



SIGURNOST ORACLE BAZE PODATAKA

STUDENT: UROŠ PEŠIĆ 1465

PROFESROR: ALEKSANDAR STANIMIROVIĆ



SIGURNOST

Sigurnost baza podataka predstavlja široku oblast koja se bavi problemima zaštite podataka od neautorizovanog pristupa, sproveđenja specifičnih sigurnosnih ograničenja, organizacije podataka i korisnika u više sigurnosnih klasa, enkripcije podataka, itd.





GLAVNI CILJEVI

Očuvanje integriteta podataka

Podaci u sistemu moraju biti zaštićeni od neovlašćene modifikacije. Modifikacija podataka podrazumeva dodavanje, ažuriranje i brisanje podataka. Narušavanje integriteta dovodi do kontaminacije podataka u bazi, koja dalje može dovesti do prevara ili donošenja pogrešnih poslovnih odluka.

Obezbeđivanje dostupnosti

Podaci u sistemu moraju biti zaštićeni od neovlašćene modifikacije. Modifikacija podataka podrazumeva dodavanje, ažuriranje i brisanje podataka. Narušavanje integriteta dovodi do kontaminacije podataka u bazi, koja dalje može dovesti do prevara ili donošenja pogrešnih poslovnih odluka.

Zaštita poverljivih podataka

Podrazumeva zaštitu poverljivih podataka od neovlašćenog čitanja i njihovog javnog iznošenja. Posledice narušavanja poverljivosti variraju u zavisnosti od stepena tajnosti otkrivenih podataka, važećih državnih zakona, i sličnih faktora.



MERE KONTROLE

01

Kontrola pristupa

Kontrola pristupa se odnosi na mehanizme zaštite od neautorizovanog pristupa, kako sistem u celini, tako i njegovih objekata (tabela, atributa, itd).

02

Kontrola toka podataka

Kontrolom toka se sprečava da podaci u sistemu teku sa viših na niže sigurnosne nivoje u sistemima koji su ovako organizovani. Kontrolom toka se sprečava pojava **prikrivenih kanala** (*covert channels*).

03

Kontrola zaključivanja

Mera kontrole koja je bitna za statističke baze podataka. Kontrolom zaključivanja se sprečava izvođenje zaključaka o podacima pojedinačnih korsnika iz populacije nad kojom se obavljaju statistička izračunavanja.

04

Enkripcija podataka

Enkripcija je proces konverzije podataka u šifrovani tekst (eng. Cyphertext), korišćenjem nekog algoritma za enkripciju. Korisna je kao dodatni vid zaštite u slučaju da podaci dospeju u pogrešne ruke.

04



KONTROLA PRISTUPA

Za ovaj aspekt sigurnosti sistema, prvenstveno je zadužen **administrator baze podataka (DBA – Database Administrator)**.

Postoje dva tipa kontrole pristupa:

- **Diskreciona kontrola pristupa - DAC**

Tehnika kontrole pristupa korišćenjem privilegija, kojima se pojedinim korisnicima dozvoljavaju, odnosno zabranjuju operacije čitanja ili modifikacije objekata u sistemu, kao što su tabele, atributi, slogovi, i slično. Takođe, moguće je selektivno dozvoliti samo neke operacije (ne funkcionišu po principu sve ili ništa).

- **Obavezna kontrola pristupa - MAC**

Koristi se za implementaciju više sigurnosnih nivoa. Ovim mehanizmom se sprečava da korisnici pristupe objektima koji su na višim nivoima sigurnosti od njihovog. Iz obavezne kontrole pristupa razvila se i kontrola pristupa zasnovana na ulogama (eng. Role-based), kojom se definišu pravila i privilegije na osnovu uloga i ograničenja koje korisnici imaju u organizaciji koja koristi sistem.



DISKRECIJONA KONTROLA PRISTUPA

Diskrepciona kontrola pristupa se zasniva na dodeljivanju i oduzimanju **privilegija** korisnicima. Postoje dva tipa privilegija:

- **Definisane na nivou korisničkog naloga**

Ovako definisane privilegije su vezane za sam korisnički nalog i ne zavise od objekata u sistemu. Neki primeri ovakvih privilegija su: dozvola za kreiranje šema, tabele, ili pogleda, dodavanje ili brisanje atributa relacija, brisanje tabele i podataka, čitanje podataka korišćenjem SELECT naredbe, itd. Privilegije na nivou naloga nisu standardni deo SQL-a, već su različito implementirane u različitim DBMS-ovima.

- **Definisane na nivou relacija**

Ovako definisane privilegije se odnose na konkretnе relacije (tabele) ili virtuelne relacije (poglede). Njima se definiše koje operacije svaki postojeći korisnik može da izvrši nad relacijama u sistemu. Takođe je moguće definisati nad kojim podskupom atributa relacija je moguće primeniti dozvoljene operacije.



DISKRECIONA KONTROLA PRISTUPA - ORACLE PRIMERI

1

Dodatak 1: Snimak ekrana prikaza upisa u tablicu EMPLOYEE i stvaranje javnog synonyma.

```
CREATE TABLE EMPLOYEE (ssn NUMBER CONSTRAINT snn_nn NOT NULL,
    fullname VARCHAR2(50),
    address VARCHAR2(50),
    sex CHAR(1));

ALTER SESSION SET CURRENT_SCHEMA=U1;
CREATE PUBLIC SYNONYM EMPLOYEE FOR U1.EMPLOYEE;

INSERT INTO EMPLOYEE VALUES (1, 'Uros Pesic', 'Addr1', 'M');
INSERT INTO EMPLOYEE VALUES (2, 'Pera Peric', 'Addr2', 'M');
SELECT * FROM EMPLOYEE;
```

Script Output x Query Result x

All Rows Fetched: 2 in 0,008 seconds

SSN	FULLNAME	ADDRESS	SEX
1	Uros Pesic	Addr1	M
2	Pera Peric	Addr2	M

2

Dodatak 2: Snimak ekrana prikaza greske ORA-00942.

```
SELECT * FROM EMPLOYEE;
```

Query Result x

Executing:SELECT * FROM EMPLOYEE in 0 seconds

ORA-00942: table or view does not exist
00942. 00000 - "table or view does not exist"
*Cause:
*Action:
Error at Line: 1 Column: 15

1

Dodatak 1: Snimak ekrana prikaza upisa u tablicu EMPLOYEE i stvaranja privilegija za upis.

```
GRANT SELECT ON EMPLOYEE TO U2;
```

Script Output x Query Result x

Task completed in 0,083 seconds

Grant succeeded.

2

Dodatak 2: Snimak ekrana prikaza upisa u tablicu EMPLOYEE i izvoza rezultata.

```
SELECT * FROM EMPLOYEE;
```

Query Result x

All Rows Fetched: 2 in 0,01 seconds

SSN	FULLNAME	ADDRESS	SEX
1	Uros Pesic	Addr1	M
2	Pera Peric	Addr2	M



DISKRECIONA KONTROLA PRISTUPA - ORACLE PRIMERI

Dodela privilegije korisniku, sa opcijom da je on dalje prosleđuje - **GRANT** opcija

The screenshot shows four separate sessions in Oracle SQL Developer, each with a red number indicating the step:

- Session 1:** A user with sysdba privileges grants the INSERT privilege on the EMPLOYEE table to user U2, including the GRANT OPTION. The output shows "Grant succeeded."
- Session 2:** User U2, who now has sysdba privileges due to the GRANT OPTION, grants the INSERT privilege on the EMPLOYEE table to user U3. The output shows "Grant succeeded."
- Session 3:** User U3, who now has sysdba privileges, inserts a new row into the EMPLOYEE table with values (3, 'Novi Korisnik', NULL, 'F'). The output shows "1 row inserted." and "Commit complete."
- Session 4:** A user with sysdba privileges runs a SELECT query on the EMPLOYEE table. The output shows the results:

SSN	FULLNAME	ADDRESS	SEX
1	3 Novi Korisnik (null)		F
2	1 Uros Pesic	Addr1	M
3	2 Pera Peric	Addr2	M



DISKRECIJONA KONTROLA PRISTUPA - ORACLE PRIMERI

Dodeljene privilegije je moguće oduzeti naredbom **REVOKE**.

The screenshot shows two Oracle SQL Developer windows side-by-side. The left window's title bar has tabs for 'Welcome Page', 'sysdba', 'U1', 'U2', and 'U3'. The 'Worksheet' tab is active, containing the command: `REVOKE INSERT ON EMPLOYEE FROM U2;`. Below it, the 'Script Output' tab shows the message: `Revoke succeeded.`. The right window also has tabs for 'Welcome Page', 'sysdba', 'U1', 'U2', and 'U3'. Its 'Worksheet' tab contains the same command. The 'Script Output' tab displays an error message: `Error starting at line : 1 in command -
INSERT INTO EMPLOYEE VALUES (5,'Novi Korisnik', NULL, 'F')
Error at Command Line : 1 Column : 13
Error report -
SQL Error: ORA-01031: insufficient privileges
01031. 00000 - "insufficient privileges"
*Cause: An attempt was made to perform a database operation without
the necessary privileges.
*Action: Ask your database administrator or designated security
administrator to grant you the necessary privileges`.

Kontrolu pristupa podskupu kolona i vrsta je moguće kontrolisati korišćenjem pogleda:

The screenshot shows two Oracle SQL Developer windows. The left window, labeled '1', contains the following commands in the 'Worksheet' tab:
`REVOKE SELECT ON EMPLOYEE FROM U2;`
`CREATE VIEW U2EMPLOYEE AS
SELECT fullname
FROM EMPLOYEE e
WHERE e.ssn > 1;`
`GRANT SELECT ON U2EMPLOYEE TO U2;`

The right window, labeled '2', shows the results of executing the view. The 'Worksheet' tab contains the command: `SELECT * FROM U1.U2EMPLOYEE;`. The 'Query Result' tab shows the output:

FULLNAME
1 Novi Korisnik
2 Pera Peric



OBAVEZNA KONTROLA PRISTUPA

Obavezna kontrola pristupa obezbeđuje implementaciju više sigurnosnih nivoa za potrebe zaštite sistema u organizacijama u kojima je to potrebno. Najčešće se koristi u kombinaciji sa diskrecionom kontrolom pristupa.

Obično se definišu 4 sigurnosne klase:

- **stroga tajna (Top secret – TS)**
- **tajna (Secret – S),**
- **poverljivo (confidential – C)**
- **javno (unclassified – U),**

pri čemu je poređak klasa po nivou sigurnosti **TS ≥ S ≥ C ≥ U**.

Model za implementaciju više sigurnosnih nivoa koji se koristi u sistemima za upravljanje bazama podataka se zove **Bell-LaPadula** model, koji svakom subjektu (korisniku, nalogu, programu) i svakom objektu (relaciji, slogu, koloni, operaciji) dodeljuje odgovarajuću sigurnosnu klasu (jednu od 4 navedene). Ako sa $\text{class}(S)$ označimo klasu subjekta S, a sa $\text{class}(O)$ klasu objekta O, dva glavna sigurnosna principa ovog modela možemo definisati na sledeći način:

- **Osnovno svojstvo sigurnosti** – Subjekat S ne može da pristupi objektu O osim u slučaju kada je **$\text{class}(S) \geq \text{class}(O)$** .
- **Star property** – Subjekat S ne može da vrši upis u objekat O, osim u slučaju kada je **$\text{class}(S) \leq \text{class}(O)$** .



ROLE-BASED KONTROLA PRISTUPA

Umesto da definišemo privilegije pojedinačno za svakog korisnika, mnogo je lakše definisati konkretne **uloge** koje korisnici imaju u sistemu – ove uloge uglavnom oslikavaju ulogu koju korisnik ima u samoj organizaciji. Tada je moguće grupisati privilegije i korisnike u uloge, čime se upravljanje privilegijama značajno olakšava. Uloge se kreiraju naredbom **CREATE_ROLE**.

The screenshot shows two windows of Oracle SQL Developer. The left window (labeled 1) displays a SQL script in the Worksheet tab:

```
CREATE ROLE hr_manager;
GRANT SELECT ON EMPLOYEE TO hr_manager;
GRANT hr_manager TO U4;
```

The right window (labeled 2) shows the results of a query in the Worksheet tab:

```
SELECT * FROM EMPLOYEE;
```

The Query Result tab shows the following data:

	SSN	FULLNAME	ADDRESS	SEX
1	3	Novi Korisnik	(null)	F
2	1	Uros Pesic	Addr1	M
3	2	Pera Peric	Addr2	M

Uloge je moguće i hijerarhijski organizovati, tako da uloge u bazi u potpunosti oslikavaju i uloge koje korisnici u samoj organizaciji.



ENKRIPCIJA PODATAKA

Kada se pomenute mere kontrole zaobiđu na bilo koji način, korisno je da podaci budu enkriptovani, kako bi se sprečilo njihovo čitanje od strane neautorizovanih korisnika. Enkripcija se može vršiti na više nivoa:

- **Transparentna enkripcija podataka (TDE)** – Svi podaci u bazi podataka koji „miruju“, tj. trenutno se ne čitaju, ažuriraju, ili šalju kroz mrežu, se čuvaju enkriptovani na disku. Pojam „transparentna“ se odnosi na činjenicu da korisnik nije svestan ove enkripcije – podaci se automatski dešifruju pre učitavanja u glavnu memoriju. Za ovaj vid enkripcije se koriste algoritmi koji koriste simetrični ključ, koji se još i naziva ključ za enkripciju baze podataka.
- **Enkripcija individualnih kolona** – Kod ovog pristupa se svaka kolona enkriptuje zasebno, čime se postiže veća fleksibilnost prilikom izbora algoritama i ključeva. Takođe je moguće korišćenje različitih ključeva za različite kolone, čime se otežava dešifrovanje podataka brute-force metodama. Glavni nedostatak ovakve enkripcije je veliki gubitak u brzini dešifrovanja prilikom učitavanja podataka, jer je potrebno dešifrovati svaku kolonu zasebno.

Oracle podržava oba tipa enkripcije, kao i enkripciju podataka koji se šalju kroz mrežu (**Native Network Encryption**).



ORACLE - ANALIZA PRIVILEGIJA

Analiza privilegija predstavlja praćenje upotrebe privilegija od strane različitih korisnika i različitih uloga u definisanom vremenskom okviru, kako bi se detektovale neiskorišćene privilegije i tako ispoštovao model **minimalnih potrebnih privilegija**.

Oracle podržava 4 tipa načina analize privilegija:

- **Analiza na osnovu konteksta (context-based)** – Korišćenjem SYS_CONTEXT funkcije se definiše uslov i evidencija o privilegijama se beleži samo ukoliko je uslov ispunjen. Na ovaj način je moguće analizirati privilegije za specifičnog korisnika, za neku adresu hosta, itd.
- **Analiza zasnovana na ulogama** – Analizira se upotreba privilegija samo za konkretnu listu uloga, koja se navodi prilikom definisanja pravila za analizu.
- **Analiza na osnovu uloge i konteksta** – Predstavlja kombinaciju prva dva tipa analize.
- **Analiza na nivou baze (bezuslovna analiza)** – Analiza privilegija se vrši na nivou cele baze: za sve korisnike, tj. uloge u svim kontekstima. Jedini nalog za koji se ne vodi evidencija o privilegijama u ovom slučaju je SYS nalog.



ORACLE - ANALIZA PRIVILEGIJA

Nalog sa koga se kreira i pokreće analiza privilegija mora da poseduje **CAPTURE_ADMIN** privilegiju. Sve funkcije neophodne za definisanje pravila analize, tipa analize, pokretanje i ostale operacije su deo DBMS_PRIVILEGE_CAPTURE PL/SQL paketa u Oracle-u.

The screenshot shows the Oracle SQL Developer interface. The top window is a Worksheet containing PL/SQL code. The code performs the following steps:

- Checks if the user has the CAPTURE_ADMIN role: `SELECT sys_context('SYS_SESSION_ROLES', 'CAPTURE_ADMIN') has_role FROM dual;`
- Creates a new capture named "All Database Capture" with type G_DATABASE: `DBMS_PRIVILEGE_CAPTURE.CREATE_CAPTURE(name => 'All Database Capture', description => 'My PA Policy', type => DBMS_PRIVILEGE_CAPTURE.G_DATABASE);`
- Enables the capture: `DBMS_PRIVILEGE_CAPTURE.ENABLE_CAPTURE (name => 'All Database Capture');`
- Checks if the capture was successfully created: `SELECT NAME, TYPE, ENABLED from DBA_PRIV_CAPTURES WHERE NAME = 'All Database Capture';`

The bottom window is a Script Output window showing the results of the execution. It displays the output of the first query (HAS_ROLE) and the last two queries (PL/SQL procedure successfully completed). A table is also shown with the following data:

NAME	TYPE	E
All Database Capture	DATABASE	Y



ORACLE - REZULTATI ANALIZE PRIVILEGIJA

Na slici možemo da vidimo mali deo rezultata analize korišćenjem Oracle Enterprise Manager-a.

Privilege Analysis: Reports

Summary [Unused](#) [Used](#) [Return](#)

Grantee	Type	Used	Revoked	System Privileges		Object Privileges	
				Unused	Used	Unused	Used
► CTXSYS	User			20		48	
► PA_ADMIN	User			11	1	42	5
► ORACLE_OCM	User			2		57	
► WMSYS	User			30		15	
► AUDSYS	User			7		25	4
► DVSYS	User			4		28	
▲ PU_PETE	User			11	1	5	5
▲ System Privileges	Folder			2	1		
UNLIMITED TABLESPACE							
SELECT ANY TABLE							
CREATE SESSION			✓				



OBJEDINJENA REVIZIJA (UNIFIED AUDITING)

DBMS vodi evidenciju o svim operacijama nad bazom koje korisnik izvršava u toku jedne **sesije**. Posebno je bitno pratiti operacije kojima se modifikuju podaci u sistemu. Svaka operacija koja se izvrši se upisuje u sistemski log fajl, zajedno sa brojem naloga i ID-em uređaja sa kog je izvršena. Ukoliko se posumnja da je došlo do nedozvoljenog manipulisanja podacima, pregledavanjem **log** fajla, tj. **revizijom baze podataka** (eng. **Database audit**) je moguće utvrditi koji korisnik je za to odgovoran.

Oracle **objedinjena revizija** omogućava prikupljanje informacija o događajima u sistemu iz više izvora u jedinstvenu **AUD\$UNIFIED** tabelu, koja je deo AUDSYS šeme. Izvori događaja za reviziju mogu biti:

- Događaji od interesa - definisani pravilima
- Database Vault događaju
- Oracle Label Security događaji
- Događaji iz Oracle Recovery Manager-a

Zahvaljujući činjenici da se svi slogovi za reviziju iz svih izvora čuvaju na jednom mestu, proces revizije je znatno lakši.



OBJEDINJENA REVIZIJA - PRIMERI

Korisnik koji kreira pravila za beleženje događaja za reviziju mora da ima privilegiju **AUDIT_ADMIN**, dok korisnici koji vrše reviziju pregledavanjem logova moraju da imaju dodeljenu privilegiju **AUDIT_VIEWER**.

Na sledećem primeru je demonstrirano kreiranje pravila za beleženje operacija koje nisu izdate korišćenjem aplikacije:

The screenshot shows the Oracle SQL Developer interface with the 'Worksheet' tab selected. The code area contains the following PL/SQL script:

```
-- proveravamo da li je unified auditing ukljucen
SELECT PARAMETER, VALUE FROM v$option WHERE PARAMETER='Unified Auditing';

-- kreira se policy koji prati sve operacije nad tabelama DEMO_HR_EMPLOYEES i DEMO_HR_USERS koje ne dolaze iz web aplikacije
-- ovaj uslov je definisan u when delu
CREATE AUDIT POLICY AUDIT_EMPLOYEESEARCH_USAGE
ACTIONS ALL ON EMPLOYEESEARCH_PROD.DEMO_HR_EMPLOYEES, ALL ON EMPLOYEESEARCH_PROD.DEMO_HR_USERS
WHEN 'SYS_CONTEXT(''USERENV'', ''SESSION_USER'') = ''EMPLOYEESEARCH_PROD''
    AND (''SYS_CONTEXT(''USERENV'', ''OS_USER'') != ''oracle''
    OR SYS_CONTEXT(''USERENV'', ''MODULE'') != ''JDBC Thin Client''
    OR SYS_CONTEXT(''USERENV'', ''HOST'') != ''dbsec-lab.dbsecvcn.oraclevcn.com'')
EVALUATE PER STATEMENT;

-- enable policy
AUDIT POLICY audit_employeesearch_usage;
```

The 'Script Output' tab at the bottom shows the results of the execution:

PARAMETER	VALUE
Unified Auditing	TRUE

The message "Audit POLICY created." is displayed at the bottom of the output.



OBJEDINJENA REVIZIJA - PRIMERI

Nakon izvršavanja upita nad tabelama DEMO_HR_EMPLOYEES, možemo pregledati evidentirane operacije:

The screenshot shows two Oracle SQL Developer sessions. The top session is titled 'empprod' and contains three queries:

```
-- primer 1
select userid, firstname, lastname, emptype, position, ssn, sin, nino
from employeesearch_prod.demo_hr_employees
where rownum < 10;

select * from EMPLOYEESEARCH_PROD.DEMO_HR_EMPLOYEES;

select firstname, lastname from EMPLOYEESEARCH_PROD.DEMO_HR_EMPLOYEES;
```

The bottom session is also titled 'empprod' and shows the audit log from the 'unified_audit_trail' table:

```
select event_timestamp, userhost, dbusername, action_name, object_schema, object_name, sql_text
from unified_audit_trail
where UNIFIED_AUDIT_POLICIES like '%AUDIT_EMPLOYEESEARCH_USAGE%'
order by 1;
```

In the 'Script Output' tab, the results of the audit query are displayed:

EVENT_TIMESTAMP	USERHOST	DBUSERNAME	ACTION_NAME	OBJECT_SCHEMA	OBJECT_NAME	SQL_TEXT
1 25-MAY-24 09.23.57.753051000 PM	dbsec-lab	EMPLOYEESEARCH_PROD	SELECT	EMPLOYEESEARCH_PROD	DEMO_HR_EMPLOYEES	select userid, firstname, lastname,
2 25-MAY-24 09.30.24.996486000 PM	dbsec-lab	EMPLOYEESEARCH_PROD	SELECT	EMPLOYEESEARCH_PROD	DEMO_HR_EMPLOYEES	select * from EMPLOYEESEARCH_PROD.DE
3 25-MAY-24 09.30.27.173532000 PM	dbsec-lab	EMPLOYEESEARCH_PROD	SELECT	EMPLOYEESEARCH_PROD	DEMO_HR_EMPLOYEES	select firstname, lastname from EMPL



OBJEDINJENA REVIZIJA - PRIMERI

Takođe, moguće je konfigurisati pravila kojima se evidentiraju operacije koje izvršavaju korisnici sa nekom specifičnom ulogom:

The screenshot shows two SQL Workspaces in Oracle SQL Developer.

Workspace 1 (Top):

```
create role mgr_role;
grant create tablespace to mgr_role;
grant mgr_role, create session to dba_nicole;
create user dba_junior identified by "Oracle123";
grant dba to dba_junior;

create audit policy aud_role_pol ROLES mgr_role;
create audit policy aud_db_a_pol ROLES dba;
audit policy AUD_ROLE_POL;
audit policy AUD_DB_A_POL;

select POLICY_NAME, AUDIT_OPTION, CONDITION_EVAL_OPT from AUDIT_UNIFIED_POLICIES where POLICY_NAME in ('AUD_ROLE_POL', 'AUD_DB_A_POL');
```

Output (Script Output):

POLICY_NAME	AUDIT_OPTION	CONDITION_EVAL_OPT
1 AUD_ROLE_POL	MGR_ROLE	NONE
2 AUD_DB_A_POL	DBA	NONE

Workspace 2 (Bottom):

```
select action_name, dbusername, object_name, OBJECT_SCHEMA, SQL_TEXT
from unified_audit_trail
where unified_audit_policies like '%AUD_DB_A_POL%'
      or unified_audit_policies like '%AUD_ROLE_POL%'
order by event_timestamp desc;
```

Output (Script Output):

ACTION_NAME	DBUSERNAME	ACTION_NAME_1	OBJECT_SCHEMA	OBJECT_NAME	SQL_TEXT
1 ALTER SYSTEM	DBA_JUNIOR	ALTER SYSTEM	(null)	(null)	alter system set job_queue_processes=100
2 ALTER SYSTEM	DBA_JUNIOR	ALTER SYSTEM	(null)	(null)	alter system set job_queue_processes=200
3 LOGON	DBA_JUNIOR	LOGON	(null)	(null)	(null)
4 DROP TABLESPACE	SYS	DROP TABLESPACE	(null)	TEST	drop tablespace test including contents and datafiles
5 CREATE TABLESPACE	SYS	CREATE TABLESPACE	(null)	TEST	create tablespace test datafile '/u01/oradata/cdb1/pdb1/test01.dbf'



DATABASE VAULT

Oracle **Database Vault (sef baze podataka)** je rešenje koje se isporučuje uz Enterprise verziju Oracle baze i predstavlja dodatni vid zaštite sistema od neautorizovanog pristupa, pomaže u sprečavanju zloupotrebe privilegija i može da zaštitи sistem od nemernih ljudskih grešaka. Komponente Database Vault-a koje služe za poboljšanje sigurnosti su:

- **Realms** – Predstavljaju „sigurnosne zone“ koje mogu da obuhvataju šeme, tabele, uloge i druge objekte sistema. Korišćenjem sigurnosnih zona je obezbeđena fleksibilnija kontrola pristupa zaštićenim objektima. Na primer, ukoliko je potrebno moguće je zaštiti šeme u sistemu koje sadrže tabele sa osetljivim podacima, i sprečiti pristup njima, čak i od strane autorizovanih korisnika (čak i od strane DBA naloga).
- **Komandna pravila** – Predstavljaju specijalni tip pravila, kojima je moguće definisati kada i pod kojim uslovima je moguće izvršiti bilo koju SQL naredbu. Na primer, moguće je sprečavati modifikaciju neke tabele u sistemu van vremenskog perioda 09h – 17h (van radnog vremena), jer bi to predstavljalo neki vid sumnjive modifikacije.
- **Skup pravila** – Skup individualnih pravila koja se koriste za kreiranje prostora zaštite i komandnih pravila. Ukoliko se skup sastoji iz više od jednog pravila, evaluaciju je moguće vršiti po principu sva pravila su ispunjena, ili bar jedno pravilo je ispunjeno
- **Faktori** – Različiti atributi, koji mogu biti IP adresa baze, username ulogovanog korisnika, čije vrednosti Database Vault koristi za evaluaciju pravila. Npr. na osnovu IP adrese sa kojeg je upućena naredba, moguće je proceniti da ona nije bezbedna za izvršenje.



DATABASE VAULT - REALM PRIMER

Oracle obezbeđuje PL/SQL pakete sa definisanim API-em za rad sa svakom od navedenih komponenti. Za rad sa Database Vault-om, potrebno je definisati par postojećih korisnika, od kojih će jednom biti dodeljena privilegija **DV_OWNER** – koja mu omogućava kreiranje i upravljanje komponentama Vault-a, dok će drugi imati privilegiju **DV_ACCTMGR** koja mu omogućava upravljanje korisničkim nalozima.

Inicijalno je potrebno configurisati Database Vault procedurom:

```
BEGIN  
DVSYS.CONFIGURE_DV(  
    dvowner_uname => 'C##DVOWNER',  
    dvacctmgr_uname => 'C##DVACCTMGR');  
END;
```

Izvršavanjem naredbe „SELECT * FROM dba_dv_status“ možemo se uveriti da je DV konfigurisan uspešno (**DV_ENABLE_STATUS** parametar):

NAME	STATUS
DV_APP_PROTECTION	NOT CONFIGURED
DV_CONFIGURE_STATUS	TRUE
DV_ENABLE_STATUS	TRUE



DATABASE VAULT - REALM PRIMER

Zatim kreiramo realm:

The screenshot shows the Oracle SQL Developer interface. The top navigation bar has tabs for 'Welcome Page', 'PDB1_SYSTEM', and 'dvowner'. The main area is a 'Worksheet' tab, which contains the following PL/SQL code:

```
begin
  DVSYS.DBMS_MACADM.CREATE_REALM(
    realm_name => 'PROTECT_EMPLOYEESEARCH_PROD'
  ,description => 'A mandatory realm to protect the EMPLOYEESEARCH_PROD schema.'
  ,enabled => DBMS_MACUTL.G_YES
  ,audit_options => DBMS_MACUTL.G_REALM_AUDIT_FAIL
  ,realm_type => 1);
END;
/
select name, description, enabled from dba_dv_realm where id# >= 5000 order by 1;
```

Below the worksheet, there are two tabs: 'Script Output' and 'Query Result'. The 'Query Result' tab shows the output of the last query:

All Rows Fetched: 1 in 0.005 seconds

NAME	DESCRIPTION	ENABLED
1 PROTECT_EMPLOYEESEARCH_PROD	A mandatory realm to protect the EMPLOYEESEARCH_PROD schema.	Y



DATABASE VAULT - REALM PRIMER

Definišemo koji objekti će biti zaštićeni realm-om:

```
begin
  DVSYS.DBMS_MACADM.ADD_OBJECT_TO_REALM(
    realm_name => 'PROTECT_EMPLOYEESEARCH_PROD'
  ,object_owner => 'EMPLOYEESEARCH_PROD'
  ,object_name => '%'
  ,object_type => '%');
end;
/
select realm_name, owner, object_name, object_type
from dvsys.dba_dv_realm_object
where realm_name in (select name from dvsys.dv$realm where id# >= 5000);
```

Script Output x Query Result x

SQL | All Rows Fetched: 1 in 0.039 seconds

REALM_NAME	OWNER	OBJECT_NAME	OBJECT_TYPE
1 PROTECT_EMPLOYEESEARCH_PROD	EMPLOYEESEARCH_PROD	%	%



DATABASE VAULT - REALM PRIMER

Kada je realm kreiran i objekti su dodeljeni, moguće je autorizovati određene korisnike, ili uloge:

```
begin
  DVSYS.DBMS_MACADM.ADD_AUTH_TO_REALM(
    realm_name => 'PROTECT_EMPLOYEESEARCH_PROD',
    grantee => 'EMPLOYEESEARCH_PROD',
    rule_set_name => '',
    auth_options => '1' );
end;
/
select realm_name, grantee, auth_options
from dvsys.dba_dv_realm_auth
where realm_name in (select name from dvsys.dv$realm where id# >= 5000);
```

The screenshot shows the Oracle SQL Developer interface. The top half displays an SQL script in the 'Script' tab, which creates a realm named 'PROTECT_EMPLOYEESEARCH_PROD' and grants it to the 'EMPLOYEESEARCH_PROD' user. The bottom half shows the 'Query Result' tab with the output of a query that lists the realm, grantee, and auth options. The result table has three columns: REALM_NAME, GRANTEE, and AUTH_OPTIONS. One row is shown: PROTECT_EMPLOYEESEARCH_PROD, EMPLOYEESEARCH_PROD, and Owner.

REALM_NAME	GRANTEE	AUTH_OPTIONS
PROTECT_EMPLOYEESEARCH_PROD	EMPLOYEESEARCH_PROD	Owner



DATABASE VAULT - REALM PRIMER

Ukoliko pokušamo da pristupimo sa neautorizovanog naloga objektu koji je zaštićen, dobijamo sledeću poruku:

The screenshot shows the Oracle SQL Developer interface. The top menu bar has tabs for 'Welcome Page', 'PDB1_SYSTEM', and 'dvowner'. Below the menu is a toolbar with various icons. The main workspace is a 'Worksheet' tab, which contains the following SQL query:

```
select userid, firstname, lastname, emptytype, position, ssn, sin, nino
from employeesearch_prod.demo_hr_employees
where rownum < 10;
```

Below the worksheet, there are two tabs: 'Script Output' and 'Query Result'. The 'Query Result' tab is active and displays the error message:

ORA-01031: insufficient privileges
01031. 00000 - "insufficient privileges"
*Cause: An attempt was made to perform a database operation without
the necessary privileges.
*Action: Ask your database administrator or designated security
administrator to grant you the necessary privileges



DATABASE VAULT - REALM PRIMER

Drugi primer demonstrira korišćenje DV-a za kontrolu putanja sa kojih je moguće prijavljivanje na sistem. Definisanjem „pouzdanih konekcija“ se sistem dodatno obezbeđuje u slučaju da hakeri uspeju da se dokopaju naloga sa jakim privilegijama.

Prvo je potrebno kreirati adekvatno pravilo, kojim se definiše koji korisnik i sa koje adrese hosta može da pristupi sistemu. Takođe potrebno je kreirati i skup pravila koji će da sadrži prethodno kreirano pravilo:

```
begin
    DVSYS.DBMS_MACADM.CREATE_RULE(
        rule_name => 'Application Connection',
        rule_expr => 'SYS_CONTEXT(''USERENV'', ''SESSION_USER'') = ''EMPLOYEESEARCH_PROD''
                      AND SYS_CONTEXT(''USERENV'', ''OS_USER'') = ''oracle''
                      AND SYS_CONTEXT(''USERENV'', ''MODULE'') = ''JDBC Thin Client''
                      AND SYS_CONTEXT(''USERENV'', ''HOST'') = ''dbsec-lab''');
end;
/

begin
    DVSYS.DBMS_MACADM.CREATE_RULE_SET(
        rule_set_name => 'Trusted Application Path'
        ,description => 'Protecting the App User'
        ,enabled => DBMS_MACUTL.G_YES
        ,eval_options => DBMS_MACUTL.G_RULESET_EVAL_ALL
        ,audit_options => DBMS_MACUTL.G_RULESET_AUDIT_FAIL
        ,fail_options => DBMS_MACUTL.G_RULESET_FAIL_SHOW
        ,fail_message => 'You cannot use the app account this way.'
        ,fail_code => -20000
        ,handler_options => null
        ,handler => null
        ,is_static => TRUE);
end;
/
```



DATABASE VAULT - REALM PRIMER

Zatim dodajemo kreirano pravilo u skup pravila:

The screenshot shows the Oracle SQL Developer interface. In the top-left corner, there are two tabs: "Worksheet" and "Query Builder". The "Worksheet" tab is active. Below the tabs, the code editor contains the following PL/SQL block:

```
begin
  DVSYS.DBMS_MACADM.ADD_RULE_TO_RULE_SET(
    rule_set_name  => 'Trusted Application Path'
  ,rule_name      => 'Application Connection'
  ,rule_order     => 1
  ,enabled        => DBMS_MACUTL.G_YES);
end;
/
SELECT rule_set_name, enabled, eval_options_meaning, audit_options, fail_message, fail_code, is_static
FROM DBA_DV_RULE_SET
WHERE rule_set_name = 'Trusted Application Path';
```

Below the code editor, there are two tabs: "Script Output" and "Query Result". The "Query Result" tab is selected. It displays the results of the SELECT query:

RULE_SET_NAME	ENABLED	EVAL_OPTIONS_MEANING	AUDIT_OPTIONS	FAIL_MESSAGE	FAIL_CODE	IS_STATIC
1 Trusted Application Path	Y	All True		1 You cannot use the app account this way.	-20000	TRUE

I kreiramo CONNECT komandno pravilo, kojim će svaka konekcija koja ne zadovoljava pravilo kreirano biti odbijena, bez obzira na privilegije korisnika:

The screenshot shows the Oracle SQL Developer interface. In the top-left corner, there are two tabs: "Worksheet" and "Query Builder". The "Worksheet" tab is active. Below the tabs, the code editor contains the following PL/SQL block:

```
begin
  DVSYS.DBMS_MACADM.CREATE_CONNECT_COMMAND_RULE(
    user_name      => 'EMPLOYEESEARCH_PROD'
  ,rule_set_name  => 'Trusted Application Path'
  ,enabled        => DBMS_MACUTL.G_YES);
end;
/
select command, object_owner, object_name, rule_set_name from dba_dv_command_rule where command = 'CONNECT';
```

Below the code editor, there are two tabs: "Script Output" and "Query Result". The "Query Result" tab is selected. It displays the results of the SELECT query:

COMMAND	OBJECT_OWNER	OBJECT_NAME	RULE_SET_NAME
1 CONNECT	EMPLOYEESEARCH_PROD %		Trusted Application Path



DATABASE VAULT - REALM PRIMER

Ukoliko probamo da se prijavimo na sistem izvan aplikacije na adresi dbsec-lab (što je definisano pravilom) dobijamo sledeću poruku:



Database Vault, osim pomenutih, poseduje veliki skup drugih mogućnosti. Još jedna od interesantnijih jeste korišćenje simulacionog moda. U ovom modu DV prati operacije shodno definisanom komandnom pravilu, međutim usled njegovog narušavanja se neće sprečiti pristup, već će se samo evidentirati. Simulacioni mod se koristi tako što se parametru *enabled* u proceduri **CREATE_COMMAND_RULE** dodeli vrednost **DVMS_MACUTL.G_SIMULATION**.



ORACLE LABEL SECURITY

Oracle labele predstavljaju Oracle-ovu implementaciju obavezne kontrole pristupa. Dostupne su samo za enterprise verziju baze podataka. Glavna ideja OLS-a (Oracle Label Security) je dodeljivanje labela individualnim vrstama i korisnicima. Prilikom pristupa nekog korisnika vrstama labelirane tabele, autorizacija se obavlja upoređivanjem njegove labele sa vrednostima labele svake vrste.

Svaka labela u OLS-u se sastoji od sledećih komponenti:

- **Nivo (Level)** – Odnosi se na sigurnosnu klasu. Predstavlja se celim brojem, gde veći broj označava veću poverljivost podataka. Svaka labela mora da ima naveden nivo.
- **Odeljak (Compartment)** – Svaka labela može imati pridružen odeljak kojem pripada vrsta uz koju ona stoji. Na primer, podaci mogu imati tri sigurnosna nivoa (javni, tajni, strogo tajni), a da pritom pripadaju različitim timova u kompaniji (IT, HR, tim za finansije)
- **Grupa (Group)** – Grupom se definiše organizacija kojoj podaci pripadaju i koja sme da im pristupi. Na primer, ukoliko imamo jedinstveni server na kojem se čuvaju podaci organizacije, koji su prikupljeni sa različitih teritorija (EU, Severna Amerika, Azija) i moraju biti tako podeljeni, moguće je pridružiti im oznaku grupe, tako da podaci koji pripadaju jednom regionu ne mogu biti dostupni korisnicima iz drugih regionala. Grupe mogu biti hijerarhijski organizovane.

Labele imaju sledeću formu: **LEVEL [:COMPARTMENT1, ... , COMPARTMENTn : GROUP1, ... , GROUPm]**



OLS - PRIMER

1. Potrebno je omogućiti OLS sledećim komandama:

```
EXEC LBACSYS.CONFIGURE_OLS;  
EXEC LBACSYS.OLS_ENFORCEMENT_ENABLE_OLS;
```

2. Kreiramo OLS procedurom **CREATE_POLICY** iz **SA_SYSDBA** PL/SQL paketa:

```
BEGIN  
LBACSYS.SA_SYSDBA.CREATE_POLICY(  
    policy_name => 'OLS_DEMO_HR_APP',  
    column_name => 'OLSLABEL',  
    default_options => 'READ_CONTROL,  
WRITE_CONTROL, LABEL_DEFAULT,  
HIDE');  
END;
```



3. Nakon kreiranja policy-ja potrebno je definisati sigurnosne nivoje:

```
BEGIN  
LBACSYS.SA_COMPONENTS.CREATE_LEVEL(  
    policy_name => 'OLS_DEMO_HR_APP',  
    level_num => 1000,  
    short_name => 'P',  
    long_name => 'Public');  
END;
```

4. Kreiramo odeljke:

```
BEGIN  
LBACSYS.SA_COMPONENTS.CREATE_COMPARTMENT(  
    policy_name => 'OLS_DEMO_HR_APP',  
    comp_num => 400,  
    short_name => 'FIN',  
    long_name => 'FINANCE');  
END;
```



OLS - PRIMER

5. Kreiramo grupe:

```
BEGIN  
LBACSYS.SA_COMPONENTS.CREATE_GROUP(  
    policy_name => 'OLS_DEMO_HR_APP',  
    group_num => 1100,  
    short_name => 'USA',  
    long_name => 'USA',  
    parent_name => 'GLOBAL');  
END;
```

6. Koristimo kreirane komponente da bismo definisali labele koje će postojati u našem sistemu:

```
BEGIN  
LBACSYS.SA_COMPONENTS.CREATE_LABEL(  
    policy_name => 'OLS_DEMO_HR_APP',  
    label_tag => 1100,  
    label_value => 'P:USA',  
    data_label => TRUE);  
END;
```

Svi primeri upita su prikazani radi demonstracije. U sistemu su kreirane labele za svaku kombinaciju nivoa, odeljka i grupe. Takođe, su kreirana dva korisnika *can_candy* kojoj je dodeljena labela **P:CAN** i *eu_evan* sa labelom **P:EU**. Podaci su nasumično labelirani.



OLS - PRIMER

7. Nakon što su svi podaci labelirani, možemo uključiti policy:

```
BEGIN  
LBACSYS.SA_POLICY_ADMIN.APPLY_TABLE_POLICY(  
    policy_name => 'OLS_DEMO_HR_APP',  
    schema_name => 'EMPLOYEESEARCH_PROD',  
    table_name => 'DEMO_HR_EMPLOYEES');  
END;
```

Rezultati pretrage tabele DEMO_HR_EMPLOYEES za različite korisnike izledaju ovako:

OLS_READ_LABEL	
P:::CAN	
COUNT(*)	162
... How many cities can be seen by app user CAN_CANDY	
CITY	COUNT_CITY
Toronto	162
OLS_READ_LABEL	
P:::EU,GER	
COUNT(*)	368
... How many cities can be seen by app user EU_EVAN	
CITY	COUNT_CITY
Berlin	125
London	122
Paris	121



TRANSPARENT SENSITIVE DATA PROTECTION - TSDP

TSDP je tehnika koju Oracle koristi za sakrivanje poverljivih podataka. Podrazumeva identifikaciju kolona osetljivih podataka i kreiranje pravila kojima se definiše na koji način će se sakriti informacije koje ovi podaci nose. TSDP koristi tehnike **redakcije podataka**. Postoje različiti načini sakrivanja podataka korišćenjem redakcije:

- **Potpuna redakcija** – Sav sadržaj kolona se sakriva. Način sakrivanja zavisi od tipa podataka kolone: kolone sa tipa NUMBER se menjaju nulom, dok se kolone tipa VARCHAR menjaju praznim stringom.
- **Parcijalna redakcija** – Prikriva se samo deo vrednosti. Na primer, broj kartice je moguće prekriti tako što će deo cifara biti zamenjen nekim specijalnim karakterom.
- **Redakcija korišćenjem regularnih izraza** – Koristi se za sakrivanje kolona koje imaju definisani strukturu, ali mogu biti različite dužine, kao što je npr slučaj sa email-ovima.
- **Nasumična redakcija** – Podaci se prikrivaju tako što se menjaju nekom nasumično generisanom vrednošću.
- **Null redakcija** – Sve vrednosti u koloni koja je označena kao „osetljiva“ se menjaju vrednošću null.



TSDP - PRIMER

1. Definišemo osetljivi tip podataka i dodeljujemo ih odgovarajućim kolonama:

```
-- kreiramo "osetljivi" tip podataka
BEGIN
    DBMS_TSDP_MANAGE.ADD_SENSITIVE_TYPE (
        sensitive_type  => 'credit_card_type',
        user_comment     => 'Type for Credit Card columns using a Varchar2 data type');
END;
/

-- oznacavamo kolonu CORPORATE_CARD kao osetljivi tip
BEGIN
    DBMS_TSDP_MANAGE.ADD_SENSITIVE_COLUMN(
        schema_name      => 'tsdp_labs',
        table_name       => 'TSDP_HR_EMPLOYEES',
        column_name      => 'CORPORATE_CARD',
        sensitive_type   => 'credit_card_type',
        user_comment     => 'Sensitive column addition of credit_card_type');
END;
/
```



TSDP - PRIMER

2. Definišemo način na koji će se parcijalna redakcija izvršiti i kada će se primeniti:

The screenshot shows a SQL developer interface with a tab titled 'TSDP_ADMIN'. The code area contains the following PL/SQL script:

```
DECLARE
    redact_feature_options DBMS_TSDP_PROTECT.FEATURE_OPTIONS;
    policy_conditions DBMS_TSDP_PROTECT.POLICY_CONDITIONS;
BEGIN
    redact_feature_options ('expression') := 'SYS_CONTEXT(''USERENV'', ''SESSION_USER'') = ''TSDP_LABS''';
    redact_feature_options ('function_type') := 'DBMS_REDACT.PARTIAL';
    redact_feature_options ('function_parameters') := 'VVVVFVVVVFVVVVFVVVV, VVVV-VVVV-VVVV-VVVV,*1,8';
    policy_conditions(DBMS_TSDP_PROTECT.DATATYPE) := 'VARCHAR2';
    DBMS_TSDP_PROTECT.ADD_POLICY ('redact_partial_cc', DBMS_TSDP_PROTECT.REDACT, redact_feature_options, policy_conditions);
END;
/

BEGIN
    DBMS_TSDP_PROTECT.ASSOCIATE_POLICY(
        policy_name => 'redact_partial_cc',
        sensitive_type => 'credit_card_type',
        associate => true);
END;
/
```



TSDP - PRIMER

3. Kreirani policy vezujemo za prethodno kreirani osetljivi tip podataka i aktiviramo ga:

```
-- primjenjujemo kreirani policy na prethodno definisani osetljivi tip
BEGIN
  DBMS_TSDP_PROTECT.ASSOCIATE_POLICY(
    policy_name => 'redact_partial_cc',
    sensitive_type => 'credit_card_type',
    associate => true);
END;
/
-- aktiviramo policy
BEGIN
  DBMS_TSDP_PROTECT.ENABLE_PROTECTION_TYPE(
    sensitive_type      => 'credit_card_type');
END;
```

Rezultati pretrage podataka osetljivog tipa:

The screenshot shows the Oracle SQL Developer interface. The top menu bar has tabs for 'Welcome Page', 'TSDP_ADMIN', and 'TSDP_LABS'. The toolbar below includes icons for running queries, saving, and connecting. The main area has two tabs: 'Worksheet' and 'Query Builder', with 'Worksheet' selected. A query is entered in the worksheet pane:

```
SELECT userid, firstname, lastname, corporate_card FROM tsdp_hr_employees WHERE length(corporate_card)=19 order by 1;
```

Below the worksheet is a 'Query Result' pane. It shows a table with three rows of data:

	USERID	FIRSTNAME	LASTNAME	CORPORATE_CARD
1	413	Kathy	Allen	****-****-5347-0000
2	449	Donna	Wright	****-****-5347-0000
3	467	Martin	Lawrence	****-****-1550-0000

The status bar at the bottom of the interface indicates: 'SQL | All Rows Fetched: 16 in 0.011 seconds'.



ORACLE DBSAT - SECURITY ASSESSMENT TOOL

DBSAT je dodatni alat, koji se danas vrlo često koristi kao prateći deo sistema koji umnogome olakšava posao održavanja sigurnosti na maksimalnom mogućem nivou. Koristi se za brzu i efikasnu procenu trenutne sigurnosti sistema, može lako da otkrije koji su osjetljivi podaci u sistemu koje je potrebno zaštititi i promoviše dobre sigurnosne prakse, koje bi trebalo primeniti. Sastoji se od 3 komponente:

- **Collector** – Izvršava SQL upite i komande operativnog sistema, kako bi prikupio neophodne informacije o trenutnom stanju sistema. Rezultat kolekcije informacija je json fajl koji reporter kasnije koristi kako bi izvršio analizu.
- **Reporter** – Analizira prikupljene podatke i generiše izveštaj u Excel, HTML, json i tekstualnom formatu.
- **Discoverer** – Koristi se za analizu i otkrivanje osjetljivih podataka u sistemu.



ORACLE DBSAT - REPORTER

Oracle Database Security Assessment

Highly Sensitive

Assessment Date & Time

Date of Data Collection	Date of Report	Reporter Version
Thu May 23 2024 14:35:16 UTC+00:00	Thu May 23 2024 14:39:07 UTC+00:00	3.1 (Jan 2024) - b73a

Database Identity

Name	Container (Type:ID)	Platform	Database Role	Log Mode	Created
CDB1	PDB1 (PDB:3)	Linux x86 64-bit	PRIMARY	NOARCHIVELOG	Wed Oct 30 2019 15:41:51 UTC+00

Summary

Section	Pass	Evaluate	Advisory	Low Risk	Medium Risk	High Risk	Total Findings
Basic Information	1	0	0	0	0	0	0
User Accounts	6	10	1	5	2	0	0
Privileges and Roles	4	25	1	0	0	0	0
Authorization Control	0	3	2	0	0	0	0
Fine-Grained Access Control	0	0	5	0	0	0	0
Auditing	6	8	2	0	0	0	0
Encryption	0	4	0	0	0	0	0
Database Configuration	8	8	0	1	3	1	0
Network Configuration	1	0	3	1	0	0	0
Operating System	2	4	0	1	2	0	0
Total	28	62	14	8	7	1	

Sample Schemas

USER.SAMPLE	
Sample schemas should be dropped	
Status	Low Risk
Summary	Found 2 sample schemas.
Details	Sample schemas: HR, SCOTT
Remarks	Sample schemas are well-known accounts provided by Oracle to serve a purpose in a production database and should be dropped. They generally serve no purpose in a production database and should be dropped to reduce the attack surface of the database.
References	Oracle Best Practice CIS Benchmark: Recommendation 4.2 DISA STIG: V-220284

Status	Advisory
Status	Advisory
Summary	Database Vault is not enabled.
Remarks	Database Vault offers customizable policies to regulate the actions of users and applications. It can limit the ability of users to exploit privileged account credentials to access sensitive information or prevent users from accessing sensitive data from unauthorized access, even by users with system privileges. Database Vault limit accidental or malicious execution of SQL commands and prevent a single user from having all system-level privileges. It also provides a set of duties to prevent a single all-powerful user and use trusted paths to access sensitive data based on system factors such as IP address, program name, and user ID. Database Vault Operations Control can be used to restrict common users from accessing pluggable databases (PDB). Database Vault Operations Control can also prevent users from accessing pluggable database (PDB) local data in autonomous environments.



ORACLE DBSAT - DISCOVERER

Summary

Sensitive Category	# Sensitive Tables	# Sensitive Columns	# Sensitive Rows
BIOGRAPHIC INFO - ADDRESS	9	36	6307209
BIOGRAPHIC INFO - EXTENDED PII	2	2	2000
FINANCIAL INFO - BANK DATA	2	2	599
FINANCIAL INFO - CARD DATA	7	7	3004
HEALTH INFO - PROVIDER DATA	1	1	149
IDENTIFICATION INFO - NATIONAL IDS	2	6	2000
IDENTIFICATION INFO - PERSONAL IDS	3	3	405
IDENTIFICATION INFO - PUBLIC IDS	9	26	2401125
IT INFO - USER DATA	13	15	12997
JOB INFO - COMPENSATION DATA	10	12	3149
JOB INFO - EMPLOYEE DATA	8	16	406
JOB INFO - ORG DATA	5	6	278
TOTAL	29*	132	8617413*

Number of Unique Tables with Sensitive Data.

* Number of Unique Rows with Sensitive Data.

Risk Level: High Risk

Security for Environments with High Value Data: Detective plus Strong Preventive Controls

Highly sensitive and regulated data should be protected from privileged users, and from users without a business need for the data. Activity of privileged accounts should be controlled to protect against insider threats, stolen credentials, and human error. Who can access the database and what can be executed should be controlled by establishing a trusted path and applying command rules. Sensitive data should be redacted on application read only screens. A Database Firewall ensures that only approved SQL statements or access by trusted users reaches the database - blocking unknown SQL injection attacks and the use of stolen login credentials.

Recommended controls include:

- Audit all sensitive operations including privileged user activities
- Audit access to application data that bypasses the application
- Encrypt data to prevent out-of-band access
- Mask sensitive data for test and development environments
- Restrict database administrators from accessing highly sensitive data
- Block the use of application login credentials from outside of the application
- Monitor database activity for anomalies
- Detect and prevent SQL Injection attacks
- Evaluate: *Oracle Audit Vault and Database Firewall, Oracle Advanced Security, Oracle Data Masking and Subsetting, Oracle Database Vault*

Tables Detected within Sensitive Category: BIOGRAPHIC INFO - ADDRESS

Risk Level	High Risk
Summary	Found BIOGRAPHIC INFO - ADDRESS within 36 Column(s) in 9 Table(s)
Location	Tables: DMS_ADMIN.MASK_DATA, EMPLOYEESEARCH_DEV.DEMO_HR_EMPLOYEES, EMPLOYEESEARCH_PROD.DEMO_HR_EMPLOYEES, HCM1.COUNTRIES, HCM1.LOCATIONS, HR_COUNTRTES_HR_LOCATTONS_LOOKUP_LOOKUPS_LOOKUP_ADDRESSES_LOOKUP_PLACES

HVALANA

PAZNJI!