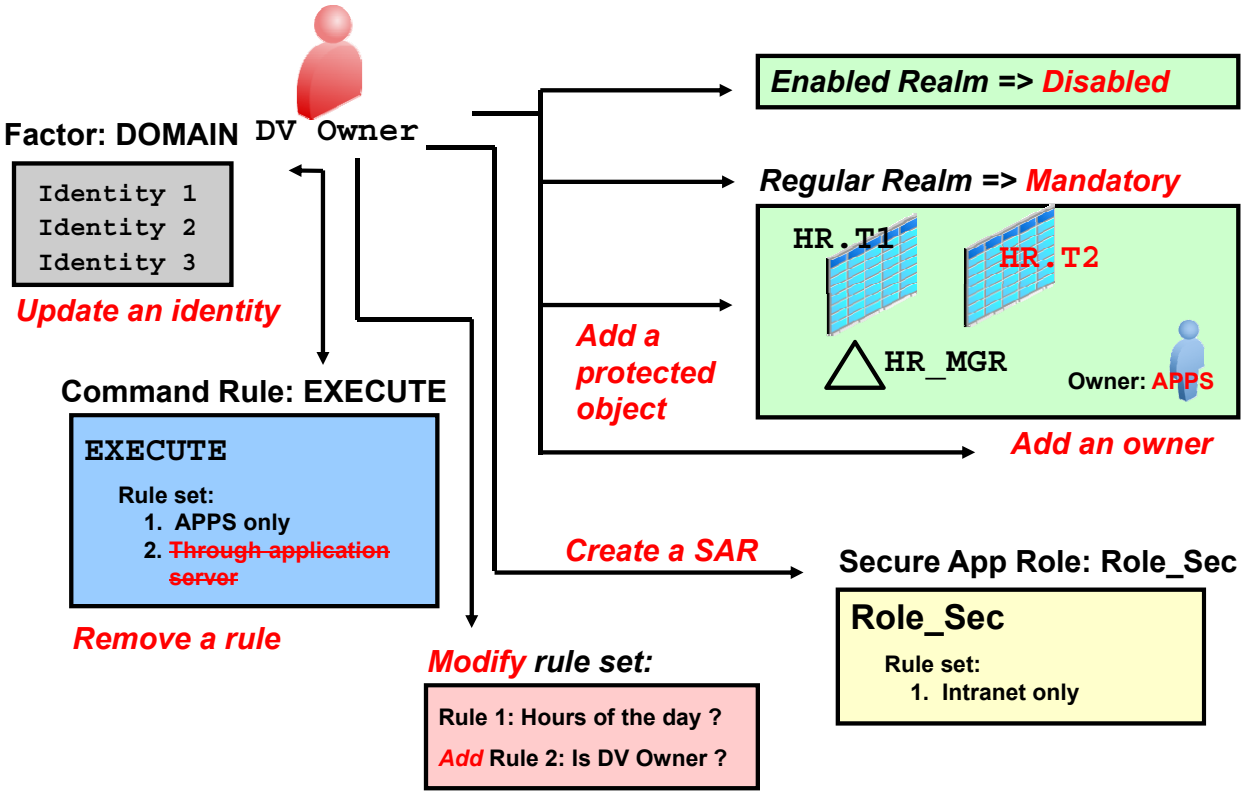
- All Database Vault Configuration Changes
- Realm Configuration Changes
- Command Rule Configuration Changes
- Rule Set Configuration Changes
- Rule Configuration Changes
- Secure Application Role Configuration Changes
- Factor Configuration Changes
- Factor Type Configuration Changes

ORACLE

A potential security breach can be detected by unauthorized configuration changes. It is important for security officers and auditors to pay special attention to Database Vault configuration changes. In addition, there are legal requirements to show that security requirements have been enforced (and no arbitrary changes have taken place).

- The **All Database Vault Configuration Changes** report displays audit records corresponding to changes made to the configuration of Database Vault components.
- The **Realm Configuration Changes** report displays audit records corresponding to changes made to the realm configuration. (*If you repeated some of the practices several times, you may have many records and this report will take a while to display the results.*)
- The **Command Rule Configuration Changes** report displays audit records corresponding to changes made to the command rule configuration.
- The **Rule Set Configuration Changes** report displays audit records corresponding to changes made to the rule set configuration.
- The **Rule Configuration Changes** report displays audit records corresponding to changes made to the rule configuration.
- The **Secure Application Role Configuration Changes** report displays audit records corresponding to changes made to the secure application role configuration.
- The **Factor Configuration Changes** report displays audit records corresponding to changes made to the factor configuration.
- The **Factor Type Configuration Changes** report displays audit records corresponding to changes made to the factor type configuration.

Mandatory Auditing of Configuration Changes



All Oracle Database Vault configuration changes are now audited in Oracle Database 12*c*. Configuration audits are generated when the Oracle Database Vault administrator executes the procedures of the `DVSYS.DBMS_MACADM` package, and you can see the configuration changes in the `SYS.DV$CONFIGURATION_AUDIT` or `UNIFIED_AUDIT_TRAIL` data dictionary view. Examples of configuration audits include `create_realm`, `update_realm`, `add_object_to_realm`, `update_rule_set`, `rename_rule`, `create_command_rule`, `delete_factor`, `authorize_datapump_user`, `create_identity`, and `create_policy_label`.

For information about Oracle Database Vault audited events, refer to the “Auditing Oracle Database Vault” chapter in Oracle *Database Vault Administrator's Guide 12c Release 1 (12.1).*

Audit Views: Configuration Changes

- Non-unified auditing:

```
SQL> SELECT USERNAME, ACTION_NAME, RETURNCODE
     FROM DVSYS.DV$CONFIGURATION_AUDIT ;

USERNAME        ACTION_NAME                      RETURNCODE
--------------- -------------------------------- ----------
SYSTEM          Disable Event Audit                    1031
SEC_ADMIN       Add Realm Object Audit                    0
SEC_ADMIN       Realm Update Audit                        0
SEC_ADMIN       Delete Realm Auth Audit                   0
SEC_ADMIN       Add Realm Auth Audit                      0
```

- Unified auditing:

```
SQL> SELECT USERID, DV_ACTION_NAME, DV_ACTION_OBJECT_NAME
     FROM SYS.DV$CONFIGURATION_AUDIT;

USERID          DV_ACTION_NAME              DV_ACTION_OBJECT_NAME
--------------- --------------------------- ---------------------
SEC_ADMIN       Add Realm Auth Audit         HR_Schema
SEC_ADMIN       CommandRule Creation Audit CREATE VIEW
```


Database Vault Audit Views in a Non-Unified Auditing Environment

The DVSYS.DV$CONFIGURATION_AUDIT data dictionary view captures DVSYS.AUDIT_TRAIL$ audit trail records for successful and unsuccessful configuration changes made to realms, rules, rule sets, factors, and other Oracle Database Vault policy configuration activities.

In the slide, the first row shows that an attempt to disable Oracle Database Vault was unsuccessful because SYSTEM does not have the appropriate DV_OWNER role. The second row shows that an object was successfully added to the list of protected objects in a realm. The third row shows that a realm was successfully updated, such as changing the type from regular to mandatory. The fourth row shows that a participant or owner was successfully removed from a realm, the fifth row shows that a participant or an owner was successfully added to a realm, and the row from the second view shows that a realm object was successfully added.

Database Vault Audit Views in a Unified Audit Environment

The SYS.DV$CONFIGURATION_AUDIT data dictionary view provides rows from the SYS.V$UNIFIED_AUDIT_TRAIL view for successful configuration changes made to realms, rules, rule sets, factors, and other Oracle Database Vault policy configuration activities. They are visible in the SYS.UNIFIED_AUDIT_TRAIL view.

Audit Views: Violated Events

- Non-unified auditing:

```
SQL> select username "US", OBJ_NAME,ACTION_NAME,
            ACTION_COMMAND, RETURNCODE
     from   DVSYS.DV$ENFORCEMENT_AUDIT;

US OBJ_NAME  ACTION_NAME           ACTION_COMMAND             RETU
-- --------- --------------------- -------------------------- ----
HR EMPLOYEES Realm Violation Audit SELECT * FROM HR.EMPLOYEES 1031
```

- Unified auditing:

```
SQL> select userid, OBJ_NAME, DV_ACTION_NAME, SQL_TEXT,
            DV_RETURN_CODE
     from   SYS.DV$ENFORCEMENT_AUDIT ;

US OBJ_NAME  DV_ACTION_NAME        SQL_TEXT                    DV_RETURN_CODE
-- --------- --------------------- --------------------------- --------------
HR EMPLOYEES Realm Violation Audit SELECT * FROM HR.EMPLOYEES            1031
```

Oracle Database Vault Audit Views in a Non-Unified Auditing Environment

The `DV$ENFORCEMENT_AUDIT` data dictionary view captures `DVSYS.AUDIT_TRAIL$` audit trail records for violations of realms, rules, rule sets, factors, and other Oracle Database Vault policy enforced objects.

In the slide, the row shows an unsuccessful HR attempt to select rows from `HR.EMPLOYEES`. The violation occurred against a realm that protected `HR.EMPLOYEES`.

Oracle Database Vault Audit Views in a Unified Auditing Environment

The `SYS.DV$ENFORCEMENT_AUDIT` data dictionary view provides information about enforcement-related audits from the `SYS.V$UNIFIED_AUDIT_TRAIL` view (for example, when users violate Oracle Database Vault policies).

The row in the `DV$ENFORCEMENT_AUDIT` data dictionary view shows that the HR user failed to perform `SELECT` on `HR.EMPLOYEES` and violated the Oracle Database Vault realm that protects the object. The realm is mandatory. Although `HR` is the owner of the object, it is not a participant of the realm.

Oracle Database Vault Audit Policy

In a unified auditing:

1. Create an audit policy on the Database Vault realm object:

```
SQL> CREATE AUDIT POLICY audpol1 ACTIONS
             COMPONENT = dv realm violation on "HR Application";
Audit policy created.
```

2. Enable the audit policy:

```
SQL> AUDIT POLICY audpol1;
Audit succeeded.
```

3. View audit results:

```
SQL> select userid, OBJ_NAME, DV_ACTION_NAME, SQL_TEXT,
             DV_RETURN_CODE
     from    SYS.DV$ENFORCEMENT_AUDIT ;

US OBJ_NAME  DV_ACTION_NAME        SQL_TEXT                    DV_RETURN_CODE
-- --------- --------------------- --------------------------- --------------
HR EMPLOYEES Realm Violation Audit SELECT * FROM HR.EMPLOYEES            1031
```

ORACLE

Oracle Database Vault Audit Views in a Unified Audit Environment

The violations are recorded only if:

1. An audit policy was created to audit the Oracle Database Vault component.

```
SQL> CONNECT system/password
SQL> CREATE AUDIT POLICY audpol1 ACTIONS
             COMPONENT = dv realm violation on "HR Application";
```

2. The audit policy is enabled.

```
SQL> audit policy audpol1;
```

Checking for Configuration Issues

Helpful Database Vault Reports:

- Command Rule Configuration Issues
- Rule Set Configuration Issues
- Realm Authorization Configuration Issues
- Factor Configuration Issues
- Identity Configuration Issues

- The **Command Rule Configuration Issues** report displays command rules for which a rule set is disabled, a rule set is incomplete, or an object owner does not exist.
- The **Rule Set Configuration Issues** report displays rule sets for which no rules are defined or enabled.
- The **Realm Authorization Configuration Issues** report displays information about a rule set for which a realm authorization is disabled, a grantee for a realm authorization does not exist, or an owner for a realm-secured object does not exist.
- The **Factor Configuration Issues** report displays the factors that have any of the following issues:
  - Rule set is disabled or incomplete.
  - Audit options are invalid.
  - No factor retrieval method/constant exists.
  - No subfactors are linked to a factor identity.
  - No subfactors are linked.
  - Oracle Label Security policy does not exist.
- The **Identity Configuration Issues** report displays Database Vault factor identities for which no map exists or for which Label identity for the Oracle Label Security label has been removed and no longer exists.

Reviewing Database Vault Configuration

Recommendations:

- Completely configure all items that are being used.
- Provide a retrieval method or constant for all factors.
- Identify all identities of all factors.
- Configure mapped factors for identity factors.
- Configure validation methods or rule sets for factors set with `SET_FACTOR`.
- Configure rule sets completely.
- Configure rule sets for realms and secure application roles.

The Database Vault configuration reports show where the configuration recommendations are not being followed. Incomplete configuration yields incomplete security.

There are times when the reports may show items as incompletely configured, which may be unavoidable. This leads you to follow other best practices:

- Create a repository rule set. This rule set is not used, except as a container for rules that are in development. These rules can be kept and not be part of any active rule set.
- Remove factors that are not used, including predefined factors. Factors that are not used can take additional unnecessary auditing effort.

Database Vault Enforcement Audit Reports

- Realm Audit
- Command Rule Audit
- Factor Audit
- Label Security Integration Audit
- Core Database Vault Audit Trail
- Secure Application Role Audit

In general, audit reports are required to monitor authorized actions and attempted unauthorized actions. They are also useful for troubleshooting configuration problems.

- The **Realm Audit** report shows audit records generated by the realm protection and realm authorization operations. A rule set performs realm authorization. This report displays the audit of the rule set processing results. This is helpful in troubleshooting rule sets and monitoring failed authorization attempts. Realm violations are also displayed in this report. This report would show when a database account attempts to perform an action on a realm object on which it is not authorized to perform that action. When you configure a realm, you set the audit options for the realm operations.
- The **Command Rule Audit** report shows audit records that are generated by the command rule processing operations. Command rules allow or disallow SQL commands on the basis of rule sets. When you configure a command rule, you can set it to audit the rule set processing results.

- The **Factor Audit report** shows audit records generated as a result of factor evaluation and factor assignment operations. You can audit instances where a factor's identity cannot be resolved and assigned (such as "no data found" or "too many rows"). A factor can have an associated rule set that assigns an identity to the factor at run time. When you configure a factor, you can set it to audit the rule set processing results.
- The **Label Security Integration Audit** report shows audit records related to the label security integration operations: the session initialization operation and the session label assignment operation. You can audit instances where the label security session fails to initialize and where the label security component prevents a session from setting a label that exceeds the maximum session label.
- The **Core Database Vault Audit Trail** report shows audit records generated by the core access security session initialization operation.
- The **Secure Application Role Audit** report shows the audit records generated by the secure application role–enabling operation. You can set secure application roles based on rule sets.

Reviewing Database Vault Audit Reports

A regular review of audit reports provides early detection of:

- Attempts against realm protections
- Violation of separation of duties
- Invalid use of factors by applications
- Configuration problems

Audit records are useless without a regular review. They just take up space. When you review these audit records, you can detect unauthorized activities if the auditing options have been set to record these activities. Normally, auditing is set for “On Failure” to reduce the number of audit records and prevent performance problems, but certain activities should be audited for both success and failure.

- Audit the Database Vault realm for failure and success. This causes auditing of the `DV_OWNER` modifications of any rule sets, realms, and so on (such as adding oneself or disabling the component).
- When you need to temporarily open a privilege, use the security manager (`secadmin`) login ID. Open the privilege, turn on (more) auditing, and then close it when the DBA has finished the task.
- Turning on auditing for success and failure can be helpful in the diagnosis of configuration problems.

  Note: If you have enabled unified auditing, this setting does not capture audit records. Instead, you must create and enable audit policies to capture this information.

General Security Reports

- Cloud Control navigation: Security (main menu) > Reports
- Groups of security-related reports:
  - Database Account Password Reports
  - Privileged Database Accounts and Roles Reports
  - Initialization Parameter and Operating System Directory Permission Reports
  - General Database Privilege and Resource Profile Reports
  - Database Audit and Privilege Reports
  - Object Privilege Reports
  - Sensitive Objects Reports
  - Unified Audit Trail
  - Other Security Reports

These reports enable you to check the status of object privileges, database account system privileges, sensitive objects, privilege management, powerful database accounts and roles, initialization parameters and profiles, account passwords, security audits, and other security vulnerability reports. You are required to use Cloud Control, SQL*Plus, or a SQL-based command-line tool to correct many of the issues shown by the general security reports.

The groups of reports are listed in the slide.

Database Account Password Reports

- Database Account Default Password
  - Change default passwords
- Database Account Status
  - Lock accounts for special schemas
  - Confirm the use of your organizations tablespaces, special passwords and resource profiles (if applicable)

This group of reports focuses on password issues. Most of these issues are prevented by implementing a reasonable password policy. Database Vault provides a password policy by default.

- The **Database Account Default Password Report** lists the database accounts that have default passwords. Default passwords are provided during the Oracle Database installation.
- The **Database Account Status Repor**t provides a quick view of the account status for each account, which helps you identify accounts that must be locked or accounts not using organizationally defined default tablespaces or accounts that use external passwords or accounts that are not using a special password and resource secure profile (if defined).

Privileged Database Accounts and Roles Reports

- Database Accounts With EXEMPT ACCESS POLICY Privilege
- Database Accounts With BECOME USER Privilege
- Database Accounts With ALTER SYSTEM OR ALTER SESSION Privileges
- Database Accounts With CATALOG Roles
- Database Accounts With Privileged Roles
- Database Accounts With ANY System Privilege

- The **Database Accounts With The EXEMPT ACCESS POLICY Privilege** report shows database (but not Database Vault) accounts and roles that have the EXEMPT ACCESS POLICY system privilege granted to them. Accounts that have this privilege can bypass all Virtual Private Database (VPD) policy filters and any Oracle Label Security policies that use Oracle Virtual Private Database indirectly.
- The **Database Accounts With BECOME USER Privilege** report shows all database accounts roles that have the BECOME USER system privilege.
- The **Database Accounts With ALTER SYSTEM OR ALTER SESSION Privileges** report shows all database accounts and roles that have the ALTER SYSTEM or ALTER SESSION privilege. Oracle recommends that you restrict these privileges only to those accounts and roles that truly need them.
- The **Database Accounts With CATALOG Roles** report displays all database accounts and roles that have the %CATALOG% roles granted to them.
- The **Database Accounts With Privileged Roles** report shows the Database Accounts With Password File Authentication. These are the accounts with special system privileges like SYSDBA/SYSOPER and are authenticated using a password file. Note: Operating system authentication takes precedence over password file authentication if you are a member of the OSDBA or OSOPER group.
- The **Database Accounts With ANY System Privilege** report shows all ANY system privileges granted to the specified database account or role.

Initialization Parameter and Operating System Directory Permission Reports

- Security Related Database Parameter Settings In init(sid).ora File
- Permissions On Operating System Directory Objects

- The **Security Related Database Parameter Settings In init(sid).ora File** report displays database parameters that can cause security vulnerabilities, if not set correctly. This report can be used to compare the recommended settings with the current state of the database parameter values.
- The **Permissions On Operating System Directory Objects** report shows all directory objects that exist in the database, whether they are available to PUBLIC, and what their privileges are. Directory objects should exist only for secured operating system (OS) directories, and access to them within the database should be protected.

General Database Privilege and Resource Profile Reports

- Privileges Distribution By Grantee
- Privileges Distribution By Grantee, Owner
- Privileges Distribution By Grantee, Owner, Privilege
- Roles And Accounts That Have A Given Role
- Resource Profiles
- System Resource Limits

The following reports display the count of privileges granted to a database account or role with various groupings. These can provide an insight into accounts and roles that may have powerful privileges.

- The **Privileges Distribution By Grantee** report displays the count of privileges granted to a database account or role. This provides insight into accounts and roles that may have powerful privileges.
- The **Privileges Distribution By Grantee, Owner** report displays a count of privileges based on the grantee and the owner of the object. This provides insight into accounts or roles that may have powerful privileges.
- The **Privileges Distribution By Grantee, Owner, Privilege** report displays a count of privileges based on the privilege, the grantee, and the owner of the object. This provides insight into the accounts or roles that may have powerful privileges.
- The **Roles and Accounts That Have A Given Role** report displays the database accounts and roles to which a role has been granted. This report is provided for dependency analysis.
- The **Resource Profiles** report provides a view of resource profiles, such as CPU_PER_SESSION and IDLE_TIME, that may be allowing unlimited resource consumption.
- The **System Resource Limits** report provides insight into the current system resource usage by the database. This helps determine whether any of these resources are approaching their limits under the existing application load.

Database Audit and Privilege Reports

- Database Report On Core Database Audit Trail
- Database Accounts With AUDIT Privileges

- The **Database Report On Core Database Audit Trail** report returns audit records for the audit policy defined and any auditing records that are generated for audit statements that you have defined. This report only displays audit records that are captured if the database initialization parameter AUDIT_TRAIL has been set to DB.
- The **Database Accounts With AUDIT Privileges** report displays all database accounts and roles that have the AUDIT ANY or AUDIT SYSTEM privilege. This privilege can be used to disable auditing, which could be used to eliminate the audit trail record of an intruder who has compromised the system.

Object Privilege Reports

- Object Access By PUBLIC
- Object Access Not By PUBLIC
- Direct Object Privileges
- Object Dependencies

These reports enable you to monitor the source of user privileges and provide information for the design of roles.

- The **Object Access By PUBLIC** report lists all objects granted to `PUBLIC`. The report details all the object access privileges that the database account has through grants to `PUBLIC`. On the Reports Parameters page, you can filter the results based on the privilege, object owner, or object name. This report is useful for tracking the privileges that a particular user has through `PUBLIC`.
- The **Object Access Not By PUBLIC** report details all the object privileges that a database account has through direct grants to the account or through a role, but not through `PUBLIC`. You specify a single database account on the Report Parameters page, and you can filter the results based on the privilege, object owner, or object name.
- The **Direct Object Privileges** report shows the direct object privileges granted to nonsystem database accounts. It is provided as a tool to help determine where password-protected roles can be implemented.
- The **Object Dependencies** report describes all dependencies in the database between procedures, packages, functions, package bodies, and triggers, including dependencies on views created without any database links. It can help you develop a security policy using the principle of least privilege for existing applications.

Sensitive Objects Reports

- System Privileges by Privilege
- Execute Privileges to Strong SYS Packages
- Access to Sensitive Objects
- Public Execute Privilege to SYS PL/SQL Procedures

- The **System Privileges by Privilege** report displays the database accounts and roles that have the system privilege selected on the Report Parameters page.
- The **Execute Privileges to Strong SYS Packages** report shows the database accounts and roles that have execute privileges on system packages that can be used to access operating system (OS) resources or other powerful system packages. The following system packages are included:

DBMS_ALERT	UTL_FILE	DBMS_CAPTURE_ADM
DBMS_DDL	UTL_HTTP	DBMS_DISTRIBUTED_TRUST_ADMIN
DBMS_FGA	UTL_SMTP	DBMS_RESOURCE_MANAGER
DBMS_JOB	UTL_TCP	DBMS_BACKUP_RESTORE
DBMS_LDAP	DBMS_LOGMNR	DBMS_ORACLE_TRACE_AGENT
DBMS_LOB	DBMS_LOGMNR_D	DBMS_RESOURCE_MANAGER_PRIVS
DBMS_RLS	DBMS_REPAIR	DBMS_OBFUSCATION_TOOLKIT
DBMS_PIPE	DBMS_REPCAT	DBMS_REPCAT_ADMIN
DBMS_RANDOM	DBMS_SESSION	DEBUG_EXTPROC

- The **Access to Sensitive Objects** report shows the database accounts and roles that have object privileges on system tables or views that contain sensitive information. It includes the following system tables and views:

ALL_SOURCE	LINK$	DBMS_BACKUP_RESTORE
ALL_USERS	OBJ$	STATS$SQL_SUMMARY
APPROLE$	OBJAUTH$	STATS$SQLTEXT
AUD$	OBJPRIV$	STREAMS$_PRIVILEGED_USER
AUDIT_TRAIL$	PROFILE$	SYSTEM_PRIVILEGE_MAP
DBA_ROLE_PRIVS	SOURCE$	TABLE_PRIVILEGE_MAP
DBA_ROLES	TRIGGER$	PROXY_ROLE_DATA$
DBA_TAB_PRIVS	USER$	PROXY_ROLE_INFO$
DEFROLE$	FGA_LOG$	ROLE_ROLE_PRIVS
USER_HISTORY$	USER$	USER_TAB_PRIVS

- The **Public Execute Privilege to SYS PL/SQL Procedures** report shows all the database accounts and roles that have execute privileges on packages owned by `SYS`. This can be used to determine which privileges can be revoked from `PUBLIC` or from other accounts and roles. This reduces vulnerabilities as part of an overall least-privileges security policy implementation.

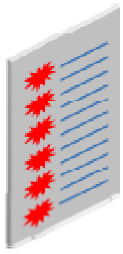
Unified Audit Trail

- Unified Audit Trail Generic
- Audit Configuration Changes
- Failed Login Attempts
- DDL Actions
- DML Actions
- Oracle Label Security Actions
- User Account Actions
- Data Pump Actions
- RMAN Actions
- Privileges Exercised

- The **Unified Audit Trail Generic** report displays the first 2000 audit records from the unified audit trail. If there are more audit records, it prompts you to use “Search” to refine the result list.
- The **Audit Configuration Changes** report displays audit records corresponding to the changes made to the audit configuration.
- The **Failed Login Attempts** report displays audit records corresponding to the failed login attempts.
- The **DDL Audit Actions** report displays audit records corresponding to the DDL actions.
- The **DML Audit Actions** report displays audit records corresponding to the DML actions.
- The **Oracle Label Security Actions** report displays the audit trail for OLS actions.
- The **User Account Actions Report** displays audit records corresponding to the user account actions.
- The **Data Pump** report displays audit records corresponding to the Data Pump actions.
- The **RMAN Actions** report displays audit records corresponding to the RMAN actions.
- The **Privileges Exercised** report displays audit records showing various privileges exercised.

Other Security Vulnerability Reports

- OS Security Vulnerability Privileges
- Java Policy Grants
- Objects Dependent on Dynamic SQL
- Unwrapped PL/SQL Package Bodies
- User Name OR Password Tables
- Tablespace Quotas
- Non-Owner Object Trigger
- Password History Access
- WITH GRANT Privilege Grants



Each of these reports focuses on a possible security vulnerability:

- The **OS Security Vulnerability Privileges** report shows the database accounts and roles that have the required system privileges to export sensitive or otherwise protected information to the operating system.
- The **Java Policy Grants** report shows the Java policy permissions stored in the database. It helps to reveal violations to the least privilege principle. Look for `GRANT`, `READ`, or `WRITE` privileges to `PUBLIC` or other accounts and roles that do not necessarily need the privileges. It is advisable to disable Java loading privileges from `PUBLIC`, if Java is not required in the database.
- The **Objects Dependent on Dynamic SQL** report shows objects that use dynamic SQL. Such objects can be the target for a SQL injection attack. After determining the objects that use dynamic SQL, you need to check the privileges that client applications (for example, a Web application) have over the object. You also need to check the access granted for the object to `PUBLIC` or a wider account base. These actions can limit the scope of an attack.
- The **Unwrapped PL/SQL Package Bodies** report displays PL/SQL package procedures that are not wrapped. Oracle provides a wrap utility that obfuscates code to the point where it cannot be read in the data dictionary. This prevents a hacker from reading source code and determining how to circumvent data protection.

- The **User Name or Password Tables** report helps to identify application tables in the database that store usernames and password strings by displaying tables with `column_names` with the strings `USER%NAME` or `PASSWORD` embedded. These tables should be examined to determine whether the information is encrypted, and if not, the code and applications using them should be modified to protect them from being visible to database sessions.
- The **Tablespace Quotas** report shows database accounts that have unlimited quotas on one or more tablespaces. These tablespaces can become potential targets for DoS attacks.
- The **Non-Owner Object Trigger** report helps to reveal triggers that are owned by a database account that is different from the account that owns the database object on which the trigger acts. If the trigger is not part of a trusted database application, it can steal sensitive data, possibly from the tables protected through Oracle Label Security (OLS) or Virtual Private Database (VPD), and place it into an unprotected table for subsequent viewing or export. This kind of trigger requires that the owner of the trigger have privileges granted directly on the object that the trigger accesses.
- The **Password History Access** report shows database accounts that have access to the `USER_HISTORY$` table that stores hashed passwords that were previously used by each account.
- The **WITH GRANT Privileges** report shows all database accounts that have been granted privileges with the `WITH GRANT` clause.

Quiz

A user with the `DV_SECANALYST` role can view reports and change the Database Vault configuration.

a. True
b. False

Answer: b

Quiz

The Database Vault home page depends on information in an audit trail. Either the `AUDIT_TRAIL` initialization parameter is set to `DB` or `"DB, EXTENDED"` or Unified Auditing must be enabled.

a. True
b. False

Answer: a

Summary

In this lesson, you should have learned how to:

- Perform security analysis in Cloud Control
- Set up a Database Vault reporting user and view Database Vault reports
- Check for configuration issues and changes
- Report security vulnerabilities

Practices

- 10-1: Viewing Configuration Issues Reports
- 10-2: Viewing Enforcement Audit Reports
- 10-3: Viewing Database Vault Configuration Changes
- 10-4: Viewing General Security Reports (*optional*)

11

Implementing Best Practices

Objectives

After completing this lesson, you should be able to do the following:

- Identify your security requirements and determine how to implement separation of duty
- Describe the suggested best practices for implementing Database Vault
- Describe the first steps in securing a database by using Database Vault
- Identify the Database Vault components needed to protect against accidental object loss
- List special considerations for dealing with an application server: connection pooling and limiting connections
- Identify the performance issues to be considered

ORACLE

Identifying Your Security Requirements

- What needs to be protected?
  - Oracle applications
  - Partner applications
  - Custom applications
- Who should be authorized and for what type of access?
  - Data access for application owners and users
  - System access to database structures without access to data by database administrators
  - System access without access to data by security administrators

ORACLE

Before you begin using Database Vault, you must identify your security requirements. This process involves determining the answer to questions, such as:

- What needs to be protected?
  - Oracle applications
  - Partner applications
  - Custom applications
- Who should be authorized and for what type of access?
  - **Data access for application owners and users:** Application owners require access through whatever middle tier processes are in place. Business users require access through the application interfaces.
  - **System access to database structures without access to data by DBAs:** DBAs need access to database structures for performing backups, patching, tuning, and monitoring activities. They do not need access to the application data itself.

- **System access without access to data by security administrators:** Security administrators need access to the system to perform security tasks, such as creating database accounts, granting privileges, running security audit reports, and administering Database Vault configuration. They do not need access to the application data itself.

Separation of Duty Best Practices

- Separation of duty can be customized on a per-company basis.
- Have separate accounts for:
  - Database account management
  - Database security administration
  - Backup management
- Auditors look for implementations, such as:
  - Separate database accounts for different responsibilities
  - Being able to track the actions of each account
- Database Vault–protected auditing prevents tampering and provides a clear picture of compliance status.

Separation of duty can be customized on a per-company basis. It is recommended that you have separate accounts for each of the various duties that exist in your environment, such as:

- One named account for database account management
- One named account for database security administration
- Additional named accounts for backup management

Auditors evaluating compliance status look for implementations, such as:

- Separate database accounts for different responsibilities. The number of people doing the tasks is less important than the separation of these responsibilities.
- Being able to track the actions of each account

Database Vault–protected auditing prevents tampering and the available reports provide a clear picture of the compliance status.

Since cyber threats mostly target privileged users, multiple DBAs can proxy into predefined SOD accounts. For example, you could set up an account for Account Management, but still have multiple DBAs using it. It would just be that each DBA would authenticate as themselves and proxy into the appropriate responsibility.

```
$ sqlplus DBA_DEBRA[acct_management]/DBA_DEBRA_PWD
```

The realms still block `DBA_DEBRA` from having default access to application data and if she should or a cyber threat should attempt to use her account, an audit record would be generated.

Separation of Duty Matrix

Responsibility		Resource Management					
User, Process or Application	Account Creation	SYSDBA	Backup	Tuning	Patching	Monitoring	Security Admin
DAUSTIN	Y	N	N	N	N	N	N
AHUTTON	N	N	N	N	N	N	Y
BEVERETT	N	N	Y	N	N	N	N
DLEE	N	N	N	N	Y	N	N
JCHEN	N	N	N	Y	N	Y	N
SYSTEM	N	N	N	N	Y	N	N
RMAN	N	Y	Y	N	N	N	N
…							

ORACLE

Creating a separation of duty matrix is an excellent way to determine the best way to incorporate separation of duty into your environment. It provides a clear picture of all the elements that require access (whether it is a user, process, or application), the responsibilities involved, and the type of access required.

Application Protection Matrix

Protection Type \ Authorized with Rule Set	SYSADM	PSFTDBA	SYSTEM	DBA
PeopleSoft Realm	OWNER	OWNER	No Access	No Access
Select Command Rule	Not Restricted	Limit PSFTDB Rule Set	No Access	No Access
Connect Command Rule	PeopleSoft Access Rule Set	Not Restricted	Not Restricted	Not Restricted
Drop Tablespace Command Rule	Disabled Rule Set	Disabled Rule Set	Disabled Rule Set	Disabled Rule Set

ORACLE

An application protection matrix is another useful tool to help you to determine and document the types of protection required by a particular application for each of the users interacting with that application and its components. The example in the slide shows a partial picture of how a PeopleSoft application is protected.

Building and Documenting Your Implementation

- Build your security policies by using API scripts.
- Document the application security policies in a separation of duty matrix and an application protection matrix.
- Document processes and procedures for daily use cases (including backup, patching, tuning, and monitoring).
- For each production database account, document responsibilities, whether or not they are locked by default, and when to use `sys` or `system` logins.
- Document emergency or “Break the Glass” scenarios.
- Define reporting rules in a production environment, including what reports, when to run, and whom to send them to.

ORACLE

It is important while implementing Database Vault security to document everything. It is recommended that, as part of your documentation, you complete the following tasks:

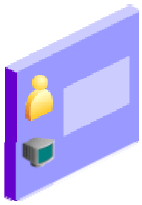
- Build your security policies by using API scripts. These provide not only a very fast and easy method of recreating your security policies, but also becomes part of your documentation.
- Document the application security policies with:
  - A separation of duty matrix
  - An application protection matrix
- Document processes and procedures for daily use cases, such as backups, patching, tuning, and monitoring.

- For each production database account, document the following:
  - The responsibilities of each account
  - Which accounts should be locked by default
  - When to use `sys` or `system` logins
- Document emergency or “Break the Glass” scenarios.
- Define reporting rules in a production environment:
  - Which reports need to be run and who runs them?
  - How often should the reports be run?
  - Who should the reports be sent to?

Trusted People for Trusted Positions

Choose trusted individuals for trusted accounts and roles:

- OS administrator (`root`)
- Oracle software owner
- Member of `DBA` or `OINSTALL` groups
- Users with the `SYSDBA` privilege
- Users with the `SYSOPER` privilege
- Users with the `DV_OWNER` role
- Users with the `DV_ACCTMGR` role



ORACLE

Database Vault restricts access to application data for many privileged users and roles in the database. However, in some cases, Database Vault trusts certain accounts and roles.

The accounts and roles that are listed in the slide have far-reaching privileges and should be granted only to trusted individuals.

The OS administrator (`root` user) can view unencrypted database files, move and delete any file, log in as any user, including the Oracle software owner, and start and stop any process.

The Oracle software owner and members of the `DBA` and `OINSTALL` groups can view unencrypted database files, move and delete database files, disable Database Vault, start and stop database processes, and modify the initialization parameter file.

Users with the `SYSBDA` privilege are required for certain Oracle utilities: Recovery Manager, Data Guard and Data Guard Broker, the Real Application Clusters `svrctl` utility, and the Automatic Storage Management command-line utilities. If these utilities are required in your environment, enable the `SYSDBA` account and monitor the audit trail. `SYSDBA` is audited by default.

Users with the SYSOPER privilege can start up and shut down the database and change initialization parameters, but they have limited access to the data dictionary.

Actions by users with the DV_OWNER or DV_ACCTMGR roles are audited. These roles have limited privileges but are still quite powerful.

Recommended Naming Conventions

- Realms
  - Use the application name as the realm name.
  - Provide a detailed realm description.
- Rule Sets
  - Start the name with a noun.
  - End the name with the relevant realm or command rule name
  - Document business requirements in the description.
- Rules
  - Start the name with a verb.
  - Complete the name with the purpose of the rule.
- Factors
  - Start the name with a noun.
  - Complete the name with a description of the derived value.

Naming conventions are important because they provide consistency and ease of maintenance. Some suggested naming conventions are shown in the slide.

Transition to Production

Perform the following important tasks:

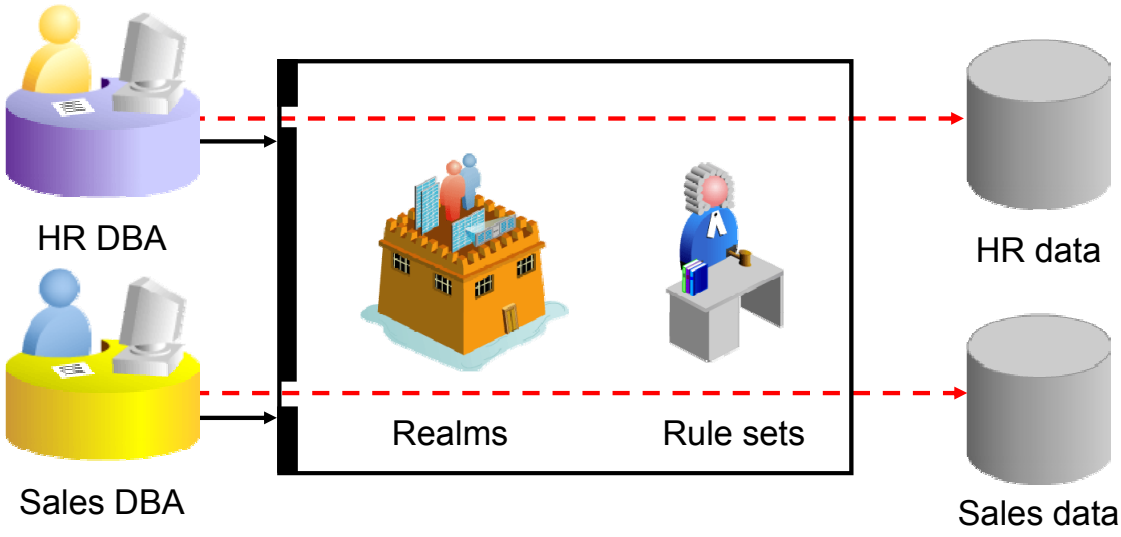
- Run a full test of your application.
- Monitor the performance and tune your rule expressions.
- Apply your Database Vault API scripts to the production environment.
- Hand over appropriate responsibilities to the administrative groups identified during the implementation of separation of duties.
- Back up your Database Vault API scripts and restrict access to them.

As you transition your Database Vault security implementation to a production environment, it is important to perform the following tasks:

- Run a full test of your application to make sure that users have access to what they need, that administrators can perform their duties, and that applications function appropriately.
- Regularly monitor performance and tune your rule expressions if necessary.
- To deploy your Database Vault security solution to production, run the Database Vault API scripts in your production environment.
- Hand over appropriate responsibilities to the production support and security groups that you identified during the implementation of separation of duties. For example, hand over:
  - The Security responsibility to the Database Security Administrator
  - Account Management to the Database Account Manager
  - Resource Management to the DBAs
- Include the Database Vault API scripts in your regular backup scenario and restrict access to them. This ensures successful recovery if you ever need to rebuild.

Implementing Separation of Duties

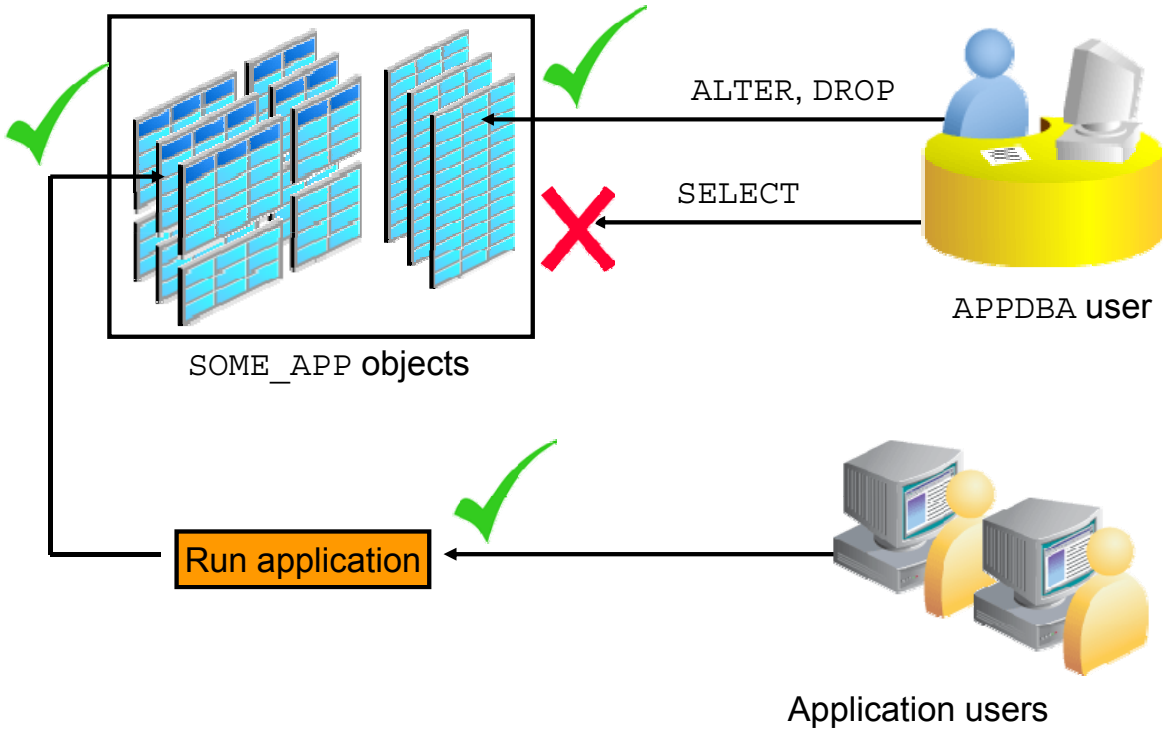
Use realms and rule sets together to implement separation of duties.



As described previously in this course, realms protect your database objects from anyone exercising system privileges. A user must be authorized to a realm to get around that protection. Further, rule sets can be attached to the authorizations defined for a realm. You can define the rule set to further qualify under what conditions the realm's objects can be accessed. This could be the time of the day or the IP address of a client machine.

The first and the simplest line of defense is to place schemas into separate realms. This model fits most environments. Then to the realm, add only those DBAs that require access to manage the database objects. You may optionally add rule sets to the realm authorizations if additional and more granular checks need to be made on the access scenario. For example, the realm may allow the `HR` DBA to drop a table in the `HR` schema, but it may also have a rule set that requires that this be done during the weekend.

Application DBA: Attributes



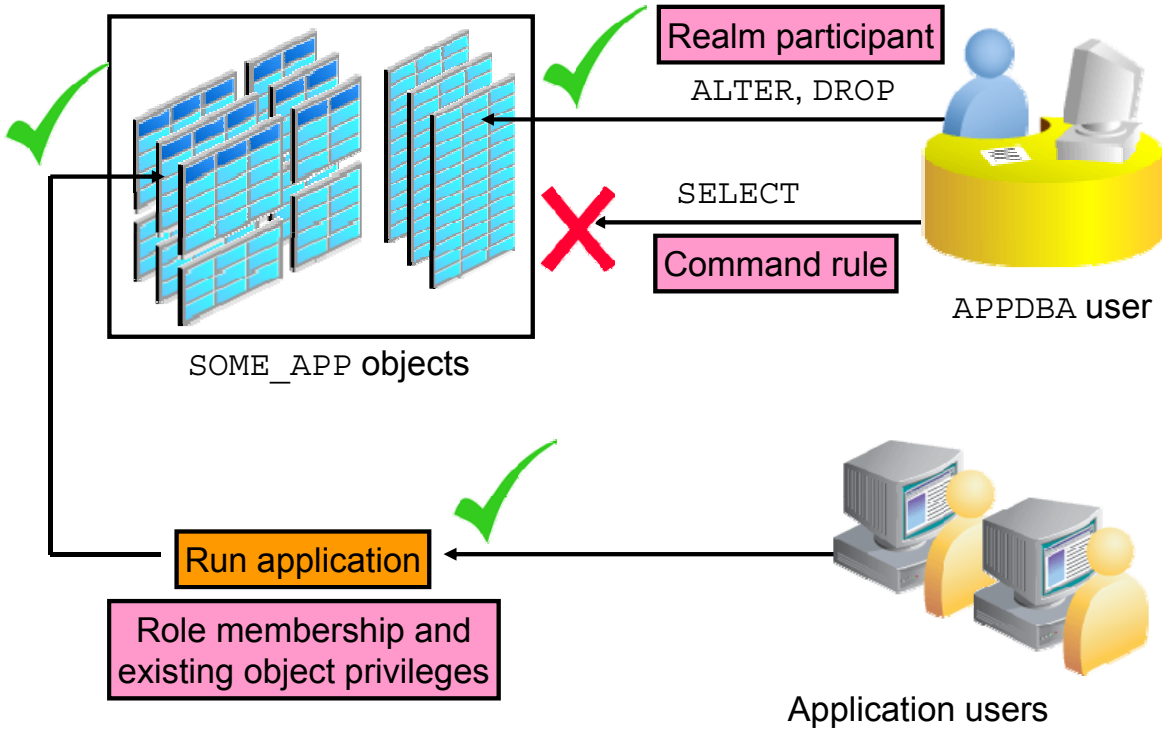
An application DBA is a user that is able to perform the typical administration duties for the database objects related only to a specific application or functional area of the database. An example is a sales DBA, who is responsible for all the sales data. There may be other type of data in the database, such as HR data, but the sales DBA is responsible only for the sales data.

The sales DBA:

- Can view the schema definitions of the protected objects
- Can modify definitions of the protected objects, including performing these commands, among others:
  - `ALTER TABLE`
  - `ALTER TABLESPACE`
  - `CREATE INDEX`
- Cannot view data in the application tables
- Cannot make a copy of the application tables

An application user must still be able to run the application, and view and modify data.

Application DBA: Implementation



Put the application data (tables, views, and so on) into a realm. This protects it from access by other DBAs in the database. Then the application DBA can be made a participant in the realm, enabling him or her to access the database objects. But the application DBA can also view the data and it is established that this is not allowed. Therefore, you implement a command rule that restricts the `SELECT` operation on these objects to be performed only by users with a specific role assigned to them.

It is assumed that the application users still have the object privileges necessary to override the realm protection. If this is not the case, they can also be made realm participants.

Application DBA: Workflow

1. Create the SALES_APP realm to protect the SH schema.
2. Make the SALES_DBA user a realm participant.
3. Create the SALES_APP_USER role.
4. Grant the SALES_APP_USER role to application users.
5. Create the Can Select SH rule set requiring the SALES_APP_USER role.
6. Create a SELECT command rule referring to the Can Select SH rule set.
7. Secure the SALES_APP_USER role in a realm.

The steps to establish an application DBA called SALES_DBA are as follows. The sales data is in the SH schema.

1. Create a realm to protect the schema. This means that only the realm members (participants or owners) can access the application's database objects.
2. Add SALES_DBA (which is the user ID for the application DBA) to the realm, enabling the application DBA to access the database objects. Now the application DBA can manage the database objects as needed.
3. Create a role to be granted only to application users. This is the role that is required to pass the SELECT command rule.
4. Grant the role to the application users, so that their access is not affected.
5. Create a rule set that has a single rule that checks for the SALES_APP_USER role.
6. Create a command rule that allows only SELECT on the SH objects when the preceding rule set is true.
7. Now that the SALES_APP_USER role exists, you need to protect it from being granted to inappropriate users. Create a realm that will hold this role and choose a single user to be the owner of this realm. In this example, the SYSTEM user is made the owner and thus, is the only one who can grant this role.

Application DBA: Result

SALES_DBA can perform administrative tasks, such as creating an index:

```
SQL> CREATE INDEX ix_ct ON sh.customers(country_id);

Index created.
```

But SALES_DBA cannot view the application data:

```
SQL> SELECT * FROM sh.customers;
SELECT * FROM sh.customers
              *
ERROR at line 1:
ORA-01031: insufficient privileges
```


In this example, the following steps took place:

- The application data was put into a realm.
- The SALES_DBA user was made a participant of that realm.
- The SELECT command rule was created to prevent SELECT on the data.

The result of these steps is the SALES_DBA user can perform such duties as index creation, but cannot view the application data.

Dual Key Security

Dual key security can be implemented by using Database Vault rule sets that check whether another user is logged in.

Dual key security means that there must two people involved when performing a specific task. The task could be granting access to database objects, executing a procedure, or executing any other database command.

Database Vault provides the tools necessary to implement this by enabling you to write a rule set that determines whether another user is currently logged in. The user ID to be checked would belong to a person other than the person attempting the operation. You can set aside a user ID that is intended to serve only this kind of purpose. The user with this ID would be granted only the ability to connect, and then have no further privileges. The very fact that this user ID is logged in allows certain things to happen. Some examples are shown in the following slides.

Dual Key Security: Use Case 1

Sample requirement: Only when the HR user is logged in, may other realm participants execute queries.

1

```
$ sqlplus BERNST
Password:
SQL> select count(*) from hr.jobs;
ORA-01031: insufficient privileges
```

2

```
$ sqlplus hr
Password:
SQL>
```

3

```
SQL> /
  COUNT(*)
----------
        19
```


In this example, the BERNST user is a participant in the HR realm. But there is an authorization rule set attached to the authorization for BERNST; the UTIL.USER_IS_ON function is called to see whether the HR user is currently logged in. If HR is logged in, BERNST can access the realm-secured objects.

The source code for the UTIL.USER_IS_ON function is as follows:

```
CREATE OR REPLACE FUNCTION UTIL.USER_IS_ON(p_username VARCHAR2)
RETURN VARCHAR2 AS
v_cnt number;
BEGIN
  select count(*) into v_cnt from
    (select null from sys.v_$session
     where username = p_username and rownum < 2);
  if v_cnt > 0 then
    return 'Y';
  else
    return 'N';
  end if;
END;
```

Dual Key Security: Use Case 2

Procedure Execution Example: Allow the execution of the `HR.GIVE_RAISE` procedure only when the `PAYROLL_MASTER` user is logged in.

- Create a command rule on `EXECUTE` of the `HR.GIVE_RAISE` procedure.
- Use the rule set `PAYROLL_MASTER is logged on` with the rule expression:

```
util.user_is_on('PAYROLL_MASTER')='Y'
```


You may have a situation in which a procedure should be run, but you want additional checks on the circumstances under which it is run. This can be implemented using dual key security by creating a command rule that controls the execution of the procedure.

In this example, a user is assigned the duty of running the `HR.GIVE_RAISE` procedure. However, there is a requirement that another person must have signed in for it to be run. This is implemented by creating a command rule on `EXECUTE` of the `HR.GIVE_RAISE` procedure. The command rule calls the rule set used in the previous example, checking for the `PAYROLL_MASTER` user to be logged in. If this user is not logged in, the `HR.GIVE_RAISE` procedure cannot be run. It may be that the only purpose for the `PAYROLL_MASTER` user is to log in and allow this procedure to be run, and then log out. Possibly, the manager of human resources (HR) and the chief financial officer (CFO) are the only people who have the password for this.

Note: The `PAYROLL_MASTER` user cannot execute the `HR.GIVE_RAISE` procedure. This user's only purpose is to allow another user to run it.

Database Account Considerations

To further enhance the security of your Database Vault configuration, adhere to the following guidelines:

- Define two separate user IDs for the Database Vault Owner and Database Vault Account Manager.
- Assign these accounts to separate people:
  - `SYS`
  - Database Vault Owner
  - Database Vault Account Manager
- Document and follow the naming conventions for account names.
- Audit the Database Vault realm.

ORACLE

When you install Database Vault, it is best to define separate accounts for the Owner and Account Manager for Database Vault. This prevents the Owner user from creating new accounts, putting them into realms for the purpose of performing suspect tasks, and then dropping the accounts.

If the Account Manager account is required to create and drop accounts, there is accountability and separation of duties. The `SYS` account should also be assigned to a separate person. This is the only way the system privileges of `SYS` can be controlled. Otherwise, the `SYS` user may log on as Database Vault Administrator, disable a realm, and then bypass the realm protections. Follow the suggested naming conventions for the Database Vault Owner and Account Manager accounts (such as `dsmith_secadmin` and `mjones_acctmgr`).

To monitor what Database Vault configuration changes are made, audit the Database Vault realm for success or failure situations. This produces an audit trail of any activity with Database Vault objects, including the case where the Owner account adds itself to a realm or disables a realm or rule set.

If you choose unobvious names for the Database Vault accounts, you can lessen the chance of Denial of Service (DoS) attacks, which may lock these accounts given enough attempts. If an attacker knows the account name, repeated attempts to log in with the wrong password can lock the account. Locking the Database Vault Owner account would render Database Vault Administrator unusable.

Dynamic Auditing

Use a rule set's custom event-handler logic to dynamically turn on auditing.

- A rule set can have a procedure invoked whenever it is evaluated for any reason.
- The procedure can call procedures that update the Database Vault components, setting their audit options appropriately.

Having full auditing on constantly is usually not good for performance or for managing an audit trail efficiently. Therefore, you may want to have the auditing of certain components turned on (for example, for success and failure) when certain rule sets are evaluated. If a rule set that allows connections to come through a virtual private network (VPN) IP address gets invoked, the connection is allowed to be made, but the rule set that is evaluated can have a custom event-handler logic procedure set. This procedure may make calls to update other rule sets that increase their auditing granularity, simply because you want to audit activity coming through VPN more thoroughly than that coming from internal clients.

Protecting from Accidental Object Loss

Use command rules to prevent inadvertent loss of production objects.

Command	Owner	Object	Rule Set Name
DROP TABLE	%	%	Disabled
TRUNCATE TABLE	%	%	Disabled
DROP DATABASE LINK	%	%	Disabled

```
SQL> DROP TABLE employees;
DROP TABLE employees
*
ERROR at line 1:
ORA-00604: error occurred at recursive SQL level 1
ORA-47400: Command Rule Violation for drop table
   on HR.EMPLOYEES
ORA-06512: at "DVSYS.AUTHORIZE_EVENT", line 55
```

ORACLE

In a production database, you may want to prevent accidental dropping of objects in certain schemas or in all schemas. If you rely on allowing only a small subset of users to perform the commands that drop objects, that lessens the likelihood of its occurrence, but it can still happen.

For example, if you define command rules to prevent dropping of tables, to drop the object, the DBA would have to request that the rule set be disabled. This separation of duties provides accountability and requires more consideration regarding the action being taken.

Quiz

Dual key security, requiring that two people must be logged in to perform a specific task, can be implemented using Database Vault rule sets.

a. True
b. False

Answer: a

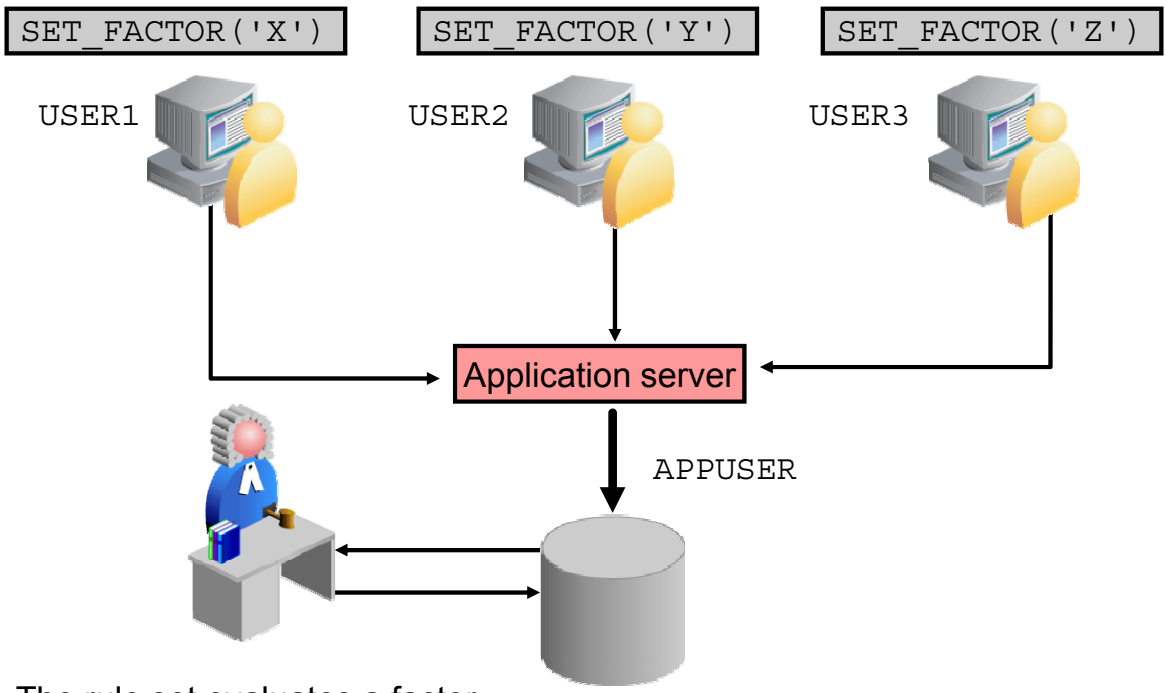
Quiz

Realms and rule sets together can be used to implement separation of duty.

a. True
b. False

Answer: a

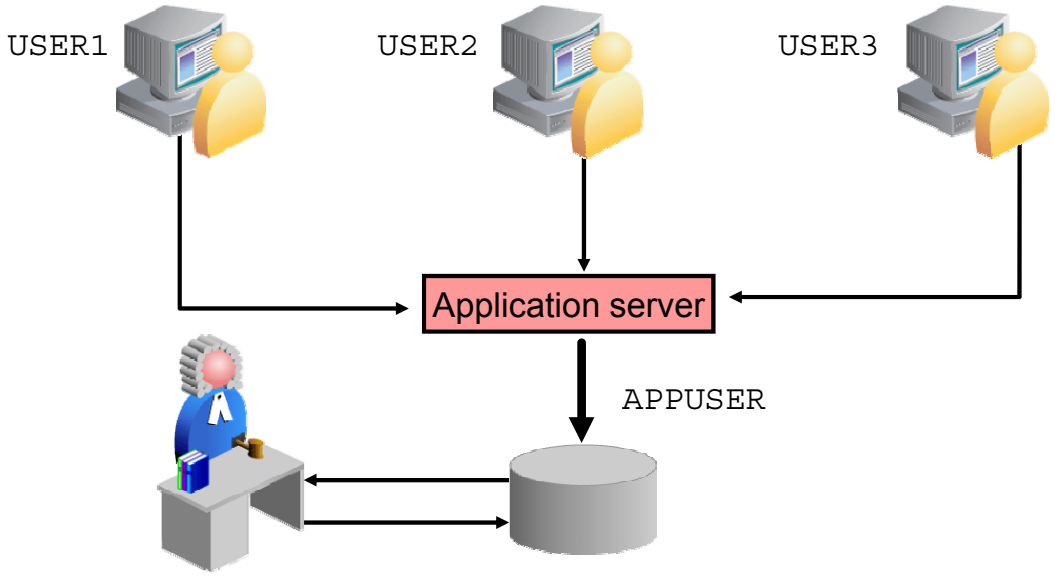
Connection Pooling Considerations



The rule set evaluates a factor.

When there is a connection pooling mechanism, the database may see all connections coming from application users as the same database account. This makes it difficult to mitigate access when decisions specific to end users need to be made. This is where the application can take advantage of `DVSYS.SET_FACTOR`. This allows the application to communicate user-specific data to the database through the connection pool. When the connection is made in Database Vault, the rule sets that refer to the factor are driven with the identities set by the application on behalf of the specific client.

Enforcing Connections from an Application Server



The rule set ensures that the IP address is acceptable before connecting.

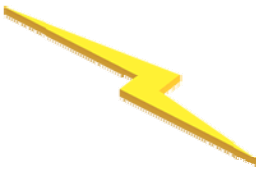
There may be circumstances where you know that all connections coming into Database Vault are from one or a set of application servers. You can code into a rule set a check on the IP address factor to ensure that the connection comes from a known application server. The expression for the rule in such a rule set is as follows:

```
DV.F$CLIENT_IP IN('111.111.111.111','111.111.111.222')
```

Fast Response to Policy Changes

You can respond quickly to security policy changes by using the delivered ON/OFF components to toggle:

- Factors: Use Enabled and Disabled assignment rule sets.
- Rule sets:
  - Enable or disable the rule set.
  - Add the `True` or `False` rule to the rule set.
- Realms: Enable or disable the realm.
- Command rules: Set the rule set to Enabled or Disabled.
- Secure application roles: Set the rule set to Enabled or Disabled.

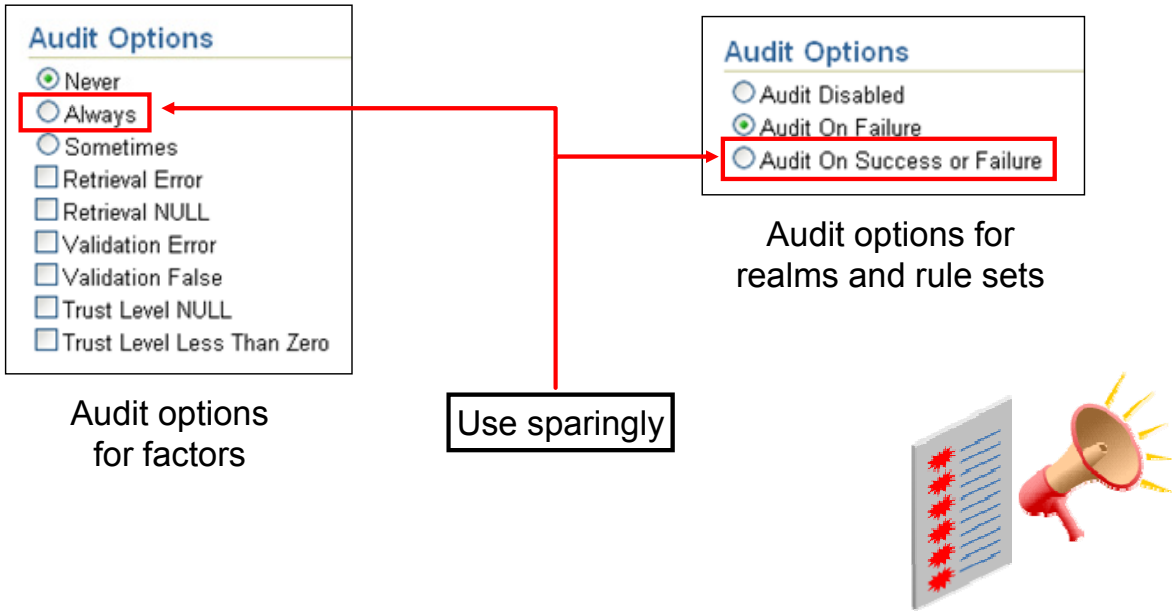


ORACLE

There are settings in the Database Vault components that make it very easy to quickly open or close access. Most components can be enabled or disabled. And the rules `True` and `False` can be used to change the rule set behavior.

Performance Considerations: Auditing

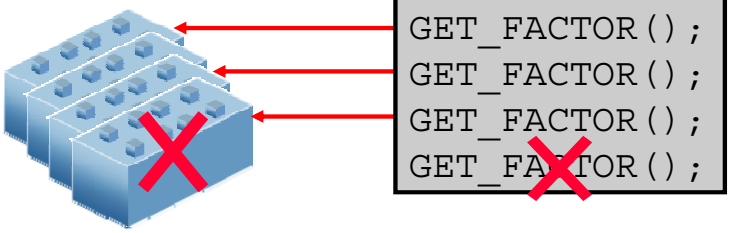
Audit only on failure.

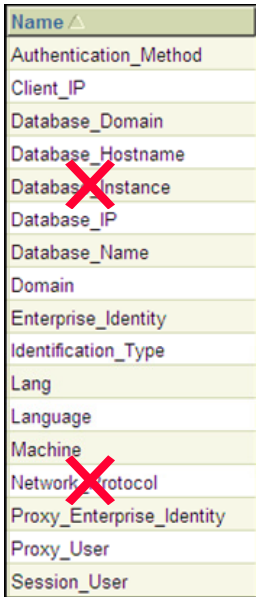


To maintain best performance in Database Vault, you should audit only on failure if possible. Otherwise, many audit records are generated when they may not be necessary. Therefore, use success auditing only when necessary. Also remember that the evaluation of rule sets and factors may be invoked when accessing realms, setting secure application roles, or resolving command rule access. The performance impact can be significant if too many components are being audited.

Performance Considerations: Unused Factors

Delete unnecessary factors.





Delivered session factors

Delete factors that you do not need. This is important for those factors defined with the For Session option. To mitigate performance issues for frequently used factors, these are evaluated and cached every time a user connects to the database. This makes subsequent accesses faster because it is known that they need to be evaluated only once for the session. However, whether they are ever referenced or not, they are evaluated once at connect time. If your system has a high frequency of session creation, depending on the number and complexity of your defined factors, the speed of connections and other operations may be noticeably impacted.

Note: If there is a factor that, by its nature, will not change during the lifetime of a connection, make sure that it is set to be evaluated For Session, rather than By Access. If a factor is reevaluated every time it is accessed, but it will never change, it is wasting resources.

Diagnosing Database Vault Using Trace Files

A session encounters a Database Vault access restriction:

```
SQL> alter session set events '47998 trace name context
     forever, level 12';

Session altered.

SQL> select * from hr.jobs;
select * from hr.jobs
*
ERROR at line 1:
ORA-01031: insufficient privileges
```

The trace file shows that the cause is a realm authorization failure:

```
...
DATVAL: kzvaudevent() start
DATVAL: Realm Auth Failed.  Current User: SYSTEM
DATVAL: Realm Auth Failed.  Object Owner:HR, Object Name:JOBS
```

ORACLE

You can monitor your Database Vault database instance for server and background process events. To do so, check the trace files of the database instance. Trace files can reveal events, such as tracing information for the logic that the Database Vault security enforcement engine executes, as well as internal errors, block corruption errors, deadlock errors, administrative actions that may have taken place, values of parameters that had nondefault settings when the database instance started, and other information.

To enable tracing, log in with any account that has the `ALTER SESSION` privilege and issue the following statement:

```
SQL> ALTER SESSION SET EVENTS
  2  '47998 trace name context forever, level 12'
```

In the example in the slide, the error message clearly says that a realm violation took place, so examining the trace file just confirms the issue. For other scenarios, you may get a more generic error (such as `ORA-01031: insufficient privileges`). You could turn on the trace event, reissue the command that caused the error, and then look at the trace file to see that it was a realm authorization violation.

Note: For more information about how to manage trace files, see the *Oracle Database Administrator's Guide* and the *Oracle Database Performance Tuning Guide*.

Password Policy Considerations

It is recommended that you make the following changes to your password policy:

	Default	Recommended
FAILED_LOGIN_ATTEMPTS	10	7
PASSWORD_GRACE_TIME	7	3
PASSWORD_LIFE_TIME	180	90
PASSWORD_REUSE_TIME	UNLIMITED	180

To further tighten security measures, it is recommended that you make these changes to the password policy in a Database Vault instance:

- **FAILED_LOGIN_ATTEMPTS:** The number of failed login attempts allowed before the account is locked. The recommended value is seven attempts.
- **PASSWORD_GRACE_TIME:** The number of days after which the grace period begins (for a password change requirement) during which a warning is issued and login is allowed. If the password is not changed during the grace period, the password expires. The recommended value is three days.
- **PASSWORD_LIFE_TIME:** The number of days that a password can be used. The recommended value is 90 days.
- **PASSWORD_RESUSE_TIME:** The number of days between reuse of the same password. The recommended value is 180 days.

Guidelines for Procedures and Packages

Concern	Remedy
Java Stored Procedures	Use invoker's rights and realms to protect data.
`UTL_FILE` Package	Grant **`EXECUTE`** only to users that require it. Use command rules to control the creation of directory objects.
`DBMS_FILE_TRANSFER` Package	Grant **`EXECUTE`** only to users that require it. Use command rules to control `EXECUTE` on the `DBMS_FILE_TRANSFER` package. Use command rules to control the creation of directory objects and database links.
LogMiner Packages	Grant **`EXECUTE`** privileges directly. Revoke privileges when finished. Audit the use of the LogMiner packages.

ORACLE

Database Vault does not protect everything. There are certain areas that the users of Database Vault should consider. Care must be taken to protect the following procedures and packages. For detailed instructions to implement these recommendations, see *Oracle Database Vault Administrator's Guide 11g Release 1 (11.1)*.

- **Java Stored Procedures:** For definers' rights Java stored procedures, the execution of the stored procedure is not blocked and realm protection is not enforced. However, underlying objects accessed by the Java stored procedure are protected by the Database Vault command rules. For invoker rights Java stored procedures, the execution of the stored procedure is not blocked. However, the underlying objects accessed by the Java stored procedure are protected by both Database Vault realms and command rules.
- **`UTL_FILE` Package:** It is recommended that you grant the `EXECUTE` permission for the `UTL_FILE` package to specific application owners, and then revoke this package from `PUBLIC`. Use command rules to control the creation of directory objects and thus further control access to the file system.
- **`DBMS_FILE_TRANFER` Package:** Grant `EXECUTE` privileges to only those users that require the use of this package. Revoke privileges from all others. Use command rules to control the creation of directory objects and database links.

- **LogMiner Packages:** If LogMiner packages are needed, grant `EXECUTE` privileges directly and revoke immediately after the task is completed. Audit the use of the LogMiner packages. A user with privileges to execute the LogMiner packages can view, insert, update, and delete records.

Other Guidelines

Concern	Remedy
Recycle Bin	Disable the Recycle Bin feature or have realm users use `DROP … PURGE`.
`ALTER SYSTEM` `ALTER SESSION`	Use the delivered `ALTER SYSTEM` command rule to prevent dumps.
`CREATE ANY JOB` `CREATE JOB`	Revoke `CREATE ANY JOB` from all users. Use `CREATE JOB` in place of `CREATE ANY JOB`.
Using Data Pump	To authorize users who need to use Oracle Data Pump, use `DVSYS.DBMS_MACADM.AUTHORIZE_DATAPUMP_USER`.
Scheduling database jobs	To authorize users who need to schedule database jobs, use `DVSYS.DBMS_MACADM.AUTHORIZE_SCHEDULER_USER`.

ORACLE

There are situations in which unauthorized users may see realm-protected data, if they have certain privileges. These guidelines help you protect against such scenarios. For detailed instructions about implementing these recommendations, see *Oracle Database Vault Administrator's Guide 11g Release 2 (11.2)*.

Recycle Bin: `SELECT_CATALOG_ROLE` has the `SELECT` privilege on the `DBA_RECYCLEBIN` view, which contains all dropped objects. A user with this role can see the contents of the recycle bin. It is recommended that you disable the `RECYCLE BIN` feature. If this feature is required, schema owners in protected realms should purge their recycle bins whenever they drop any objects.

`ALTER SYSTEM` / `ALTER SESSION` privileges: Users with the `ALTER SESSION` or `ALTER SYSTEM` privilege can set events and dump blocks and then read unencrypted data even from realm-protected objects. Use command rules to prevent this.

`CREATE ANY JOB` / `CREATE JOB`: Users with the `CREATE ANY JOB` privilege can create a job in a schema that has realm access privileges. Revoke the privilege when it is not required. Grant the `CREATE JOB` privilege instead of `CREATE ANY JOB` whenever feasible. `CREATE JOB` allows the user to create jobs only in his or her own schema.

Oracle Streams, Oracle Database Enterprise Manager Grid Control, and Advanced Replication depend on the CREATE ANY JOB privilege. If you use these products, grant this privilege only to SYS.

The CREATE ANY JOB recommendations described in this section also apply to the following privileges:

- CREATE EXTERNAL JOB
- EXECUTE ANY PROGRAM
- EXECUTE ANY CLASS
- MANAGE_SCHEDULER

Using Data Pump: In addition to the standard Oracle Data Pump privileges, any user who needs to use Oracle Data Pump for export and import in a Database Vault environment must be authorized to do so. Authorize desired users by using the DVSYS.DBMS_MACADM.AUTHORIZE_DATAPUMP_USER procedure. For example:

```
EXEC DVSYS.DBMS_MACADM.AUTHORIZE_DATAPUMP_USER('DP_MGR');
```

Scheduling database jobs: In addition to the standard scheduling privileges, any user who needs to schedule database jobs in a Database Vault environment must be authorized to do so. Authorize desired users using the DVSYS.DBMS_MACADM.AUTHORIZE_SCHEDULER_USER procedure. For example:

```
EXEC DVSYS.DBMS_MACADM.AUTHORIZE_SCHEDULER_USER('JOB_MGR');
```

Miscellaneous Recommendations

Consider the following as you work with Database Vault:

- Do not put spaces in factor names:

```
SQL> select "DVF"."F$A B" from dual;
ORA-00904: "DVF"."F$A B": invalid identifier
```

- To create rules that are not yet needed in a rule set, define a repository rule set to store them in until they are needed.
- Test any expressions for rules and factors by putting them into this statement:

```
SQL> SELECT 1 from dual WHERE <expression>;
```


Some other considerations when working with Database Vault are as follows:

- Do not put spaces in factor names. It is possible to create a factor with a space in its name, but when you refer to the factor name by using the `DVF.F$` syntax, it fails because of the space.
- If you need to create some rules that you know you will need at some point, but do not have a rule set to contain them yet, consider creating a repository rule set. There is nothing special about this rule set; it just will not be referenced anywhere. However, if you create this, you can proceed to create any rule you think that you will need at some point and have a place to store it. When you use Database Vault Administrator, the only way to create a rule is to edit a rule set. Rather than risk making changes to a rule set that is being referenced by other components, create a rule set called Repository (or something similar) and put these rules there. Then they are available for use in other rule sets when you need them.
- Sometimes, the expressions in rules can get complex. You can test them before saving them by appending the expression to the SQL statement:

  `SELECT 1 FROM dual WHERE <expression>`

  If `1` is returned, the expression is true. If no rows are returned, it is false.

Oracle Database Vault

Application Certification

• PeopleSoft	✓	Done
• E-Business Suite 11*i*/R12	✓	Done
• Siebel	✓	Done
• Oracle Content DB	✓	Done
• Oracle Internet Directory	✓	Done
• iFlex Flexcube UBS	✓	Done

Security policies for the certified applications can be downloaded from:

`http://www.oracle.com/technology/software/products/database_vault/index.html`

Quiz

Assume that the PROD database has Database Vault configured. On the PROD database, Joe needs to be able to schedule jobs and Emily needs to use Oracle Data Pump. What should you do?

a. Nothing. No additional configuration is required.
b. Authorize Joe to schedule database jobs by using DVSYS.DBMS_MACADM.AUTHORIZE_SCHEDULER_USER and authorize Emily to use Oracle Database Pump by using DVSYS.DBMS_MACADM.AUTHORIZE_DATAPUMP_USER.
c. No additional configuration is required for Emily, but you must authorize Joe to schedule database jobs by using DVSYS.DBMS_MACADM.AUTHORIZE_SCHEDULER_USER.
d. No additional configuration is required for Joe, but you must authorize Emily to use Oracle Database Pump by using DVSYS.DBMS_MACADM.AUTHORIZE_DATAPUMP_USER.

ORACLE

Answer: b

Quiz

The provided Enabled and Disabled rule sets can be used to quickly turn on and off restrictions implemented by Database Vault components.

a. True
b. False

Answer: a

Summary

In this lesson, you should have learned how to:

- Identify your security requirements and determine how to implement separation of duty
- Describe the suggested best practices for implementing Database Vault
- Describe the first steps in securing a database by using Oracle Database Vault
- Identify the Database Vault components needed to protect against accidental object loss
- List special considerations for dealing with an application server: connection pooling and limiting connections
- Identify the performance issues to be considered

ORACLE

Practices

- 11-1: Protecting Data from `SELECT ANY TABLE` Access
- 11-2: Restricting `OE DBA` Activities to Nonbusiness Hours
- 11-3: Locking Down the DBA Roles
- 11-4: Preventing Data Loss
- 11-5: Allowing Temporary `ALTER SYSTEM` Command Access

This practice presents a workshop scenario covering the topics listed in the slide.

A

Your Learning

Overview

This appendix provides:

- Cheat sheets:
  - DOS/ UNIX Commands
  - vi Editor
- Further training options
- Personalized, additional resources

This appendix provides a few “cheat sheets” that might be helpful during the course.
It also adds details about further training options.
Your instructor might suggest additional resources for your learning.
See also Appendix B for more information on *Using Enterprise Manager Cloud Control*.

DOS/UNIX Commands, a brief starter:

DOS	UNIX	English
dir	ll	list long (name, date, size, owner, etc)
dir/w	ls	list wide (no details)
dir/s	locate	find a file anywhere
del	rm	delete or remove files
copy	cp	copy file1 to file2
move	mv	move file1 to file2
ren	mv	rename file1 to file2
cd	pwd	print working directory
cd ..	cd ..	change directory UP one level
cd \	cd /	change directory to TOP level (root)
C-A-D	ps -ef	process statistics (often used with grep)
	top	dynamic list of top processes by percent
md	mkdir	make directory
rd	rmdir	remove directory
edit	vi	full-screen character-based editor (see below)
more	more	list a file and pause (space/enter to continue)
	tail -20 file1	list the last 20 lines of a file
type	cat	list a file and don't pause
	strings	same as cat but for files with binary chars
set	set	display all environment variables such as $HOME
help	man	manual (help) pages
find	grep	find a word in a line in a larger list of lines
prompt	PS1='$PWD >'	change the prompt to include current dir
logoff	su -	switch user (usually to Super User)
chkdsk	df -h	how much free space is left on disk (in GB)
(n/a)	which file1	finds executables along paths
ver	uname –a	version of operating system software

Remember: Everything in UNIX is case-sensitive.

To change to a "ReallyLongDirectoryName“, type "cd Rea*".

Gedit

gedit is a visual wysiwyg editor. The first time gedit runs, it needs the &:

```
gedit filename &
```

then you can use:

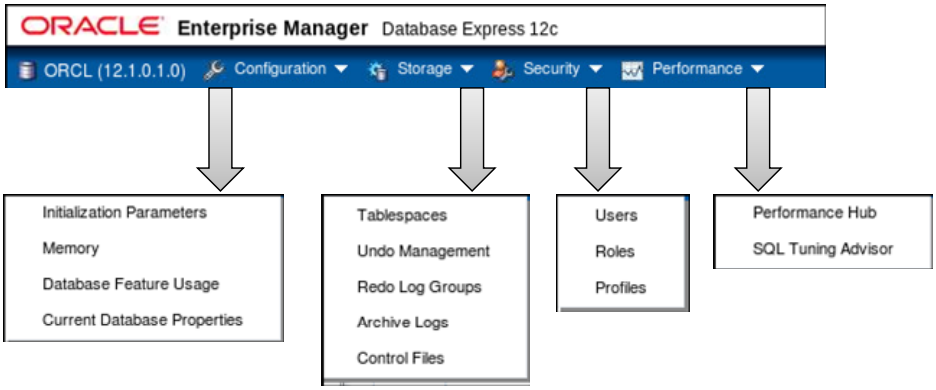
```
gedit filename
```

Vi Commands

Command	Description
:wq	write, quit
:q!	quit, no save
[esc]	get out of current mode back to command mode
a	append mode
A	Append mode at end of current line
dd	delete current line
i	insert mode (can also insert line feeds)
o	insert blank line below cursor
p	paste buffer after cursor
r	replace single character
:s/a/b/	substitute (change) "a" to "b"
.	repeat last substitution
u	undo last change
Y	yank (copy) current line
x	delete single character
dd	delete whole line

Enterprise Manager Database Express Menus

http://<*hostname*>:<*port*>em



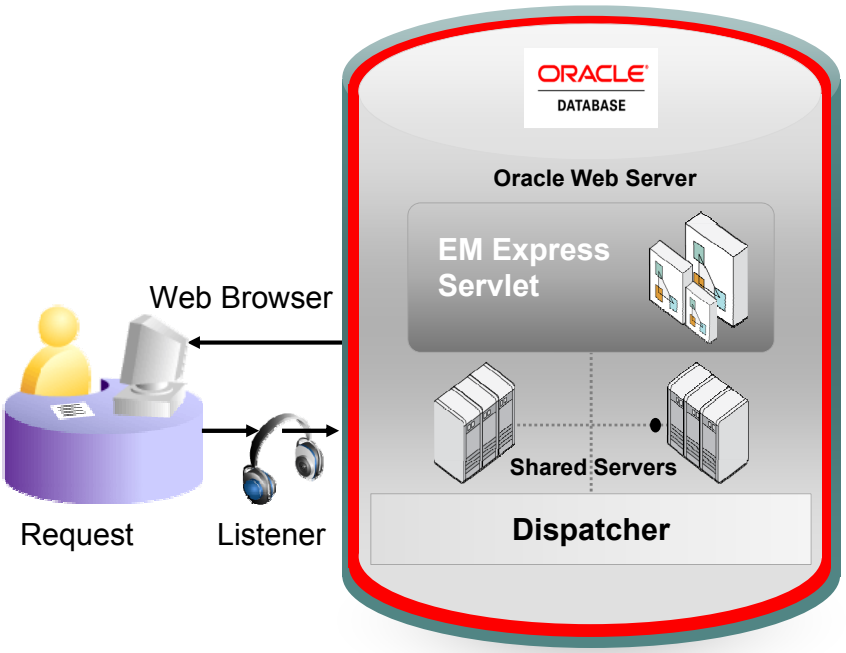
Enter your URL and then a valid Oracle username and password.

- To display the hostname in Linux: `hostname --long`
- The port is by default `5500`.
- In Oracle classrooms, you may use `SYS` as User Name, `oracle_4U` as Password, acting `as sysdba`.

The slide shows Enterprise Manager Database Express (EM Express) menus for:

- Configuration
- Storage
- Security
- Performance

Request Handling in EM Express



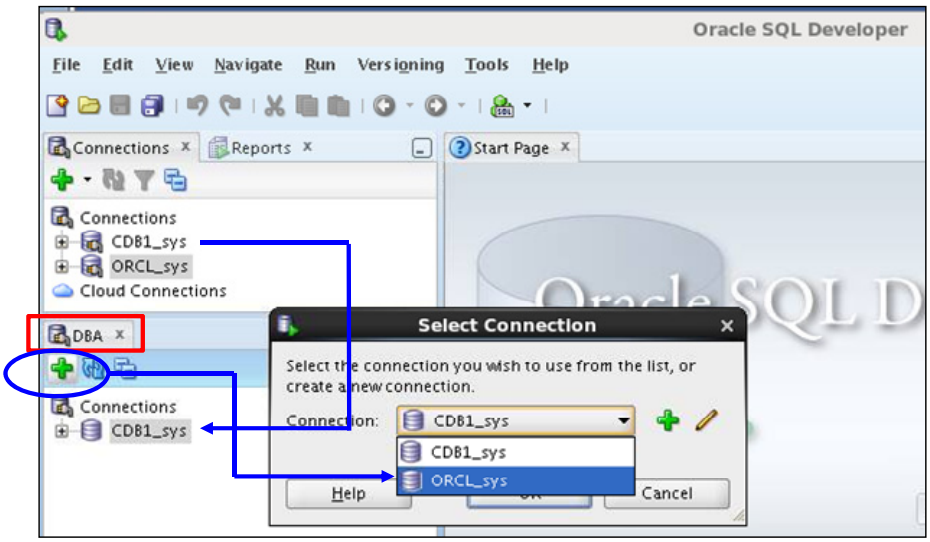
EM Express Servlet

- Authenticates and validates the request
- Serves the request by executing queries inside the database
- Writes the output to the response stream

This slide provides an overview of how EM Express handles requests.

Oracle SQL Developer: Connections

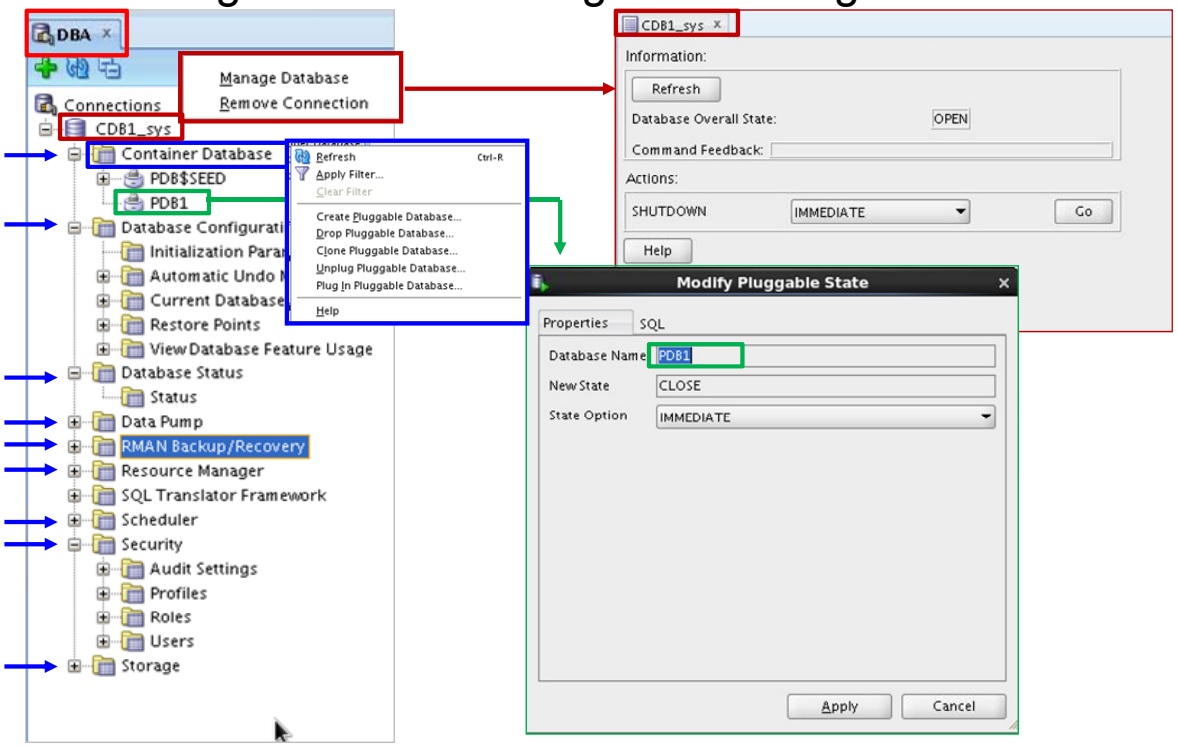
Perform DBA operations in the DBA navigator using DBA connections:



Oracle SQL Developer is a tool that allows stand-alone graphical browsing and development of database schema objects, as well as execution of database administrative tasks.
SQL Developer enables users with database administrator privileges to view and edit certain information relevant to DBAs and perform DBA operations. To perform DBA operations, use the DBA navigator, which is similar to the Connections navigator in that it has nodes for all defined database connections. If the DBA navigator is not visible, select View and then DBA. You should add only connections for which the associated database user has DBA privileges or at least privileges for the desired DBA navigator operations on the specified database.

Oracle SQL Developer: DBA Actions

Performing DBA tasks through DBA navigator:

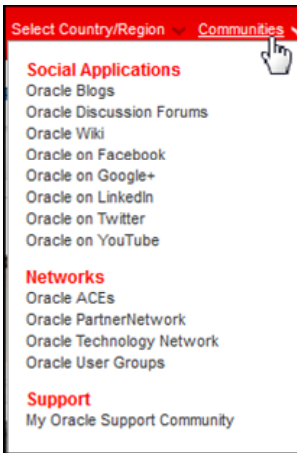


The DBA operations that can be performed are the following:

- Pluggable database startup/shutdown
- Database configuration: Initialization Parameters, Automatic Undo Management, Current Database Properties, Restore Points, View Database Feature Usage
- Database status view
- Data Pump Export and Import jobs
- RMAN Backup/Recovery actions
- Resource Manager configuration
- Scheduler setting
- Security configuration like audit settings, profiles, roles, users
- Storage configuration for archive logs, control files, data files, redo log groups, tablespaces, temporary tablespace groups

Continuing Your Learning

- Documentation
- Oracle Technology Network (OTN)
  - Oracle Database > Options > Oracle Database Vault
  - Oracle communities on Social Applications
- Oracle University training courses
- Oracle University self-studies
- Oracle Learning Library (OLL)
- My Oracle Support (MOS)
- YouTube videos on the Oracle Leaning channel



Documentation:

- Oracle Database Vault Administrator's Guide

Oracle Technology Network (OTN): http://www.oracle.com/technetwork/index.html

Oracle University training courses:

- Oracle Database 12*c*: Security

Oracle University self-studies: Oracle Database 12*c* New Features series

Oracle Learning Library (OLL): https://www.oracle.com/goto/oll

Further Information

For more information about topics that are not covered in this course, refer to the following:

- Other Oracle University Oracle Database 12*c* ILT courses
- Oracle Database 12*c*: New Features Self-Studies
  - A comprehensive series of self-paced online courses covering all new features in detail
  - Demonstrations for all topics covered in self-paced online courses: http://www.oracle.com/goto/oll
- Oracle By Example series: Oracle Database 12*c*
  - http://www.oracle.com/technology/obe/demos/admin/demos.html
  - http://www.oracle.com/technology/obe/start/index.html
- Oracle OpenWorld events
  - http://www.oracle.com/openworld/index.html

ORACLE

For more information about topics that are not covered in this course, refer to the following:

- Oracle University Oracle Database 12*c* instructor-led courses
- Oracle Database 12*c*: New Features self-paced online courses
- Oracle By Example series: Oracle Database 12*c*
- Oracle OpenWorld events

B

Using Enterprise Manager Cloud Control

Objectives

After completing this lesson, you should be able to:

- Describe the different components of Cloud Control
- Explain the architecture of Cloud Control
- List the target types managed by Cloud Control
- Explore the Oracle Enterprise Manager Cloud Control interface

Note

For a complete understanding of Oracle Enterprise Manager Cloud Control and Database Express installation and usage, refer to the following guides in the Oracle documentation:

- *Oracle Enterprise Manager Cloud Control Basic Installation Guide 12c Release 1*
- *Oracle Enterprise Manager Cloud Control Advanced Installation and Configuration Guide 12c Release 1*
- *Oracle Enterprise Manager Cloud Control Administrator's Guide 12c Release 1*
- *Oracle Enterprise Manager Licensing Information 12c Release 1*

Key Challenges for Administrators

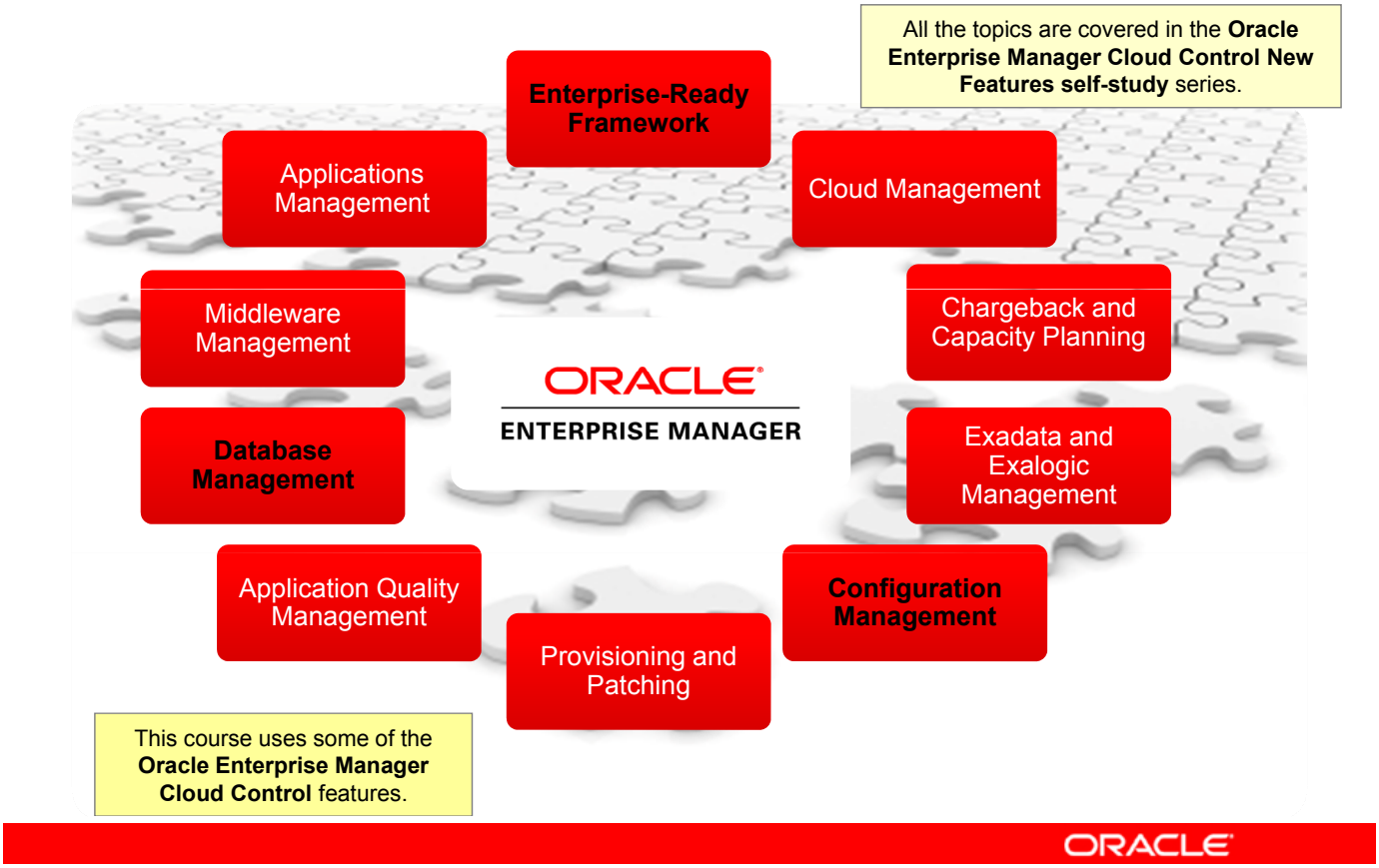
As the composition of the data center broadens into the cloud environment, the challenges to manage it also increase. The key challenges for managing a data center include:

- Monitoring performance and availability
- Resolving problems quickly
- Containing operating costs
- Aligning IT with business priorities

As data centers grow with the growth in business, so do the challenges. An administrator is faced with challenges that include:

- Monitoring high levels of performance and availability of applications
- Identifying and resolving problems quickly and effectively
- Enabling IT professionals to use resources effectively, thereby reducing costs
- Aligning IT with business priorities to ensure that businesses are agile enough to meet the changing needs

Enterprise Manager Cloud Control



Key Objectives in Enterprise Manager Cloud Control Design

- Designing a management framework that is capable of providing next-generation functionality
- Enhancing application-to-disk manageability
- Providing a complete enterprise private cloud solution

Enterprise Manager Cloud Control includes the following features:

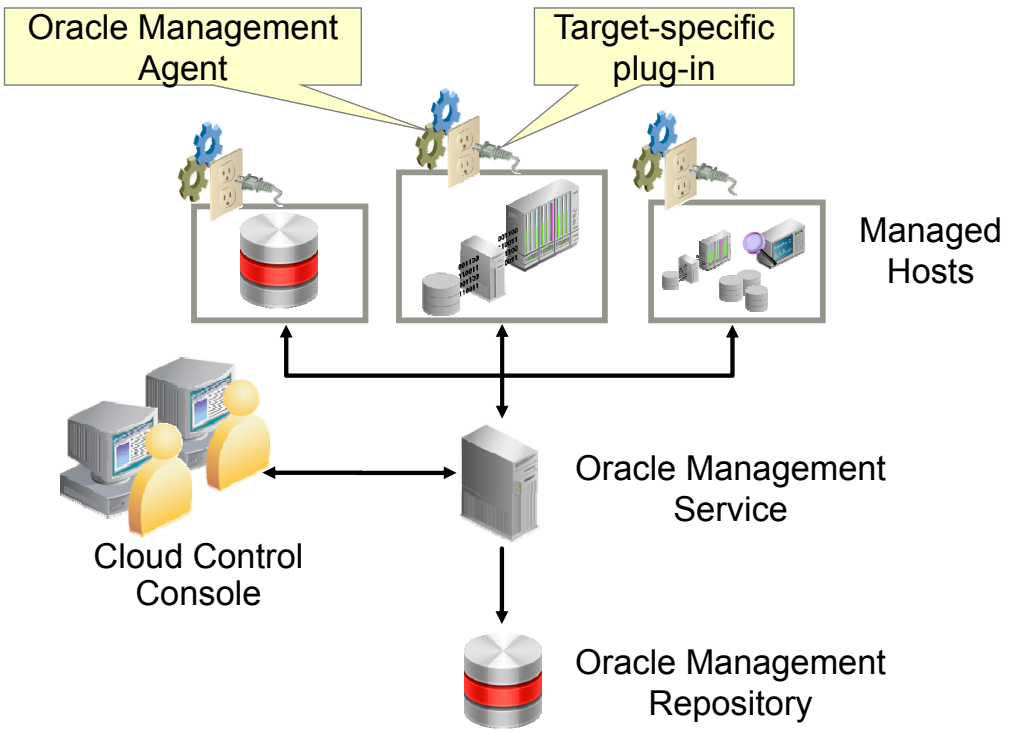
- **Enterprise-Ready Framework:** Provides modular and extensible architecture, target plug-ins, self-updateable entities, integrated Support Workbench, and centralized incident console
- **Cloud Management:** Provides complete cloud lifecycle management
- **Chargeback and Capacity Planning:** Provides chargeback based on target types, and uses Automatic Workload Repository (AWR) Warehouse to consolidate AWR reports from multiple databases across the enterprise
- **Exadata and Exalogic Management:** Provides an integrated view of the hardware and software in an Exadata machine, and complete lifecycle management for Exalogic systems
- **Configuration Management:** Provides an integrated set of tools, agent-less discovery, integration with My Oracle Support, and custom configuration capabilities

- **Provisioning and Patching:** Provides profiles for provisioning known configurations, user-defined deployment procedures, and a software library integrated with self-updating capabilities
- **Application and Quality Management:** Database Replay, Application Server Replay, Real Application Testing integrated with Data Masking, and test-database management including Application Data Model
- **Database Management:** Provides management of Oracle Database systems, including performance management and change lifecycle management. Some aspects of Database Management are covered in detail in this course.
- **Middleware Management:** Provides management of Fusion Middleware systems
- **Applications Management:** Provides management of Fusion Applications

Note: For a complete understanding of Oracle Enterprise Manager Cloud Control usage, refer to the following sources of information:

- *Using Oracle Enterprise Manager Cloud Control* course
- *Oracle Enterprise Manager Cloud Control: Install and Upgrade* course
- *Oracle Enterprise Manager Cloud Control New Features* Self-Study series
- *Oracle Enterprise Manager 12c* demonstrations in the Oracle Learning Library (URL www.oracle.com/goto/oll)

Cloud Control Components

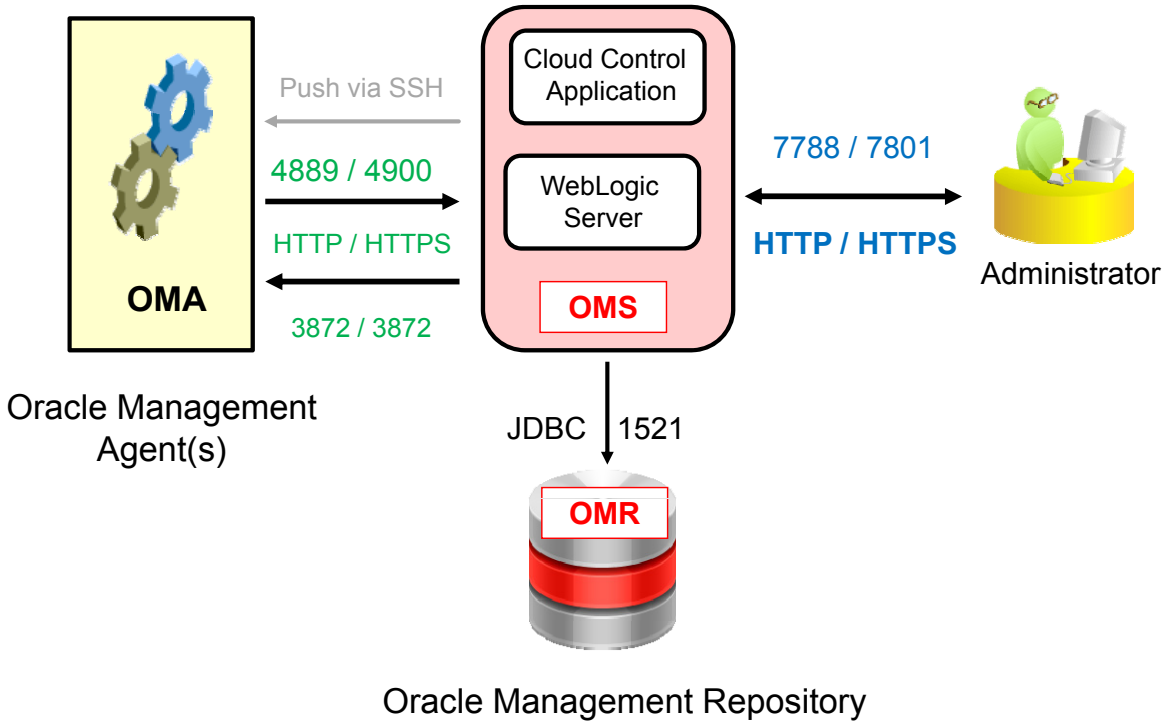


Enterprise Manager Cloud Control is composed of four main components as illustrated:

- The Oracle Management Repository (OMR)
- The Oracle Management Service (OMS)
- The Oracle Management Agent (OMA or agent) with target-specific plug-ins
- The Cloud Control Console

The Oracle Management Agent runs on hosts, gathering metric data about those host environments as well as using plug-ins to monitor availability, configuration, and performance and to manage targets running on the host. The agents communicate with the Oracle Management Service to upload metric data collected by them and their plug-ins. In turn, the OMS stores the data that it collects in the Oracle Management Repository where it can be accessed by the OMS for automated and manual reporting and monitoring. The OMS also communicates with the agents to orchestrate the management of their monitored targets. As well as coordinating the agents, the OMS runs the Cloud Control Console web pages that are used by administrators and users to report on, monitor, and manage the computing environment that is visible to Cloud Control via the agents and their plug-ins.

Components and Communication Flow



The communication flow between the Cloud Control components is illustrated using directional arrows. Communication between an agent and the OMS, and the OMS and the console is bi-directional. All the ports shown and listed in the slide are default values that can be changed during installation, either by the installer as it searches for available ports, or explicitly by you. You can also change ports after installation.

- An agent uploads data to the OMS via HTTP on port 4889 or via HTTPS on port 4900 (designed to be able to work with WAN.\).
- The OMS communicates with the agent via HTTP or HTTPS on port 3872.
- The reason for the separate ports for OMS to OMA communications is that they can communicate asynchronously and simultaneously with one another.
- The OMS communicates with the OMR via JDBC on port 1521. Although the OMR will return data to the OMS, this is not considered to be a separate communication between the two; therefore, the flow is shown to be unidirectional from OMS to OMR.
- Cloud Control console users access the Cloud Control webpages via HTTPS on port 7801 or via HTTP on port 7788.

Knowing the ports used in your Cloud Control installation is important, especially if you are managing hosts behind firewalls or where other network restrictions apply, because communication will need to be allowed on these ports and in the directions shown.

Oracle Management Repository

The Oracle Management Repository (OMR):

- Resides in an Oracle database
- Includes schema objects belonging to `SYSMAN`
- Must be installed in a preexisting database
- Can be installed in a RAC database

Note: Uses a restricted-use license of the Oracle Database

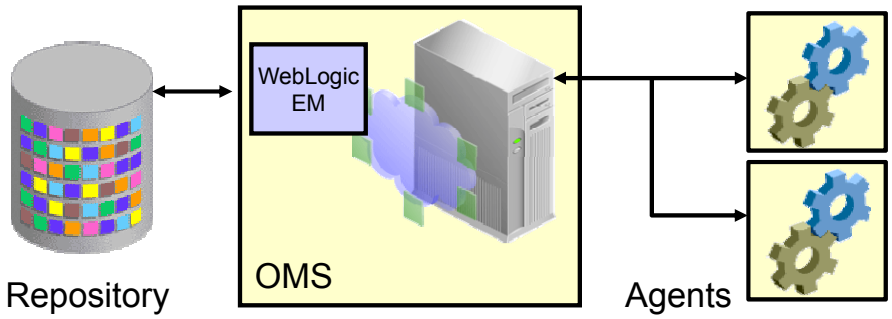
The OMR is installed in an Oracle database as a group of approximately 4,000 schema objects belonging to the `SYSMAN` user stored in three tablespaces: `MGMT_ECM_DEPOT_TS`, `MGMT_TABLESPACE`, and `MGMT_AD4J_TS`. These schema objects contain information about Enterprise Manager Cloud Control users and administrators, targets and applications that are monitored and managed by Enterprise Manager Cloud Control, and groups, systems, incidents, and other Enterprise Manager Cloud Control artifacts. The OMR is created during installation in a preexisting database, and for scalability requirements can be installed in a Real Application Clusters (RAC) database. In this case, customers must license the second node for the database, and both nodes require an Oracle Real Application Clusters license.

The database used to house the OMR should not be used for any other applications for the following reasons:

- Enterprise Manager Cloud Control's usage of the database should not have to compete with any other usage.
- Using the OMR database for other applications may restrict your ability to upgrade and patch the OMR schema and database as required
- Enterprise Manager Cloud Control includes a restricted-use database license that can be used for the OMR only

Note: Refer *Oracle Enterprise Manager Licensing Information 12c Release 1,* Section *Enterprise Manager Restricted-Use License*

Controlling the Enterprise Manager Cloud Control Framework



Component Control Utilities		
Repository	OMS	Agent
`sqlplus` or `srvctl`	`emctl`	`emctl`
`lsnrctl`		

Each component of the Enterprise Manager Cloud Control framework has its own utility or utilities that can be used to monitor, start, and stop the component. In many cases, these utilities also provide some capability to configure the component beyond the simple start-and-stop functionality.

RAC databases require the use of the Server Control commands; for single instances, there is a choice between SQL*Plus and Server Control. Server Control is usable when Oracle Restart is installed and the database is registered with the OLR.

To start and stop the listener, use either the Server Control utility or the `lsnrctl` command.

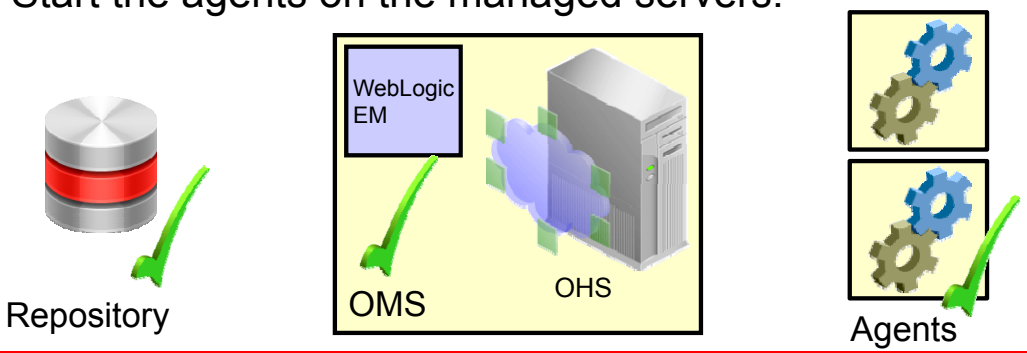
Examples:

```
srvctl stop database -d orcl -o immediate
srvctl start database -d orcl -o open
```

Starting the Enterprise Manager Cloud Control Framework

To start the Cloud Control framework, perform the following steps:

1. Start the repository database listener.
2. Start the repository database instance.
3. Start the OMS.
4. Start the agent on the OMS/repository server.
5. Start the agents on the managed servers.



ORACLE

To start the whole Enterprise Manager Cloud Control framework, follow the steps below:

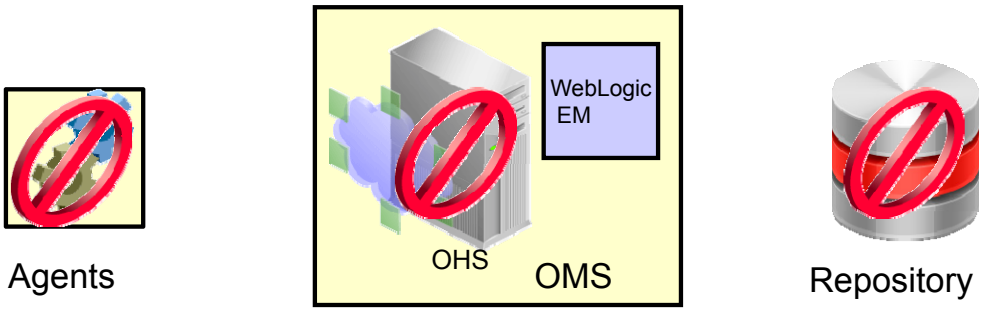
1. Start the repository listener:
   ```
   $ORACLE_HOME/bin/lsnrctl start
   ```
2. Start the repository database instance:
   ```
   $ORACLE_HOME/bin/sqlplus / as sysdba
   SQL> startup
   ```
3. Start the OMS (including OHS and WebLogic Managed Server):
   ```
   $OMS_HOME/bin/emctl start oms
   ```
4. Start the agent (on the OMS/repository host):
   ```
   $AGENT_HOME/bin/emctl start agent
   ```
5. Start the agent on the managed servers:
   ```
   $AGENT_HOME/bin/emctl start agent
   ```

Note: Use the `SRVCTL` command if you have a RAC instance for the repository.

Stopping the Enterprise Manager Cloud Control Framework

To stop the Enterprise Manager Cloud Control framework, perform the following steps:

1. Stop the agents on the managed servers.
2. Stop the agent on the OMS/repository server.
3. Stop the OMS.
4. Stop the repository database instance.



To stop the whole Enterprise Manager Cloud Control framework, follow the steps below:

1. Stop the agent on the managed servers:
   `$AGENT_HOME/bin/emctl stop agent`
2. Stop the agent (on OMS/repository host):
   `$AGENT_HOME/bin/emctl stop agent`
3. Stop the OMS (including OHS and WebLogic Managed Server):
   `$OMS_HOME/bin/emctl stop oms`
4. Stop the repository database instance:
   ```
   $ORACLE_HOME/bin/sqlplus / as sysdba
   SQL> shutdown immediate
   ```

Note: Use the `SRVCTL` command if you have a RAC instance for the repository.

Different Target Types

Enterprise Manager Cloud Control can monitor, administer, maintain, and manage many different types of targets including:

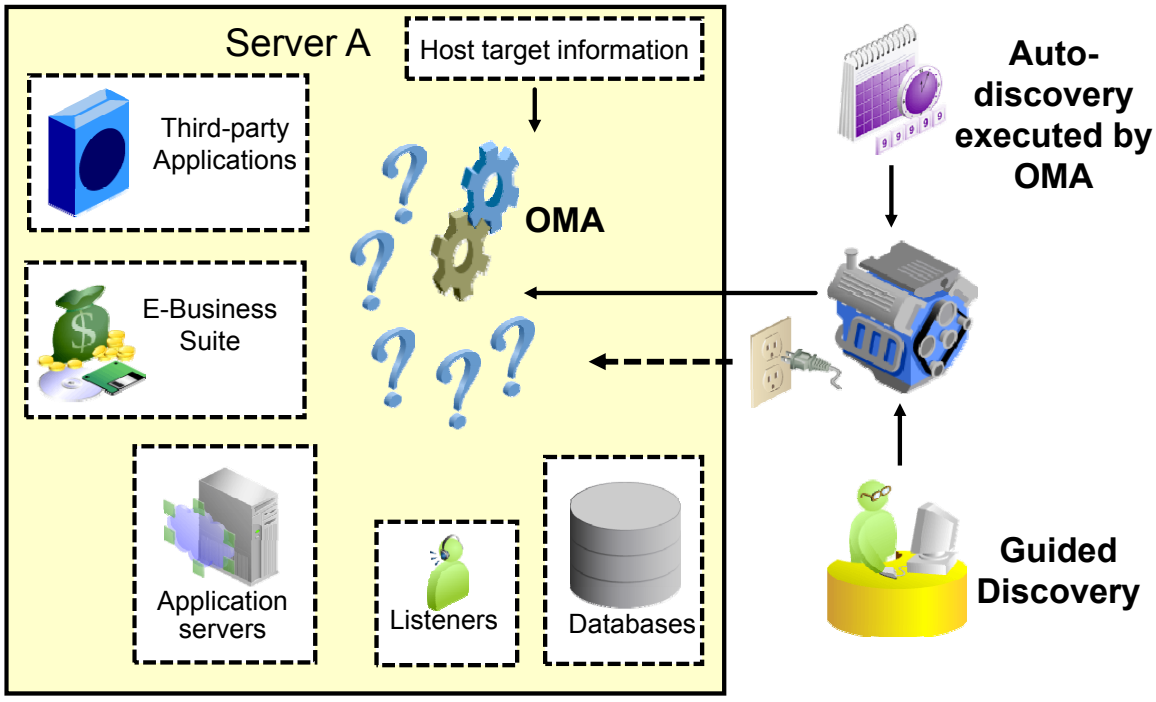
- Oracle Databases
- Oracle Database Listener
- Oracle Fusion Middleware products
- Oracle Application Server
- Oracle WebLogic Server
- Oracle applications, including E-Business Suite, SOA, Siebel, and PeopleSoft
- Exadata and Exalogic
- Cloud Control Components: OMR and OMS
- Third-party products

ORACLE

Targets are the entities that Enterprise Manager Cloud Control manages. To do so, it uses target-type–specific plug-ins and host-specific agents.

Enterprise Manager Cloud Control can monitor, administer, maintain, and manage different types of targets as listed in the slide. As your environment changes, you can add and remove targets from Enterprise Manager Cloud Control as needed. The commonly used Oracle targets (including Enterprise Manager Cloud Control components, such as the OMR and OMS) are predefined as part of the base Enterprise Manager Cloud Control product, but Enterprise Manager Cloud Control has an open API that enables you to create custom targets.

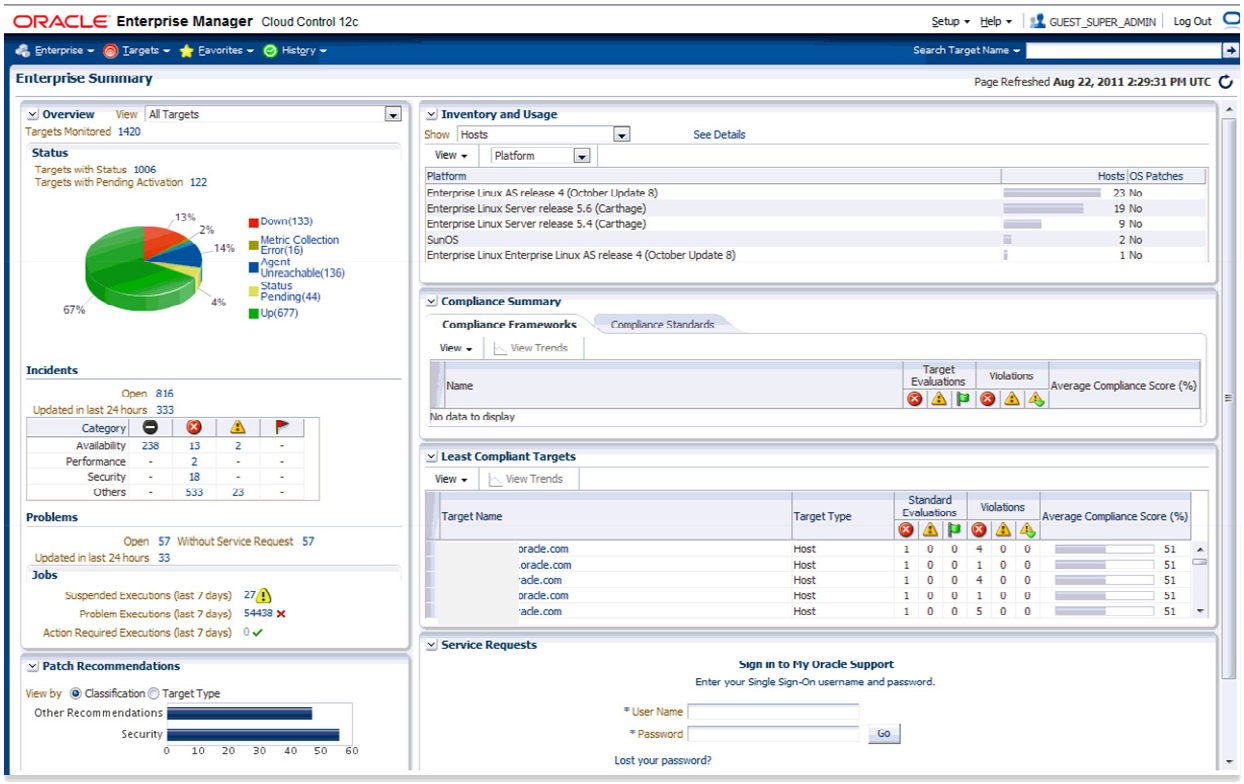
Target Discovery



After an agent has been installed on a host, it needs to look for targets that it can manage. As an Enterprise Manager Cloud Control administrator, you can guide that process from the Enterprise Manager Cloud Control console pages. Guided discovery allows you to nominate a family of target types that you want to search for, such as database and listeners, and then the agents where you want that search to be executed. If any new targets are discovered, the appropriate plug-in will be pushed from the OMS if it is not already installed on the agent, the target will be recorded in the OMR, and monitoring will begin.

You can also configure auto-discovery to run at regular intervals and get an agent to search for known targets unattended, allowing you to review the results at a later stage and promote discovered targets to become managed targets.

Enterprise Manager Cloud Control



The image in the slide is of the Enterprise Summary page of Oracle Enterprise Manager Cloud Control. The user interface (UI) functionality includes:

- Information displayed in graphs and tables
- Summary information with drilldown capability to relevant details
- User-selected home page from a predefined set, or based on any page in the console
- Menu-driven navigation
- Global target search
- History and favorites
- Customizable target home pages (per-user basis)

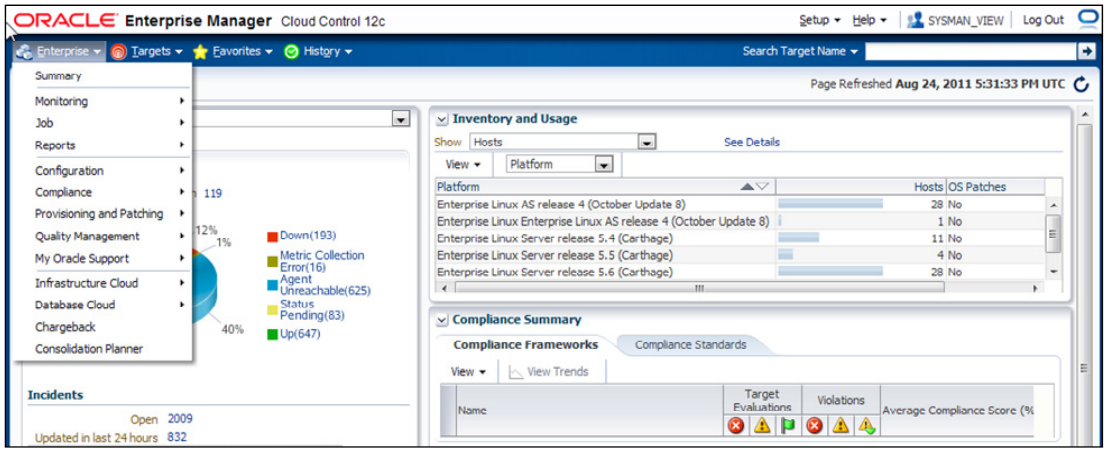
User Interface

Setting your home page:

- Predefined home page based on roles
- Any page

Menu-based navigation:

- Make any page a favorite for quick access.



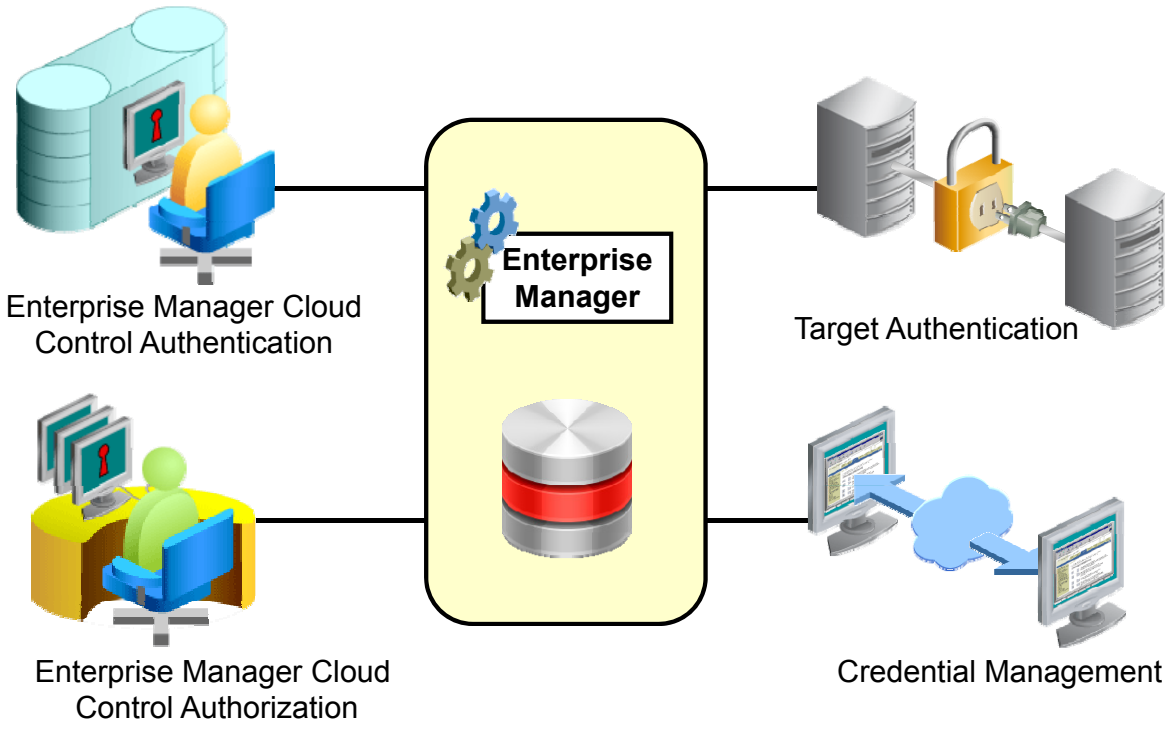
User Interface Enhancements

The user interface in Enterprise Manager Cloud Control has been rewritten in Application Development Framework (ADF). When you log in to the new UI, use drop-down menus to navigate from one place to another in the product.

- Choose your own home page: When you first log in to Enterprise Manager, you are provided with a selection of predefined home pages based on roles. If you are managing databases, you can choose the database home page.
  If those are not suitable, you can select any page in the UI to be your home page instead.
- Mark any page as a “favorite” for quick access. Because you manage certain targets frequently, you mark these target home pages as favorites in much the same way you mark a favorite in a browser. However, because the favorites you mark in Enterprise Manager are stored in the repository, you can move from client machine to client machine and your favorites are still available to you.

For information about how to customize your Enterprise Manager Cloud Control console, follow the demonstration *Oracle Enterprise Manager 12c: Console Overview and Customization* in the Oracle Learning Library.

Security: Overview



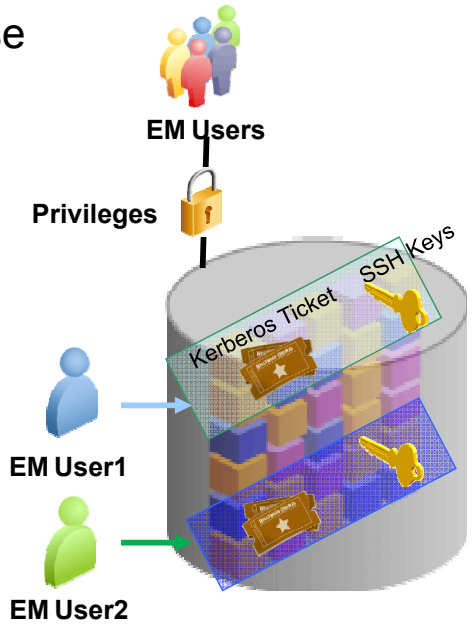
Enterprise Ready Framework: Security

The Enterprise Manager security system can be divided into four parts (as shown in the graphic):

- Enterprise Manager Cloud Control authentication
- Enterprise Manager Cloud Control authorization and privileges
- Credential management (host credentials can be defined here)
- Target authentication (host credentials can be used here)

Managing Securely with Credentials

- Centralized credential store for ease of management
- Support for managing password-less and strong authentication credentials
  - Kerberos tickets
  - SSH keys
- Reuse and sharing among users (without disclosing the sensitive content of credentials)
- Controlled and protected access
- Support for Sudo/PowerBroker



Security with Credentials

As a management tool that handles a lot of scripts and powerful actions like patching, Enterprise Manager has to work with a lot of credentials for hosts, databases, and a range of other objects. Managing all these credentials can be a real challenge.

- The centralized store facilitates this task because you can name and store credentials there.
- Password-less and strong authentication credentials are supported, such as the Kerberos tickets and SSH key pairs.
- Credentials can be reused and shared among users (without disclosing sensitive content, such as a password). Users are granted access to these credentials by the use of privileges, and so they can be reused without knowing what the contents of the credentials themselves are.
- Access to the credentials is controlled and protected by privileges.
- The Enterprise Manager credential subsystem enables you to securely store credentials as preferences or operation credentials, which can then be used to perform different system management activities. Enterprise Manager also supports Sudo/PowerBroker–based impersonation.

Distinguishing Credentials

- Named credentials
- Preferred credentials
- Default credentials
- Access level:
  - **View:** access to use the credential
  - **Edit:** to change the credential (including changing its name and password)
  - **Full:** for complete access (including the ability to delete the credential)
- Usage classification: job, collection, and monitoring

Types of Credentials

As the Enterprise Manager administrator, you can also store credentials (username/password, a public key-private key pair, or an X509v3 certificate) as **named credentials** in Enterprise Manager to use when performing operations like running jobs, patching, and other system management tasks. Objects refer or point to named credentials. They are “placeholders” to facilitate, for example, the changing of passwords.

You can store, access, and modify a fixed number of username/password–based credentials as **preferred credentials** to simplify access to managed targets by storing target login credentials in the Management Repository.

Default credentials can be set for a particular target type and will be available for all the targets of the target type.

The three levels of access that can be granted are:

- **View access:** To use the credentials
- **Edit access:** To change the credentials, including changing its name and password
- **Full access:** For complete access, including the ability to delete the named credential

Credentials can also be classified by their usage, such as job credentials (used by the job system), collection credentials, and monitoring credentials (used by OMA).

Core concepts and definitions:

- **Credential type** is the type of authentication supported by a target type. For example, a host can support a username/password–based authentication, public key authentication, or Kerberos authentication. Various authentication schemes are supported, including native agent authentication and SSH.
- **Credential set** is a placeholder for a credential. Credential sets can be used to decouple credentials from the system that uses a credential. A credential set enables you to change its mapping to named credentials for a target without editing the system that uses the credential. For example, you could have a credential set for patching tasks.
- **Credential store** is a logical store for all the named credentials of an Enterprise Manager administrator.

Defining credentials by:

- **Credential name:** The credential is referenced using the name of the credential in the credential store.
- **Credential set:** The credential is referenced using the credential set name and the target name. The lookup gets the credential associated with the credential set name and target name.
- **Direct value:** The credential is specified by providing the values of the attributes. This reference does not refer to a credential in the credential store.

For information about how to set credentials, follow the demonstrations:

- *Oracle Enterprise Manager 12c: Create and Use Named Credentials* in the Oracle Learning Library
- *Oracle Enterprise Manager 12c: Create SSH Key Named Credentials* in the Oracle Learning Library
- OBE *Enterprise Ready Framework: Create and Use Credentials*

Quiz

Which targets can be managed by using Enterprise Manager Cloud Control?

a. Hosts
b. Databases
c. Application servers
d. Web applications
e. OMS and OMR
f. All of the above

Answer: f

Practices

Videos cover the following topics:

- B-1: Accessing Enterprise Manager Cloud Control
- B-2: Setting the Summary page as the home page
- B-3: Adding a database instance as a new target monitored by Cloud Control
- B-4: Creating a new named credential
- B-5: Using the named credential

ORACLE

View the “Oracle Enterprise Manager 12*c*: Console Overview and Customization” demonstration (unless your instructor just demonstrated those topics) (8 minutes).

Optional demonstrations about related topics are available:

- Oracle Enterprise Manager 12*c*: Create an Enterprise Manager Administrator (6 minutes)
- Oracle Enterprise Manager 12*c*: Create and Use Named Credentials (6 minutes)
- Oracle Enterprise Manager 12*c*: Create SSH Key Named Credentials (3 minutes)
- Oracle Enterprise Manager 12*c*: Discover and Promote Unmanaged Hosts and Targets

Optional Oracle by Example (OBE) tutorials about related topics are available:

- Oracle Enterprise Manager 12*c* Enterprise Ready Framework: Create and Use Credentials (60 minutes)
- Oracle Enterprise Manager 12*c* Enterprise Ready Framework: Create a Super Administrator Account (5 minutes)