

Università di Pisa

**Laurea Magistrale in Sicurezza Informatica:
Infrastrutture ed Applicazioni**

**Confidenzialità ed integrità di basi di dati
a.a. 2009/2010**

ORACLE

DATABASE VAULT

Marco Alamanni

Introduzione

Negli ultimi anni normative e standard quali il **Sarbanes-Oxley Act (SOX)**, la **European Union Data Protection Directive (95/46/EC)**, l' **Health Insurance Portability and Accountability Act (HIPAA)** e il **Payment Card Industry Data Security Standard (PCI-DSS)** hanno decisamente spinto aziende private ed istituzioni pubbliche a considerare con grande attenzione le problematiche relative al controllo degli accessi e la protezione dei dati sensibili nelle loro infrastrutture informatiche.

La sicurezza dei server database ha così assunto maggiore importanza, poiché essi costituiscono le "casseforti" dove sono custoditi dati cruciali per il business (o la mission) dell'azienda o dell'ente pubblico.

Le esigenze più sentite sono in particolare la protezione dei dati sensibili dagli utenti con privilegi amministrativi e la **separazione dei ruoli** nell'amministrazione del database.

Oracle Database Vault (in seguito **DBV**) è un'opzione di sicurezza per database Oracle la cui realizzazione è stata guidata proprio da questi obiettivi.

E' costruita su di un framework dichiarativo che, attraverso componenti quali **realms**, **factors**, **command rules**, **rules set** e **secure applications roles**, consente di definire politiche di sicurezza in modo flessibile e trasparente alle applicazioni.

DBV è configurabile e gestibile tramite interfaccia web con **Oracle Database Vault Administrator** e fornisce anche una API PL/SQL attraverso la quale creare e gestire tutti i componenti e configurare tutti i vari parametri di DBV.

Problematiche di sicurezza affrontate da DBV

Oracle ha due principali account di tipo amministratore: SYSTEM e SYS.

SYSTEM è l'account amministratore standard con cui l'amministratore del database (DBA) normalmente si connette per svolgere le sue funzioni.

SYS invece non è usato di solito come account utente vero e proprio, a differenza di SYSTEM.

Possiede gli oggetti che costituiscono il nucleo del motore database e ha pertanto anche tutti i privilegi su di essi, cosa che si traduce in un controllo completo del database.

E' un' account necessario per importanti funzioni di amministrazione e di manutenzione del database, come installare patches, effettuare aggiornamenti, backup e recovery, abilitare e disabilitare meccanismi di sicurezza.

Il rischio per la sicurezza del database non è rappresentato solo dalla possibile compromissione di questi accounts, ma soprattutto da vulnerabilità nelle applicazioni per cui, attraverso tecniche come SQL injection, un attaccante può eseguire codice arbitrario in un contesto amministrativo. Non tutte le applicazioni sono infatti scritte tenendo conto del principio del 'privilegio minimo' e richiedono diritti amministrativi per funzionare correttamente.

Agli account amministrativi sono concessi privilegi di tipo ANY (SELECT ANY TABLE, DROP ANY TABLE ecc...), per cui un DBA può accedere, modificare o cancellare dati sensibili a cui non dovrebbe avere accesso, non rispettando la conformità alle linee guida stabilite dalle normative in materia.

DBV introduce nuovi ruoli amministrativi:

- DV_SECANALYST, DV_ADMIN e DV_OWNER per la gestione di DBV e delle politiche di sicurezza;
- DV_REALM_OWNER per la gestione dei realms e dei ruoli associati;

- DV_ACCTMGR per la gestione degli accounts e dei profili.

Questi ruoli separano la gestione della sicurezza e degli account dal ruolo tradizionale del DBA, implementando il concetto della separazione dei ruoli. Le funzioni e i privilegi che prima erano propri di un unico 'super-utente' vengono così suddivisi tra diversi ruoli, in modo che nessun utente abbia più il controllo completo sui dati e sulla configurazione del database.

A differenza di opzioni di sicurezza come **Virtual Private Database (VPD)** e **Label Security (OLS)**, che forniscono un meccanismo di controllo degli accessi a livello di records di una tabella, Database Vault restringe l'accesso a livello di comandi e di oggetti. Come vedremo, si può integrare anche con esse.

Componenti

Realms

Un **realm** è un raggruppamento funzionale di schemi e ruoli per cui l'utilizzo di privilegi amministrativi di tipo ANY è limitato solo agli utenti e ai ruoli autorizzati. In altre parole, sono gruppi di schemi e ruoli protetti dall'accesso non autorizzato da parte di utenti con privilegi amministrativi di tipo ANY.

Sono definiti i seguenti realms di default:

- **Database Vault Account Management:** definisce il realm per l'amministratore che crea e gestisce gli accounts e i profili (DV_ACCTMGR);
- **Oracle Data Dictionary:** definisce il realm per i seguenti schemi di Oracle Catalog:
ANONYMOUS; DBSNMP; MDSYS; SYS; BI; EXFSYS; MGMT_VIEW; SYSMAN;
CTXSYS; MDDATA; OUTLN; SYSTEM
Questo realm controlla anche la capacità di concedere privilegi e ruoli amministrativi;
- **Oracle Database Vault:** definisce il realm per gli schemi di Database Vault (DVSYS, DVF e LBACSYS);
- **Oracle Enterprise Manager:** definisce il realm per gli account di Enterprise Manager (SYSMAN, DBSNMP).

Si possono creare nuovi realms, ad esempio uno per tutti gli schemi usati da un dipartimento aziendale. Un realm può includere uno o più schemi del database e uno schema può appartenere a più realms. Può essere creato solo da un utente che ha un ruolo amministrativo di Database Vault (Security Administrator), come DV_OWNER o DV_ADMIN.

Creando un realm, possono essere anche specificate le opzioni di auditing, stabilendo ad esempio di registrare nel log di audit i comandi non permessi dal meccanismo di protezione del realm (realm violations). Creato un realm, possono esservi aggiunti o tolti anche successivamente schemi e ruoli. Le autorizzazioni possono essere concesse solo da un Security Administrator e definiscono gli accounts e i ruoli che sono autorizzati ad accedere e manipolare gli oggetti protetti dal realm.

Un utente o un ruolo autorizzato può essere di due tipi: **owner** o **participant**.

Un participant può creare, accedere e modificare gli oggetti protetti dal realm, a patto che gli siano stati concessi i diritti per farlo dal sistema DAC standard del database. Un realm può avere più participants.

Un utente owner ha in più la facoltà di concedere o revocare privilegi su oggetti del realm e di assegnare o revocare ruoli ad altri utenti. Non può però aggiungere altri utenti owner o participant al realm; ciò è permesso solo ai ruoli amministrativi di Database Vault. Gli utenti che hanno questi ruoli, come DV_OWNER, non possono essere autorizzati nei realm creati.

Le autorizzazioni non aggiungono privilegi agli utenti che già dispongono di diritti di accesso agli oggetti del realm, ma limitano l'uso dei privilegi amministrativi solo agli utenti autorizzati.

Forniscono un meccanismo di protezione trasparente alle applicazioni che controlla a run-time (prima che il comando venga passato al motore SQL del database) se l'utente che ha impartito un comando SQL ha l'autorizzazione ad eseguirlo. Se non è autorizzato viene generato un errore di tipo 'realm violation' che può essere registrato nei log di audit.

Un'applicazione dei realms è anche quella come meccanismo di sicurezza per il consolidamento di server database. In molte aziende, infatti, esistono server database distinti per ogni dipartimento (amministrativo, risorse umane, ricerca ecc...). Essi possono essere raggruppati in un unico server database, con riduzione dei costi di esercizio e manutenzione delle macchine. Tramite i realms si possono comunque mantenere separati i dati ed i ruoli amministrativi dei vari dipartimenti, impedendo, ad esempio, che l'amministratore del database di risorse umane possa accedere ai dati del dipartimento ricerca.

Command rules e rules sets

Quando viene impartito un comando SQL, Database Vault intercetta il comando e innanzitutto verifica le autorizzazioni del realm per consentire o meno la sua esecuzione. Se viene autorizzato, un secondo controllo è effettuato tramite le **command rules**. Esse sono regole che possono essere applicate alla maggior parte dei comandi DDL e DML e ne consentono l'esecuzione solo se la condizione specificata nella regola si verifica. Una command rule può quindi bloccare un comando eseguito da un utente autorizzato in un realm, se la condizione non è rispettata. Le command rules non sono legate ai realms; una command rule può essere anche riferita ad un oggetto non protetto da alcun realm.

Le command rules sono suddivise nelle seguenti categorie:

- **Command rules di sistema:** sono applicate a comandi come CONNECT e ALTER SYSTEM e se ne può creare solo una per ogni istanza del database;
- **Command rules associate ad uno schema:** un esempio è creare una command rule per il comando DROP TABLE;
- **Command rules associate ad un oggetto:** un esempio è creare una command rule per il comando DROP TABLE per una specifica tabella inclusa nella regola;

Quando si crea una command rule, si devono specificare:

- Il comando SQL a cui si riferisce;
- Il nome e il proprietario dell'oggetto a cui si riferisce il comando (il carattere speciale % viene usato per selezionare tutti gli oggetti o tutti i proprietari);
- Il **rule set** associato;

Non si possono creare command rules per gruppi di comandi, come CREATE TABLE, ALTER TABLE e DROP TABLE, ma si possono definire più command rules per lo stesso comando. Ad esempio per il comando DROP TABLE possono esserne create una (o più) per specifici proprietari di oggetti e una per tutti gli altri.

Un **rule set** è un insieme di regole, ognuna formata da un'espressione PL/SQL che restituisce un valore booleano.

I rules set sono usati per restringere le autorizzazioni sui realms (specificando le condizioni sotto le quali sono attive), per definire quando permettere una command rule e per definire quando assegnare l'identità di un factor.

Una regola può essere associata a più rules set e si possono così creare librerie di regole da usare per definire in modo flessibile la propria politica di sicurezza.

I rules sets possono essere configurati in modo tale da richiedere che tutte le regole associate siano

vere (ALL TRUE) o che almeno una di esse sia vera (ANY TRUE).

Database Vault fornisce un insieme di rule sets predefiniti, che possono essere personalizzati a piacimento.

Factors

I **factors** sono variabili o attributi riferiti ad una sessione del database di cui sono utili a stabilire il contesto, per autorizzare comandi come i seguenti:

- Connessione al database, quando usati come parte di una command rule associata a CONNECT;
- Eseguire un comando SQL su di un oggetto, quando usati come parte di una command rule o di autorizzazioni di un realm;
- Filtrare comandi SELECT e DML, quando usati come integrazione a VPD e OLS.

DBV fornisce un insieme di factors predefiniti. A ciascun factor sono associati un tipo ed una funzione PL/SQL che restituisce il suo valore (**identità**). Tali funzioni appartengono allo schema DVF e hanno la forma `DVF.F$factor_name`. L'identità può anche essere una costante o dipendere dall'identità di altri factors associati. Alcuni dei tipi predefiniti di factors sono:

Authentication_method, Client_IP, Domain, Database_hostname, Database_IP, Session_User.

Secure application roles

I **secure applications roles** sono ruoli che possono essere abilitati solo all'interno di procedure PL/SQL, le quali effettuano controlli per stabilire se le condizioni per l'abilitazione del ruolo sono verificate. Per definire queste condizioni si usano i rule sets.

Un esempio applicativo

Supponiamo di voler restringere l'accesso di un amministratore al database usando i seguenti criteri:

- Garantire che l'amministratore acceda dal corretto indirizzo IP;
- Limitare l'accesso solo durante l'orario di lavoro.

Abbiamo un utente *john* a cui sono stati concessi privilegi amministrativi da un utente con ruolo DV_ACCTMGR.

Innanzitutto un utente con ruolo DV_OWNER o DV_ADMIN si connette a Oracle DBV Administrator. Il primo passo è quello di aggiungere identità al factor 'Domain'; si creano due identità, SECURE e NOT SECURE, rispettivamente con 'Trust Level' VERY TRUSTED e UNTRUSTED.

Poi si associano le identità di 'Domain' al factor 'Client_IP': all'identità SECURE si associa l'indirizzo IP della macchina dell'amministratore (ad esempio 192.168.1.100) e a NOT SECURE tutti gli altri indirizzi.

Il terzo passo è quello di creare un rule set per limitare l'orario di connessione e si chiama ad esempio 'DBA WORKING HOURS', specificando come opzione di valutazione delle regole ALL TRUE. Si creano poi le regole seguenti:

Name: Internal DBA ; Rule Expression: DVF.F\$SESSION_USER='JOHN'

Name: Secure IP ; Rule Expression: DVF.F\$DOMAIN='SECURE'

Name: Week Day ; Rule Expression: TO_CHAR(SYSDATE, 'D') BETWEEN '2' AND '6'

Name: Week Working Day Hours ; Rule Expression: TO_CHAR(SYSDATE, 'HH24') BETWEEN '08' AND '19'

Il passo successivo è creare una command rule che usi il rules set definito sopra, per controllare ad esempio l'esecuzione di DROP TABLE su oggetti di proprietà dell'account 'SH' protetti da un realm; nella pagina 'Create Command Rule' si seleziona il comando DROP TABLE, come 'Object owner' 'SH' e come rule set 'DBA WORKING HOURS'.

Quando l'utente *john* si connette ed invia il comando DROP TABLE SH.SALES; DBV verifica se l'utente è autorizzato per il realm; in caso negativo viene generato un errore 'realm violation for drop table on SH.SALES', altrimenti verifica la command rule associata. Se una delle regole del rule set 'DBA WORKING HOURS' non è vera, perchè *john* si è connesso da un altro indirizzo IP o fuori dell'orario di lavoro, il comando viene bloccato e viene generato l'errore 'command rule violation for drop table on SH.SALES'.

Integrazione con Oracle Label Security e Virtual Private Database

OLS è un'opzione di sicurezza attraverso cui si possono definire politiche di sicurezza multi-livello di tipo Mandatory Access Control (MAC).

Consente di restringere l'accesso ai records delle tabelle del database, confrontando il livello di sicurezza assegnato all'utente con quello assegnato alla label del record.

DBV permette di associare le identità dei factors alle labels di OLS, integrando così i due meccanismi di sicurezza.

Il seguente è un esempio di questa integrazione per assegnare a due utenti amministratori *john* e *robert* diversi livelli d'accesso ai dati.

Per prima cosa creiamo le policies di OLS. Ci connettiamo come LBACSYS (amministratore di OLS) e creiamo la seguente policy:

```
EXEC SA_SYSDBA.CREATE_POLICY('PRIVACY','PRIVACY_COLUMN','NO_CONTROL');
```

Poi creiamo i livelli per la policy 'PRIVACY':

```
EXEC SA_COMPONENTS.CREATE_LEVEL('PRIVACY',2000,'S','SENSITIVE');
```

```
EXEC SA_COMPONENTS.CREATE_LEVEL('PRIVACY',1000,'C','CONFIDENTIAL');
```

Creiamo il compartimento 'PII':

```
EXEC SA_COMPONENTS.CREATE_COMPARTMENT('PRIVACY',100,'PII','PERS_INFO');
```

Concediamo agli utenti *john* e *robert* le seguenti labels:

```
EXEC SA_USER_ADMIN.SET_USER_LABELS('PRIVACY','john','S:PII');
```

```
EXEC SA_USER_ADMIN.SET_USER_LABELS('PRIVACY','robert','C');
```

Dopo essersi collegati come utente con ruolo DV_OWNER si crea un rule set per controllare le autorizzazioni OLS:

```
EXEC DVSYS.DBMS_MACADM.CREATE_RULE_SET('PII Rule Set','Protect PII data from privileged users','Y',1,0,2,NULL,NULL,0,NULL);
```

e vi si aggiunge la regola 'Check OLS Factor':

```
EXEC DVSYS.DBMS_MACADM.CREATE_RULE('Check OLS Factor',
'dominates(sa_utl.numeric_label('PRIVACY'),char_to_label('PRIVACY','S:PII'))
= '1');
```

Si aggiorna la command rule associata al comando ALTER SYSTEM per aggiungervi la rule set 'PII Rule Set':

```
EXEC DVSYS.DBMS_MACADM.UPDATE_COMMAND_RULE('ALTER SYSTEM','PII Rule Set',
'%' , '%' , 'Y');
```

Quando i due amministratori si connettono ed eseguono ad esempio il comando ALTER SYSTEM SET AUDIT_TRAIL=OS; l'utente *john* sarà autorizzato ad eseguirlo, poiché ha un livello di sicurezza 'S' (Secret) mentre l'utente *robert* no poiché ha assegnato un livello 'C' (Confidential).

VPD permette di definire politiche di sicurezza a livello di record di tabelle in modo tale che, quando un utente effettua una query, il server modifica in modo dinamico e trasparente all'utente la clausola WHERE, in base ad una funzione PL/SQL. Le policies di VPD possono essere applicate a comandi come SELECT, INSERT, UPDATE e DELETE. Le funzioni DVF.F\$ dei factors possono essere usate nelle funzioni PL/SQL che implementano le politiche di VPD.

Conclusioni

DBV è un framework per la gestione della sicurezza nei database Oracle i cui componenti si integrano tra di loro e con altre opzioni di sicurezza come OLS, consentendo di definire politiche di sicurezza che implementano i concetti della separazione dei ruoli amministrativi e della protezione dei dati sensibili dagli utenti amministratori, in conformità alle normative e agli standard vigenti.

Bibliografia

- ✓ Patricia Huey, **Oracle Database Vault Administrator's Guide 11g Release 2 (11.2)**, Oracle, 2010

- ✓ David C. Knox et al, **Applied Oracle Security: Developing Secure Database and Middleware Environments**, Oracle Press/McGraw-Hill, 2010.