Oracle® Database Vault

Database Vault Getting Started Guide

23ai
F97756-01
July 2024

Oracle Database Vault Database Vault Getting Started Guide, 23ai

F97756-01

Contents

Preface

This quick start guide will help you configure Oracle Database Vault on a single instance database.

For more advanced configurations, such as those using Oracle Real Application Clusters or Oracle Data Guard, see the *Oracle Database Vault Administrator's Guide*.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customer access to and use of Oracle support services will be pursuant to the terms and conditions specified in their Oracle order for the applicable services.

Related Documents

The following documents contain information that may be of use when getting started with Database Vault

- *Oracle Database Vault Administrator's Guide*
- *Oracle Database Security Guide*
- *Oracle Label Security Administrator's Guide*
- *Oracle Database Administrator's Guide*
- *Oracle Database SQL Language Reference*
- *Oracle Multitenant Administrator's Guide*

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1

Overview of Oracle Database Vault

Oracle Database Vault offers data security controls within an Oracle database to restrict access to application data by privileged users.

Database Vault can help you reduce the risk of insider and outside threats, address compliance requirements, including separation of duties, as well as minimize the potential of human error on application data and objects. Database Vault is built into the kernel of the Oracle database, thus ensuring you are implementing your security controls as close to the data as possible.

In this quick start guide, you will learn how to configure and enable Oracle Database Vault, how to use Database Vault realms and command rules to protect data in the sample schema `HR`, and how to prevent the use of destructive commands, such as `DROP TABLE` on `HR` objects. This guide will also show you how to configure Unified Audit policies to audit violations of your realms and command rules.

Oracle Database Vault can do much more than the examples this quick guide describes. For more information of Oracle Database Vault, refer to the *Oracle Database Vault Administrator's Guide*.

2

Configuring and Enabling Oracle Database Vault

To get started with Oracle Database Vault, you must configure it, enable it, and then restart the Oracle database. In an Oracle multitenant environment, you complete this on the container database (CDB) and then each of the pluggable databases (PDB).

Oracle recommends creating four accounts to manage key roles within Oracle Database Vault. Two are primary accounts and two are backup accounts. Store the passwords for these accounts in a safe place so that you do not lose or forget them. The accounts with the `DV_OWNER` role, specifically, become your most critical accounts to never lock and never lose the passwords to. To minimize the risk of locking the accounts, Oracle recommends creating specific database profiles to assign to these four accounts.

2.1 Configuring Database Vault on the Container Database

Perform the following steps to configure Database Vault on the container database.

1. Connect as a database user that can create accounts and grant privileges in both the container root and pluggable database:

   ```
   connect / as sysdba
   ```

2. Grant the following privileges:

   ```
   GRANT CREATE SESSION, SET CONTAINER TO c##dvowner
       IDENTIFIED BY <password> CONTAINER = ALL;
   GRANT CREATE SESSION, SET CONTAINER TO c##dvowner_backup
       IDENTIFIED BY <password> CONTAINER = ALL;
   GRANT CREATE SESSION, SET CONTAINER TO c##dvacctmgr
       IDENTIFIED BY <password> CONTAINER = ALL;
   GRANT CREATE SESSION, SET CONTAINER TO c##dvacctmgr_backup
       IDENTIFIED BY <password> CONTAINER = ALL;
   ```

3. Oracle recommends creating a profile that will not permanently lock the accounts you created for Oracle Database Vault. You can adjust these to meet your requirements, but Oracle recommends using the `PASSWORD_LOCK_TIME` profile parameter to an acceptable length to allow the account to be unlocked after a reasonable time has passed. In this example, the following parameter is set to one minute:

   ```
   CREATE PROFILE C##DV_PROFILE LIMIT
       FAILED_LOGIN_ATTEMPTS 5
       PASSWORD_VERIFY_FUNCTION ora12c_verify_function
       PASSWORD_LOCK_TIME 1/1440
       CONTAINER=ALL;
   ```

4. Assign the profile to the Oracle Database Vault accounts you created:

```
ALTER USER c##dvowner PROFILE c##dv_profile CONTAINER=ALL;
ALTER USER c##dvowner_backup PROFILE c##dv_profile CONTAINER=ALL;
ALTER USER c##dvacctmgr PROFILE c##dv_profile CONTAINER=ALL;
ALTER USER c##dvacctmgr_backup PROFILE c##dv_profile CONTAINER=ALL;
```

5. Perform the configuration on the container database:

```
connect / as sysdba

BEGIN
    CONFIGURE_DV (
        dvowner_uname         => 'c##dvowner',
        dvacctmgr_uname       => 'c##dvacctmgr',
        force_local_dvowner   => FALSE);
END;
/
```

6. Recompile invalid objects. This step is not required but it's recommended to keep invalid objects to a minimum.

```
@$ORACLE_HOME/rdbms/admin/utlrp.sql
```

7. As `C##DVOWNER`, enable Oracle Database Vault:

```
CONNECT c##dvowner
EXEC DBMS_MACADM.ENABLE_DV;
```

8. Restart the Oracle database. If you are in an Oracle Real Application Cluster, you can minimize the downtime by using Oracle Real Application Clusters to perform a rolling enablement, but in a single instance environment you must take the downtime.

```
CONNECT / as sysoper
shutdown immediate;
startup;
```

9. From the container database, verify Oracle Database Vault is configured and enabled in the CDB root:

```
connect / as sysdba

SELECT * FROM DBA_DV_STATUS;
```

Both `DV_CONFIGURE_STATUS` and `DV_ENABLE_STATUS` should show `TRUE` and `APP_PROTECTION` will show `NOT CONFIGURED` because you have not enabled Oracle Database Vault operations control.

For example:

```
NAME                 STATUS
-------------------- --------------
DV_CONFIGURE_STATUS           TRUE
```

```
DV_ENABLE_STATUS              TRUE
DV_APP_PROTECTION    NOT CONFIGURED
```

10. Grant the backup accounts the appropriate Database Vault roles.
For example, the user who is the backup for Oracle Database Vault Owner has the `DV_OWNER` role:

```
CONNECT c##dvowner
GRANT DV_OWNER TO c##dvowner_backup WITH ADMIN OPTION CONTAINER=ALL;
```

11. Grant the backup accounts the appropriate Database Vault roles.
For example, the user who is the backup for Oracle Database Vault Account Manager has the `DV_ACCTMGR` role:

```
CONNECT c##dvacctmgr
GRANT DV_ACCTMGR TO c##dvacctmgr_backup WITH ADMIN OPTION CONTAINER=ALL;
```

12. To verify the Database Vault related users have the appropriate roles, query the role privileges view as a user who has the DBA role:

```
connect / as sysdba

column grantee format a25
column granted_role format a25
select grantee, granted_role, admin_option, common
    from dba_role_privs
where granted_role in ('DV_ACCTMGR','DV_OWNER')
order by 1,2,3;
```

The output should look like the following:

```
GRANTEE                   GRANTED_ROLE     ADMIN_OPTION     COMMON
_________________________ ________________ ________________ _________
C##DVACCTMGR              DV_ACCTMGR       YES              YES
C##DVACCTMGR_BACKUP       DV_ACCTMGR       YES              YES
C##DVOWNER                DV_OWNER         YES              YES
C##DVOWNER_BACKUP         DV_OWNER         YES              YES
```

2.2 Configuring Database Vault on a Pluggable Database

Perform the following steps to configure Database Vault on a pluggable database. You can enable Database Vault on zero or more pluggable databases.

The following configuration will use common database users and grant them the `DV_OWNER` and `DV_ACCTMGR` roles, respectively. This is a common configuration however, not the only configuration. You may create, and use, database users local to the pluggable database instead of common users. For more information on the different configuration options, see the *Oracle Database Vault Administrator's Guide*.

1. On the pluggable database, as a user with the `SYSDBA` administrative privilege, perform the configuration:

```
alter session set container=pdb_name;
```

```
BEGIN
    CONFIGURE_DV (
        dvowner_uname         => 'c##dvowner',
        dvacctmgr_uname       => 'c##dvacctmgr');
END;
/
```

2. Recompile invalid objects. This step is not required but it's recommended to keep invalid objects to a minimum.

```
$ORACLE_HOME/rdbms/admin/utlrp.sql
```

3. Next, as `C##DVOWNER`, enable Oracle Database Vault in the pluggable database:

```
CONNECT c##dvowner@pdb_name

EXEC DBMS_MACADM.ENABLE_DV;
```

4. As a user with the `SYSOPER` role, restart the Oracle pluggable database. If you are in an Oracle Real Application Cluster, you can minimize the downtime by performing rolling enablement, but in a single instance environment you must take the downtime.

```
connect / as sysoper

ALTER PLUGGABLE DATABASE pdb_name CLOSE IMMEDIATE;
ALTER PLUGGABLE DATABASE pdb_name OPEN;
```

5. If you wish to check the status of Oracle Database Vault in all container and pluggable databases, you can run the following command from the container database. Now, Database Vault should be enabled on the container and pluggable databases.

```
connect / as sysdba

SELECT * FROM CDB_DV_STATUS;
```

The output should look like the following. Notice the red highlighted rows, demonstrating both the container and pluggable database have Oracle Database Vault enabled:

```
NAME                          STATUS               CON_ID
_____________________         _________________    _________
DV_CONFIGURE_STATUS           TRUE                 1
DV_ENABLE_STATUS              TRUE                 1
DV_APP_PROTECTION             NOT CONFIGURED       1
DV_CONFIGURE_STATUS           TRUE                 3
DV_ENABLE_STATUS              TRUE                 3
DV_APP_PROTECTION             NOT CONFIGURED       3
```

Alternatively, if you want to see the container name instead of the container ID, run the following query:

```
SELECT CON_ID_TO_CON_NAME(CON_ID) CON_NAME, NAME, STATUS FROM
CDB_DV_STATUS;
```

The output should look like the following:

```
NAME                           STATUS                 CON_ID
_____________________________  _____________________  _________
CDB$ROOT                       DV_CONFIGURE_STATUS    TRUE
CDB$ROOT                       DV_ENABLE_STATUS       TRUE
CDB$ROOT                       DV_APP_PROTECTION      NOT CONFIGURED
pdb_name                       DV_CONFIGURE_STATUS    TRUE
pdb_name                       DV_ENABLE_STATUS       TRUE
pdb_name                       DV_APP_PROTECTION      NOT CONFIGURED
```

For more information on the `CON_ID_TO_CON_NAME` function, review the *Oracle Database SQL Language Reference*

3
Managing Database Users

Oracle recommends using named accounts instead of the generic `SYS` and `SYSTEM` accounts. This section explains how to create database users and separate duties through roles, realms, command rules, and authorizations.

Once Oracle Database Vault is enabled, separation of duties is enforced through roles, realms, command rules, and authorizations. For example:

- To create a user, you must have the `DV_ACCTMGR` role which is granted the `CREATE USER` system privilege.
- To protect a schema from database users granted `SELECT ANY TABLE` system privilege, create a realm to limit access to only authorized users.
- To stop a user with the `DROP TABLE` privilege from performing this action, create a command rule on `DROP TABLE`.
- To perform an Oracle Data Pump import of a table that is protected by a Database Vault realm or command rule, you must have the ability to import with Oracle Data Pump and be granted the Database Vault authorization.

For more information on these controls, see the *Oracle Database Vault Administrator's Guide*.

After you have enabled Oracle Database Vault, you will see that `SYS` is no longer able to perform certain actions. This is intentional because `SYS` should not be an account used except for patching, upgrading, and special circumstances. The `SYSTEM` account is also an account that should not be used unless necessary. `SYSTEM` is a highly privileged account that is difficult to assign to a single user. Oracle recommends using named accounts (for example, jsmith, cmack, gkramer, and so on) instead of shared, or generic, accounts.

For example, named accounts can be set up like those in the following table to ensure separation of duties. You will learn how to set up these accounts in the following topic.

Table 3-1 Example Named Accounts

Username	Location	Responsibilities
C##DVOWNER	CDB & PDBs	Database Vault owner
C##DVOWNER_BACKUP	CDB & PDBs	Database Vault owner backup account
C##DVACCTMGR	CDB & PDBs	Database Vault account management
C##DVACCTMGR_BACKUP	CDB & PDBs	Database Vault account management backup account
C##JSMITH	CDB & PDBs	DBA, Database Vault owner and account manager
C##CMACK	CDB & PDBs	Audit administration
GKRAMER	PDB	DBA
HR	PDB	Application owner

Oracle Database Vault attempts to protect database user accounts from being misused or abused by privileged users. Once you have configured and enabled Oracle Database Vault,

you must have the Oracle Database Vault role DV_ACCTMGR to create a user. This applies to the ALTER USER and the DROP USER system privileges as well as PROFILE management system privileges.

3.1 Creating Named Database Accounts

Learn how to create named database accounts to replace the generic, SYS and SYSTEM, accounts.

Prerequisites

Have an account that has been granted the DV_ACCTMGR role, such as the C##DVACCTMGR user that you created during Configuring Database Vault on the Container Database. This user should have the privileges to create accounts and grant the DV_ACCTMGR role to other users.

1. Connect as a user that has been granted the DV_ACCTMGR role:

   ```
   connect c##dvacctmgr
   ```

2. Create a named user account, C##JSMITH and grant them the DV_ACCTMGR role:

   ```
   CREATE USER C##JSMITH IDENTIFIED BY <password> CONTAINER=ALL;
   GRANT DV_ACCTMGR TO C##JSMITH WITH ADMIN OPTION CONTAINER=ALL;
   ```

3. Connect as a user that has been granted the DV_OWNER role:

   ```
   connect c##dvowner
   ```

4. Grant JSMITH the role of DV_ADMIN with the ability to pass the role on to other users:

   ```
   GRANT DV_ADMIN TO C##JSMITH WITH ADMIN OPTION CONTAINER=ALL;
   ```

   Granting DV_ADMIN WITH ADMIN OPTION will allow JSMITH to create, manage and delete policies, realms, command rules, rules and rule sets but not disable Oracle Database Vault. These privileges are a subset of the privileges granted to DV_OWNER.

   Granting this role to a named account should allow you to securely store the shared accounts (C##DVOWNER, C##DVOWNER_BACKUP, C##DVACCTMGR, C##DVACCTMGR_BACKUP) and only use them for emergencies. Day-to-day operations should be completed by database users who are using their own named credentials.

5. Connect as SYSDBA user:

   ```
   connect / as sysdba
   ```

6. Grant the DBA role to C##JSMITH and include WITH ADMIN OPTION so the user can forward-grant privileges to other database users:

   ```
   GRANT DBA TO C##JSMITH WITH ADMIN OPTION CONTAINER=ALL;
   GRANT RESOURCE TO C##JSMITH WITH ADMIN OPTION CONTAINER=ALL;
   GRANT AUDIT_ADMIN TO C##JSMITH WITH ADMIN OPTION CONTAINER=ALL;
   ```

> **Note:**
>
> Oracle recommends creating a subset of system and object privileges in a custom role, rather than using the `DBA` role.

4
Oracle Database Vault Default Realms

Oracle Database Vault Realms restrict what actions users can take on the database.

In addition to Oracle Database Vault's goal to protect your sensitive data, reduce human error, and limit insider threats, Oracle Database Vault protects components of the Oracle Database dictionary and its own dictionary objects.

There are times when, to access certain objects, perform certain administrative tasks, or grant certain Oracle database roles, you must be authorized by Database Vault to do so.

Oracle Database Vault provides two types of realms: traditional and mandatory. Both realm types can protect either an entire schema, individual database roles or crucial objects within a schema selectively, such as tables and indexes.

Traditional realms will respect database users using their direct-object grants.

Mandatory realms require the grant, either direct or granted through a role, and the Database Vault realm authorization.

In this example, you will authorize `C##JSMITH` to forward-grant the resource role they have been granted. To do so, you must add `C##JSMITH` to a Database Vault default realm.

4.1 Traditional Realms

With a traditional realm, an object owner or users who has been granted object privileges can perform queries or DML operations without realm authorization but must have realm authorization to perform DDL operations.

4.1.1 Demonstrating Realm Violations

Because Oracle Database Vault Realms restrict actions that a user can take on the database, some actions will fail without proper authorization. This example shows how the `C##JSMITH` user is unable to grant the `RESOURCE` role because they are not in the Oracle System Privilege and Role Management Realm.

1. Attempt to create the `C##CMACK` user and grant them the `RESOURCE` role:
   a. Connect as `C##JSMITH`:

      ```
      connect c##jsmith
      ```

   b. Create `C##CMACK` role and grant them the `RESOURCE` role:

      ```
      CREATE USER C##CMACK IDENTIFIED BY <password> CONTAINER=ALL;
      GRANT RESOURCE TO C##CMACK WITH ADMIN OPTION CONTAINER=ALL;
      ```

You will receive the following error as C##JSMITH:

```
ORA-47410: Insufficient realm privileges to GRANT on RESOURCE
```

2. Identify which realm is protecting the RESOURCE role:
   a. Connect as C##JSMITH:

```
connect c##jsmith
```

   b. Run the following:

```
SELECT REALM_NAME, OBJECT_NAME
    FROM DBA_DV_REALM_OBJECT
    WHERE OBJECT_TYPE = 'ROLE'
        AND OBJECT_NAME = 'RESOURCE'
    ORDER BY 1,2;
```

   You should see the following output:

```
REALM_NAME                                                  OBJECT_NAME
___________________________________________________________ _____________
Oracle System Privilege and Role Management Realm           RESOURCE
```

3. Identify which database users are authorized to access the realm protected objects and, more specifically, which users have the OWNER realm authorization which will allow them to forward-grant privileges on objects they hold WITH ADMIN OPTION privileges:

```
SELECT GRANTEE, AUTH_OPTIONS
    FROM DBA_DV_REALM_AUTH
    WHERE REALM_NAME = 'Oracle System Privilege and Role Management Realm'
    ORDER BY 1;
```

   You should see the following output:

```
GRANTEE       AUTH_OPTIONS
_____________ _______________
SYS           Owner
```

You should not use SYS unless it is necessary. Instead, you should authorize C##JSMITH to the Oracle System Privilege and Role Management Realm as an owner, thus allowing C##JSMITH to grant the resource role to other database users.

You have completed the steps to authorize C##JSMITH to the Oracle System Privilege and Role Management Realm as an owner, thus allowing C##JSMITH to grant the resource role to other database users.

4.1.2 Granting Realm Authorization

Authorize a named user to the Oracle System Privilege and Role Management Realm as an owner, allowing the user to grant the resource role to other database users. If you are granting privileges to a database user in all database containers, authorize a named user on each of the pluggable databases (PDB) where Oracle Database Vault is enabled.

For example, add C##JSMITH to the Database Vault realm:

- On a Container Database
- On a Pluggable Database

On a Container Database

1. Connect as C##JSMITH:

```
connect c##jsmith
```

2. Add C##JSMITH to the realm:

```
BEGIN DVSYS.DBMS_MACADM.ADD_AUTH_TO_REALM (
    realm_name     => 'Oracle System Privilege and Role Management Realm'
    ,grantee        => 'C##JSMITH'
    ,rule_set_name  => null
    ,auth_options   => DBMS_MACUTL.G_REALM_AUTH_OWNER
END;
/
```

3. Confirm that C##JSMITH has been added to the realm:

```
SELECT GRANTEE, AUTH_OPTIONS
    FROM DBA_DV_REALM_AUTH
    WHERE REALM_NAME = 'Oracle System Privilege and Role Management Realm'
    ORDER BY 1;
```

You should see the following output:

```
GRANTEE       AUTH_OPTIONS
_____________ _______________
C##JSMITH     Owner
SYS           Owner
```

On a Pluggable Database

1. Connect as C##JSMITH on the PDB:

```
connect c##jsmith@pdb_name
```

2. Add C##JSMITH to the realm:

```
BEGIN DVSYS.DBMS_MACADM.ADD_AUTH_TO_REALM (
    realm_name     => 'Oracle System Privilege and Role Management Realm'
    ,grantee        => 'C##JSMITH'
    ,rule_set_name  => null
    ,auth_options   => DBMS_MACUTL.G_REALM_AUTH_OWNER
    ,auth_scope     => DBMS_MACUTL.G_SCOPE_LOCAL);
END;
/
```

3. Confirm that C##JSMITH has been added to the realm:

```
SELECT GRANTEE, AUTH_OPTIONS
    FROM DBA_DV_REALM_AUTH
    WHERE REALM_NAME = 'Oracle System Privilege and Role Management Realm'
    ORDER BY 1;
```

You should see the following output:

```
GRANTEE       AUTH_OPTIONS
_____________ _______________
C##JSMITH     Owner
SYS           Owner
```

> **Tip:**
>
> An advanced configuration would be to create a common role and add that role to the realm authorized owners list instead of individual database usernames.
> For example, create C##ACME_DBA role and add it to Oracle System Privilege and Role Management Realm as an authorized owner. You can then create a Database Vault realm to protect who can grant the C##ACME_DBA role.

4.1.3 Granting RESOURCE Role to Other Users

After a named user has the privileges and realm authorization, they can grant the RESOURCE to other users.

For example, as C##JSMITH, grant the RESOURCE role to C##CMACK:

1. Connect as C##JSMITH:

```
connect c##jsmith
```

2. Create C##CMACK role and grant them the RESOURCE role:

```
CREATE USER C##CMACK IDENTIFIED BY <password> CONTAINER=ALL;
GRANT RESOURCE TO C##CMACK WITH ADMIN OPTION CONTAINER=ALL;
```

3. As C##JSMITH, create the GRKAMER account in the pluggable database:

```
ALTER SESSION SET CONTAINER=pdb_name;
CREATE USER GKRAMER IDENTIFIED BY <password>;
GRANT CREATE SESSION TO GKRAMER;
```

4. Use the C##JSMITH database user to grant several system privileges to GRKAMER:

```
GRANT SELECT ANY TABLE, UPDATE ANY TABLE, CREATE ANY TABLE, DROP ANY TABLE
TO GKRAMER;
```

```
GRANT CREATE ANY INDEX, DROP ANY INDEX TO GKRAMER;
GRANT RESOURCE TO GKRAMER;
```

For additional information on default realms and command rules, refer to the *Oracle Database Vault Administrator's Guide*.

4.2 Mandatory Realms

A mandatory realm provides stronger protection than traditional realms for objects within a realm. Mandatory realms block both object privilege and system privilege access and will not allow users with object privileges to perform queries, DML, or DDL operations without realm authorization. In other words, if the objects are protected by mandatory realms, even the object owner cannot access their own objects without proper realm authorization.

4.2.1 Querying Application Data Before Applying Realms and Command Rules

Prior to applying realms and command rules, you can query the `HR.EMPLOYEES` table to see what actions users are able to perform. After applying realms and command rules the results will be different.

Query the `HR.EMPLOYEES` table as each of the following user and compare the results to the table below:

- `SYS`
- `SYSTEM`
- `C##DVOWNER`
- `C##DVACCTMGR`
- `C##JSMITH`
- `C##CMACK`
- `GKRAMER`
- `HR`

1. Connect as `<user>`:

   ```
   connect <user>
   ```

2. Attempt the following commands:

   ```
   SELECT COUNT(*) FROM HR.EMPLOYEES;
   CREATE TABLE HR.EMP2 AS SELECT * FROM HR.EMPLOYEES;
   CREATE INDEX HR.TEST_IDX ON HR.EMP2(HIRE_DATE,LAST_NAME);
   DROP INDEX HR.TEST_IDX;
   DROP TABLE HR.EMP2;
   ```

Command	SYS	SYSTEM	C##DVOWNER	C##DVACCTMGR	C##JSMITH	C##CMACK	GKRAMER	HR
SELECT COUNT(*) FROM HR.EMPLOYEES;	Success	Success	ORA-00942: table or view does not exist	ORA-00942: table or view does not exist	Success	ORA-00942: table or view does not exist	Success	Success
CREATE TABLE HR.EMP2 AS SELECT * FROM HR.EMPLOYEES;	Success	Success	ORA-00942: table or view does not exist	ORA-00942: table or view does not exist	Success	ORA-00942: table or view does not exist	Success	Success
CREATE INDEX HR.TEST_IDX ON HR.EMP2(HIRE_DATE,LAST_NAME);	Success	Success	ORA-00942: table or view does not exist	ORA-00942: table or view does not exist	Success	ORA-00942: table or view does not exist	Success	Success
DROP INDEX HR.TEST_IDX;	Success	Success	ORA-01418: specified index does not exist	ORA-01418: specified index does not exist	Success	ORA-01418: specified index does not exist	Success	Success
DROP TABLE HR.EMP2;	Success	Success	ORA-00942: table or view does not exist	ORA-00942: table or view does not exist	Success	ORA-00942: table or view does not exist	Success	Success

If your results be different from the table, verify you have configured and enabled Oracle Database Vault in the container database and created the users and granted the appropriate privileges.

Notice the Database Vault related accounts `C##DVOWNER` and `C##DVACCTMGR` do not have the system or object privileges required to access the HR.EMPLOYEES table. These accounts are not intended to access data, only perform Database Vault administrative activities

4.2.2 Creating a Mandatory Realm to Protect Tables and Views

Mandatory realms require the grant, either direct or granted through a role, and the Database Vault realm authorization. By creating a mandatory realm, only users who have the appropriate system or object privileges and realm authorization can access the specified tables and views.

Prerequisites

Have an account that has been granted the `DV_ACCTMGR` or `DV_OWNER`role, such as the `C##JSMITH` user that you created during Creating Named Database Accounts.

1. Connect as `C##JSMITH` on the pluggable database:

```
connect c##jsmith@pdb_name
```

2. Create the mandatory realm:

```
BEGIN
    DVSYS.DBMS_MACADM.CREATE_REALM(
        realm_name     => 'Protect HR tables'
        ,description    => 'Mandatory realm to protect HR tables'
        ,enabled        => dbms_macutl.g_yes
        ,audit_options  => null
        ,realm_type     => dbms_macadm.mandatory_realm);
END;
/
```

> **Note:**
>
> If you are using Oracle Database 19c or earlier, and traditional auditing, you can specify a value for `audit_options` instead of null. For more information, see the *Oracle Database Vault Administrator's Guide* for Oracle Database 19c.

3. Add the objects you wish to protect with the new realm. You could specify a wildcard (%) for the `OBJECT_TYPE` parameter and protect all existing and new objects in a schema.
In this example, you will separate `TABLES` and `VIEWS` into a single realm, and `INDEXES` into a separate realm in the following example.

```
BEGIN
    DVSYS.DBMS_MACADM.ADD_OBJECT_TO_REALM(
        realm_name     => 'Protect HR tables'
        ,object_owner   => 'HR'
        ,object_name    => '%'
        ,object_type    => 'TABLE');
END;
/

BEGIN
    DVSYS.DBMS_MACADM.ADD_OBJECT_TO_REALM(
        realm_name     => 'Protect HR tables'
        ,object_owner   => 'HR'
        ,object_name    => '%'
        ,object_type    => 'VIEW');
END;
/
```

4. Authorize the users to access the realm-protected tables and views. In a mandatory realm you will also authorize the object owner.
In this example, you will authorize `HR` to access its own objects.

```
BEGIN
    DVSYS.DBMS_MACADM.ADD_AUTH_TO_REALM(
        realm_name     => 'Protect HR tables'
        ,grantee        => 'HR'
```

```
        ,rule_set_name => ''
        ,auth_options  => dbms_macutl.g_realm_auth_owner);
END;
/
```

5. Determine if you will be enforcing the realm or not:

   - Simulation (non-enforcement) mode: Oracle Database Vault simulation mode allows you to identify which users are accessing data you want to protect with an Oracle Database Vault realm. Only violations of the Database Vault realm authorization list will be recorded.
     Set the `ENABLED` parameter to `DBMS_MACUTL.G_SIMULATION`

     ```
     BEGIN
         DVSYS.DBMS_MACADM.UPDATE_REALM(
             realm_name     => 'Protect HR tables'
             ,description   => 'Mandatory realm to protect HR tables'
             ,enabled       => dbms_macutl.g_simulation
             ,audit_options => null
             ,realm_type    => dbms_macadm.mandatory_realm);
     END;
     /
     ```

   - Enforcement mode: Oracle Database Vault enforcement mode means the mandatory realm will enforce the controls on the objects it is protecting and allow database users access if they maintain the appropriate system or object privileges and authorization to the realm.
     Set the `ENABLED` parameter to `DBMS_MACUTL.G_YES`

     ```
     BEGIN
         DVSYS.DBMS_MACADM.UPDATE_REALM(
             realm_name     => 'Protect HR tables'
             ,description   => 'Mandatory realm to protect HR tables'
             ,enabled       => dbms_macutl.g_yes
             ,audit_options => null
             ,realm_type    => dbms_macadm.mandatory_realm);
     END;
     /
     ```

For more information on Oracle Database Vault simulation mode, see the Troubleshooting and Tracing Errors section in the *Oracle Database Vault Administrator's Guide*.

4.2.3 Creating a Mandatory Realm to Protect Indexes

By separating the tables and indexes into separate realms, you can allow index maintenance without access to table data. There is still a risk because data could be accessed indirectly, but if index maintenance is a task of the DBA and this is a risk you are comfortable assuming, then this separation is a common methodology.

For example:

1. Connect as `C##JSMITH`:

   ```
   connect c##jsmith
   ```

2. Create the mandatory realm:

```
BEGIN
    DVSYS.DBMS_MACADM.CREATE_REALM(
        realm_name     => 'Protect HR indexes'
        ,description    => 'Mandatory realm to protect HR indexes'
        ,enabled        => dbms_macutl.g_yes
        ,audit_options  => null
        ,realm_type     => dbms_macadm.mandatory_realm);
END;
/
```

> **Note:**
>
> If you are using Oracle Database 19c or earlier, and traditional auditing, you can specify a value for `audit_options` instead of null. For more information, see the *Oracle Database Vault Administrator's Guide*.

3. Add the index to the realm:

```
BEGIN
    DBMS_MACADM.ADD_OBJECT_TO_REALM(
        realm_name    => 'Protect HR indexes'
        ,object_owner  => 'HR'
        ,object_name  => '%'
        ,object_type  => 'INDEX');
END;
/
```

4. Add authorization to the realm:

```
BEGIN
    DVSYS.DBMS_MACADM.ADD_AUTH_TO_REALM(
        realm_name    => 'Protect HR indexes'
        ,grantee      => 'HR'
        ,rule_set_name => null
        ,auth_options  => dbms_macutl.g_realm_auth_owner);
END;
/
```

5. Add the DBA, in this case `GKRAMER`, as a realm-authorized owner so that they can perform index creation, deletion, and updates:

```
BEGIN
    DVSYS.DBMS_MACADM.ADD_AUTH_TO_REALM(
        realm_name    => 'Protect HR indexes'
        ,grantee      => 'GKRAMER'
        ,rule_set_name => null
        ,auth_options  => dbms_macutl.g_realm_auth_owner);
END;
/
```

5
Command Rules

A command rule protects Oracle Database SQL statements that affect one or more database objects. These statements can include `SELECT`, `ALTER SYSTEM`, database definition language (DDL), and data manipulation language (DML) statements. To customize and enforce the command rule, you associate it with a rule set, which is a collection of one or more rules. The command rule is enforced at run time. Command rules affect anyone who tries to use the SQL statements it protects, regardless of the realm in which the object exists.

5.1 Creating a Command Rule to Prevent Destructive Actions

Creating a command rule is optional, but it can help protect database objects and data against malicious activity or mistakes by privileged users. Command rules can prevent destructive commands like `TRUNCATETABLE`, `DROP INDEX`, `DROP PROCEDURE`.

For example, prevent the `HR` schema from dropping a table:

1. Connect as `C##JSMITH`:

   ```
   connect c##jsmith
   ```

2. Disable `HR`'s ability to `DROP TABLE`:

   ```
   BEGIN
       DVSYS.DBMS_MACADM.CREATE_COMMAND_RULE(
           command        => 'DROP TABLE'
           ,object_owner  => 'HR'
           ,object_name   => '%'
           ,rule_set_name => 'Disabled'
           ,enabled       => dbms_macutl.g_yes);
   END;
   /
   ```

   As a result of this command rule, all users who perform the `DROP TABLE` command on `HR` schema objects will be considered in violation of the Database Vault command rule as the rule set specifies the command should be `Disabled`.

   Because the result of the `Disabled` rule set will always return false, the command rule will disable `DROP TABLE` commands on the `HR` schema for all database users, including the `HR` database schema itself.

   The `rule_set_name` of `Disabled` may appear counterintuitive because neither the Database Vault command rule nor the rule set are disabled. Instead, the `Disabled` rule set contains a rule with a rule expression that will always returns false, never allowing the command associated with the rule set to run.

You can view the rule expression SQL using the following:

```
select rule_name, rule_expr from dba_dv_rule_set_rule where rule_set_name
= 'Disabled';
```

You will see the following output:

```
RULE_NAME            RULE_EXPR
-------------------- --------------------
False                1=0
```

If you do not want to disable `DROP TABLE` commands on the `HR` schema in all situations, you can use a custom Database Vault rule set to identify when the command can be used.

3. If you want to switch the command rule to be in simulation mode, instead of enforced, set the `ENABLED` parameter to `DBMS_MACUTL.G_SIMULATION` instead of `DBMS_MACUTL.G_YES`:

```
BEGIN
    DVSYS.DBMS_MACADM.UPDATE_COMMAND_RULE(
        command        => 'DROP TABLE'
        ,object_owner  => 'HR'
        ,object_name   => '%'
        ,rule_set_name => 'Disabled'
        ,enabled       => dbms_macutl.g_simulation);
END;
/
```

Simulation (non-enforcement) mode: Oracle Database Vault simulation mode allows you to identify which users are accessing data you want to protect with an Oracle Database Vault realm. Only violations of the Database Vault realm authorization list will be recorded.

4. To identify the users who violated the command rule, query the `DVSYS.DBA_DV_SIMULATION_LOG` view.

```
select username, command, violation_type, sqltext from
DVSYS.DBA_DV_SIMULATION_LOG;
```

For more information on Oracle Database Vault simulation mode, see the Troubleshooting and Tracing Errors section in the *Oracle Database Vault Administrator's Guide*.

5.2 Creating a Command Rule That Allows Actions from Specified IP Addresses Only

You can create a command rule that allows specific commands to be performed from limited IP addresses.

For example, the `HR` schema to drop a table only if the command is run from an approved IP address:

1. Connect as `C##JSMITH`:

```
connect c##jsmith
```

2. Create a rule that meets your acceptance criteria:

```
BEGIN
    DBMS_MACADM.CREATE_RULE(
        rule_name  => 'Trusted IP Address'
        ,rule_expr => 'sys_context(''userenv'',''ip_address'') = ''<IP
Address>'' ');
END;
/
```

The rule will return `TRUE` only if the IP address of the connected user's session equals the IP address in the rule expression. This could be an `IN` list or a not equals. You could compare a hostname instead or a portion of a hostname.

3. Create a rule set:

```
BEGIN
    DVSYS.DBMS_MACADM.CREATE_RULE_SET(
        rule_set_name    => 'Trusted Rule Set'
        ,description     => 'A rule set for controlling access by IP
address'
        ,enabled         => 'Y'
        ,eval_options    => dbms_macutl.g_ruleset_eval_any
        ,audit_options   => null
        ,fail_options    => dbms_macutl.g_ruleset_fail_show
        ,fail_message    => 'Access is blocked. Contact the IT helpdesk.'
        ,fail_code       => '-20000'
        ,handler_options => dbms_macutl.g_ruleset_handler_off
        ,handler         => null
        ,is_static       => true);
END;
/
```

Rule sets have the option of returning the general error message or a custom error message. In the example below, you provide a detailed error message to the end user that tells them to speak to their security team.

4. Add the rule from step two to the rule set:

```
BEGIN
    DVSYS.DBMS_MACADM.ADD_RULE_TO_RULE_SET(
        rule_set_name  => 'Trusted Rule Set'
        ,rule_name     => 'Trusted IP Address');
END;
/
```

Rule sets can consist of multiple rules that all must be met (true) or at least one rule must be true. Rules can be used in multiple rule sets. However, if you change the rule then the changes will apply to all rule sets the rule is used in.

5. Update the existing command rule to use the new rule set:

```
BEGIN
    DVSYS.DBMS_MACADM.UPDATE_COMMAND_RULE(
        command        => 'DROP TABLE'
```

```
        ,object_owner  => 'HR'
        ,object_name   => '%'
        ,rule_set_name => 'Trusted Rule Set'
        ,enabled       => dbms_macutl.g_yes);
END;
/
```

5.3 Creating a Command Rule to Control Application Authentication

Minimize the risk stolen or misused database application credentials may pose, by creating a Database Vault command rule to control how application credentials are used.

Stolen application credentials are one of the biggest risks to your organization's data. Application credentials often stored in configuration files, or scripts, and are frequently known by developers and administrators, making it difficult to identify the person using those credentials.

As `C##JSMITH` you will create a Database Vault rule, rule set, and command rule to limit how the HR application schema can connect to the Oracle Database.

1. Connect as `C##JSMITH`:

   ```
   connect c##jsmith@pdb_name
   ```

2. Create these four rules:

   ```
   BEGIN
       DBMS_MACADM.CREATE_RULE(
           rule_name  => 'Trusted Application IP Address'
           ,rule_expr => 'sys_context(''userenv'',''ip_address'') = ''<IP
   Address>'' ');
   END;
   /


   BEGIN
       DBMS_MACADM.CREATE_RULE(
           rule_name  => 'Trusted Application Hostname'
           ,rule_expr => 'sys_context(''userenv'',''host'') = ''appserver''
   ');
   END;
   /


   BEGIN
       DBMS_MACADM.CREATE_RULE(
           rule_name  => 'Trusted Application OS USer'
           ,rule_expr => 'sys_context(''userenv'',''os_user'') = ''appuser''
   ');
   ```

```
END;
/


BEGIN
    DBMS_MACADM.CREATE_RULE(
        rule_name  => 'Trusted Application Program'
        ,rule_expr => 'sys_context(''userenv'',''client_program_name'') =
''<client program name>'' ');
END;
/
```

3. Create the rule set:

```
BEGIN
    DVSYS.DBMS_MACADM.CREATE_RULE_SET(
        rule_set_name    => 'Trusted Application Path'
        ,description     => 'Controlling access to the application'
        ,enabled         => 'Y'
        ,eval_options    => dbms_macutl.g_ruleset_eval_all
        ,audit_options   => null
        ,fail_options    => dbms_macutl.g_ruleset_fail_show
        ,fail_message    => 'Unauthorized application usage. Contact the
IT helpdesk.'
        ,fail_code       => '-20000'
        ,handler_options => dbms_macutl.g_ruleset_handler_off
        ,handler         => null
        ,is_static       => true);
END;
/
```

By setting the `EVAL_OPTIONS` parameter to `ALL`, all rules must evaluate to true for this rule set to evaluate to true.

4. Add each of the rules from step to the rule set:

- ```
  BEGIN
      DVSYS.DBMS_MACADM.ADD_RULE_TO_RULE_SET(
          rule_set_name  => 'Trusted Application Path'
          ,rule_name     => 'Trusted Application IP Address');
  END;
  /
  ```

- ```
  BEGIN
      DVSYS.DBMS_MACADM.ADD_RULE_TO_RULE_SET(
          rule_set_name  => 'Trusted Application Path'
          ,rule_name     => 'Trusted Application Hostname');
  END;
  /
  ```

- ```
  BEGIN
      DVSYS.DBMS_MACADM.ADD_RULE_TO_RULE_SET(
          rule_set_name  => 'Trusted Application Path'
          ,rule_name     => 'Trusted Application OS User');
  ```
  ```

```
END;
/
```

- ```
  BEGIN
      DVSYS.DBMS_MACADM.ADD_RULE_TO_RULE_SET(
          rule_set_name  => 'Trusted Application Path'
          ,rule_name     => 'Trusted Application Program);
  END;
  /
  ```

5. Create a command rule that will evaluate the Trusted Application Path rule set when `HR` attempts to connect:

   ```
   BEGIN
       DVSYS.DBMS_MACADM.CREATE_CONNECT_COMMAND_RULE(
           user_name      => 'HR'
           ,rule_set_name => 'Trusted Application Path'
           ,enabled       => dbms_macutl.g_yes);
   END;
   /
   ```

6. Attempt to connect as `HR`:

   ```
   connect hr@pdb_name
   ```

   You will see the custom error message:

   ```
   ORA-47306: 20000: Unauthorized application usage. Contact the IT helpdesk.
   ```

7. Disable the command rule you just created to continue with other examples in this guide:

   a. Connect as `C##JSMITH`:

      ```
      connect c##jsmith@pdb_name
      ```

   b. Disable the command:

      ```
      BEGIN
          DVSYS.DBMS_MACADM.UPDATE_CONNECT_COMMAND_RULE(
              user_name      => 'HR'
              ,rule_set_name => 'Trusted Application Path'
              ,enabled       => dbms_macutl.g_no);
      END;
      /
      ```

   c. Verify the command rule on connect is disabled:

      ```
      SELECT ENABLED FROM DBA_DV_COMMAND_RULE WHERE COMMAND = 'CONNECT' AND
      OBJECT_OWNER = 'HR';
      ```
      ```

You will see:

```
ENABLED
_____________
N
```

6

Creating Unified Audit Policies and Accessing Audit Records

In Oracle Database Vault 23ai, the unified audit trail is protected and not accessible unless the proper Oracle Database Vault authorization is given. This protection extends to privileged users, such as `SYS` and `SYSTEM`, users with DBA role, and users with `AUDIT_VIEWER` or `AUDIT_ADMIN` roles.

To perform this task in Oracle Database 23ai, you will do the following:

1. Grant the `AUDIT_ADMIN` role `WITH ADMIN OPTION`
2. Use the `DV_OWNER` role to authorize `AUDIT_ADMIN` for the user

Because you are enforcing separation of duties, you will use two distinct database users to create this new database user. For example:

Grant and Authorize the `AUDIT_ADMIN` role `WITH ADMIN OPTION`

1. Connect as `C##JSMITH`:

   ```
   connect c##jsmith
   ```

2. Grant the `AUDIT_ADMIN` role `WITH ADMIN OPTION` to `C##CMACK`:

   ```
   GRANT AUDIT_ADMIN TO C##CMACK WITH ADMIN OPTION CONTAINER=ALL;
   ```

   As `C##JSMITH` has the appropriate `DV_ACCTMGR` and `AUDIT_ADMIN` roles, `AUDIT_ADMIN` granted `WITH ADMIN OPTION`, two steps can be completed by a single database user.

   However, to authorize `C##CMACK` to use their `AUDIT_ADMIN` role, a user with the `DV_ADMIN` role must perform the authorization. This ensures database users with highly privileged roles, such as viewing or managing audit data, cannot do so without explicit authorization.

3. Attempt to query the `UNIFIED_AUDIT_TRAIL` data dictionary view using both `C##JSMITH` and `C##CMACK`:

   ```
   connect c##jsmith@pdb_name

   SELECT COUNT(*) FROM UNIFIED_AUDIT_TRAIL;


   connect c##cmack@pdb_name

   SELECT COUNT(*) FROM UNIFIED_AUDIT_TRAIL;
   ```

   The expected outcome for both users, on Oracle Database 23ai, is `ORA-1031, insufficient privileges.`

4. Authorize C##CMACK to use their AUDIT_ADMIN role on the container database and each pluggable database:

```
connect c##jsmith

EXEC DBMS_MACADM.AUTHORIZE_AUDIT_ADMIN('C##CMACK');


connect c##jsmith@pdb_name

EXEC DBMS_MACADM.AUTHORIZE_AUDIT_ADMIN('C##CMACK');
```

C##CMACK can now query and managed the unified auditing operations in both the container database and each pluggable database. However, C##JSMITH still can't.

> **Caution:**
>
> C##JSMITH could grant themselves authorization to AUTHORIZE_AUDIT_ADMIN. This is a simple example with minimal separation of duties. To fully protect C##JSMITH from granting themselves the authorization, the user should not have both AUDIT_ADMIN and DV_ADMIN roles granted to them. To minimize the risk and enforce separation of duties, you would designate a separate user to grant the AUDIT_ADMIN role.

5. As C##CMACK, query the unified auditing operations to confirm authorization:

```
connect c##cmack@pdb_name

SELECT COUNT(*) FROM UNIFIED_AUDIT_TRAIL;
```

You will see the number of unified audit trails.

Create Audit Policies

1. Connect as C##CMACK:

```
connect c##cmack@pdb_name
```

2. Create these audit policies:

```
CREATE AUDIT POLICY aud_protect_hr_tables
ACTIONS COMPONENT=DV REALM VIOLATION ON "Protect HR tables";
AUDIT policy aud_protect_hr_tables;


CREATE AUDIT POLICY aud_protect_hr_indexes
ACTIONS COMPONENT=DV REALM VIOLATION ON "Protect HR indexes";
AUDIT policy aud_protect_hr_indexes;


CREATE AUDIT POLICY aud_protect_rule_set_trs
ACTIONS COMPONENT=DV RULE SET ON "Trusted Rule Set";
AUDIT policy aud_protect_rule_set_trs;
```

3. Verify the Unified Audit policies exist and are enabled:

```
SELECT POLICY_NAME
    FROM AUDIT_UNIFIED_ENABLED_POLICIES
WHERE POLICY_NAME LIKE 'AUD%'
ORDER BY 1;
```

   The output will be similar to:

```
POLICY_NAME

_____________________________________
AUD_PROTECT_HR_INDEXES
AUD_PROTECT_HR_TABLES
AUD_PROTECT_RULE_SET_TRS
```

4. To generate audit events, query the `HR.EMPLOYEES` table as each of the following user:

   - `SYS`
   - `SYSTEM`
   - `C##DVOWNER`
   - `C##DVACCTMGR`
   - `C##JSMITH`
   - `C##CMACK`
   - `GKRAMER`
   - `HR`

   a. Connect as `<user>`:

   ```
   connect <user>
   ```

   b. Attempt the following commands:

   ```
   SELECT COUNT(*) FROM HR.EMPLOYEES;
   CREATE TABLE HR.EMP2 AS SELECT * FROM HR.EMPLOYEES;
   CREATE INDEX HR.TEST_IDX ON HR.EMP2(HIRE_DATE,LAST_NAME);
   ```

```
DROP INDEX HR.TEST_IDX;
DROP TABLE HR.EMP2;
```

5. As C##CMACK, query the Unified Audit view to identify Database Vault related records and compare the results to the table below:

```
SELECT EVENT_TIMESTAMP, DBUSERNAME, OBJECT_SCHEMA, OBJECT_NAME, SQL_TEXT,
RETURN_CODE
    FROM UNIFIED_AUDIT_TRAIL
WHERE AUDIT_TYPE = 'Database Vault'
    AND OBJECT-SCHEMA = 'HR';
```

Command	SYS	SYSTEM	C##DVOWNER	C##DVACCTMGR	C##JSMITH	C##CMACK	GKRAMER	HR
SELECT COUNT(*) FROM HR.EMPLOYEES;	ORA-01031: insufficient privileges	ORA-01031: insufficient privileges	ORA-00942: table or view does not exist	ORA-00942: table or view does not exist	ORA-01031: insufficient privileges	ORA-41900: missing READ privilege on "HR"."EMPLOYEES"	ORA-01031: insufficient privileges	Success
CREATE TABLE HR.EMP2 AS SELECT * FROM HR.EMPLOYEES;	ORA-01031: insufficient privileges	ORA-01031: insufficient privileges	ORA-00942: table or view does not exist	ORA-00942: table or view does not exist	ORA-01031: insufficient privileges	ORA-41900: missing READ privilege on "HR"."EMPLOYEES"	ORA-01031: insufficient privileges	Success
CREATE INDEX HR.TEST_IDX ON HR.EMP2(HIRE_DATE,LAST_NAME);	ORA-47401: realm violation for CREATE INDEX on HR.TEST_IDX	ORA-47401: realm violation for CREATE INDEX on HR.TEST_IDX	ORA-00942: table or view does not exist	ORA-00942: table or view does not exist	ORA-47401: realm violation for CREATE INDEX on HR.TEST_IDX	ORA-41900: missing READ privilege on "HR"."EMPLOYEES"	ORA-47415: Insufficient Oracle Database Vault authorization for DDL	Success
DROP INDEX HR.TEST_IDX;	ORA-01418: specified index does not exist	ORA-01418: specified index does not exist	ORA-01418: specified index does not exist	ORA-01418: specified index does not exist	ORA-01418: specified index does not exist	ORA-01418: specified index does not exist	ORA-01418: specified index does not exist	Success

Command	SYS	SYSTEM	C##DVOWNER	C##DVACCTMGR	C##JSMITH	C##CMACK	GKRAMER	HR
DROP TABLE HR.EMP2;	ORA-47401: realm violation for CREATE INDEX on HR.TEST_IDX	ORA-47401: realm violation for CREATE INDEX on HR.TEST_IDX	ORA-47401: realm violation for CREATE INDEX on HR.TEST_IDX	ORA-47401: realm violation for CREATE INDEX on HR.TEST_IDX	ORA-47401: realm violation for CREATE INDEX on HR.TEST_IDX	ORA-47401: realm violation for CREATE INDEX on HR.TEST_IDX	ORA-47401: realm violation for CREATE INDEX on HR.TEST_IDX	ORA-47306: 20000: Access is blocked. Please speak to your security team

7

Authorizing DDL on a Different Schema

In an Oracle Database Vault environment, when a schema is protected by a realm or has access to realm protected objects, through realm authorization or object privileges, then the schema automatically has DDL controls applied to it to prevent a malicious actor from performing DDL modifications.

The only unexpected failure from the previous test is `GKRAMER` should be authorized to perform index creation in the `HR` schema.

For example, if a malicious user had privileges to modify a procedure in the `HR` schema, then they could insert malicious code into the procedure, and it would be trusted by the Database Vault realm. To avoid this situation, Oracle Database Vault enforces controls on DDL statements.

To allow `GKRAMER` to perform `CREATE INDEX` for `HR` objects, you must authorize `GKRAMER` to perform DDL:

1. Connect as `C##JSMITH` to the pluggable database:

   ```
   connect c##jsmith@pdb_name
   ```

2.
   ```
   SELECT * FROM DBA_DV_DDL_AUTH ORDER BY 1;

   EXEC DBMS_MACADM.AUTHORIZE_DDL('GKRAMER','HR');

   SELECT * FROM DBA_DV_DDL_AUTH ORDER BY 1;
   ```

3. Since you only changed the DDL authorization for `GKRAMER`, you will only retest the commands for `GRKAMER`:

   a. Connect as `GRKAMER`:

      ```
      connect GRKAMER
      ```

   b. Attempt the following commands:

      ```
      SELECT COUNT(*) FROM HR.EMPLOYEES;
      CREATE TABLE HR.EMP2 AS SELECT * FROM HR.EMPLOYEES;
      CREATE INDEX HR.TEST_IDX ON HR.EMP2(HIRE_DATE,LAST_NAME);
      DROP INDEX HR.TEST_IDX;
      DROP TABLE HR.EMP2;
      ```

   c. As `C##CMACK`, query the Unified Audit view to identify Database Vault related records and compare the results to the table below:

      ```
      SELECT EVENT_TIMESTAMP, DBUSERNAME, OBJECT_SCHEMA, OBJECT_NAME,
      SQL_TEXT, RETURN_CODE
          FROM UNIFIED_AUDIT_TRAIL
      ```

```
WHERE AUDIT_TYPE = 'Database Vault'
    AND OBJECT-SCHEMA = 'HR';
```

You will also see `GKRAMER` cannot drop the index they created. This is because of the Database Vault command rule you created preventing `DROP INDEX` commands on the `HR` schema.

Command	GKRAMER Without DDL Authorization	GKRAMER With DDL Authorization
`SELECT COUNT(*) FROM HR.EMPLOYEES;`	ORA-01031: insufficient privileges	ORA-01031: insufficient privileges
`CREATE TABLE HR.EMP2 AS SELECT * FROM HR.EMPLOYEES;`	ORA-01031: insufficient privileges	ORA-01031: insufficient privileges
`CREATE INDEX HR.TEST_IDX ON HR.EMP2(HIRE_DATE,LAST_NAME);`	ORA-47415: Insufficient Oracle Database Vault authorization for DDL	Success
`DROP INDEX HR.TEST_IDX;`	ORA-01418: specified index does not exist	Success
`DROP TABLE HR.EMP2;`	ORA-47401: realm violation for CREATE INDEX on HR.TEST_IDX	ORA-47401: realm violation for CREATE INDEX on HR.TEST_IDX

> **Note:**
>
> If you have upgraded from an earlier Oracle Database release, you may see (%,%) in the `DBA_DV_DDL_AUTH` view. As this authorization was added in a later release, Oracle chose to allow the existing DDL-allowed behavior to continue. If you are performing a new installation of Oracle Database 19c or later, you will not have the (%,%) authorization.

8

Performing a Data Pump Export of an Application Schema

When you want to export data that is protected by Oracle Database Vault realms or command rules, you must authorize the user to perform the task. Because Database Vault is enforcing a mandatory access control policy, only having the system privileges to perform an Oracle Data Pump export is not enough.

If the user is expected to perform Oracle Data Pump full database export, import into another schema, or transportable tablespace operations, then they must have additional authorizations and configuration. For more information on these scenarios, please see DBA Operations in an Oracle Database Vault Environment in the *Oracle Database Vault Administrator's Guide*.

This quick start guide will focus on two types of Oracle Data Pump exports:

1. Full-schema exports
2. Table-only exports

If you are not exporting the schema or table as the schema owner, then you must have the appropriate Oracle Database system roles to perform an export. You will create a dedicated database user to perform these two export operations.

1. As `C##JSMITH`, create a PDB database user named `DP_MGR`:
   a. Connect as `C##JSMITH` to the pluggable database:

      ```
      connect c##jsmith@pdb_name
      ```

   b. Create the `DP_MGR` named user:

      ```
      CREATE USER DP_MGR IDENTIFIED BY password;
      GRANT CONNECT, EXP_FULL_DATABASE TO DP_MGR;
      GRANT UNLIMITED TABLESPACE TO DP_MGR;
      ```

2. Attempt to perform an Oracle Data Pump Export as `DP_MGR`:

   ```
   expdp dp_mgr@pdb_name SCHEMAS=HR directory=DATA_PUMP_DIR REUSE_DUMPFILES=y
   dumpfile=test1.dmp logfile=test1.log
   ```

   You will encounter the following error for each of the table data objects you are attempting to export. This is because the objects are protected by the Database Vault realm, `Protect HR tables`.

   ```
   ORA-31693: Table data object "HR"."COUNTRIES" failed to load/unload and is
   being skipped due to error:
   ORA-47415: Insufficient Oracle Database Vault authorization for DATAPUMP.
   ```

3. Identify the database users who have system privileges to perform Data Pump exports or imports:

a. Connect as `C##JSMITH` to the pluggable database:

```
connect c##jsmith@pdb_name
```

b. Run the following command:

```
SELECT GRANTEE, GRANTED_ROLE
    FROM DBA_ROLE_PRIVS
WHERE GRANTED_ROLE IN ('EXP_FULL_DATABASE','DATAPUMP_EXP_FULL_DATABASE')
ORDER BY 1,2;
```

4. Authorize `DP_MGR` to only export a single table based on their database privileges. With Oracle Database Vault enabled on the pluggable database, it is necessary to authorize users to use their privileges. System privileges are not enough to perform the Data Pump export.

```
EXEC DBMS_MACADM.AUTHORIZE_DATAPUMP_USER('DP_MGR', 'HR', 'EMPLOYEES');
```

5. Review the Oracle Database Vault Data Pump authorizations:

```
SELECT * FROM DBA_DV_DATAPUMP_AUTH;
```

Only `TABLE` exports by `DP_MGR` should be authorized.

```
GRANTEE    SCHEMA    OBJECT       TYPE    ACTION
__________ _________ ____________ ________ _________
DP_MGR     HR        EMPLOYEES    TABLE    %
```

6. Run the Data Pump Export command:

```
expdp dp_mgr@pdb_name SCHEMAS=HR directory=DATA_PUMP_DIR REUSE_DUMPFILES=y
dumpfile=test1.dmp logfile=test1.log
```

You will find the `HR.EMPLOYEES` table has been exported but you will still receive `ORA-31693` errors for all other schema tables and views.

```
ORA-31693: Table data object "HR"."COUNTRIES" failed to load/unload and is
being skipped due to error:
ORA-47415: Insufficient Oracle Database Vault authorization for DATAPUMP.
. . exported "HR"."EMPLOYEES" 17.32 KB 107 rows
```

7. To perform the Data Pump Export, without the Database Vault violations, you must specify the `HR.EMPLOYEES` table for export:

```
expdp dp_mgr@pdb_name TABLES=HR.EMPLOYEES directory=DATA_PUMP_DIR
REUSE_DUMPFILES=y dumpfile=test1.dmp logfile=test1.log
```

8. To authorize `DP_MGR` to export a schema, based on their database privileges, revoke the table-specific authorization, and authorize the schema export:

```
EXEC DBMS_MACADM.UNAUTHORIZE_DATAPUMP_USER('DP_MGR', 'HR', 'EMPLOYEES');
EXEC DBMS_MACADM.AUTHORIZE_DATAPUMP_USER('DP_MGR', 'HR');
```

9. Review the Oracle Database Vault Data Pump authorizations. Only HR schema exports by `DP_MGR` should be authorized:

```
SELECT * FROM DBA_DV_DATAPUMP_AUTH;


GRANTEE     SCHEMA     OBJECT        TYPE     ACTION
__________ __________ _____________ _________ __________
DP_MGR      HR         %             %         %
```

10. Authorize `DP_MGR` to export schema, based on their database privileges, run the following command:

```
EXEC DBMS_MACADM.UNAUTHORIZE_DATAPUMP_USER('DP_MGR', 'HR');
EXEC DBMS_MACADM.AUTHORIZE_DATAPUMP_USER('DP_MGR');
```

> **Note:**
>
> This does not include performing a full Oracle Data Pump export. To perform a full export, the `DP_MGR` requires the `DV_OWNER` role. It is recommended to be more specific than full Data Pump exports. Export specific schemas or objects based on your requirements rather than the full database.

11. Review the Oracle Database Vault Data Pump authorizations for `DP_MGR`:

```
SELECT * FROM DBA_DV_DATAPUMP_AUTH;


GRANTEE     SCHEMA     OBJECT        TYPE     ACTION
__________ __________ _____________ _________ __________
DP_MGR      %          %             %         %
```

Now, your `DP_MGR` user can perform a Data Pump export like this example:

```
expdp dp_mgr@pdb_name SCHEMAS=HR,SCOTT,SH directory=DATA_PUMP_DIR
REUSE_DUMPFILES=y dumpfile=test1.dmp logfile=test1.log
```

12. To revoke the authorizations for `DP_MGR` to use their database privileges, run the following command:

```
exec dbms_macadm.unauthorize_datapump_user('DP_MGR');
```

The query to review your Data Pump authorizations should return no row:

```
SELECT * FROM DBA_DV_DATAPUMP_AUTH;
```

9

Separating Container Administrators from application data

Beginning with Oracle Database Vault 19c, operations control allows you to separate containers administrators (C## users) from the non-Oracle-maintained data in pluggable databases.

For example, `C##JSMITH` would not be able to query the `HR.EMPLOYEES` table in the `PDB1` pluggable database.

After you have configured and enabled Oracle Database Vault in the container database, it is easy to enable operations control.

To check the current enablement status of operations control:

1. Connect to the `sysdba` user account:

   ```
   connect / as sysdba
   ```

2. Check the Database Vault configurations on the container database:

   ```
   select * from cdb_dv_status
   ```

   The following result should appear:

   ```
   NAME                    STATUS             CON_ID
   ______________________  _________________  _________
   DV_CONFIGURE_STATUS     TRUE               1
   DV_ENABLE_STATUS        TRUE               1
   DV_APP_PROTECTION       NOT CONFIGURED     1
   DV_CONFIGURE_STATUS     TRUE               3
   DV_ENABLE_STATUS        TRUE               3
   DV_APP_PROTECTION       NOT CONFIGURED     3
   ```

   `DV_APP_PROTECTION` is the parameter for Oracle Database Vault operations control.

To enable Database Vault operations control:

1. Connect as a user with the `DV_OWNER` role:

   ```
   connect c##dvowner
   ```

2. Enable Database Vault operations control:

   ```
   EXEC DBMS_MACADM.ENABLE_APP_PROTECTION;
   ```

3. Verify that Database Vault operations control was enabled:

a. Connect to the `sysdba` user account:

```
connect / as sysdba
```

b. Check the Database Vault configurations on the container database:

```
select * from cdb_dv_status
```

The following result should appear:

```
NAME                    STATUS             CON_ID
_____________________ _________________ _________
DV_CONFIGURE_STATUS     TRUE               1
DV_ENABLE_STATUS        TRUE               1
DV_APP_PROTECTION       ENABLED            1
DV_CONFIGURE_STATUS     TRUE               3
DV_ENABLE_STATUS        TRUE               3
DV_APP_PROTECTION       ENABLED            3
```

`DV_APP_PROTECTION` is the parameter for Oracle Database Vault operations control.

A valid configuration is to have Database Vault operations control enabled in the CDB and all PDBs while Database Vault is not enabled in a PDB. If Database Vault is enabled in the container database, operations control can be used to separate container users (C##) from data in pluggable databases.

For example, if there was a second PDB in this environment, these results of `select * from cdb_dv_status` are a valid configuration:

```
NAME                    STATUS             CON_ID
_____________________ _________________ _________
DV_CONFIGURE_STATUS     TRUE               1
DV_ENABLE_STATUS        TRUE               1
DV_APP_PROTECTION       ENABLED            1
DV_CONFIGURE_STATUS     TRUE               3
DV_ENABLE_STATUS        TRUE               3
DV_APP_PROTECTION       ENABLED            3
DV_CONFIGURE_STATUS     FALSE              4
DV_ENABLE_STATUS        FALSE              4
DV_APP_PROTECTION       ENABLED            4
```

10
Cleaning Up After the Quick Start Guide

If you've completed the tasks in this Quick Start Guide in your databases then there are a number of changes that should be reverted.

As a user with the `DV_OWNER` or `DV_ADMIN` role, perform the following:

1. Disable Database Vault operations control from the container database:

```
connect c##dvowner


EXEC DBMS_MACADM.DISABLE_APP_PROTECTION;


connect / as sysdba


SELECT * FROM CDB_DV_STATUS;
```

2. Delete the command rules, realms, and associated rules and rule sets

```
connect c##jsmith@pdb_name


BEGIN
    DBMS_MACADM.DELETE_COMMAND_RULE(
        command         => 'DROP TABLE'
        ,object_owner   => 'HR'
        ,object_name    => '%'
        ,scope          => DBMS_MACUTL.G_SCOPE_LOCAL);
END;
/


BEGIN
    DBMS_MACAM.DELETE_RULE_SET(
        rule_set_name => 'Trusted Rule Set');
END;
/


BEGIN
    DBMS_MACAM.DELETE_RULE(
        rule_name => 'Trusted IP Address');
END;
/


BEGIN
    DVSYS.DBMS_MACADM.DELETE_REALM_CASCADE(realm_name => 'Protect HR
```

```
tables');
END;
/


BEGIN
    DVSYS.DBMS_MACADM.DELETE_REALM_CASCADE(realm_name => 'Protect HR
indexes');
END;
/
```

3. As a user who has the privileges to administer unified audit policies, delete the unified audit policies:

```
connect c##cmack@pdb_name


NOAUDIT POLICY AUD_PROTECT_HR_TABLES;
NOAUDIT POLICY AUD_PROTECT_HR_INDEXES;
NOAUDIT POLICY AUD_PROTECT_RULE_SET_TRS;

DROP AUDIT POLICY AUD_PROTECT_HR_TABLES;
DROP AUDIT POLICY AUD_PROTECT_HR_INDEXES;
DROP AUDIT POLICY AUD_PROTECT_RULE_SET_TRS;
```

4. To drop the users in this example, perform the following as a user with the `DV_ACCTMGR` role:

```
connect c##dvacctmgr


DROP USER C##CMACK CASCADE;


ALTER SESSION SET CONTAINER=pdb_name;

DROP USER GKRAMER CASCADE;
```

5. Before you can drop `JSMITH`, you must revoke `DV_ADMIN`. This is a mechanism to prevent the accidental, or intentional, destruction of privileged Database Vault users:

```
connect c##dvowner


REVOKE DV_ADMIN FROM C##JSMITH CONTAINER=ALL;


connect c##dvacctmgr


DROP USER C##JSMITH CASCADE;
```

6. Disable Oracle Database Vault on the pluggable and container databases. You will perform the disablement in reverse order of the enablement. To enable, you started with the

container database and moved to the pluggable databases. To disable, you will start on the pluggable databases then move to the container database.

a. In the pluggable database, as a user with the `DV_OWNER` role, perform the following:

```
CONNECT c##dvowner@pdb_name


SELECT * FROM DBA_DV_STATUS;
EXEC DBMS_MACADM.DISABLE_DV;
```

b. Restart the pluggable database for the changes to take effect:

```
connect / as sysdba


ALTER PLUGGABLE DATABASE pdb_name CLOSE IMMEDIATE;
ALTER PLUGGABLE DATABASE pdb_name OPEN;
```

c. Check the Oracle Database Vault enablement status:

```
SELECT CON_ID_TO_CON_NAME(CON_ID) CON_NAME, NAME, STATUS
    FROM CDB_DV_STATUS
ORDER BY 1;
```

The output should be:

```
CON_NAME    NAME                    STATUS
___________ _______________________ ___________________
CDB$ROOT    DV_CONFIGURE_STATUS     TRUE
CDB$ROOT    DV_ENABLE_STATUS        TRUE
CDB$ROOT    DV_APP_PROTECTION       DISABLED
pdb_name    DV_CONFIGURE_STATUS     TRUE
pdb_name    DV_ENABLE_STATUS        FALSE
pdb_name    DV_APP_PROTECTION       DISABLED
```

d. Now disable Database Vault on the container database:

```
CONNECT c##dvowner


EXEC DBMS_MACADM.DISABLE_DV;
```

e. Restart the container database for the changes to take effect:

```
connect / as sysdba


SHUTDOWN IMMEDIATE;
STARTUP;
```

f. Check the Oracle Database Vault enablement status:

```
SELECT CON_ID_TO_CON_NAME(CON_ID) CON_NAME, NAME, STATUS
    FROM CDB_DV_STATUS
ORDER BY 1;
```

The output should be:

```
CON_NAME    NAME                    STATUS
___________ _______________________ ___________________
CDB$ROOT    DV_CONFIGURE_STATUS     TRUE
CDB$ROOT    DV_ENABLE_STATUS        FALSE
CDB$ROOT    DV_APP_PROTECTION       DISABLED
pdb_name    DV_CONFIGURE_STATUS     TRUE
pdb_name    DV_ENABLE_STATUS        FALSE
pdb_name    DV_APP_PROTECTION       DISABLED
```

11

Applying Oracle Database Patches

You can upgrade or apply patches to the Oracle database while Database Vault is enabled by granting a special role to `SYS`. As a user with the `DV_OWNER` role, grant `DV_PATCH_ADMIN` to `SYS` for all pluggable databases.

From the container database:

1. Connect as a user with the `DV_OWNER` role:

   ```
   connect c##dvowner
   ```

2. Grant `DV_PATCH_ADMIN` to `SYS` for all pluggable databases:

   ```
   GRANT DV_PATCH_ADMIN TO SYS CONTAINER=ALL;
   ```

3. Once patching is complete, revoke `DV_PATCH_ADMIN` from the `SYS` user:

   a. Connect as a user with the `DV_OWNER` role:

      ```
      connect c##dvowner
      ```

   b. Revoke `DV_PATCH_ADMIN`:

      ```
      REVOKE DV_PATCH_ADMIN FROM SYS CONTAINER=ALL;
      ```

12

Troubleshooting and Tracing Errors

Occasionally, you will need to troubleshoot authorizations or the lack of authorization to application objects, database system privileges or roles, or other activities. Here are some steps you can follow to help you identify the problem.

12.1 Most Common Database Vault Views

It is important to understand how to navigate the views associated with Oracle Database Vault.

As a user with the `DV_ADMIN` role, you should familiarize yourself with the views with this query:

```
connect c##jsmith@pdb_name

SELECT VIEW_NAME
    FROM DBA_VIEWS
WHERE VIEW_NAME LIKE 'DBA_DV_%'
ORDER BY 1;
```

The most common views you will work with are as follows:

Table 12-1 Common Database Vault Views

View Name	Purpose
`DBA_DV_%_AUTH`	Look for views with `AUTH` in the name. These views list which users are authorized for various activities, such as `PROXY_AUTH`, `DDL_AUTH`, `JOB_AUTH`, and so on.
`DBA_DV_COMMAND_RULE`	Lists command rules created.
`DBA_DV_REALM`	Lists realms created. Pay attention to the `REALM_TYPE` column. `REGULAR` respects direct object grants to access realm protected data, but `MANDATORY` does not. Mandatory requires you to be an authorized participant/owner in the realm.
`DBA_DV_REALM_AUTH`	Lists authorizations to realms created. Pay attention to `AUTH_RULE_SET_NAME` (restrictions to using authorization) and `AUTH_OPTIONS` (owner vs. participant).
`DBA_DV_REALM_OBJECT`	Lists objects protected by realms. `%` means all, including object names or types not yet created.
`DBA_DV_RULE`	Lists rules and the rule expression.
`DBA_DV_RULE_SET`	Lists rule sets, status, and if a fail message is returned to the user. Also lists whether it is static or dynamic (`IS_STATIC`) and whether all rules have to be true or just one (`EVAL_OPTIONS_MEANING`).
`DBA_DV_RULE_SET_RULE`	Lists rule sets and their associated rules.

Table 12-1 (Cont.) Common Database Vault Views

View Name	Purpose
DBA_DV_SIMULATION_LOG	Lists the actions that would have violated a command rule or realm if the command rule or realm was in enforcement mode.
DBA_DV_STATUS	Lists the configuration and enablement status of Oracle Database Vault and whether operations control is in use (DV_APP_PROTECTION).
DBA_ROLES	Lists the roles in the database. Oracle Database Vault creates 12+ roles that help enforce separation of duties. Look for DV_% roles.

```
DV_OWNERDV_ADMINDV_SECANALYST

connect c##jsmith@pdb_name

SELECT * FROM (
    SELECT REALM_NAME, 'PROTECTED OBJECTS' COL2, OWNER COL3, OBJECT_TYPE
COL4 ,OBJECT_NAME COL5
    FROM DVSYS.DBA_DV_REALM_OBJECT
    UNION
    SELECT REALM_NAME, 'AUTHORIZATIONS' COL2, GRANTEE COL3,
AUTH_RULE_SET_NAME COL4, AUTH_OPTIONS COL5
    FROM DVSYS.DBA_DV_REALM_AUTH)
    WHERE REALM_NAME IN (SELECT NAME FROM DBA_DV_REALM WHERE ORACLE_SUPPLIED
= 'NO')
    ORDER BY REALM_NAME ASC, COL2 DESC;
```

If you followed the examples in this quick start guide, you would end up with results like this:

```
REALM_NAME            COL2                  COL3    COL4     COL5
____________________ ____________________ _______ ________ ________
Protect HR indexes    protected objects    HR      INDEX    %
Protect HR indexes    authorizations       HR               Owner
Protect HR tables     protected objects    HR      TABLE    %
Protect HR tables     protected objects    HR      VIEW     %
Protect HR tables     authorizations       HR               Owner
```

To view user-defined Database Vault command rules, and their associates rule set and rules, you could use a query like this:

```
SELECT A.COMMAND, A.OBJECT_OWNER, A.OBJECT_NAME, B.RULE_SET_NAME, B.RULE_NAME
    FROM DVSYS.DBA_DV_COMMAND_RULE A
        , DVSYS.DBA_DV_RULE_SET_RULE B
    WHERE A.RULE_SET_NAME = B.RULE_SET_NAME
        AND A.ORACLE_SUPPLIED != 'YES'
ORDER BY 1,2,3;
```

If you followed the examples in this quick start guide, you would end up with results like this:

```
COMMAND    OBJECT_OWNER OBJECT_NAME RULE_SET_NAME    RULE_NAME
__________ ____________ ___________ _________________ ____________________
DROP TABLE HR           %           Trusted Rule Set Trusted IP Address
```

For more detailed scripts to collect Oracle Database Vault information, refer to to My Oracle Support Doc ID 1352556.1, Script To List The Database Vault Realms, Command Rules And Rule Sets.

12.2 Realm and Command Rule Enforcement Simulation

Oracle Database Vault simulation mode can also be used to help troubleshoot realms and command rules.

There are times, particularly at the beginning of a project, when you do not want to enforce your custom Database Vault realms or command rules immediately. Oracle Database Vault offers a simulation mode, where the enforcement activity would be logged, but not enforced, allowing you to verify you have the proper database users authorized to the realm or the correct logic in the rule, that is being enforced by the rule set, for a command rule.

For example, if you want to modify the realm you created to protected tables and allow an action to occur, you can update the realm to be in simulation mode. As a user with `DV_OWNER` or `DV_ADMIN` role, perform the following:

1. Connect as `C##JSMITH` on the pluggable database:

   ```
   connect c##jsmith@pdb_name
   ```

2. Create the mandatory realm in simulation mode:

   ```
   BEGIN
       DVSYS.DBMS_MACADM.UPDATE_REALM(
           realm_name     => 'Protect HR tables'
           ,description   => 'Mandatory realm to protect HR tables'
           ,enabled       => dbms_macutl.g_simulation
           ,audit_options => null
           ,realm_type    => dbms_macadm.mandatory_realm);
   END;
   /
   ```

3. Verify the realm is in simulation mode:

   ```
   SELECT ENABLED FROM DBA_DV_REALM WHERE NAME = 'Protect HR tables';
   ```

   You should see:

   ```
   ENABLED
   __________
   S
   ```

4. As GKRAMER, perform a query of the HR.EMPLOYEES table:

```
connect gkramer@pdb_name

SELECT COUNT(*) FROM HR.EMPLOYEES;
```

5. Once you complete your activity, you should review the simulation log to see what would have been prevented by the realm if it was still in enforcement mode:

```
connect c##jsmith@pdb_name

SELECT USERNAME,COMMAND, VIOLATION_TYPE, REALM_NAME, RETURNCODE, SQLTEXT
FROM DBA_DV_SIMULATION_LOG;
```

The following output should appear:

```
USERNAME COMMAND VIOLATION_TYPE  REALM_NAME         RETURNCODE
SQLTEXT
________ _______ _______________ ___________________ __________
_____________________________
GKRAMER  SELECT  Realm Violation Protect HR tables  1031       SELECT
COUNT(*) FROM HR.EMPLOYEES
```

> **Note:**
>
> Simulation mode does not create records for activity that would be authorized by the realm or command rule. Only activity that would have been denied is recorded in the simulation log.

6. After you have completed your simulation exercise, Oracle recommends deleting all rows from the simulation log table to not confuse yourself if you perform the action again. As a user with the DV_OWNER role, run the following commands:

```
connect c##jsmith@pdb_name

DELETE FROM DVSYS.SIMULATION_LOG$;
COMMIT;
```

7. Return your realm to enforcement mode:

```
BEGIN
    DVSYS.DBMS_MACADM.UPDATE_REALM(
        realm_name     => 'Protect HR tables'
        ,description   => 'Mandatory realm to protect HR tables'
        ,enabled       => dbms_macutl.g_yes
        ,audit_options => null
        ,realm_type    => dbms_macadm.mandatory_realm);
END;
/
```

8. Verify the realm is in enforcement mode:

```
SELECT ENABLED FROM DBA_DV_REALM WHERE NAME = 'Protect HR tables';
```

You should see:

```
ENABLED
__________
Y
```

12.3 Tracing Database Vault Activity

If you are still unsure as to why your SQL command is being blocked by Oracle Database Vault then you can enable tracing.

Trace files are created on the database server in the diagnostics directory, the same as tracing other database operations. Oracle Database Vault tracing is enabled by session or system-wide, on the container database or a specific pluggable database.

To enable tracing, you must have the following privileges:

- Database Vault role `DV_OWNER` or `DV_ADMIN`
- System privilege `ALTER SESSION` or `ALTER SYSTEM`

If you have followed the examples in this guide, you have a database user who has both the `DV_ADMIN` role and the `ALTER SYSTEM` system privilege, by having the `DBA` role.

To enable tracing:

1. Connect as `C##JSMITH` on the pluggable database:

```
connect c##jsmith@pdb_name
```

2. Enable tracing:

```
ALTER SYSTEM SET EVENTS 'TRACE[DV] DISK=HIGHEST';
```

3. Run a command that will fail and show up in the trace log:
   a. Connect as `GKRAMER` on the pluggable database:

```
connect gkramer@pdb_name
```

   b. Run a command that will fail:

```
SELECT COUNT(*) FROM HR.EMPLOYEES;
```

   You will receive Oracle error message `ORA-01031: insufficient privileges`.

4. Verify that tracing has been enabled by viewing the trace log on the database server. As the oracle operating system user on the Oracle database, run the following:

```
cd $ORACLE_BASE/diag

find . -type f -name "*_QS_DV_trace.trc"
```

```
./rdbms/free/FREE/trace/FREE_ora_225318_QS_DV_trace.trc

vi ./rdbms/free/FREE/trace/FREE_ora_225318_QS_DV_trace.trc
```

In the trace file, you will see the result of the command and the specific realm that is protecting the object. `Protect HR Tables` is the realm protecting the `HR.EMPLOYEES` table:

```
Result=Realm Authorization Failed
Realm_Name=Protect HR tables     Required_Auth_Level=0
Current_User=141
Object_Owner=HR     Object_Name=EMPLOYEES     Object_Type=TABLE
SQL_Text=select count(*) from hr.employees
```

5. When you are finished, you will disable the tracing event and, if appropriate, revoke the `DV_ADMIN role`.
   You should not revoke `DV_ADMIN` from `C##JSMITH` in this example.
   a. Connect as `C##JSMITH` on the pluggable database:

      ```
      connect c##jsmith@pdb_name
      ```

   b. Disable tracing:

      ```
      ALTER SYSTEM SET EVENTS 'TRACE[DV] OFF';
      ```

If the advice in this section does not help you resolve your issues, submit a Support Request and include the relevant information about your environment and Database Vault settings. Uploading the results of MOS 1352556.1 Script To List The Database Vault Realms, Command Rules And Rule Sets, will help your support engineer identify your issue more effectively.

Glossary

Index