Global Information Assurance Certification Paper

Row Level Security in Oracle Databases with Virtual Private Database and Label Security

Steve Enevold
January 13, 2003
GSEC version 1.4b

Abstract

A significant amount of today's information is maintained in databases, and much of this information is confidential or sensitive in nature. Databases are being consolidated to save money, increase administration efficiency, and provide better business intelligence. Many of these databases are Oracle and need to be properly secured to meet legislative, business, and other policy requirements.

Oracle has addressed the issue of segregating information at the row level using two specific technologies called Virtual Private Database and Label security. Both VPD and Label Security arrive at similar results using different methods, with benefits and challenges to each.

Introduction

Evolution of data requirements

In the early days of computing, large mainframes stored information and performed operations for users on character based terminals.

With the advent of the personal computers and the internet, information moved from the mainframes to a more distributed model, with information pieces stored on desktops and mini computers. This posed several problems, including how to secure, back up, and report on the distributed data. Companies have found that distributed information is not practical and are moving back to a centralized information model, with the entire company's data in a single, data center, if not in a single, highly available data server. The consolidation benefits include, ease of backup, disaster recovery, security, and reporting.

There are also several drawbacks to consolidation which must be considered. If all the data is in a single server, that server becomes a single point of failure. If the site goes down, there needs to be a backup server in another location with a recent copy of the primary server's data so business may continue. Finally, storing information from different departments, cities, countries, or companies, in the same set of tables makes it challenging to ensure that only authorized access is successful.

Applications and security

Traditionally, applications have not been built with security as a key development goal. The application usually requires a user to enter her login and password, then the applications opens a connection to the database using a generic login account, or pooled connection. This makes it difficult, if not impossible to audit

who is accessing, modifying, or deleting sensitive information. If the application users have access to the same database using ad-hoc reporting tools, they completely bypass the application's security policies and controls. Also, as security policies change, the applications must be modified to comply with the new policies.

Application service providers have emerged in the recent years, offering application access services, while maintaining the applications and the customer data on the ASP's servers. This poses another issue, how to keep one company's information secure from another company's users. Some ASPs use a different server for each company that is hosted. While this is a strong method of data separation, it is very costly in terms of administration, application licensing, and hardware redundancy.

Row Level Security

Virtual Private Database is included with Enterprise Edition and uses packages, procedures and policies to restrict what a user sees depending on specific attributes. Label Security is an additional cost product that creates a "label" for each row that is managed via a graphical interface and matches a user's security clearance levels to the label tags on individual rows. Both technologies are patented by Oracle and provide a unique and strong method of locking down information in the database, thereby eliminating the application security bypass issue, and reducing the risk of unauthorized access.

Much like Virtual Private Networks protect information transferred across the internet by creating a secure "tunnel" over the wire, Oracle's technologies virtually hide the information from unauthorized access at the row level. This capability has many benefits. Information for different users, departments, or companies can be stored in the same table for ease of administration, reporting, and disaster recovery, while making it "appear" that the user sees all the rows in a table.

Row level security is elegant because the user doesn't know that any restrictions have taken place. When the user asks for all the rows in a table, the query returns all the rows in the table that the user was authorized to see. This removes the issue of bypassing the application security because the access control is handled at the table level, regardless of the access method.

Also, building a data mart or data warehouse brings information from different parties together for reporting and analysis. The traditional method of segregating data within a table is to create views and has been used for years successfully. The challenge is maintaining what can become hundreds of views to comply with company or government policy. If a user has access to the underlying tables he can bypass the views entirely. Also, performance of queries accessing views can trail VPD performance significantly.

The U.S. government has enacted legislation around patient privacy (HIPAA), financial transaction privacy (Gramm, Leach, Bliley), and anti-money laundering and suspicious activity reporting (Patriot Act). For the Patriot Act, financial institutions must run specific searches on financial transactions and flag those transactions that are deemed suspicious. These suspicious transactions are to be stored and investigated, but they must not be visible to anyone except the authorized investigators or risk non-compliance with Gramm, Leach, Bliley. GLB states that a person's financial transactions must be kept private. You begin to see the challenge here.

The benefits of both of these technologies are:

1. Data security policies are maintained at the data level, so there isn't a back door to the data using an ad-hoc or other query/reporting tool as there is in application based security management
2. Procedures and policies are built once, regardless of the users, access methods or the applications used.
3. Greatly reduces the need for database triggers, views, and application logic to segregate data, thereby increasing performance and reducing maintenance requirements.

Potential issues to be evaluated before deciding to implement row level security

1. Some applications expect to see all of the information. One case in particular is Computer Aided Design systems. If a designer has access to the wheel assembly of a vehicle, but his clearance level prevents him from seeing past where the assembly connects to the frame or fuselage, what will the CAD application do when it is expecting the missing data?
2. How much of your data really needs to be secured in this manner?
3. Which technology is best for the information you need to protect?

Differences between VPD and OLS

VPD	OLS
-Included in Enterprise Edition	-Enterprise Edition security option
-Build your own security policies	-Security policy and label infrastructure
-Custom Development	-Out-of-the box
-Not Evaluated	-Evaluated against Common Criteria

Similarities between VPD and OLS

-Both supply API's
-Oracle Policy Manager can manage both
-Excellent for hosting, data warehouses
-Centralized security in the database
-Restrict access at the row level
-Patented technologies

Virtual Private Database

Benefits

Information from disparate parties can be stored in the same table, with separation of access handled behind the scenes. Most applications will not be adversely affected by the dynamic query modification. The technology is part of the database license, and writing PL/SQL procedures and policies is an elegant way to manage information access. Using “application context” even a user accessing data through a pooled connection can be given unique access controls, can have their true identity maintained, and as such, can be audited.

Challenges

If the policy is complex enough to require many hours of PL/SQL programming, Label Security should be considered. Also, Virtual Private Database will never be evaluated under Common Criteria or other information security standards, because each implementation is custom built. Also, referential integrity will bypass VPD and must be worked into the design.

How it Works

With VPD the query is re-written with qualifying statements. For example, Joan is in charge of payroll for the western region and asks for the salaries of all employees. The query would look something like:

```
> Select name, sal from emp;
```

Virtual Private Database would determine Joan’s access privileges and append the qualifying statement to the query:

```
> Select name, sal from emp
  where region = western;
```

Application Context

The user and department are not the only criteria that can be used to modify the query. Application Context is a feature of VPD that can consider many other variables before dynamically granting a user access to rows of a table. These variables include: authentication data, auth type, client info, current user, current userid, DB domain, DB name, external name (for SSL authenticated sessions), host, Language, network protocol, proxy and session user, time of day, connection type, IP address, application used and many more. (Unknown #1, table 6-2)

Building VPD (Theriault, Newman, p. 314-319)

Security policies and application contexts are used to build a Virtual Private database environment. The developer needs the following privileges:

CREATE ANY CONTEXT
EXECUTE_CATALOG_ROLE

Security Policy is a function that is used to determine how the query will be modified to restrict access to certain rows. It can be activated for all of the statements accessing the table or view, or only certain types of actions, such as UPDATE, SELECT, or DELETE. These policies are managed using the DBMS_RLS package. There are four procedures in this package:

Procedure	Purpose
dbms_rls.add_policy	Adds a policy to a table or view
dbms_rls.drop_policy	Drops a policy from a table or view
dbms_rls.refresh_policy	Forces a reparse of open cursors associated with a policy, so that a new policy or change to a policy can be implemented immediately
dbms_rls.enable_policy	Enables or disables a policy that was previously added to a table or view

These procedures make it possible to:

1. Specify the table or view the policy is added to
2. The name of the policy
3. The function that implements the policy
4. The type of statement that the policy is applied to (select, insert, update, delete)
5. Additional information you deem appropriate for the policy

For example:

Dbms_rls.add_policy
(‘nelson’,’emp_time_tab’,’emp_policy’,’secusr’,’emp_sec’,’select’);

emp_policy is the policy, secusr is the schema, and emp_sec is the function used.

Application Context is a namespace with a corresponding set of attribute/value pairs and is bound to a PL/SQL package for setting values in the context. This stops someone from setting values to grant access to unauthorized records for malicious purposes. Following are the steps required to implement application context.

1. Create a PL/SQL package that sets the context for the application
2. Create a unique context and associate it with the PL/SQL package
3. Set the context.
4. Use the context in a policy function.

1. Creating the package APP_SECURITY_CONTEXT This will set the attribute called “empno” in the user context APP_CONTEXT. The employee ID for the employee is retrieved from a table based on the current user. The current user is retrieved from the default application context (USERENV) by using the SYS_CONTEXT function.

```
Create or replace package APP_SECURITY_CONTEXT is
        Procedure SET_EMPNO;
end;
create or replace package body APP_SECURITY_CONTEXT is
        procedure SET_EMPNO
        IS
        EMP_ID number;
        Begin
                select EMPNO into EMP_ID from EMP_TAB
                        where EMPLOYEE_NAME =
                        SYS_CONTEXT('USERENV','SESSION_USER');
                dbms_session.set_context ('APP_CONTEXT','EMPLOYEE_NO',
                EMP_ID);
        end;
end;
```

The syntax for the SYS_CONTEXT function is:

Sys_context (‘NAMESPACE’,’ATTRIBUTE’, [LENGTH])

The procedure returns the value of ATTRIBUTE as defined in the package currently associated with the context NAMESPACE. It is evaluated once for each statement execution and is treated as a constant during type checking for optimization.

2. Create a unique context and associate it with the PL/SQL Package. Context names must be unique within the entire database, not just the schema. Contexts are owned by SYS.

Create context APP_CONTEXT using APP_SECURITY_CONTEXT;

3. Set the Context. You can set the user’s security automatically by using and event trigger to pull session info into the context.

4. Use the context in a policy function.

```
create or replace package body APP_SECURITY as
/*limits select statements based on employee number*/
function EMPNO_SEC (D1 varchar2, D2 varchar2) return varchar2
is
```

```
        D_PREDICATE varchar2 (2000);
        begin
                D_PREDICATE = 'EMPNO = SYS_CONTEXT
                (''APP_CONTEXT'',''EMPLOYEE_NO'');
                Return D_PREDICATE;
        end EMPNO_SEC;
end APP_SECURITY;
```

Finally, we add the policy for the user NELSON:

```
dbms_rls.add_policy
('NELSON','EMP_TIME_TAB','EMP_POLICY','SECUSR','APP_SECURITY,EMP
NO_SEC','SELECT')
```

If an employee executes a select statement on the table EMP_TIME_TAB will only return that employee's information, regardless of the application or access method. It will convert the statement from

```
Select * from EMP_TIME_TABLE;
to
select * from EMP_TIME_TABLE
        where EMPNO = SYS_CONTEXT('APP_CONTEXT','EMPLOYEE_NO);
```

As you can see from the above "simple" example, it takes some thought and effort to implement row level security using Virtual Private Database, but once it is set up, it reduces significantly the labor needed to manage information access policies in the database at the row level.

Label Security

Military and government organizations are not the only groups that can benefit from Label Security. Any company that needs to keep information centrally located, yet must ensure that only authorized access is allowed, especially where data marts and data warehouses are used, and want or need to have a graphical interface to manage it.

Label Security can use up to three sets of criteria to determine a users access privileges to individual rows of managed tables. These are Levels, Compartments, and Groups.

LEVEL - determines how sensitive the information is and the user's clearance to view the data. Military and Department of Defense levels include public, sensitive, secret, and top secret. Commercial companies can use levels like public, confidential, proprietary, etc. to set access control. Since the actual data tagging is a numeric value, the actual wording for the security level is very flexible. This is normally set up in a hierarchical access state, where a person with top secret access can view information that is secret, sensitive, or public. Someone with proprietary, can view confidential, and so on.

Level Examples:

Military
- Top Secret
- Secret
- Confidential
- Unclassified

Financial Services
- Acquisitions
- Corporate
- Client
- Operations

Commercial
- Proprietary
- Internal Only
- Company Confidential
- Public

COMPARTMENT - is a non-hierarchical value that further defines what areas the data is restricted to. Someone may have authorization for all information top secret and below and access to all groups, but if they do not have the specific compartment access, they will not see the data. In general, most commercial applications of OLS will not need compartment level granularity.

To help clarify how a compartment is used, let's look at a military application.

A defense contractor is designing and building a new transport vehicle. Different sections of the vehicle have different security clearance requirements during the design and manufacturing phases. The project supervisor has top secret clearance and is in all the groups responsible for this project. The "groups" labels can include wheel assembly, offensive and defensive systems, and self destruct technology.

Both Level and Group labels are hierarchical, meaning the project supervisor can view anything from top secret through unclassified across all the groups.

Now add a requirement to install a top secret device for Special Forces, and only Special Forces cleared personnel are authorized to know anything about the device's purpose, design, or contents. All information regarding this device would have a specific compartment designation, so even the project supervisor would not be able to see this information unless he/she also had that compartment clearance.

Compartment Examples:

- Military
  - Alpha
  - SAC
  - Sigma
- Financial
  - Insurance
  - Trusts
  - Commercial Loans
- Commercial
  - Litigation
  - AlphaDesign

GROUP - sets what projects, departments, locations or otherwise organized information can be accessed. It sets the owner of the information and further restricts access to people outside of the authorized group. This can also be set up in a hierarchical structure.

Group Examples:

- Military
  - Army
  - NATO
  - U.S.
- Financial
  - Client
  - Trustee
  - Beneficiary
- Commercial
  - Northwest
  - Pacific Rim
  - HR

To help clarify how a compartment is used, let's look at a military application example.

A defense contractor is designing and building a new transport vehicle. Different sections of the vehicle have different security clearance requirements during the design and manufacturing phases. The project supervisor has top secret clearance and is in all the groups responsible for this project. The label groups can include wheel assembly, offensive and defensive systems, and self destruct

technology. Both Level and Group labels are hierarchical, meaning the project supervisor can view anything from top secret through unclassified across all the groups.
Now add a requirement to install a top secret device for Special Forces, and only Special Forces cleared personnel are authorized to know anything about the device's purpose, design, or contents. All information regarding this device would have a compartment designation, so even the project supervisor would not be able to see this information unless he/she also had that compartment clearance.

Benefits

Label Security is managed through an easy to use, graphical interface. It has passed Common Criteria EAL-4, which helps companies using the product show due diligence of data security compliance. Also, Label Security is robust and flexible in the configuration and management of policies with up to 10,000 distinct values for each of the 3 label criteria.

Challenges

Access to rows in a particular table is based on the "Label" attached to each row, which is stored in a numeric column that is added to the tables being managed. Some applications do not accept modifications to the underlying table definitions. Even if it doesn't break the application, the support contract may become invalid. Label Security helps address this issue by making it possible to "hide" the column, thereby making it invisible to the application using the table. Before deciding to implement Label Security, it is prudent to determine how the applications accessing the underlying data will be affected.

Performance depends on implementation techniques and policy complexity. It is recommended that bitmap indexes are placed on the label column to improve performance. Partitioning can also help with performance of Oracle Label Security.

Installation

You must run the Oracle Universal Installer to add Label Security to the database. This product is not installed as part of the default configuration, as it is an additionally priced item. You should select the custom option and add OLS to the list of install options. (Czuprynski, Part2: Implementation)

Configuration

Use either the Oracle Database Configuration Assistant (DBCA) or the script $ORA_HOME/rdbms/admin/catols.sql to configure the database. A new user will be created called LBACSYS with a default password of LBACSYS (Label Based Access Control). The required objects will be created and stored in the LBACSYS schema. (Czuprynski, Part2: Implementation)

Determining what levels, compartments and groups are needed for your organization to meet policy, regulatory, or other requirements is necessary at this

point, as is a thorough data classification review. Once you have this information, you are ready to get to work.

Sensitivity Labels
As we discussed earlier, the three main components to the classification of the rows in a table are Level, Compartment, and Group and are used in unison. For example:

Level	Compartment	Group
Top Secret	SAC	Airforce
Confidential		Europe

These are long names tied to the numeric values which are stored in the Label column of the table being managed.
Within the Policy Manager, you set the numeric value and tie it to short and long names.

Level ID	Short Name	Long Name
10000	PB	Public
20000	CD	Confidential
30000	TS	Trade Secret

You will do the same for Compartments:

Compartment ID	Short Name	Long Name
100	HR	Human Resources
200	LE	Legal
300	SA	Sales
400	MK	Marketing

Finally, you set up the groups:

Group ID	Short Name	Long Name	Parent
0	CO	Corporate	(none)
10	HK	Hong Kong	CO
20	DT	Dallas Texas	CO
30	FN	Fargo North Dakota	CO

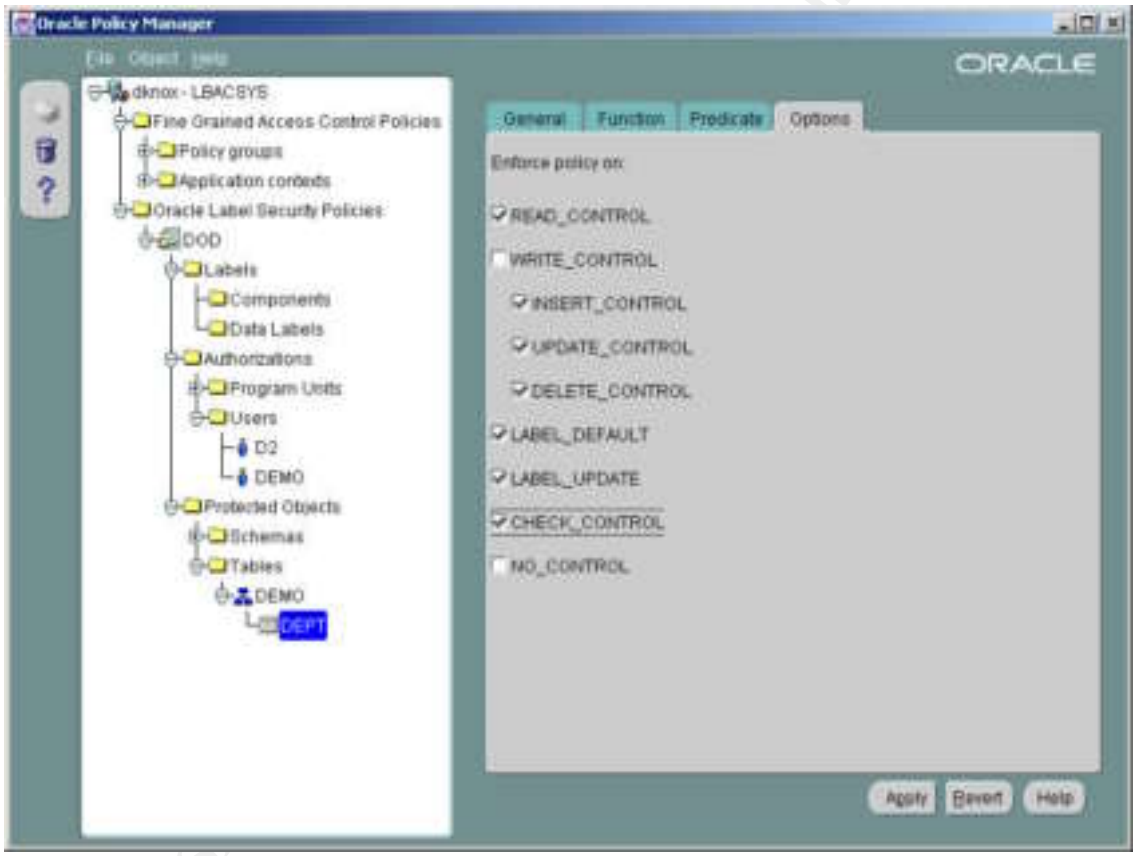
Notice that groups has an additional attribute. I have set this up in a hierarchical format, so someone with Group “CO” can view data across all child groups.

The next step is to group the various attributes of these components into labels for the various rows and for the users that will be accessing the rows.
Policy Labels

Label ID	Label Tag
10000	PB::CO
20100	CD:HR:CO
30320	TS:SA:DT

If you look at the last line above, you will see that Label ID 30320 breaks down to 30000 = Trade Secret, 300 = Sales, and 20 = Dallas, Texas. When the label is applied to a record, only a user with the Level of 30000 or higher, a group of Dallas Texas(20) or Corporate(0), and the specific compartment Sales(300) will have access to that row.

All of this can be built via OLS packages and procedures, or via Policy Manager, which is simple to use and very effective. Remember, that this product is a result of federal agency security requirements, which included the need for a graphical administration environment. Note also that once you have purchased Label Security, Policy Manager will manage your VPD policies and application contexts.



The previous image (screen capture from one of my demos) shows Oracle Policy manager and the granularity of control over tables, users, policies, etc. being managed. Also, a security administrator has the flexibility to change the policy enforcement as required with the click of a check box.

Automating the Labeling of existing Rows

Because this can all be done via procedures and packages, you can write a script that sets the row label based on attributes within the record itself or other attributes you would need to provide.

Management of inserts, updates, deletes
As shown in the graphic above, simply by checking a box, you can enforce extremely granular policies on how information is accessed updated or inserted. This is also true of the users you are managing. Privileges tied to users include: WRITEUP and WRITEDOWN, which allows a user to change the level of sensitivity of a label.
WRITEACROSS – allows a user to modify the compartments and groups within a policy's limits.

User Label Authorizations
There are a variety of configurable authorizations for an individual user. These would normally be set up and managed by a security administrator and include maximum and minimum access level, default level, row level, read and write compartments, and read and write groups. (Needham, Davidson, p 6) All of these determine how a particular user will interact with the data he or she is allowed to access or modify.

Conclusion

Oracle has put a great deal of thought and effort into these two technologies, and provides developers and security administrators robust, secure, highly configurable tools to manage security where the information is stored. VPD and Label security use slightly different methods to restrict access down to the individual rows of a table. These technologies have simplified and strengthened information security by reducing the responsibility of application security and placing the policies and responsibilities where they belong, with the data.

Virtual Private Database is recommended for situations where the requirements and security policies are not complex and the implementation does not require a recognized standard security evaluation like Common Criteria.

Label Security has passed the strictest international security standard evaluations, including Common Criteria. The Policy Manager application makes configuration and management a point and click operation, for a majority of the work, plus it can manage existing VPD policies and contexts. Label security is recommended for complex policies, or high security environments.

References

Kyte, Thomas. “Oracle expert one-on-one.” Wrox Press Ltd March, 2002: 913-924.

Theriault, Marlene. Newman, Aaron. “Oracle Security Handbook.” Oracle Press Editions from Osborne 2001: 314-319

Czuprynski, Jim. “Oracle Label Security, Part 1: Overview.” www.databasejournal.com August 29, 2003 URL: http://www.databasejournal.com/features/oracle/article.php/3065431 (January 8,2004)

Czuprynski, Jim. “Oracle Label Security, Part 2: Implementation.” www.databasejournal.com September 18, 2003 URL: http://www.databasejournal.com/features/oracle/article.php/3077761 (January 8, 2004)

Needham, Paul. Davidson, Mary Ann. “Security Solutions in Applications Hosting.” 2003 URL: http://otn.oracle.com/deploy/security/ols/pdf/597.pdf (January 12, 2004)

Unknown #1. “Oracle9*i* SQL Reference Release 2 (9.2)” 2002 Part Number A96540-02 URL: http://download-west.oracle.com/docs/cd/B10501_01/server.920/a96540/functions122a.htm#88703 (January 13, 2004)

Unknown #2. “Oracle9*i* Application Developer's Guide – Fundamentals” 2002 Release 2 (9.2) Part Number A96590-01 URL: http://download-west.oracle.com/docs/cd/B10501_01/appdev.920/a96590/adgsec02.htm#1006752 (January 9, 2004)

Unknown #3. “Oracle Label Security and Database Consolidation.” 2003 URL: http://www.oracle.com/ip/se/ols9ir2new_bwp_0303.pdf (January 5, 2004)