Addendum to the CIS Benchmark for Oracle 12c

Please note these checks are not part of the official CIS Benchmark for 12c, though they have been submitted to CIS for inclusion at some later date, as I believe the benchmark can be improved. These checks have been written using the same format with all checks numbering from 6.1 to 6.32.
David Litchfield
22nd January 2017

6.1 Ensure SYS has no INHERIT PRIVILEGES for other users

Profile Applicability:

- Level 1 - RDBMS

Description:

The INHERIT PRIVILEGES privilege allows the owner of an invoker rights procedure to inherit the privileges of the invoker during the execution of the procedure. By default, no user should have this privilege

Rationale:

If an attacking user has the INHERIT PRIVILEGES privilege for the SYS user then they can inherit SYS privileges when injecting an invoker rights function into a PL/SQL injection flaw.

Audit:

To assess this recommendation execute the following SQL statement

```
SELECT TABLE_NAME, GRANTEE FROM DBA_TAB_PRIVS
WHERE PRIVILEGE = 'INHERIT PRIVILEGES'
AND GRANTEE = 'SYS';
```

Ensure that no records are returned.

Remediation:

To remediate this, execute the following SQL statement

```
REVOKE INHERIT PRIVILEGES ON USER <username> FROM SYS;
```

References:

1. http://docs.oracle.com/database/121/DBSEG/dr_ir.htm#DBSEG658

6.2 Ensure no user has the INDEX privilege on SYS or SYSTEM owned tables

Profile Applicability:

- Level 1 - RDBMS

Description:

A user may create an index on a table owned by another user if they have the INDEX privilege on the table in question.

Rationale:

If the index is a function-based index the function will execute with the privileges of the table's owner. This may provide an attacker the opportunity to execute SQL as SYS or SYSTEM.

Audit:

To assess this recommendation execute the following SQL statement

```
SELECT GRANTEE, OWNER, TABLE_NAME FROM DBA_TAB_PRIVS WHERE PRIVILEGE =
'INDEX' AND OWNER IN ('SYS', 'SYSTEM');
```

Ensure no records are returned.

Remediation:

To remediate this, execute the following SQL statement

```
REVOKE INDEX ON <schema.table> FROM <username>;
```

References:

1. http://www.davidlitchfield.com/Privilege_Escalation_via_Oracle_Indexes.pdf

6.3 Ensure an OS user has been set for Java runtime for SYS

Profile Applicability:

- Level 1 - RDBMS

Description:

By default, if an OS command is executed from the Java Runtime it executes with the privileges of the process owner, typically oracle on Linux or LocalSystem on Windows. It is possible to set a low privileged OS user instead.

Rationale:

As the oracle user or LocalSystem account have full control over the database, it is strongly recommended that an OS user be specified for the Java Runtime for the SYS user and any user that owns Java classes that allow the execution of OS commands.

Audit:

To assess this recommendation execute the following SQL statement

```
SELECT OS_USERNAME FROM SYS.JAVA$RUNTIME$EXEC$USER$ WHERE OWNER#=0;
```

Ensure an record is returned.

Remediation:

To remediate this, execute the following SQL statement

```
EXEC
DBMS_JAVA.SET_RUNTIME_EXEC_CREDENTIALS('<username>','<password>');
```

References:

1. https://docs.oracle.com/database/121/JJDEV/chten.htm#JJDEV10300

6.4 Ensure java.io.FilePermission.execute is revoked from unauthorized 'GRANTEE'

Profile Applicability:

- Level 1 - RDBMS

Description:

The execute action for the java.io.FilePermission allows a user to execute OS commands from the Java Runtime.

Rationale:

Access to this privilege should be limited to ensure that only trusted users have the ability to execute OS commands from the Java Runtime.

Audit:

To assess this recommendation execute the following SQL statement

```
SELECT GRANTEE FROM DBA_JAVA_POLICY WHERE TYPE_NAME =
'java.io.FilePermission' AND ACTION LIKE '%execute%' AND (NAME='<<ALL
FILES>>' OR NAME LIKE '%*%') AND KIND = 'GRANT' AND ENABLED='ENABLED'
AND GRANTEE !='JAVASYSPRIV';
```

Ensure only trusted users or no records are returned.

Remediation:

To remediate this, execute the following SQL statement

```
EXEC DBMS_JAVA.REVOKE_PERMISSION ('<username>',
'java.io.FilePermission', '<<ALL FILES>>', 'execute');
```

References:

1. https://docs.oracle.com/database/121/JJDEV/chten.htm#CBBIHIGI

6.5 Ensure java.security.AllPermission is revoked from unauthorized 'GRANTEE'

Profile Applicability:

- Level 1 - RDBMS

Description:

The java.security.AllPermission grants all permissions and should not be used.

Rationale:

Anyone with java.security.AllPermission can effectively run any Java without any security controls.

Audit:

To assess this recommendation execute the following SQL statement

```
SELECT GRANTEE FROM DBA_JAVA_POLICY WHERE TYPE_NAME =
'java.security.AllPermission' AND KIND = 'GRANT' AND ENABLED='ENABLED'
AND GRANTEE != 'SYS';
```

Ensure no records are returned.

Remediation:

To remediate this, execute the following SQL statement

```
EXEC DBMS_JAVA.REVOKE_PERMISSION ('<username>',
'java.security.AllPermission');
```

References:

1. https://docs.oracle.com/database/121/JJDEV/chten.htm#CBBIHIGI
2. https://docs.oracle.com/javase/7/docs/api/java/security/AllPermission.html

6.6 Ensure wildcard java.io.FilePermission.write is revoked from unauthorized 'GRANTEE'

Profile Applicability:

- Level 1 - RDBMS

Description:

The write action of the java.io.FilePermission grants the ability to write files to the OS.

Rationale:

Anyone able to write to the OS using the write action of the java.io.FilePermission may be able to compromise the database server. Files are written with the privileges of the oracle user on Linux and LocalSystem on Windows.

Audit:

To assess this recommendation execute the following SQL statement

```
SELECT GRANTEE FROM DBA_JAVA_POLICY WHERE TYPE_NAME =
'java.io.FilePermission' AND ACTION LIKE '%write%' AND NAME='<<ALL
FILES>>' AND KIND = 'GRANT' AND ENABLED='ENABLED' AND GRANTEE !=
'JAVASYSPRIV';
```

Ensure no records are returned.

Remediation:

To remediate this, execute the following SQL statement

```
EXEC DBMS_JAVA.REVOKE_PERMISSION ('<username>',
'java.io.FilePermission', '<<ALL FILES>>', 'write');
```

References:

1. https://docs.oracle.com/database/121/JJDEV/chten.htm#CBBIHIGI

6.7 Ensure java.lang.RuntimePermission.loadLibrary is revoked from unauthorized 'GRANTEE'

Profile Applicability:

- Level 1 - RDBMS

Description:

The loadLibrary permission allows grantees to load OS libraries into the server's address space.

Rationale:

Allowing a user to load a library into the server's address space could lead to a full compromise of the database server.

Audit:

To assess this recommendation execute the following SQL statement

```
SELECT GRANTEE, NAME, SEQ FROM DBA_JAVA_POLICY WHERE TYPE_NAME =
'java.lang.RuntimePermission' AND NAME LIKE '%loadLibrary%' AND KIND =
'GRANT' AND ENABLED='ENABLED' AND GRANTEE NOT IN ('SYS', 'ORDSYS');
```

Ensure no records are returned.

Remediation:

To remediate this, execute the following SQL statement

```
EXEC DBMS_JAVA.DELETE_PERMISSION(<seq>);
```

References:

1. https://docs.oracle.com/database/121/JJDEV/chten.htm#CBBIHIGI
2. http://www.davidlitchfield.com/oracle_backdoors.pdf

6.8 Ensure membership of the JAVASYSPRIV role is revoked from unauthorized 'GRANTEE'

Profile Applicability:

- Level 1 - RDBMS

Description:

The JAVASYSPRIV role has a larger number of very powerful Java permissions. Membership of this role should be tightly controlled. Ensure that only trusted users have membership.

Rationale:

Any user with membership of the JAVASYSPRIV role can trivially abuse its permissions to take full control of the database server.

Audit:

To assess this recommendation execute the following SQL statement

```
SELECT GRANTEE FROM DBA_ROLE_PRIVS WHERE GRANTED_ROLE = 'JAVASYSPRIV'
AND GRANTEE != 'SYS';
```

Ensure no records are returned

Remediation:

To remediate this, execute the following SQL statement

```
REVOKE JAVASYSPRIV FROM <username>;
```

References:

1. https://docs.oracle.com/database/121/JJDEV/chten.htm#JJDEV13325

6.9 Ensure SELECT on Dynamic Performance Views that contain SQL is revoked from unauthorized 'GRANTEE'

Profile Applicability:

- Level 1 - RDBMS

Description:

A number of dynamic views contain previously executed SQL and values for bind variables. These may contain highly sensitive information.

Rationale:

Access to the dynamic views that contain previously executed SQL statements and bind variables should be tightly controlled because they may contain highly sensitive data.

Audit:

To assess this recommendation execute the following SQL statement

```
SELECT GRANTEE||':'||TABLE_NAME FROM DBA_TAB_PRIVS WHERE TABLE_NAME IN
('V_$SQL', 'GV_$SQL', 'V_$SQLTEXT', 'GV_$SQLTEXT',
'V_$SQLTEXT_WITH_NEWLINES', 'GV_$SQLTEXT_WITH_NEWLINES', 'V_$SQLAREA',
'GV_$SQLAREA', 'V_$SQL_SHARED_MEMORY', 'GV_$SQL_SHARED_MEMORY',
'V_$SQLAREA_PLAN_HASH', 'GV_$SQLAREA_PLAN_HASH', 'V_$SQLSTATS',
'GV_$SQLSTATS', 'WRH$_SQLTEXT', 'WRI$_ADV_SQLW_STMTS');
```

Ensure no records are returned.

Remediation:

To remediate this, execute the following SQL statement

```
REVOKE SELECT ON <tablename> FROM <username>;
```

References:

6.10 Ensure 'EXECUTE' is revoked from 'PUBLIC' on 'DBMS_XMLSTORE'

Profile Applicability:

- Level 1 - RDBMS

Description:

The DBMS_XMLSTORE package accepts a table name and XML as input and then inserts into or updates that table.

Rationale:

This package can be used as an auxiliary inject function in a SQL injection attack. Revoking the EXECUTE privilege from PUBLIC on this packages helps harden the server.

Audit:

To assess this recommendation execute the following SQL statement

```
SELECT GRANTEE FROM DBA_TAB_PRIVS WHERE TABLE_NAME = 'DBMS_XMLSTORE'
AND GRANTEE = 'PUBLIC' AND PRIVILEGE = 'EXECUTE';
```

Ensure no record is returned.

Remediation:

To remediate this, execute the following SQL statement

```
REVOKE EXECUTE ON DBMS_XMLSTORE FROM PUBLIC;
```

References:

1. http://www.davidlitchfield.com/DBMS_XMLSTORE_PLSQL_Injection.pdf

6.11 Ensure 'EXECUTE' is revoked from 'PUBLIC' on 'DBMS_XMLSAVE'

Profile Applicability:

- Level 1 - RDBMS

Description:

The DBMS_XMLSAVE package accepts a table name and XML as input and then inserts into or updates that table.

Rationale:

This package can be used as an auxiliary inject function in a SQL injection attack. Revoking the EXECUTE privilege from PUBLIC on this packages helps harden the server.

Audit:

To assess this recommendation execute the following SQL statement

```
SELECT GRANTEE FROM DBA_TAB_PRIVS WHERE TABLE_NAME = 'DBMS_XMLSAVE'
AND GRANTEE = 'PUBLIC' AND PRIVILEGE = 'EXECUTE';
```

Ensure no record is returned.

Remediation:

To remediate this, execute the following SQL statement

```
REVOKE EXECUTE ON DBMS_XMLSAVE FROM PUBLIC;
```

References:

1. http://www.davidlitchfield.com/DBMS_XMLSTORE_PLSQL_Injection.pdf

6.12 Ensure 'EXECUTE' is revoked from 'PUBLIC' on 'DBMS_AW'

Profile Applicability:

- Level 1 - RDBMS

Description:

The DBMS_AW package contains procedures and functions for interacting with Analytic Workspaces. Several functions accept OLAP DML as input which can be used to execute SQL.

Rationale:

This package can be used as an auxiliary inject function in a SQL injection attack. Revoking the EXECUTE privilege from PUBLIC on this packages helps harden the server.

Audit:

To assess this recommendation execute the following SQL statement

```
SELECT GRANTEE FROM DBA_TAB_PRIVS WHERE TABLE_NAME = 'DBMS_AW' AND
GRANTEE = 'PUBLIC' AND PRIVILEGE = 'EXECUTE';
```

Ensure no record is returned.

Remediation:

To remediate this, execute the following SQL statement

```
REVOKE EXECUTE ON DBMS_AW FROM PUBLIC;
```

References:

1. http://www.davidlitchfield.com/ExploitingPLSQLInjectionCREATESESSION.pdf

6.13 Ensure 'PUBLIC' has not been assigned membership of a role

Profile Applicability:

- Level 1 - RDBMS

Description:

PUBLIC is not assigned membership of a role by default. This check ensures this default is still in place.

Rationale:

By granting PUBLIC membership of a role, every user of the database server will have that role membership and therefore any privileges that role may have.

Audit:

To assess this recommendation execute the following SQL statement

```
SELECT GRANTED_ROLE FROM DBA_ROLE_PRIVS WHERE GRANTEE = 'PUBLIC';
```

Ensure no records are returned.

Remediation:

To remediate this, execute the following SQL statement

```
REVOKE <rolename> FROM PUBLIC;
```

References:

6.14 Ensure 'PUBLIC' has not been granted a system privilege

Profile Applicability:

- Level 1 - RDBMS

Description:

By default, PUBLIC is not assigned any system privileges. This check is to ensure this default is still in place.

Rationale:

By granting PUBLIC a system privilege, every user of the database server will have that privilege.

Audit:

To assess this recommendation execute the following SQL statement

```
SELECT PRIVILEGE FROM DBA_SYS_PRIVS WHERE GRANTEE = 'PUBLIC';
```

Ensure no records are returned.

Remediation:

To remediate this, execute the following SQL statement

```
REVOKE <privilege> FROM PUBLIC;
```

References:

6.15 Ensure 'PUBLIC' does not have write access to any directories

Profile Applicability:

- Level 1 - RDBMS

Description:

An Oracle directory object gives a user access to the server's file system when using UTL_FILE or BFILENAME.

Rationale:

Allowing PUBLIC to write to the file system could allow the server to be compromised.

Audit:

To assess this recommendation execute the following SQL statement

```
SELECT TABLE_NAME FROM DBA_TAB_PRIVS WHERE GRANTEE = 'PUBLIC' AND
PRIVILEGE = 'WRITE';
```

Ensure no records are returned.

Remediation:

To remediate this, execute the following SQL statement

```
REVOKE WRITE ON <directory> FROM PUBLIC;
```

References:

6.16 Ensure no public database links exist

Profile Applicability:

- Level 1 - RDBMS

Description:

A database link allows users to connect to a remote database server.

Rationale:

A public database link allows all users to connect to a remote database server with the privileges of the username specified for the link. Access to remote databases should be tightly controlled.

Audit:

To assess this recommendation execute the following SQL statement

```
SELECT DB_LINK FROM DBA_DB_LINKS WHERE OWNER = 'PUBLIC';
```

Ensure no records are returned.

Remediation:

To remediate this, execute the following SQL statement

```
DROP PUBLIC DATABASE LINK <linkname>;
```

References:

6.17 Ensure 'ALL' Is Revoked from Unauthorized 'GRANTEE' on FGA$

Profile Applicability:

- Level 1 - RDBMS

Description:

The FGA$ table contains columns that contains the schema and name of a procedure to execute when a table is accessed and that has a Fine Grained Auditing policy defined for it.

Rationale:

If a user has the ability to insert into or update this table they have the ability to indirectly execute arbitrary SQL. Granting privileges on this specific table can be used as a means of backdooring an Oracle database server.

Audit:

To assess this recommendation execute the following SQL statement

```
SELECT GRANTEE, PRIVILEGE FROM DBA_TAB_PRIVS WHERE TABLE_NAME =
'FGA$';
```

Ensure no records are returned.

Remediation:

To remediate this, execute the following SQL statement

```
REVOKE <privilege> ON FGA$ FROM <username>;
```

References:

1. http://www.davidlitchfield.com/oracle_backdoors.pdf

6.18 Ensure 'ALL' Is Revoked from Unauthorized 'GRANTEE' on RLS$

Profile Applicability:

- Level 1 - RDBMS

Description:

The RLS$ table contains columns that contains the schema and name of a procedure to execute when a table is accessed and that has a Row Level Security policy defined on it.

Rationale:

If a user has the ability to insert into or update this table they have the ability to indirectly execute arbitrary SQL. Granting privileges on this specific table can be used as a means of backdooring an Oracle database server.

Audit:

To assess this recommendation execute the following SQL statement

```
SELECT GRANTEE, PRIVILEGE FROM DBA_TAB_PRIVS WHERE TABLE_NAME =
'RLS$';
```

Ensure no records are returned.

Remediation:

To remediate this, execute the following SQL statement

```
REVOKE <privilege> ON RLS$ FROM <username>;
```

References:

1. http://www.davidlitchfield.com/oracle_backdoors.pdf

6.19 Ensure 'ALL' Is Revoked from Unauthorized 'GRANTEE' on RADM$

Profile Applicability:

- Level 1 - RDBMS

Description:

The RADM$ table contains a column that contains SQL which will execute when a table is accessed and that table has a redaction policy defined for it.

Rationale:

If a user has the ability to insert into or update this table they have the ability to indirectly execute arbitrary SQL. Granting privileges on this specific table can be used as a means of backdooring an Oracle database server.

Audit:

To assess this recommendation execute the following SQL statement

```
SELECT GRANTEE, PRIVILEGE FROM DBA_TAB_PRIVS WHERE TABLE_NAME =
'RADM$';
```

Ensure no records are returned.

Remediation:

To remediate this, execute the following SQL statement

```
REVOKE <privilege> ON RADM$ FROM <username>;
```

References:

1. http://www.davidlitchfield.com/oracle_backdoors.pdf

6.20 Ensure 'ALL' Is Revoked from Unauthorized 'GRANTEE' on CONTEXT$

Profile Applicability:

- Level 1 - RDBMS

Description:

The CONTEXT$ table contains columns for the schema and name for a PL/SQL package that will execute for a given application context.

Rationale:

If a user has the ability to insert into or update this table they have the ability to indirectly execute arbitrary SQL. Granting privileges on this specific table can be used as a means of backdooring an Oracle database server.

Audit:

To assess this recommendation execute the following SQL statement

```
SELECT GRANTEE, PRIVILEGE FROM DBA_TAB_PRIVS WHERE TABLE_NAME =
'CONTEXT$';
```

Ensure no records are returned.

Remediation:

To remediate this, execute the following SQL statement

```
REVOKE <privilege> ON CONTEXT$ FROM <username>;
```

References:

1. http://www.davidlitchfield.com/oracle_backdoors.pdf

6.21 Ensure 'ALL' Is Revoked from Unauthorized 'GRANTEE' on COL$

Profile Applicability:

- Level 1 - RDBMS

Description:

The DEFAULT$ column of the COL$ table contains the SQL for tables with virtual columns and function-based indexes.

Rationale:

If a user has the ability to insert into or update this table they have the ability to indirectly execute arbitrary SQL. Granting privileges on this specific table can be used as a means of backdooring an Oracle database server.

Audit:

To assess this recommendation execute the following SQL statement

```
SELECT GRANTEE, PRIVILEGE FROM DBA_TAB_PRIVS WHERE TABLE_NAME =
'COL$';
```

Ensure no records are returned.

Remediation:

To remediate this, execute the following SQL statement

```
REVOKE <privilege> ON COL$ FROM <username>;
```

References:

1. http://www.davidlitchfield.com/oracle_backdoors.pdf

6.22 Ensure 'ALL' Is Revoked from Unauthorized 'GRANTEE' on TRIGGER$

Profile Applicability:

- Level 1 - RDBMS

Description:

The ACTION# column of the TRIGGER$ table contains contains the SQL of each trigger.

Rationale:

If a user has the ability to insert into or update this table they have the ability to indirectly execute arbitrary SQL. Granting privileges on this specific table can be used as a means of backdooring an Oracle database server.

Audit:

To assess this recommendation execute the following SQL statement

```
SELECT GRANTEE, PRIVILEGE FROM DBA_TAB_PRIVS WHERE TABLE_NAME =
'TRIGGER$';
```

Ensure no records are returned.

Remediation:

To remediate this, execute the following SQL statement

```
REVOKE <privilege> ON TRIGGER$ FROM <username>;
```

References:

1. http://www.davidlitchfield.com/oracle_backdoors.pdf

6.23 Ensure 'ALL' Is Revoked from Unauthorized 'GRANTEE' on SOURCE$

Profile Applicability:

- Level 1 - RDBMS

Description:

The SOURCE$ table contains the PL/SQL source for all packages, procedures and functions.

Rationale:

If a user has the ability to insert into or update this table they have the ability to indirectly execute arbitrary SQL. Granting privileges on this specific table can be used as a means of backdooring an Oracle database server.

Audit:

To assess this recommendation execute the following SQL statement

```
SELECT GRANTEE, PRIVILEGE FROM DBA_TAB_PRIVS WHERE TABLE_NAME =
'SOURCE$';
```

Ensure no records are returned.

Remediation:

To remediate this, execute the following SQL statement

```
REVOKE <privilege> ON SOURCE$ FROM <username>;
```

References:

1. http://www.davidlitchfield.com/oracle_backdoors.pdf

6.24 Ensure 'ALL' Is Revoked from Unauthorized 'GRANTEE' on OBJAUTH$

Profile Applicability:

- Level 1 - RDBMS

Description:

The OBJAUTH$ table stores details about which users or roles have which privileges on which objects.

Rationale:

If a user has the ability to insert into or update this table they have the ability to grant access to any object. Granting privileges on this specific table can be used as a means of backdooring an Oracle database server.

Audit:

To assess this recommendation execute the following SQL statement

```
SELECT GRANTEE, PRIVILEGE FROM DBA_TAB_PRIVS WHERE TABLE_NAME =
'OBJAUTH$';
```

Ensure no records are returned.

Remediation:

To remediate this, execute the following SQL statement

```
REVOKE <privilege> ON OBJAUTH$ FROM <username>;
```

References:

1. http://www.davidlitchfield.com/oracle_backdoors.pdf

6.25 Ensure 'ALL' Is Revoked from Unauthorized 'GRANTEE' on SYSAUTH$

Profile Applicability:

- Level 1 - RDBMS

Description:

The SYSAUTH$ table contains details about which users and roles have what system privileges.

Rationale:

If a user has the ability to insert into or update this table they have the ability to grant arbitrary system privileges. Granting privileges on this specific table can be used as a means of backdooring an Oracle database server

Audit:

To assess this recommendation execute the following SQL statement

```
SELECT GRANTEE, PRIVILEGE FROM DBA_TAB_PRIVS WHERE TABLE_NAME =
'SYSAUTH$';
```

Ensure no records are returned.

Remediation:

To remediate this, execute the following SQL statement

```
REVOKE <privilege> ON SYSAUTH$ FROM <username>;
```

References:

1. http://www.davidlitchfield.com/oracle_backdoors.pdf

6.26 Ensure 'ALL' Is Revoked from Unauthorized 'GRANTEE' on OBJ$

Profile Applicability:

- Level 1 - RDBMS

Description:

The OBJ$ table contains details about all objects on the server.

Rationale:

If a user has the ability to insert into or update this table they have the ability to create or modify existing objects which could lead to a server compromise. Granting privileges on this specific table can be used as a means of backdooring an Oracle database server

Audit:

To assess this recommendation execute the following SQL statement

```
SELECT GRANTEE, PRIVILEGE FROM DBA_TAB_PRIVS WHERE TABLE_NAME =
'OBJ$';
```

Ensure no records are returned.

Remediation:

To remediate this, execute the following SQL statement

```
REVOKE <privilege> ON OBJ$ FROM <username>;
```

References:

1. http://www.davidlitchfield.com/oracle_backdoors.pdf

6.27 Ensure 'ALL' Is Revoked from Unauthorized 'GRANTEE' on JAVA$POLICY$

Profile Applicability:

- Level 1 - RDBMS

Description:

The JAVA$POLICY$ table contains details about which users and roles have been granted what Java permissions.

Rationale:

If a user has the ability to insert into or update this table they have the ability to grant arbitrary Java privileges. Granting privileges on this specific table can be used as a means of backdooring an Oracle database server

Audit:

To assess this recommendation execute the following SQL statement

```
SELECT GRANTEE, PRIVILEGE FROM DBA_TAB_PRIVS WHERE TABLE_NAME =
'JAVA$POLICY$';
```

Ensure no records are returned.

Remediation:

To remediate this, execute the following SQL statement

```
REVOKE <privilege> ON JAVA$POLICY$ FROM <username>;
```

References:

1. http://www.davidlitchfield.com/oracle_backdoors.pdf

6.28 Ensure 'ALL' Is Revoked from Unauthorized 'GRANTEE' on FGA_LOG$

Profile Applicability:

- Level 1 - RDBMS

Description:

The FGA_LOG$ table contains audit records for tables with a defined Fine Grained Auditing policy.

Rationale:

This table may contain highly sensitive data so access should be tightly controlled.

Audit:

To assess this recommendation execute the following SQL statement

```
SELECT GRANTEE, PRIVILEGE FROM DBA_TAB_PRIVS WHERE TABLE_NAME =
'FGA_LOG$';
```

Ensure no records are returned.

Remediation:

To remediate this, execute the following SQL statement

```
REVOKE <privilege> ON FGA_LOG$ FROM <username>;
```

References:

6.29 Ensure no user has the EXECUTE privilege on DBMS_PDB_EXEC_SQL

Profile Applicability:

- Level 1 - RDBMS

Description:

The DBMS_PDB_EXEC_SQL procedure takes an SQL query as a parameter and executes it.

Rationale:

When the DBMS_PDB_EXEC_SQL procedure executes it does so with SYS privileges allowing the user to execute SQL as SYS. This could lead to a full compromise of the database server and, as such, the execute privilege on this procedure should be tightly controlled.

Audit:

To assess this recommendation execute the following SQL statement

```
SELECT GRANTEE FROM DBA_TAB_PRIVS WHERE TABLE_NAME =
'DBMS_PDB_EXEC_SQL' AND PRIVILEGE = 'EXECUTE';
```

Ensure no records are returned.

Remediation:

To remediate this, execute the following SQL statement

```
REVOKE EXECUTE ON DBMS_PDB_EXEC_SQL FROM <username>;
```

References:

6.30 Ensure _dbms_sql_security_level is set to 1 or 2

Profile Applicability:

- Level 1 - RDBMS

Description:

The _dbms_sql_security_level parameter controls how access to cursors used by DBMS_SQL is controlled. The default setting of 1 ensures that the effective user ID for both parse and execute operations are the same; this prevents cursor snarfing attacks. A setting 2 is more restrictive; a setting of 0 or 384 turns off security checks thus opening the server to cursor snarfing attacks.

Rationale:

To ensure the server is not vulnerable to cursor snarfing attacks ensure the _dbms_sql_security_level parameter is set to 1 or 2.

Audit:

To assess this recommendation execute the following SQL statement

```
SELECT c.ksppstvl FROM x$ksppi a, x$ksppsv c WHERE a.indx = c.indx AND
a.ksppinm = '_dbms_sql_security_level';
```

Ensure the value 1 or 2 is returned.

Remediation:

To remediate this, execute the following SQL statement

```
ALTER SYSTEM SET "_dbms_sql_security_level" = 1 SCOPE=SPFILE;
```

References:

1. http://docs.oracle.com/cd/B28359_01/appdev.111/b28419/d_sql.htm#i1027587
2. http://www.davidlitchfield.com/cursor-snarfing.pdf

6.31 Ensure _sys_logon_delay is set to 1 or higher

Profile Applicability:

- Level 1 - RDBMS

Description:

The _sys_logon_delay hidden parameter determines the length in seconds of the delay between failed logon attempts for the SYS user (and other users controlled by the password file such as SYSBACKUP, SYSDG and SYSKM)

Rationale:

If the _sys_logon_delay parameter is set to 0 then there is no enforced delay between logon attempts in a brute force attack. As SYS cannot be logged out, by setting this parameter to 1 or higher attackers can be slowed down.

Audit:

To assess this recommendation execute the following SQL statement

```
SELECT c.ksppstvl FROM x$ksppi a, x$ksppsv c WHERE a.indx = c.indx AND
a.ksppinm = '_sys_logon_delay';
```

Ensure the value returned is 1 or higher.

Remediation:

To remediate this, execute the following SQL statement

```
ALTER SYSTEM SET "_sys_logon_delay" = 1 SCOPE=SPFILE;
```

References:

6.32 Ensure _fifteenth_spare_parameter is set to 'all'

Profile Applicability:

- Level 1 - RDBMS

Description:

The _fifteenth_spare_parameter determines whether the oradebug facility has been disabled or not.

Rationale:

The oradebug facility can be used to read and write to memory, run arbitrary functions (and as such OS commands via the system() function call) and turn off auditing. Disabling the oradebug facility will help harden the server.

Audit:

To assess this recommendation execute the following SQL statement

```
SELECT c.ksppstvl FROM x$ksppi a, x$ksppsv c WHERE a.indx = c.indx AND
a.ksppinm = '_fifteenth_spare_parameter';
```

Ensure the value returned is 'all'

Remediation:

To remediate this, execute the following SQL statement

```
ALTER SYSTEM SET "_fifteenth_spare_parameter" = all SCOPE=SPFILE;
```

References:

1. http://blog.red-database-security.com/2013/09/13/fix-for-oradebug-disable-auditing-available-11-2-0-3/