ORACLE DATABASE VAULT

implement oracle vault in database:

Step 1-1: Verify Prerequisites:

-Verify that your database is in ARCHIVELOG mode:

SQL> SELECT log_mode FROM v$database;

```
SQL> SELECT log_mode FROM v$database;

LOG_MODE
------------
ARCHIVELOG
```

-Check if Oracle Label Security (OLS) is enabled, as Database Vault requires it, If it is not installed, you must enable it.

SQL>

SELECT COMP_NAME, VERSION, STATUS FROM dba_registry WHERE comp_id='OLS';

```
COMP_NAME
----------------------------------------
VERSION                        STATUS
------------------------------ -----------
Oracle Label Security
19.0.0.0.0                     VALID
```

Step 1-2: Check whether DB Vault is enabled:

SQL> select * from v$option where lower(PARAMETER) like '%vault%';

```
PARAMETER
----------------------------------------
VALUE
----------------------------------------
Oracle Database Vault
FALSE
```

SQL> select * from dba_dv_status;

```
NAME
------------------
STATUS
------------------------------
DV_APP_PROTECTION
NOT CONFIGURED

DV_CONFIGURE_STATUS
FALSE

DV_ENABLE_STATUS
FALSE
```

Step 2-1: Users to manage database vault:

SQL>

create user c##dvowner identified by <password>;

create user c##dvactmgr identified by <password>;

to set configuration for users

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

Step 2-2: to enable database vault:

SQL> conn c##dvowner/dvowner

EXEC DBMS_MACADM.ENABLE_DV;

execute dvsys.dbms_macadm.enable_app_protection(NULL);

EXEC DBMS_MACADM.DISENABLE_APP_PROTECTION;

EXEC DBMS_MACADM.ENABLE_APP_PROTECTION ('PDB_NAME');

Step 2-3: restart database to confige database vault:

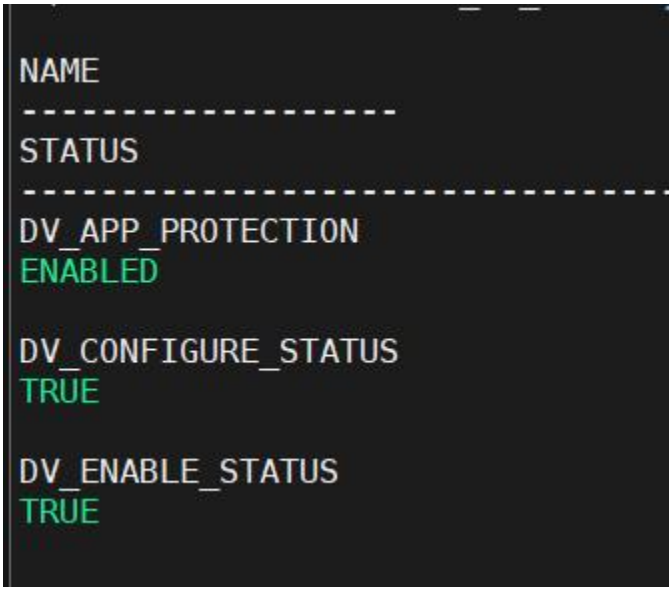
SQL> conn / as sysdba

SQL> shutdown immediate

SQL> startup

SQL> alter pluggable database all open;

SQL> select * from dba_dv_status;



Step 3-1: Define Realms, Command Rules, and Factors:

Steps to Prevent Access Example: -

1. Create a Database Vault Realm

Create a realm specifically to protect these 3 tables.

```
BEGIN
  DBMS_MACADM.CREATE_REALM (
    realm_name       => 'Protect_bank_Tables',
    description      => 'Protect table1, table2, and table3 in the bank schema',
    enabled          => DBMS_MACUTL.G_YES
  );
END;
/
```

2. Add the Specific Tables to the Realm

Add only the 3 tables to the realm

```
BEGIN
  -- Add table1
  DBMS_MACADM.ADD_OBJECT_TO_REALM(
```

```
    realm_name      => 'Protect_bank_Tables',
    object_schema   => 'BANK',
    object_name     => 'TABLE1',
    object_type     => 'TABLE'
  );

  -- Add table2
  DBMS_MACADM.ADD_OBJECT_TO_REALM(
    realm_name      => 'Protect_bank_Tables',
    object_schema   => 'BANK',
    object_name     => 'TABLE2',
    object_type     => 'TABLE'
  );

  -- Add table3
  DBMS_MACADM.ADD_OBJECT_TO_REALM (
    realm_name      => 'Protect_bank_Tables',
    object_schema   => 'BANK',
    object_name     => 'TABLE3',
    object_type     => 'TABLE'
  );
END;
/
```

3. Use Database Roles for Fine-Grained Privileges

Create a role for users who need access

```
CREATE ROLE bank_read_role;
GRANT SELECT ON bank. <table_name> TO bank_read_role;
```

2.Assign this role only to specific users:

GRANT bank_read_role TO user_allowed_access;

4. audit

AUDIT SELECT, INSERT, UPDATE, DELETE ON bank. <table_name> BY ACCESS;

Step 3-2: Optionally Restrict SYS from Performing Administrative Actions:

A) Create a Database Vault Realm for SYS

- Switch to secure_admin (Database Vault Owner) and create a realm to restrict SYS from modifying system objects:

```
BEGIN
  DVSYS.DBMS_MACADM.CREATE_REALM(
    realm_name    => 'Restrict SYS Realm',
    realm_desc    => 'Prevents SYS from accessing sensitive data',
    enabled       => DBMS_MACUTL.G_YES
  );
END;
/
```

B) Add SYS-Related Objects to the Realm

- Protect important schemas (like SYS, SYSTEM):

- This prevents SYS from modifying any table in the SYS schema.

```
BEGIN
  DVSYS.DBMS_MACADM.ADD_REALM_OBJECT(
    realm_name  => 'Restrict SYS Realm',
    object_owner => 'SYS',
    object_name  => '%',
    object_type  => 'TABLE'
  );
END;
/
```

Prevent SYS from Running Critical Commands

==========

- Create a Command Rule to block SYS from executing certain commands:

A) Block SYS from Creating Users Example

```
BEGIN
  DVSYS.DBMS_MACADM.CREATE_COMMAND_RULE(
    command        => 'CREATE USER',
    rule_name      => 'Block SYS Create User',
    object_owner   => NULL,
    object_name    => NULL,
    rule_expr      => 'SYS_CONTEXT (''USERENV'', ''SESSION_USER'')!= ''SYS''',
    enabled        => DBMS_MACUTL.G_YES
  );
END;
/
```

B) Block SYS from Executing DDL Statements, This prevents SYS from executing ALTER SYSTEM commands.

```
BEGIN
  DVSYS.DBMS_MACADM.CREATE_COMMAND_RULE (
    command        => 'ALTER SYSTEM',
    rule_name      => 'Block SYS Alter System',
    object_owner   => NULL,
    object_name    => NULL,
    rule_expr      => 'SYS_CONTEXT(''USERENV'', ''SESSION_USER'') != ''SYS''',
    enabled        => DBMS_MACUTL.G_YES
  );
END;
/
```