ORACLE VPD U PRIMJENI

Nataša Dvoršak
ULJANIK IRI d.o.o., Pula
e-mail: natasa.dvorsak@uljanik.hr

SAŽETAK

Razvoj i sve raširenija primjena informacijskih tehnologija, a posebno BI rješenja, dovode do situacija kada aplikacijska zaštita podataka više nije dovoljna. Bazi se pristupa iz različitih sustava/alata, te se otvara pitanje zaštite podataka spremljenih u bazi.
Zašto ne primijeniti već gotovo Oracle rješenje VPD (Virtual Private Database)?
U Oracle ver. 10gR2 uvedena su značajna poboljšanja performansi koja VPD sustav čine iskoristivim nad transakcijskim podacima.
Prikazat će se dizajn i izrada sustava zaštite u kojem djeluje nekoliko različitih sigurnosnih grupa, tehnike koje se primjenjuju za bolje performanse, različite mogućnosti filtriranja i prikaza podataka, bilježenje kritičnih aktivnosti na bazi uz FGA (Fine Grained Auditing).

Oracle VPD in the implementation

The development of information technology, and specifically BI solutions, are leading to situations where the application data protection is no longer sufficient. Database can be accessed from different systems / tools, the solution is in data protection in the database. Why not apply the Oracle out of the box solution VPD?
Oracle in 10gR2 introduced significant performance improvements that make the VPD usable over the transaction data.
It will be demostrated the design and implementation of security policies in which there are several different policy groups, the techniques applied to enhance performance, a variety of filtering options and masking data, recording of critical activities on the database using the FGA.

UVOD

Razvojem informacijske tehnologije rastu i apetiti korisnika za novim informacijama i izvještajima u svim oblicima. Sve je više informatički pismenih korisnika koji znaju raditi s izvještajnim alatima poput Oracle BI, Oracle Discoverera, SQL developera, MS Excela i slično. Zaštita podataka u takvom okruženju veoma je izazovni zadatak. U našem okruženju korisnici imaju jedinstveno korisničko ime (user na Oracle bazi zaštićen baznim rolama) koje koriste za rad s ERP sustavom, ali također i za konekciju na bazu s ostalim dostupnim alatima. Tablice kojima se pristupa namijenjene su za pohranu podataka različitih poslovnih subjekata. Pojedini korisnici ovlašteni su za pristup podacima jednog ili više poduzeća u sustavu. Sigurnost ERP sustava koncipirana je prema modelu zaštite na nivou poduzeća, dok je pristup podacima drugim kanalima manjkav. Osim interne zaštite podataka poslovnih subjekata, pred informatičare se stavljaju i zahtjevi za zaštitu od neovlaštenog pristupa osjetljivim podacima (npr. prema Zakonu o zaštiti osobnih podataka). Jedno od rješenja koje Oracle nudi je VPD (Virtual Private Database). VPD dolazi u standardnom paketu Enterprise edicije Oracle baze.

1 UVOD U VPD

1.1 Osnovne informacije

Oracle Virtual Private Database (VPD) je sigurnosni okvir koji je prvi puta implementiran u verziji baze 8i pod nazivom Fine Grained Access Control (FGAC). Osnovna značajka VPD-a je postizanje razine sigurnosti na nivou retka (Row Level Security – RLS). U izdanju baze 10g u VPD se dodaje i mogućnost zaštite na nivou stupaca. Standardno ponašanje zaštite stupaca svodi se na skrivanje redova s osjetljivim stupcima, a postoji i mogućnost prikazivanja svih redova sa skrivanjem osjetljivih stupaca (column masking). Skrivanje stupaca može se koristiti samo u SELECT naredbama. VPD pravila na razini stupaca ne mogu se primjenjivati nad synonymima.

VPD mehanizam temelji se na dinamičkoj modifikaciji naredbi za čitanje (SELECT) ili ažuriranje

(INSERT, UPDATE, DELETE) podataka tablica, viewa ili synonyma. Server baze podataka automatski modificira naredbu koju je korisnik poslao bazi tako da u WHERE klauzulu dodaje uvjete povezujući ih AND operatorom. Dodatne uvjete vraća funkcija koja je implementirana u okviru sigurnosnih pravila VPD-a. Ukoliko je nad istim objektom istovremeno aktivno više VPD sigurnosnih funkcija rezultati se povezuju AND operatorom u WHERE klauzuli. Dakle, postoji mogućnost da jedna sigurnosna politika uvjetuje isključenje druge i rezultat je tada prazan skup podataka.

1.2 Aplikacijski kontekst

Uz pojam VPD-a vezan je i pojam aplikacijskog konteksta (Application Context). Aplikacijski kontekst sastoji se od parova ključ-vrijednost koji se smještaju u sigurnu predmemoriju (cache) podataka. Ovisno o tome u kojem je memorijskom prostoru smješten razlikujemo globalni (System Global Area) i lokalni aplikacijski kontekst (User Global Area). Lokalni kontekst vidljiv je samo u okviru jedne bazne sesije, dok je globalni vidljiv svim sesijama baze. Kontekstne vrijednosti povezuju se i smještaju u grupe, tzv. namespace. Svaka sesija Oracle baze uvijek kreira zadani USERENV namespace sa svojim kontekstnim atributima, ali moguće je kreirati i vlastite grupe (namespace) sa svojim atributima.

U kontekstu VPD-a aplikacijski kontekst zanimljiv je iz dva glavna razloga:

- optimizacija izvođenja,
- implementacija vlastitih grupa sigurnosti.

Optimizacija izvođenja uz korištenje aplikacijskog konteksta moguća je zahvaljujući:

- brzom pristupu atributima konteksta,
- korištenju vrijednosti kontekstnih atributa u predikatima WHERE klauzule,
- mogućnosti kreiranja uvjeta koji imaju atribute slične "bind" varijablama.

Atributi konteksta smješteni su u posebnom memorijskom prostoru koji omogućava brzi pristup. Umjesto da se neprestano pristupa vrijednostima u baznim tablicama, ubrzanje se može postići čitanjem vrijednosti iz aplikacijskog konteksta. Dobra je praksa koristiti predikate WHERE klauzule s funkcijama koje vraćaju kontekstne vrijednosti, jer se na taj način uspijeva izbjeći korištenje konstanti u predikatima i kreiranje jednakog predikata za više korisnika sustava.

Primjer 1:.

```
select * from employees t where t.department_id = sys_context('EMP',
'DEPARTMENT_ID')
```

U verziji baze 10g uveden je i novi parametar "Policy Type" funkcije za kreiranje VPD sigurnosne politike DBMS_RLS.ADD_POLICY kojim se može utjecati na performanse servera baze.

Tabela 1: DBMS_RLS.ADD_POLICY utjecaj Policy Type parametra [1]

Policy Types	When Policy Function Executes...	Usage Example	Shared Across Multiple Objects?
STATIC	Once, then the predicate is cached in the SGA[1]	View replacement	No
SHARED_ STATIC	Same as STATIC	Hosting environments, such as data warehouses where the same predicate must be applied to multiple database objects	Yes
CONTEXT_ SENSITIVE	▪ At statement parse time ▪ At statement execution time when the local application context has changed since the last use of the cursor	3-tier, session pooling applications where policies enforce two or more predicates for different users or groups	No
SHARED_ CONTEXT_ SENSITIVE	First time the object is reference in a database session Predicates are cached in the private session memory UGA so policy functions can be shared among objects.	Same as CONTEXT_SENSITIVE, but multiple objects can share the policy function from the session UGA	Yes
DYNAMIC	Policy function re-executes every time a policy-protected database object is accessed.	Applications where policy predicates must be generated for each query, such as time-dependent policies where users are denied access to database objects at certain times during the day	No

[1] Each execution of the same cursor could produce a different row set even for the same predicate because the predicate may filter the data differently based on attributes such as SYS_CONTEXT or SYSDATE.

Još jedan razlog korištenja aplikacijskog konteksta u okviru VPD-a krije se u korištenju vlastitih grupa sigurnosne politike. Zadana (default) sigurnosna grupa je SYS_DEFAULT. Sigurnosna politika VPD-a definirana na grupi SYS_DEFAULT uvijek se izvodi. Vlastite VPD grupe kreiraju se kada se želi uvjetovati izvođenje neke VPD sigurnosne politike. Kreiranjem posebnih sigurnosnih grupa stvara se mogućnost istovremene aktivnosti sigurnosne politike samo jedne od ne-SYS_DEFAULT grupa. U jednoj korisničkoj sesiji uvijek je aktivna sigurnosna politika SYS_DEFAULT grupe i samo jedne ili svih vlastitih grupa. Koja je grupa aktivna definira se posebnim atributom aplikacijskog konteksta (Driving Context). Ukoliko Driving Context atribut nije definiran ili je vrijednost NULL tada će VPD mehanizam aktivirati sve sigurnosne politike.

1.3 Administracija VPD-a

Administracija VPD sigurnosnih pravila u potpunosti se obavlja uz pomoć paketa DBMS_RLS. Pomoću ovog paketa mogu se dodavati, aktivirati/deaktivirati, osvježavati i brisati sigurnosna pravila ili grupe VPD-a, te dodavati ili brisati aplikacijski kontekst. VPD pravila mogu se definirati nad tablicama, viewima ili synonymima baze.

Osnovni postupak kreiranja VPD pravila svodi se na:

- Kreiranje funkcije koja vraća predikat WHERE uvjeta (VARCHAR2), a prima dva parametra vlasnik i ime tablice.
- Pridruživanje funkcije tablici (DBMS_RLS.ADD_POLICY).

U radu s VPD grupama postupak je nešto složeniji:

- potrebno je kreirati aplikacijski kontekst (lokalni ili globalni)
- pridružiti tablici grupu (DBMS_RLS.CREATE_POLICY_GROUP)
- pridružiti tablici funkciju (DBMS_RLS.ADD_GROUPED_POLICY)
- pridružiti tablici atribut aplikacijskog konteksta, tzv. "driving context" tablici (DBMS_RLS.ADD_POLICY_CONTEXT)

Tabela 2: DBMS_RLS popis funkcija i procedura [1]

Procedure	Purpose
DBMS_RLS.ADD_POLICY	To add a policy to a table, view, or synonym
DBMS_RLS.ENABLE_POLICY	To enable (or disable) a policy you previously added to a table, view, or synonym
DBMS_RLS.REFRESH_POLICY	To invalidate cursors associated with non-static policies
DBMS_RLS.DROP_POLICY	To drop a policy from a table, view, or synonym
For Handling Grouped Policies	
DBMS_RLS.CREATE_POLICY_GROUP	To create a policy group
DBMS_RLS.DELETE_POLICY_GROUP	To drop a policy group
DBMS_RLS.ADD_GROUPED_POLICY	To add a policy to the specified policy group
DBMS_RLS.ENABLE_GROUPED_POLICY	To enable a policy within a group
DBMS_RLS.REFRESH_GROUPED_POLICY	To reparse the SQL statements associated with a refreshed policy
DBMS_RLS.DISABLE_GROUPED_POLICY	To disable a policy within a group
DBMS_RLS.DROP_GROUPED_POLICY	To drop a policy which is a member of the specified group
For Handling Application Context	
DBMS_RLS.ADD_POLICY_CONTEXT	To add the context for the active application
DBMS_RLS.DROP_POLICY_CONTEXT	To drop the context for the application

Osim ovog paketa za rad s VPD-em potreban je i DBMS_SESSION koji omogućava ažuriranje kontekstnih varijabli.

Primjer 2.

```
dbms_session.set_context(namespace => 'CONTEXT_VPD',
                         attribute => 'POLICY_GROUP',
                         value     => 'FIN');
```

1.4 Informacije o VPD okruženju

Sve informacije o postavljenom VPD okruženju na bazi smještene su u nekoliko tablica/viewa kataloga baze. Administrator VPD-a svakako treba konzultirati ove tablice kako bi dobio točnu sliku definiranog sustava zaštite, kao i za provjeru funkcioniranja sustava VPD-a (V$VPD_POLICY). U V$VPD_POLICY zapisani su predikati koji se dodaju WHERE klauzuli tablice/viewa/synonyma. View koji se u literaturi gotovo i ne spominje je USER/ALL/DBA_SEC_RELEVANT_COLS u kojem se nalaze podaci o stupcima tablica koje su uključene u VPD sigurnost.

Tabela 3: Popis "data dictionary" view-a koji opisuju VPD postavke [2]

View	*Description*
ALL_POLICIES	Shows all VPD policies on objects accessible to the current user
ALL_POLICY_CONTEXTS	Shows the contexts used within VPD policies defined for objects accessible to the current user
ALL_POLICY_GROUPS	Shows VPD policy groups defined for objects accessible to the current user
DBA_POLICIES	Shows all VPD policies
DBA_POLICY_CONTEXTS	Shows all contexts used within VPD policies
DBA_POLICY_GROUPS	Shows all VPD policy groups
USER_POLICIES	Shows all VPD policies on objects owned by the current user
USER_POLICY_CONTEXTS	Shows the contexts used within VPD policies defined for objects owned by the current user
USER_POLICY_GROUPS	Shows VPD policy groups defined for objects owned by the current user
V$VPD_POLICY	Shows all the policies and predicates associated with the cursors currently in the library cache

Za potrebe testiranja sustava vjerojatno ćete konzultirati i zapise u viewima V$CONTEXT i V$GLOBALCONTEXT.

```
select * from v$context;
select * from v$globalcontext;
select * from dba_policies;
select * from dba_policy_groups;
select * from dba_policy_contexts;
select * from dba_sec_relevant_cols;
select sql_id,
       object_name,
       policy_group,
       policy,
       policy_function_owner,
       predicate
  from v$vpd_policy;
```

Select v$context | Select v$globalcontext | Select dba_policies | Select dba_policy_groups | Select dba_policy_contexts | Select dba_sec_relevant_cols | Select v$vpd_policy

	SQL_ID	OBJECT_NAME	POLICY_GROUP	POLICY	POLICY_FUNCTION_OWNER	PREDICATE
1	2qqdsyqy4850h	KB_PROJEKTI	SYS_DEFAULT	KB_PROJEKT_ID	TABLEBOSS	1=1
2	2qqdsyqy4850h	KB_PROJEKTI	SYS_DEFAULT	KB_KALK_NAB	TABLEBOSS	1=1
3	art850nnzw5nn	KA_RADNICI	VPD_PL_SQL	KA_RADNICI_COL	TABLEBOSS	1=1
4	d9wzxmdqas71r	KA_RADNICI	VPD_PL_SQL	KA_RADNICI_COL	TABLEBOSS	1=1
5	837tp2m3jc8cd	KO_KOOPERANTI	SYS_DEFAULT	DRUSTVO_ID	TABLEBOSS	DRUSTVO_ID = SYS_CONTEXT('VPD_C
6	837tp2m3jc8cd	KO_KOOPERANTI	SYS_DEFAULT	DRUSTVO_ID	TABLEBOSS	1=1

Slika 1: Pogled na podatke u V$VPD_POLICY

2 VLASTITO RJEŠENJE

Implementacija VPD-a u našem okruženju imala je za cilj ugradnju mehanizama zaštite podataka na razini poduzeća koji neće zahtijevati velike zahvate u programskom rješenju. Sigurnosni zahtjevi trebaju biti zadovoljeni prilikom svakog pristupa podacima, dakle s bilo kojim alatom za manipuliranje podacima baze.

Nakon prijelaza na bazu 10g, VPD se sporadično koristi za zaštitu podataka pojedinih tablica i viewa koji su kreirani za potrebe korištenja u Oracle Discoverer analizama, te u iznimnim slučajevima zaštite izrazito osjetljivih podataka. Novo rješenje trebalo bi pokriti sve opće zahtjeve zaštite podataka na razini poduzeća (retka tablice) i zaštite osjetljivih podataka u tablicama. Osim VPD-a u našem okruženju koristi se i Fine grained auditing (FGA) za praćenje manipulacija podacima nad izrazito osjetljivim tablicama (npr. plaće).

U realizaciji rješenja vodili smo se idejom da poslovna pravila VPD sigurnosti budu što jednostavnija (tako savjetuju i u literaturi koju smo proučavali). Takvo razmišljanje išlo je u smjeru da se izrade dvije funkcije, jedna za zaštitu redaka i druga za zaštitu osjetljivih stupaca te se one pridruže svim tablicama/viewima koji zahtijevaju sigurnost podataka. Unutar funkcija mogu se razraditi mehanizmi pristupa ovisno o postavljenim ovlaštenjima korisnika. U konačnoj verziji rješenja, funkcije za vraćanje predikata VPD mehanizma prilično su jednostavne. U after logon triggeru baze ugrađen je mehanizam punjenja atributa aplikacijskog konteksta s vrijednostima predikata za svaku od tablica. Ovlasti korisnika za rad u VPD okruženju definirane su u posebnoj baznoj tablici. Na temelju zapisa u tablici kreira se predikat VPD mehanizma.

```
  function row_vpd(p_schema in varchar2, p_table in varchar2)
    return varchar2 is
    v_result varchar2(1000);
  begin
    v_result := nvl(sys_context('NEKI_CONTEXT', upper(p_table)), '1=0');
    return v_result;
  end vrati_drustva_vpd;

  function col_vpd(p_schema in varchar2, p_table in varchar2)
    return varchar2 is
    v_rv varchar2(2000);
  begin
    v_rv   :=   nvl(sys_context('NEKI_CONTEXT',   upper(p_table)   ||'_COL'),
'1=0');
    return v_rv;
  end;
```

Jednostavnost je primijenjena i za odabir tablica nad kojima će se dignuti VPD zaštita. Uključili smo svega nekoliko ključnih tablica u kojima se može primijeniti zaštita redaka na razini poduzeća. U tablicama kćerima, tamo gdje je bilo potrebno, primijenili smo samo zaštitu na nivou stupaca. Zaštitom

na razini tablica automatski su u VPD uključeni i svi viewi koji se vežu uz odabrane tablice.

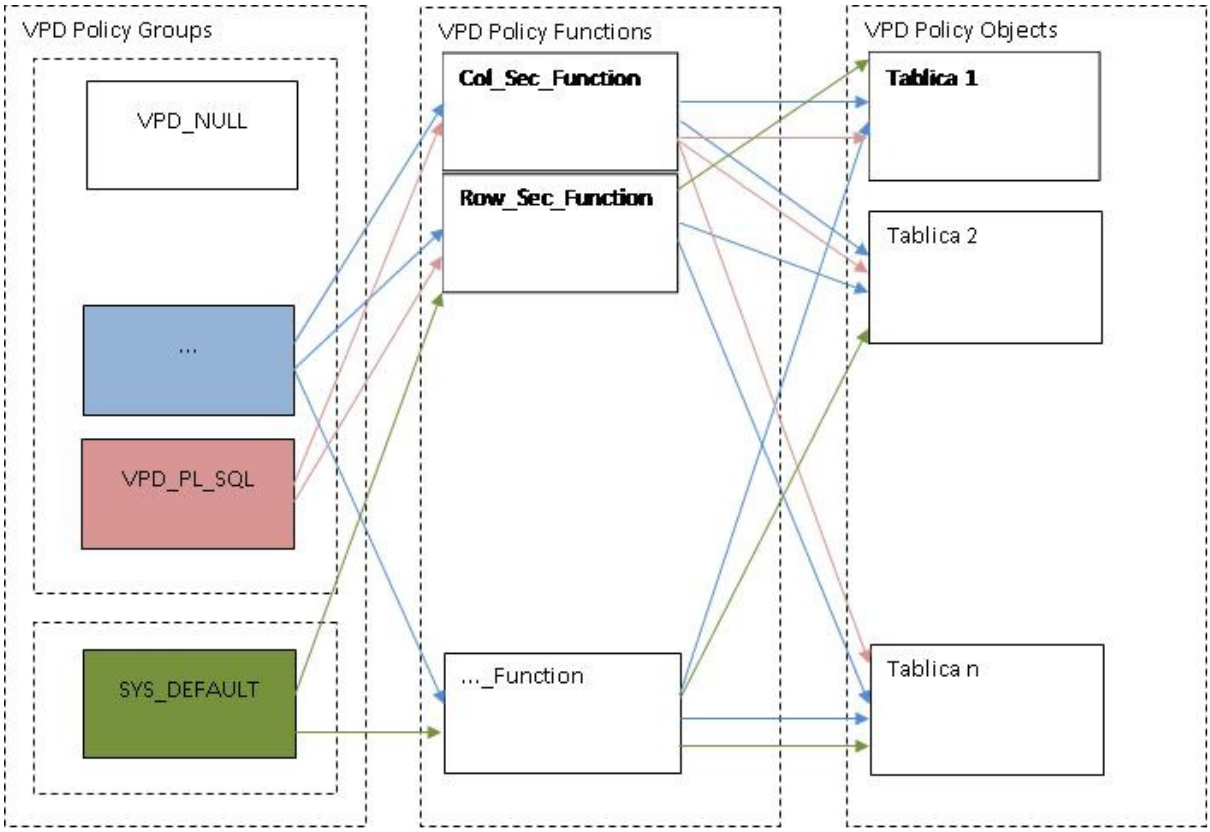
Testiranjem ove početne zamisli naišlo se na sljedeće probleme u radu:

- VPD sigurnosna politika provodi se uvijek, čak i u baznim procedurama i funkcijama čiji je vlasnik korisnik koji ima ovlaštenja pristupa svim redcima ili stupcima
- korištenje "data masking" opcije zaštite uzrokuje grešku (FRM 40654:Record has been updated by another user ...) prilikom pokušaja ažuriranja podataka u Oracle Forms aplikaciji.

Kako bi se izbjegli dodatni problemi u radu ERP sustava, koji se bazira na Oracle Forms/Reports rješenju, odlučeno je da je u radu s aplikacijom najbolje izbjeći aktivaciju VPD sigurnosnih pravila. Razmatrana su dva moguća rješenja:

- u VPD funkciju ugraditi logiku izbjegavanja VPD-a vraćanjem uvijek istinitog predikata, npr. "1=1"
- iskoristiti mogućnosti vlastitih VPD sigurnosnih grupa (policy groups) tako da se ovisno o aplikacijskom kontekstu aktivira zadovoljavajuća grupa.

Odlučeno je da se iskoristi ova druga mogućnost, jer prvi način uvijek aktivira neku sigurnosnu politiku nad tablicama, dok u radu s grupama postoji mogućnost izbjegavanja aktivacije VPD mehanizma. Identificirane su tablice čije retke i/ili stupce treba zaštititi. Implementacija s vlastitim VPD grupama zahtjeva nekoliko koraka. Svakoj od tablica treba se pridružiti kontekst aplikacijske grupe (DBMS_RLS.ADD_POLICY_CONTEXT), zatim svakoj tablici pridružiti svaku od vlastitih VPD grupa (SYS_DBMS_RLS.CREATE_POLICY_GROUP) i na kraju, opcionalno, pridružiti policy funkciju (DDMS_RLS.ADD_GROUPED_POLICY). Slijedeći te korake, svakoj od relevantnih tablica pridružen je kontekst aplikacijske grupe.



Slika 2: Shematski prikaz modela VPD implementacije

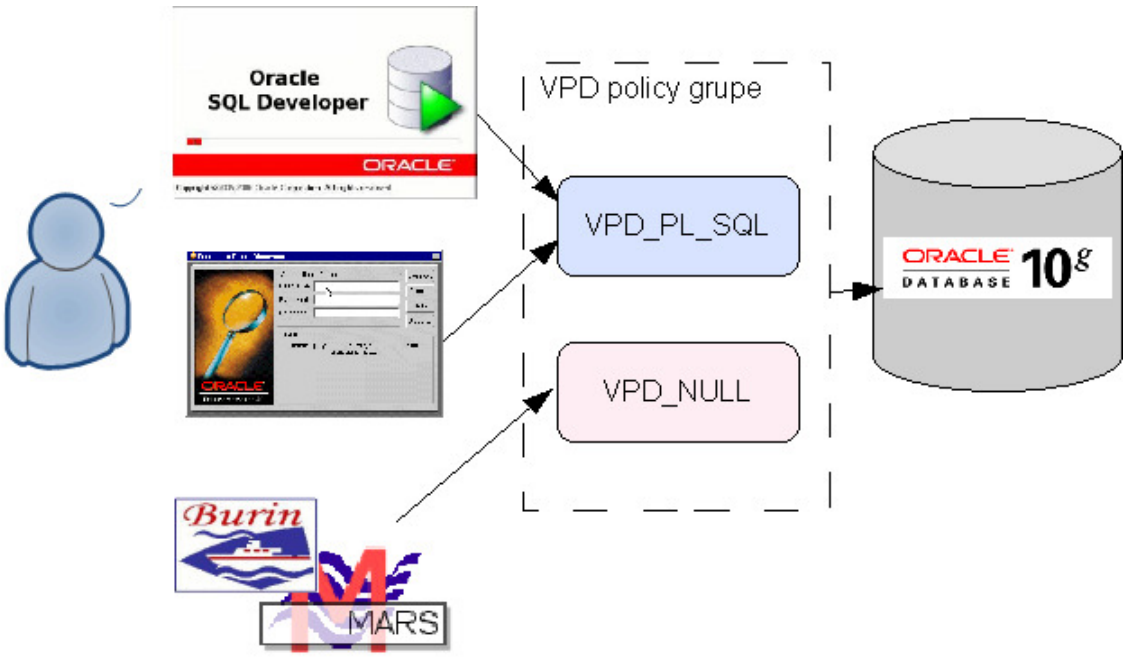
```
begin
  dbms_rls.add_policy_context(object_schema => 'VLASNIK',
                              object_name   => 'TABLICA',
                              namespace     => 'VPD_CONTEXT',
                              attribute     => 'VPD_POLICY_GROUP');
end;
/
```

Kreirane su dvije VPD sigurnosne grupe "VPD_PL_SQL" i "VPD_NULL" za svaku od tablica:

```
begin
  sys.dbms_rls.create_policy_group(object_schema => 'VLASNIK',
                                   object_name   => 'TABLICA',
                                   policy_group  => 'VPD_PL_SQL');
end;
/
begin
  sys.dbms_rls.create_policy_group(object_schema => 'VLASNIK',
                                   object_name   => 'TABLICA',
                                   policy_group  => 'VPD_NULL');
end;
/
```

Za grupu "VPD_NULL" nije kreiranja nikakva sigurnosna politika, tj. nije pridružena ni jedna funkcija za vraćanje predikata, dok je za grupu "VPD_PL_SQL" svakoj od tablica pridružena bar jedna od dviju funkcija.

```
begin
dbms_rls.add_grouped_policy(object_schema        =>'VLASNIK',
                            object_name          =>'TABLICA',
                            policy_group         =>'VPD_PL_SQL',
                            policy_name          =>'TABLICA_col',
                            function_schema      =>'VLASNIK',
                            policy_function      =>'vpd_p.col_vpd',
                            sec_relevant_cols    =>'OPC,NAS,UL,KBR',
                            sec_relevant_cols_opt=>dbms_rls.all_rows,
                            policy_type      =>dbms_rls.CONTEXT_SENSITIVE);
end;
/
```



Slika 3: Ovisno o modelu konekcije na bazu korisniku se 'podmeće' odgovarajuća VPD grupa

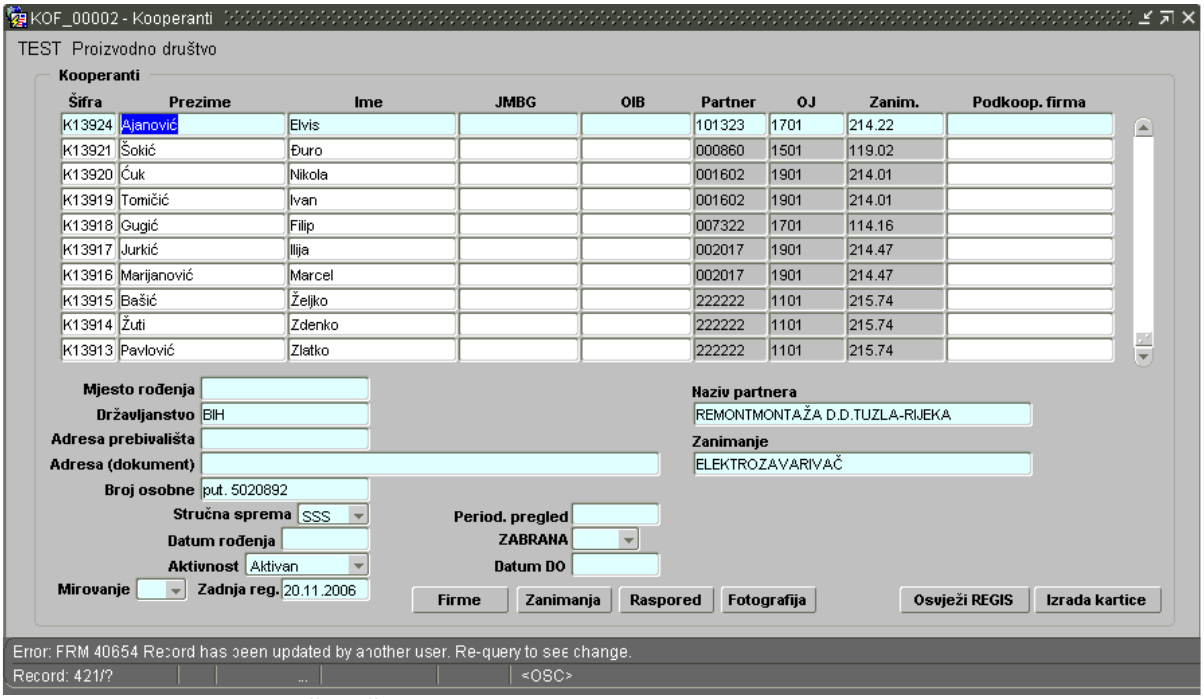
Za evidenciju podataka o grupama koje će se aktivirati prilikom konekcije na bazu kreirana je tablica u koju se za specifične atribute konekcije (korisnik baze, program, računalo/server, korisnik OS-a) zapisuje koja će se VPD grupa aktivirati. Prilikom konekcije na bazu okida se after logon trigger baze u kojem se poziva procedura za postavljanje konteksta VPD okruženja. U kontekstne atribute zapisuju se predikati svake sigurnosne grupe, te grupa koja predstavlja Driving Context. Postavljeno rješenje zadovoljava zahtjeve koji su postavljeni, ali je administracija dijelom zahtjevna.

Sljedeći korak poboljšanja bio bi izgradnja grafičkog sučelja za potpunu administraciju VPD sigurnosne politike. Treba napraviti sučelje ("dashboard") na kojem će se vidjeti statusi svih definiranih VPD pravila i na kojem će biti jednostavne "komande" za aktivaciju i deaktivaciju pravila, te mogućnost definiranja novih.

3 MOGUĆI PROBLEMI

3.1 Oracle Forms aplikacija

Kada se koristi VPD zaštita stupaca maskiranjem podataka javlja se problem ažuriranja podataka u Oracle Forms modulu, jer je vrijednost polja na ekranu (NULL) različita od one zapisane u bazi. Ova greška dovodi do otkrića još jednog problema sigurnosti VPD-a. 'Column masking' opcija zaštite radi samo nad SELECT naredbama. Kada nad istom tablicom izvedete SELECT ... FOR UPDATE tada 'column masking' zaštita više ne funkcionira. SELECT FOR UPDATE je upravo ono što forma radi kada pokuša rezervirati slog za ažuriranje u izvornom ON-LOCK triggeru i tada se uspoređuju učitane vrijednosti s vrijednostima na bloku forme. Kada vrijednosti nisu jednake javlja grešku FRM 40654. Ova greška može se zaobići samo izmjenama u Forms modulu.



Slika 4: Pokušaj ažuriranja podataka na formi koja ima maskirane stupce

U našem rješenju oslonili smo se na sigurnost implementiranu u aplikaciji, te smo kod prijave u aplikaciju aktivirali VPD grupu (Driving Context) koja nema implementirane VPD funkcije.

Istraživanjem po forumima, na linku http://www.orafaq.com/forum/t/30540/2/ postoji zapis o još jednoj mogućnosti pojave greške FRM 40654. Problem je dokumentiran kao Oracle Bug 7344505 (metalink ID 759252.1). Greška se javlja samo kod dinamičkih VPD predikata. Na forumu je naveden primjer ažuriranja tablice kćeri D na koju je podignut VPD predikat koji provjerava postojanje retka u roditeljskoj tablici M. VPD pravila dozvoljavaju ažuriranje tablice D, dok ažuriranje tablice M nije dozvoljeno VPD-em. Kada se na formi pokrene ažuriranje retka tablice D forms modul izvodi zaključavanje sa SELECT FOR UPDATE koji kao rezultat vraća 0 redova i navodi forms modul na grešku kao da je netko obrisao slog.

Greška se može zaobići zahvatom u ON-LOCK triggeru koji bi trebao na trenutak isključiti VPD.

3.2 PL/SQL kôd na bazi

Naišli smo i na problem u radu baznih procedura i funkcija u kojima nam je VPD zaštita neželjena pojava. Npr. željeli smo kontrolirati da li se JMBG radnika pojavljuje u tablicama ostalih poslovnih subjekata kao "nepoželjni" radnik. I ovaj problem zaobiđen je aktivacijom posebne VPD grupe prilikom prijave u aplikaciju. Rješenje se moglo postići i poigravanjem s atributima aplikacijskog konteksta, ali to je zahtjevalo zahvate u kôdu koje smo nastojali izbjeći.

3.3 Rad s definiranim VPD grupama

Prilikom definiranja vlastitih VPD grupa potrebno je u svaku od grupa uključiti sve tablice koje se pojavljuju u bilo kojoj od grupa. Mogući scenarij je npr.: kreirali ste grupu G1 i nju pridružili tablicama T1 i T2, zatim grupu G2 i nju priključili tablici T1. Kada korisnik u sesiji na bazi ima pridruženu grupu G2 i pokuša čitati tablicu T2 javit će se greška:

```
ORA-28123 Driving context 'string,string' contains invalid group 'string'.
```

Uvijek provjerite podatke u tablici DBA/USER/ALL_POLICY_GROUPS, tj. da li je svakoj od tablica pridružena svaka od grupa.

3.4 Greške u VPD funkcijama

Ukoliko je funkcija koja vraća predikat VPD-a u statusu "invalid" ili ima grešku javit će se poruka

greške prilikom rada s tablicom/viewom:

```
ORA-28112 Failed to execute policy function.
```

Kada funkcija vrati predikat s neispravnom sintaksom javit će se:

```
ORA-28113: policy predicate has error
```

3.5 DBMS_RLS Policy Type

Ukoliko u predikatima VPD pravila koristite aplikacijski kontekst tada parametar policy type prilikom kreiranja VPD pravila (DBMS_RLS.ADD_POLICY, DBMS_RLS.ADD_GROUPED_POLICY) ne smije imati vrijednost STATIC ili SHARED_STATIC. Na primjerima koji su testirani predikat je bio oblika:

```
ime_kolone = SYS_CONTEXT('NEKI_CONTEXT','ATRIBUT').
```

Rezultat VPD pravila bio je nepredvidiv, ovisio je o vrijednosti inicijalno pohranjenoj u SGA međumemoriju. Kada se parametar Polici Type postavlji kao DYNAMYC, CONTEXT_SENSITIVE, SHARED_CONTEXT_SENSITIVE pravila VPD-a s kontekstnim varijablama u predikatima rade ispravno.

3.6 Rekurzija u VPD definiciji

Potrebno je naglasiti mogućnost pojave rekurzije. Oprezno koristite tablice/viewe koji imaju podignutu VPD zaštitu u funkcijama za vraćanje VPD predikata. Moguć je scenarij beskonačne petlje kada npr. funkcija F1 poziva proceduru P2, koja zatim poziva proceduru P3 koja čita iz tablice koja ima implementiranu VPD zaštitu funkcijom F1.
Za bolje performanse analizirajte predikate koje kreiraju VPD funkcije čitajući zapise u V$VPD_POLICY, te podignite odgovarajuće indexe.

3.7 Web aplikacija

U web aplikacijama praksa je da se konekcija na bazu radi s fiksnim aplikacijskim korisničkim imenom, npr. APPKORISNIK. U takvoj situaciji VPD sigurnost se ne može oslanjati na korisnika baze (APPKORISNIK), već na korisnika koji se prijavio u aplikaciju. Za takvu situaciju postoji relativno jednostavno rješenje. Kreirati će se novi atribut globalnog aplikacijskog konteksta REALUSER čija će vrijednost biti stvarni korisnik aplikacije.

```
begin
   sys.dbms_session.set_context(namespace => 'VPD_GLOB_CTX',
                                attribute => 'REALUSER',
                                value     => upper(p_user),
                                username  => user,
                                client_id => upper(p_user));
   dbms_session.set_identifier(p_user);
end;
```

Kreirati će se nova VPD sigurnosna grupa VPD_WEB_SEC čije funkcije će se oslanjati na korisnika zapisanog u globalnom aplikacijskom kontekstu.

Tabela 4: Predloženi scenarij VPD-a ovisno o modelu konekcije na bazu [1]

User Model Scenario	Individual DB Connection	Separate Application Context per User	Single DB Connection	Application Must Switch User Name
Application users are also database users	Yes	Yes	No	No
Proxy authentication using OCI or thick JDBC	Yes	Yes	No	No
Proxy authentication integrated with Enterprise User Security[1]	No	No	Yes	Yes
One Big Application User	No	No[2]	Yes	Yes[2]
Web-based applications	No	No	Yes	Yes

[1] User roles and other attributes, including globally initialized application context, can be retrieved from Oracle Internet Directory to enforce VPD.

[2] Application developers can create a global application context attribute representing individual application users (for example, `REALUSER`), which can then be used for controlling each session attributes, or for auditing.

Budući da web rješenja u našem okruženju nemaju široku primjenu, a korisnik za konekciju na bazu koristi se isključivo u aplikaciji, odlučili smo u tim rješenjima za sada ostati na VPD zaštiti jedinstvenog korisnika koji se konektira na bazu.

4 ZAOBILAŽENJE VPD-A

Dobro je znati da VPD pravila ne pogađaju sve korisnike. To je učinjeno namjerno iz više razloga. Jedan od razloga je kako bi se osigurao potpuni i točni backup svih podataka. Također, bilo bi problema kada bi se dogodila neka greška u nekoj od funkcija VPD-a koja ne bi dozvoljavala konekciju na bazu ni jednom korisniku.

Zadana je postavka Oracle baze da su svi korisnici koji imaju SYSDBA privilegiju isključeni iz VPD-a. Također, i korisnici kojima je dodijeljena sistemska privilegija "EXEMPT ACCESS POLICY" ne prolaze kroz VPD sigurnosni sustav. Dobro je periodično provjeriti korisnike kojima je ta privilegija dodijeljena:

```
select * from dba_sys_privs where privilege = 'EXEMPT ACCESS POLICY';
```

Dodjeljivanje ove sistemske privilegije treba izbjegavati u praksi, a pogotovo uz opciju "with admin option". Na primjeru vlastitog rješenja prikazano je kako se VPD sigurnost može postaviti tako da se ovisno o kontekstu konekcije na bazu pojedina pravila VPD-a uključe ili isključe.

5 SIGURNOST VPD OKRUŽENJA

5.1 Potencijalni problemi

VPD nije savršen i potpuno siguran mehanizam. Postoji niz slabih karika koje treba dodatno osigurati. Moguće slabe točke VPD-a su:

- informacije o predikatima
- informacija o VPD definicijama
- zaštita VPD strukture baznim rolama
- mogućnosti zaobilaženja VPD-a
- SQL injekcija
- pristup podacima izvan baze.

5.2 Informacije o predikatima

Iz viewa v$vpd_policies moguće je dobiti popis izvedenih sigurnosnih pravila VPD-a i njihovih predikata. Ovo je zanimljiv view, pogotovo ako se poveže s v$sqlarea ili v$sql tako da se dobije originalni SQL i korišteni predikat. Ovo je primjer jednog takvog upita na bazu:

```
select sql_text, predicate, policy, object_name
  from v$sqlarea, v$vpd_policy
 where hash_value = sql_hash;
```

Zbog sigurnosti, pristup ovim podacima baze treba dozvoliti samo administratorima sustava.

Postoji i nekoliko načina da se "trejsanjem" dođe do podataka o predikatima VPD pravila. Jedan od jednostavnijih načina je uz korištenje "Event 10730":

```
alter session set sql_trace=true;
alter session set events '10730 trace name context forever';
select * from neka_tablica_koja_ima_vpd;
alter session set events '10730 trace name context off';
alter session set sql_trace=false;
```

Datoteka s rezultatima izgleda otprilike ovako:

```
 *** 2003-10-19 16:30:59.000 *** SESSION ID:(7.64) 2003-10-19 16:30:59.000
------------------------------------------------------------  Logon  user
: VPD Table or View  : VPD.TRANSACTIONS Policy name     : VPD_TEST_POLICY
Policy  function:  VPD.VPD_POLICY.VPD_PREDICATE  RLS  view  :  SELECT
"TRNDATE","CREDIT_VAL","DEBIT_VAL","TRN_TYPE","COST_CENTER"            FROM
"VPD"."TRANSACTIONS"  "TRANSACTIONS" WHERE (cost_center='ACCOUNTS')
```

Iz podataka se može isčitati predikat VPD-a. Slično se može iskoristiit i "Event 10060".

5.3 Informacije o VPD definicijama

Iz viewa %_POLICY_GROUPS, %_POLICY_CONTEXTS, %_POLICIES, %_SEC_RELEVANT_COLS moguće je saznati detalje o VPD konfiguraciji. Pristup ovim podacima treba ograničiti isključivo na administratore sustava. Također treba zaštititi i pristup podacima o izvornom kodu funkcija koje se koriste za vraćanje predikata (OBJ$, SOURCE$, PROCEDURE$, ARGUMENT$, %_SOURCE, V$CONTEXT, v$GLOBALCONTEXT i ostalo).

5.4 Zaštita VPD strukture baznim rolama

U prethodnom poglavlju spomenuti su neki od viewa koje treba dodatno zaštititi. Osim toga treba poraditi i na zaštiti paketa DBMS_RLS i DBMS_SESSION. Potrebno je razmisliti o zaštiti lokalnog i globalnog konteksta. Preporuka je da se kontekst definira tako da se dozvoli ažuriranje isključivo preko dedicirane bazne procedure, koju treba zaštititi od neovlaštenog izvođenja.

Vlastite tablice, procedure, triggeri u kojima se definiraju dodatni parametri VPD okruženja trebaju se uključiti u strogi mehanizam zaštite. Nikako se ne smiju zanemariti i sistemske privilegije kojima se može utjecati na navedene objekte.

5.5 Mogućnost zaobilaženja VPD-a

Već je spomenuto kako se VPD u potpunosti može zaobići s "EXEMPT ACCESS POLICY" sistemskim ovlaštenjem. Osim toga u praktičnoj implementaciji VPD-a koriste se vlastite tablice i kontekstne varijable. Ove podatke moguće je zlonamjerno očitati ili izmijeniti i na taj način prevariti VPD. Korisnici s ovlastima nad paketom DBMS_RLS imaju mogućnost rušenja (drop) ili kreiranja VPD pravila.

5.6 SQL injekcija

U knjizi "Oracle Hacker's Handbook" [4] prikazano je kako se SQL injekcijom može npr. izvršiti

rušenje (drop) VPD pravila. Ukratko, preko procedure XDB_PITRIG vlasnika XDB koji ima ovlasti za izvođenje DBMS_RLS napravljena je SQL injekcija naredbe DBMS_RLS.DROP_POLICY.

```
SQL> CONNECT SCOTT/TIGER
Connected.
SQL> SELECT * FROM VPD.VPDTESTTABLE;
CLASSIFICATION ORDER_TEXT RANK
-------------------- -------------------- --------------
UNCLASSIFIED UPDATE DUTY ROTA CORPORAL
UNCLASSIFIED POLISH BOOTS MAJOR

SQL> CREATE OR REPLACE FUNCTION F RETURN NUMBER AUTHID CURRENT_USER IS
2 PRAGMA AUTONOMOUS_TRANSACTION;
3 BEGIN
4 DBMS_OUTPUT.PUT_LINE('HELLO');
5 EXECUTE IMMEDIATE 'BEGIN
SYS.DBMS_RLS.DROP_POLICY(''VPD'',''VPDTESTTABLE'',''SECRECY''); END;';
6 RETURN 1;
7 COMMIT;
8 END;
9 /
Function created.
SQL> CREATE TABLE FOO (X NUMBER);
SQL> EXEC XDB.XDB_PITRIG_PKG.PITRIG_DROP('SCOTT"."FOO" WHERE
1=SCOTT.F()--','BBBB');
PL/SQL procedure successfully completed.
SQL> SELECT * FROM VPD.VPDTESTTABLE;
CLASSIFICATION ORDER_TEXT RANK
-------------------- -------------------- --------------------
SECRET CAPTURE ENEMY BASE GENERAL
UNCLASSIFIED UPDATE DUTY ROTA CORPORAL
SECRET INVADE ON TUESDAY COLONEL
UNCLASSIFIED POLISH BOOTS MAJOR
```

5.7 Pristup podacima izvan baze

Direktnim pristupom "sirovim" podacima na disku mogu se očitati podaci koj su VPD-em zaštićeni. U "Oracle Hacker's Handbook" [4] prikazan je jedan takav primjer Java procedure na bazi koja čita "sirove" podatke i ispisuje ih. Ovakvoj opasnosti podvrgnute su "backup" i "export" datoteke. Osjetljive podatke trebalo bi dodatno zaštititi enkripcijom.

5.8 Column Masking slabosti

Column Masking opcija zaštite stupaca funkcionira sam uz SELECT naredbe, i to na prvi pogled izgleda u redu. Međutim, jedna od grešaka koja je uočena testiranjem sustava, a nema zapisa o tome u literaturi je problem naredbe SELECT ... FOR UPDATE. Takvim dohvatom podataka stupci tablice nisu maskirani kao što bi se očekivalo.

Propust u sugurnosti "Column Maskinga" može se uočiti i pozivom naredbe UPDATE, tako na primjer naredbom:

```
update radnici t
   set t.neko_polje = ''
 where id = 2413 returning jmbg into :jmbg;
```

može se očitati zapis stupca koji očekujemo da bude skriven VPD-om.
Na temelju prikazanih primjera zaključuje se da je maskiranje stupaca sigurno samo u slučaju kada korisnik nema pridružene bazne ovlasti za ažuriranje zaštićene tablice.

ZAKLJUČAK

VPD nije svemoguć! Prikazali smo neke od mogućnosti zlouporabe. Mehanizam maskiranja stupaca ima iznenađujućih propusta. Za efikasnu zaštitu servere i bazu treba dodatno pratiti i zaštititi, te primijeniti višeslojnu zaštitu. Treba znati da je VPD samo jedan od segmenata zaštite. Primjenjiv je i efikasan u radu korisnika koji bazi pristupaju samo s ovlastima za čitanje podataka. VPD nikako ne pruža kvalitetnu zaštitu od zlonamjernih napada na podatke. Glavna prednost je u tome što se implementira direktno na bazi i ne zahtjeva dodatne izmjene u aplikacijama, a koristi ga svaka aplikacija koja pristupa bazi.

Ukoliko se, unatoč prikazanim nedostacima, ipak odlučite koristiti Oracle VPD, nastojite primijeniti zlatno pravilo jednostavnosti kako se ne biste izgubili u džungli administracije VPD pravila.

LITERATURA

1. Oracle® Database Security Guide 10*g* Release 2 (10.2)
2. Ben-Natan, R. (2009): HOWTO Secure and Audit Oracle 10g and 11g, Auerbach Publications
3. Knox, D. C. (2004): Effective Oracle Database 10g Security by Design, McGraw-Hill
4. David Litchfield (2007): Oracle Hacker's Handbook, Wiley
5. Thomas Kyte: Expert one-on-one Oracle, Apress

LINKOVI

6. https://forums.oracle.com/forums/
7. http://www.petefinnigan.com/Oracle_Security_VPD6Slides.pdf
8. http://www.symantec.com/connect/articles/oracle-row-level-security-part-2