Database Command Line Scanner (v1.2)

Contents

1. Overview

This document describes "Database Command Line Credit Card Scanning" (DCL Scanner) software. This module enables the Scanner to be run in a "command line/stand alone" mode as opposed to using a web server that would required a separate install. This is useful in high security areas where running a web server may not be feasible.

The Scanner searches the database for clear text payment card numbers and provides a PDF report with the detailed result of the scan.

The scanning is done in non-intrusive way: no data is stored in the target databases and no agents are installed on database servers.

It provides important information for PCI (Payment Card Industry) related audits. The generated reports can be used as proof that no clear payment card data exists in a given database.

1.1 Architecture

The command line scanner is a standalone module. Extract it into a directory on the database server or any other server/client. It just required a TNS based connection to the databases to be scanned. There is no web server and no web interface. All actions are controlled by a single configuration file.

1.2 Key features

The scanner has a number of key features:

- It is specifically written to work with Oracle databases and takes advantage of advanced Oracle features;
- It is designed to run with minimal performance impact. So for example it can be run on a mission critical production database safely;
- It provides full control of the degree of parallelism to be used;
- A scan can be paused and resumed as required;
- It can be run on multi-terabyte databases;
- It runs on Oracle RAC databases and on Oracle Exadata machines;
- A PDF report can be generated from a partial, failed or unfinished scan. So, for example when scanning a very large database, interim reports can be obtained.
- It has features for enabling a "quick initial or sample scan" to be conducted. For example:
    - specific tables can be included or excluded,
    - for large partitioned tables the scan can be limited to the latest "n" range partitions
    - scans can be limited to only the data that has changed in the last 2 days

- large CLOB columns can be omitted

2. Installation

The command line scanner runs on Linux, Solaris, AIX and HPUX. The software is distributed in the form of self-extracting packages.

2.1 Unix Installation

Required Privileges: None, if installed in the local home directory. Otherwise – root privileges may be required.

Unix Installation package can be downloaded from this URL:
http://www.dbscanlabs.com /dcls_download.html

To install the software the following steps need to be performed:

1. Create a directory in which DCL Scanner will reside
2. Download the DCL Scanner Installer and save it into the created directory (for example, for Linux 32bit OS it will be dclscanner-x.x-Linux-32b-install.sh, where x.x is the version).
3. The installer is a self-extracting archive, so first make it executable "**chmod u+x dclscanner-1.2-Linux-32b-install.sh**"
4. Run the installer: "**./dclscanner-1.2-Linux-32b-install.sh**"
5. After the installation has completed, create a config file for your database (use sample.cfg as a template) and then run the command: "**dclscanner.sh -c <config_file_name>**" to start scanning.

Basic Usage

3.1 Scanning Parameters

The scanning parameters are specified in the .cfg file. The following are the key parameters. Please refer to Appendix C for the complete list of configuration parameters and their syntax.

- tns_name
  - TNS connection to the database (can be in the form of a TNS name or an Easy Connect String, please see section 3.1.1 Configuring database connection for more details)
- db_user
  - Database user under which the scan will be run
  - This user will require select privileges on all the tables to be scanned
- password
  - Password of db user. This will be encrypted automatically in the config file
- schema_list
  - List of schemas to be scanned

3.1.1 Configuring database connection

The easiest way to configure a database connection is to use Oracle EZ Connect URL (//hostname:port/SERVICE_NAME). However, standard canonical TNS names can be used as well.

It is normally recommended to install the DCL Scanner on a database server to eliminate network latency. So, if DCL Scanner has read access to the set of *.ora configuration files on the database server (e.g. in $ORACLE_HOME/network/admin directory) the scanner can use them via TNS_ADMIN environment variable.

If for any reason the *.ora configuration files are not available to use, an independent TNS configuration can be created. DCL Scanner has its own set of standard Oracle configuration files located at **<DCLScanner_Install_Dir>/instantclient_11_2\network\admin** directory. These *.ora files can be used for any custom TNS configuration.

3.2 Running a Scan

To launch a scan run "**./dclscanner.sh -c filename.cfg**"

A detailed log file that provides an ongoing progress report can be found at <**installation_directory>/logs/dclscanner.log** (for example, dclscanner-1.2/logs/dclscanner.log)

A sample log file is shown in Appendix A.

After the scan has successfully finished a PDF report is generated and stored in the **<installation_directory>/reports** directory. Since reports contain sensitive data location of the reports directory cannot be changed for security reasons.

If scan is stopped due to a database error (e.g. target database became unavailable), it can be resumed later after database has become available.

A sample PDF Report file is shown in Appendix B.

4. Advanced Usage

4.1 Advanced Scanning Parameters

The advanced scanning parameters fall under the following categories:

- Setup
- DB Connections
- Schema Scanning Parameters
- Table Scanning Parameters
- Excluded Tables

Setup :

Parallelism can be controlled in two ways. The number of individual threads that will be run (**parallel_threads**) and the "in database" parallelism (**db_parallelism**).

So for example if
parallel_threads = 2
db_parallelism = 4

Two separate worker threads will be launched and the tables will be allocated to each. As they scan the tables a database parallel degree of 4 will be used by ALL the queries.

This provides the user with the flexibility to allocate more resources to the scanning operations during times where the database load is low and to allocate fewer resources when the database load is high (for example during a batch window).

If the parameter **"resume_unfinished_scans"** is set to YES then failed or aborted scans can be resumed from the point they failed at. For example, if the database was shutdown for maintenance, the scan can be resumed once it was available.

The parameter **"update_runtime_stats_intvl"** controls the frequency at which the log file is updated.

The parameter **"ignore_truncated_cards"** allows truncated cards to be ignored. For example patterns like `123456-000000-8765.`

The parameters **"scan_last_2_days_data"** allows the scan to be limited to only data that has changed in the last two days.

The parameter **"scan_clob_columns"** allows large CLOB columns to be omitted as part of an initial scan.
All the parameters in this section can be changed on the fly by terminating the scan, changing the parameters and resuming the scan.

Schema Scanning Parameters:

The Schema scanning parameter (**schema_list**) allows the user to provide a list of schemas to be scanned. It also controls the number of last partitions to be scanned for partitioned tables.

Table Scanning Parameters:

Table scanning parameters (**table_list** and **exclude_table_list**) are designed to configure various exceptions to scan database tables when Schema scanning parameters are too broad and/or not suitable.

Table scanning parameters can be used for:

- Excluding a database table from scanning (the object falls under Schema scanning parameters)
- Excluding individual columns of a table from scanning (for instance, to avoid unnecessary false positives)
- Scanning just a single database table (or a small group of tables) as oppose to scanning the whole schema via Schema scanning parameters.
- Overriding scanning rules for a database table. For example, a schema is configured with number of Range partitions to scan set to "Last 2". A particular table can be configured so that all its range partitions are scanned.

Excluded Tables:

This parameter (**exclude_table_list**) allows specific tables to be omitted from a scan. This is useful when running repeated scans or when trying to obtain a quick scan by omitting some of the largest tables in a schema. Also, individual table columns can be excluded from the scan.

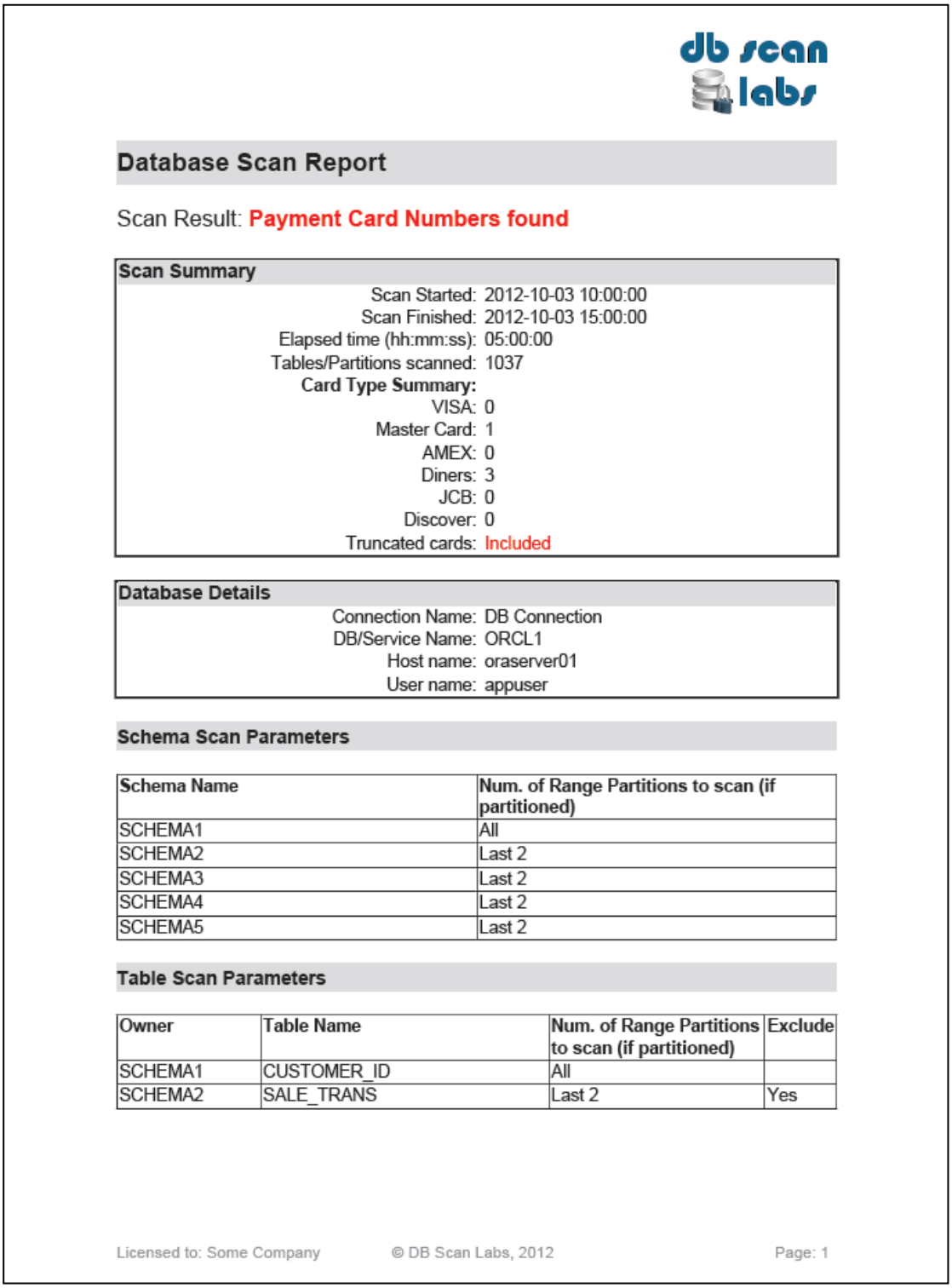
Appendix A

Below, there is a sample log file from DCL Scanner run.

```
2012-12-19 23:30:59     INFO  Database Credit Card Scanner, v1.2
2012-12-19 23:30:59     INFO  Reading config file: orile.cfg
2012-12-19 23:31:02     INFO  Local database opened OK.
2012-12-19 23:31:03     INFO  Checking DB connection details
(usr1@//dbhost01:1521/ORADB.WORLD):
2012-12-19 23:31:04     INFO        Connected OK.
2012-12-19 23:31:04     INFO  License check OK. Days left: 3
2012-12-19 23:31:04     INFO  Loading scanning parameters:
2012-12-19 23:31:04     INFO        Schema parameter: Schema: USR1;
Object type: Tables
2012-12-19 23:31:05     INFO        Table  parameter: USR1.PKT_SML;
Exclude: Yes
2012-12-19 23:31:05     INFO        Table  parameter: USR1.EVENT_LOG;
Exclude: Yes
2012-12-19 23:31:05     INFO  Starting Scanning Run...
2012-12-19 23:31:06     INFO  Scanning configuration:
2012-12-19 23:31:06     INFO        Parallel Threads: 2
2012-12-19 23:31:06     INFO        DB Parallelism: 1
2012-12-19 23:31:06     INFO        Resume Unfinished Scans: Yes
2012-12-19 23:31:06     INFO        Ignore Truncated Cards: No
2012-12-19 23:31:06     INFO        Scan CLOB Columns: No
2012-12-19 23:31:06     INFO        Scan Last 2 Days Data only: No
2012-12-19 23:31:06     INFO  Created parallel slaves: 2
2012-12-19 23:31:06     INFO        SLAVE_1: Got object: USR1.TABLE1
2012-12-19 23:31:06     INFO        SLAVE_2: Got object: USR1.TABLE2
2012-12-19 23:31:06     INFO        SLAVE_2: Got object: USR1.TABLE3
2012-12-19 23:31:06     INFO        SLAVE_1: Got object:
USR1.another_table
2012-12-19 23:31:06     INFO        SLAVE_1: completed.
2012-12-19 23:31:07     INFO        SLAVE_2: completed.
2012-12-19 23:31:14     INFO  PDF Report has been successfully
generated: /home/user1/dclscanner-
1.2/reports/dcls_ORADB_scan_20121219_233109.pdf
2012-12-19 23:31:14     INFO  Scanning Run has been closed with
status = C/Cards found
2012-12-19 23:31:14     INFO  Scanning run statistics:
2012-12-19 23:31:14     INFO        Total tables/partitions checked:
4
2012-12-19 23:31:14     INFO        Errors: 0
2012-12-19 23:31:14     INFO        Cards found:
2012-12-19 23:31:14     INFO              VISA: 770
2012-12-19 23:31:14     INFO              Master Card: 384
2012-12-19 23:31:14     INFO              AMEX: 384
2012-12-19 23:31:14     INFO              Diners: 0
2012-12-19 23:31:14     INFO              JCB: 0
2012-12-19 23:31:14     INFO              Discover: 0
2012-12-19 23:31:14     INFO        Elapsed time : 00:00:04
2012-12-19 23:31:14     INFO  Scanning run completed.
2012-12-19 23:31:16     INFO  Local database shut down.
```

Appendix B

Below there are screenshots showing sample PDF report produced by DCLScanner.



db scan labs

Database Scan Report

Scan Result: **Payment Card Numbers found**

Scan Summary

Scan Started:	2012-10-03 10:00:00
Scan Finished:	2012-10-03 15:00:00
Elapsed time (hh:mm:ss):	05:00:00
Tables/Partitions scanned:	1037
Card Type Summary:	
VISA:	0
Master Card:	1
AMEX:	0
Diners:	3
JCB:	0
Discover:	0
Truncated cards:	Included

Database Details

Connection Name:	DB Connection
DB/Service Name:	ORCL1
Host name:	oraserver01
User name:	appuser

Schema Scan Parameters

Schema Name	Num. of Range Partitions to scan (if partitioned)
SCHEMA1	All
SCHEMA2	Last 2
SCHEMA3	Last 2
SCHEMA4	Last 2
SCHEMA5	Last 2

Table Scan Parameters

Owner	Table Name	Num. of Range Partitions to scan (if partitioned)	Exclude
SCHEMA1	CUSTOMER_ID	All	
SCHEMA2	SALE_TRANS	Last 2	Yes

Licensed to: Some Company © DB Scan Labs, 2012 Page: 1

Tables with Card Numbers

SCHEMA1.CUSTOMER_ID partition (P_1)
SCHEMA1.SALE_SAMPLE

Card Number Details

Table: SCHEMA1.CUSTOMER_ID partition (P_1); Cards: 2

Column Name	Number	Card Type	ROWID
CARD_ID	528812XXXXXX3251	MASTER_CARD	AAAV0BAAbAAIYaiAAJ
STORED_CARD_NUMBER	301047XXXX4622	DINERS	AAAWyTAASAAEe5hAA+

Table: SCHEMA1.SALE_SAMPLE; Cards: 2

Column Name	Number	Card Type	ROWID
CARD_NUMBER	382031XXXX3456	DINERS	AAAWyUAASAAEclMAAY
CARD_NUMBER	382030XXXX3801	DINERS	AAAWyUAASAAI3YFAAe

Please note:

1. Clear card numbers are shown in the masked form.
2. Only the following datatypes have been scanned: CHAR, VARCHAR2, CLOB, NCHAR, NVARCHAR2, NUMBER.

Appendix C

Below there is the full list of DCLScanner configuration parameters along with comments and their syntax.

```
#
# This is a configuration file for the Database Credit Card Command Line Scanner
(DCLScanner)
#
# This file defines only one database connection and scanning parameters
# associated with it.
#
# Only one instance of DCLScanner can be active at any moment in time.
#
# File format    : parameter=value
# Comment symbol : #
# List delimiter : ;
# Multiline parameter values are NOT supported.
#
#
# ===[ Setup ]================================================================
#
# This section defines general parameters to run DCLScanner.
#
#

# parallel_threads - the number of parallel threads that are used for scanning
#

parallel_threads = 1

# resume_unfinished_scans = YES|NO
#                            YES - if unfinished (interrupted) scan has been
#                                  found, it will be resumed from the point
#                                  where it stopped.
#                            NO  - The new scanning run will be started from the
#                                  beginning.
#

resume_unfinished_scans = yes

# update_runtime_stats_intvl - the number of minutes after which the current
#                              scanner runtime statistics is displayed.

update_runtime_stats_intvl = 2

# ignore_truncated_cards = YES|NO
#                          YES - means card numbers like 123456-000000-8765 will
#                                be skipped.
#                          NO  - all the cards numbers are reported.
#

ignore_truncated_cards = no

# db_parallelism = <empty_value> | 1 | 2..N
#                          <empty_value> - means that default table PARALLEL degree
will be used
#                                          (defined during table creation)
#                               1       - PARALLEL options will be disabled for ALL
the tables
#                                          to be scanned.
#                            2..N       - Specified parallel degree will be forced
for EACH table
#                                          to be scanned.

db_parallelism = 1

# scan_last_2_days_data: This parameter allows to avoid scanning very large
# tables and scan ONLY the data that has been created/modified during the
# last 2 days. Note: database must have been running for at least 2 days.
#
```

```
# scan_last_2_days_data = YES|NO
#                         YES - Only the data that has been created/changed
#                               during the last 2 days will be scanned for
#                               the credit card numbers.
#                         NO  - Table default PARALLEL degree will be used if
#                               defined.
#

scan_last_2_days_data = no

#
# scan_clob_columns = YES|NO
#                     YES - scan CLOB columns for credit card data.
#                     NO  - ignore CLOB columns.
#

scan_clob_columns = no

# ===[ DB connection ]=================================================

# tns_name parameter can be either a standard TNS name from tnsnames.ora
# file (located in the <DCLS_DIRECTORY>/instantclient_11_2/network/admin
# directory or an EZConnect URL: //hostname:portNo/SERVICE_NAME

tns_name =
db_user  =

# When database password gets changed, DCLScanner detects the new password and
# encrypts it, so the password is stored in the clear text only until
# DCLScanner has been run for the first time after the change.

password =

# ===[ Schema scanning parameters ]====================================

#
# This parameter defines the list of schemas to be scanned for credit card
# numbers.
#
# format: schema_list=SCHEMA_NAME[[:X|ALL];SCHEMA_NAME[:X|ALL]...]
#
# where: :X|all - the number of range partitions to be scanned for the schema
#                 (if table is partitioned by range):
#
#                   - X last partitions to be scanned
#                   - ALL range partitions to be scanned.
#
#        If ommitted, default number of partitions to scan is 2.
#
# For example: schema_list=app_schema_1;app_schema_2:1;app_schema_3:all
#

schema_list =

# ===[ Table scanning parameters ]=====================================

#
# This parameter defines the list of individual tables to be scanned for
# credit card numbers.
#
# format: table_list=SCHEMA.TABLE_NAME[[:X|ALL];SCHEMA.TABLE_NAME[:X|ALL]...]
#
# where: :X|all - the number of range partitions to be scanned for the table
#                 (if table is partitioned by range):
#
#                   - X last partitions to be scanned
#                   - ALL range partitions to be scanned.
#
#        If ommitted, default number of partitions to scan is 2.
#
# For example:
table_list=app_schema_1.table1;app_schema_2.table2:1;app_schema_3.table3:all

table_list =

# ===[ Excluded tables ]===============================================
```

```
#
# This parameter defines the list of individual tables (or individual table
# columns) to be excluded from scanning. If column list is not specified, the
# whole table is excluded from scan.
#
# format:
table_list=SCHEMA.TABLE_NAME[(COLUMN_NAME1,COLUMN_NAME2...)][;SCHEMA.TABLE_NAME...]
#
# For example:
table_list=app_schema_1.table1;app_schema_2.table2;app_schema_3.table3(col1,col2)
#

exclude_table_list =
```