

An Oracle White Paper
February 2010

Oracle Advanced Security TDE Tablespace Encryption Implementation Guide for Oracle Peoplesoft Enterprise

Introduction	1
Target Audience	1
How to begin	3
Pre-Requisites.....	3
Part One: Extract information to re-build the database	3
Preparing the owner of the application.....	3
Extract tablespace names and DDL commands	4
Extract table names and DDL commands (Part I).....	6
Verify that all application tables can be re-defined online	8
Extract table names and DDL commands (Part II).....	8
Preparing Indexes for Online Table Redefinition	10
Part Two: Generate tablespace migration scripts	12
Start the Online Table Redefinition process.....	14
Create indexes in encrypted tablespaces	15
Copy and automatically register dependent objects	17
Synchronize tables and finalize Online Table Redefinition	18
Part Three: Migrate tablespaces	20
Migrate the PSINDEX tablespace	20
Migrate the PSDEFAULT tablespace.....	20
Migrate your application tablespaces	21
Verify and attest	22
Appendix A: Using a Hardware Security Module (HSM).....	23

Introduction

TDE tablespace encryption, part of the Oracle Advanced Security Option, transparently encrypts data stored in PeopleSoft application tablespaces, using industry standard encryption algorithms. It provides strong protection from malicious access to database files, and built-in key management.

Because existing tablespaces cannot be encrypted, it is necessary to move the application data from clear-text tablespaces to encrypted copies of the original tablespaces. This document explains the entire procedure in great detail and consists of three parts: The first part explains how to extract the information needed from the original database; the 2nd part creates a set of SQL scripts for each tablespace; these scripts contain all necessary information to re-create an exact copy of the Oracle PeopleSoft database layout in encrypted tablespaces, including tables, indexes, and all other dependent objects; in the 3rd and last part, the migration to encrypted tablespaces is performed. All steps can be done while the application is fully available, there is no downtime required. No changes to the application are necessary; your Oracle PeopleSoft application remains unchanged and fully supported.

Target Audience

This document and the complimentary SQL scripts and procedures are written for the Database Security Administrator (DSA) or DBA who's task it is to migrate an Oracle PeopleSoft Enterprise installation from clear-text tablespaces to encrypted tablespaces. This document outlines how this can be achieved without downtime, without data loss, and transparent to application users and the application itself, using the Online Table Redefinition feature of the Oracle Database Enterprise Edition.

This document applies to customers using Oracle Database 11g and Oracle PeopleTools version 8.48 and later.

How to begin

The accompanying script ('`tse_psft.sql`') examines your database and creates SQL scripts that can be run to eventually do the migration and encryption work. There will be six scripts for each tablespace that will migrate that tablespace with all its content and dependent objects to a new encrypted tablespace. When the scripts are finished, the original clear-text tablespace can be deleted.

Pre-Requisites

- Transparent Data Encryption needs to be enabled for your database. The TDE master encryption key needs to be available to the database to process encrypted data. The master key is stored in an external security module (ESM), either in the Oracle Wallet (an encrypted file outside of the database), or a Hardware Security Module (HSM).
- Please verify available temporary disk space; the amount of additional disk space needed during the migration is determined by the largest tablespace used by your application, **plus** the size of the 'PSINDEX_ENC' tablespace.
- Understanding Online Table Redefinition, documented here:
http://download.oracle.com/docs/cd/E11882_01/server.112/e10595/tables007.htm
- Please see Appendix A if you want to use a Hardware Security Module (HSM)

Part One: Extract information to re-build the database

Preparing the owner of the application

After expanding the compressed file that also contained this document, start '`tse_psft.sql`' with:

```
SYSTEM> @tse_psft.sql
```

At first, it will prompt you for the Oracle database user who owns the application; the default name is 'SYSADM'; this name is also used in this document. The second and final question to answer is the encryption algorithm to be used for the encrypted tablespaces; please choose from 'AES256', 'AES192', 'AES128', or '3DES168'.

The first part of the script grants necessary privileges for Online Table Redefinition to this user:

```
$ sqlplus sys as sysdba
SYS> grant execute on DBMS_REDEFINITION to SYSADM;
SYS> grant CREATE ANY TABLE, ALTER ANY TABLE, DROP ANY TABLE, LOCK ANY
```

```
TABLE, SELECT ANY TABLE, CREATE ANY TRIGGER, CREATE ANY INDEX to SYSADM;
SYS> connect SYSADM
```

Extract tablespace names and DDL commands

In the next section, a table, that will store the information necessary to assemble the SQL scripts for the new (encrypted) tablespaces, will be created:

```
SYSADM> create table      otr_log (
tbs_name          varchar2(32)
, tbs_ddl         clob
, enc_tbs_name    varchar2(32)
, enc_tbs_ddl     clob);
```

In the next step, the columns ‘tbs_name’ and ‘enc_tbs_name’ will be populated with information gathered from the data dictionary:

```
SYSADM> insert into otr_log (tbs_name, enc_tbs_name) select distinct
        tablespace_name, tablespace_name||'_ENC' from user_tables;
```

This query finds the names of all tablespaces owned by the application owner, and at the same time makes sure that the database default tablespaces ‘SYSTEM’, ‘SYSAUX’, as well as temporary and undo tablespaces are not included, since they cannot be encrypted.

Tablespaces that do not contain tables are not found with this query; for example the tablespace ‘PSINDEX’ does not appear in ‘user_tables’ and needs to be added manually:

```
SYSADM> insert into otr_log (tbs_name, enc_tbs_name)
        values ('PSINDEX', 'PSINDEX_ENC');
```

Some tablespaces are not used at all (do not contain any tables) and don’t appear in ‘user_tables’. If these should be encrypted regardless, add them manually to the ‘otr_log’ table.

In order to successfully extract the DDL commands, which were used to initially create the tablespaces, the following SQL*Plus settings need to be modified:

```
SYSADM> set long 100000;
SYSADM> set heading off;
SYSADM> set feedback off;
SYSADM> set echo off;
SYSADM> set pages 100;
SYSADM> set trimspool on;
SYSADM> set linesize 2500;
```

The next command appends a ‘;’ to each extracted DDL command:

```
SYSADM> exec dbms_metadata.set_transform_param
(dbms_metadata.session_transform,'SQLTERMINATOR',true);
```

Format the CLOB column:

```
SYSADM> column tbs_ddl format a2500 word_wrapped;
```

Populate the ‘tbs_ddl’ column:

```
SYSADM> update otr_log set tbs_ddl =
dbms_metadata.get_ddl('TABLESPACE',tbs_name);
```

The original data is now stored in ‘tbs_name’ and ‘tbs_ddl’:

```
oracle@dbsec-psft:~$ DPLARGE
CREATE TABLESPACE "DPLARGE" DATAFILE
'/opt/oracle/oradata/psft/dplarge.dbf' SIZE 9437184
AUTOEXTEND ON NEXT 5242880 MAXSIZE 32767M
LOGGING ONLINE PERMANENT BLOCKSIZE 8192
EXTENT MANAGEMENT LOCAL AUTOALLOCATE DEFAULT NOCOMPRESS SEGMENT SPACE MANAGEMENT AUTO;

EOBFAPP
CREATE TABLESPACE "EOBFAPP" DATAFILE
'/opt/oracle/oradata/psft/eobfapp.dbf' SIZE 3145728
AUTOEXTEND ON NEXT 5242880 MAXSIZE 32767M
LOGGING ONLINE PERMANENT BLOCKSIZE 8192
EXTENT MANAGEMENT LOCAL AUTOALLOCATE DEFAULT NOCOMPRESS SEGMENT SPACE MANAGEMENT AUTO;
```

Figure 1: Original DDL for clear-text tablespaces

In order to create encrypted counterparts of the clear text tablespaces, their DDL needs to be modified and stored in ‘enc_tbs_ddl’.

Change the tablespace name:

```
SYSADM> update otr_log set enc_tbs_ddl =
replace (tbs_ddl,'" DATAFILE','_ENC" DATAFILE');
```

Change the name of the datafile(s):

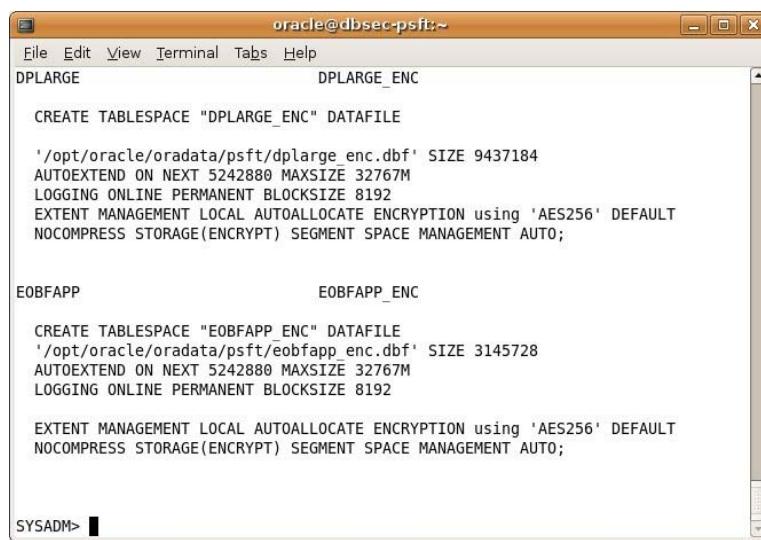
```
SYSADM> update otr_log set enc_tbs_ddl =
replace (enc_tbs_ddl,'.dbf','_enc.dbf');
```

Add the syntax for encrypted tablespaces, including the encryption algorithm selected at the beginning of this script:

```
SYSADM> update otr_log set enc_tbs_ddl =
replace (enc_tbs_ddl,'DEFAULT NOCOMPRESS ','ENCRYPTION using
'&alg' DEFAULT'||CHR(10)||' NOCOMPRESS STORAGE(ENCRYPT)');
```

Inserting a ‘new-line’ character (CHR(10)) is necessary to avoid a line break within the word ‘NOCOMPRESS’ later when the information is exported to a SQL script.

The information for the old and new tablespaces is complete:



The screenshot shows a terminal window titled "oracle@dbsec-psft:~". The window contains two sets of DDL commands. The first set, under the heading "DPLARGE", creates a tablespace named "DPLARGE_ENC" with a datafile located at "/opt/oracle/oradata/psft/dplarge_enc.dbf" of size 9437184, autoextend on next 5242880, maxsize 32767M, logging online permanent, blocksize 8192, extent management local autoallocate encryption using 'AES256' default, and nocompress storage(encrypt) segment space management auto. The second set, under the heading "EOBFAPP", creates a tablespace named "EOBFAPP_ENC" with a datafile located at "/opt/oracle/oradata/psft/eobfapp_enc.dbf" of size 3145728, autoextend on next 5242880, maxsize 32767M, logging online permanent, blocksize 8192, extent management local autoallocate encryption using 'AES256' default, and nocompress storage(encrypt) segment space management auto. The prompt "SYSADM>" is visible at the bottom left.

```

CREATE TABLESPACE "DPLARGE_ENC" DATAFILE
  '/opt/oracle/oradata/psft/dplarge_enc.dbf' SIZE 9437184
  AUTOEXTEND ON NEXT 5242880 MAXSIZE 32767M
  LOGGING ONLINE PERMANENT BLOCKSIZE 8192
  EXTENT MANAGEMENT LOCAL AUTOALLOCATE ENCRYPTION using 'AES256' DEFAULT
  NOCOMPRESS STORAGE(ENCRYPT) SEGMENT SPACE MANAGEMENT AUTO;

CREATE TABLESPACE "EOBFAPP_ENC" DATAFILE
  '/opt/oracle/oradata/psft/eobfapp_enc.dbf' SIZE 3145728
  AUTOEXTEND ON NEXT 5242880 MAXSIZE 32767M
  LOGGING ONLINE PERMANENT BLOCKSIZE 8192
  EXTENT MANAGEMENT LOCAL AUTOALLOCATE ENCRYPTION using 'AES256' DEFAULT
  NOCOMPRESS STORAGE(ENCRYPT) SEGMENT SPACE MANAGEMENT AUTO;

```

Figure 2: Modified DDL for encrypted tablespaces

Extract table names and DDL commands (Part I)

For all application tables, the DDL needs to be extracted, stored, and modified as well; this information is stored in ‘tables_in_tbs’:

```

SYSADM> create table      tables_in_tbs (
  table_name          varchar2(64)
, tbs_name            varchar2(64)
, enc_tbs_name        varchar2(64)
, can_redef           varchar2(3)
, table_ddl           clob
, enc_table_ddl       clob
, has_lob             varchar2(3)
, owner               varchar2(64));

SYSADM> create table      index_log(
  index_name          varchar2(64)
, int_index_name      varchar2(64)
, index_ddl           clob
, int_index_ddl       clob
, table_name          varchar2(64))

```

```
, int_table_name          varchar2(64)
, owner                  varchar2(64));
```

The following steps are similar to the ones before:

```
SYSADM> set long 100000;
SYSADM> set heading off;
SYSADM> set feedback off;
SYSADM> set echo off;
SYSADM> set pages 100;
SYSADM> set trimspool on;
SYSADM> set linesize 2500;

SYSADM> exec dbms_metadata.set_transform_param
        (dbms_metadata.session_transform,'SQLTERMINATOR',true);

SYSADM> column x format a2500 word_wrapped;
```

The following command extracts the DDL for potentially tens of thousands of tables, which means it may take one or two hours to complete.

```
SYSADM> insert into tables_in_tbs
        (table_name, table_ddl, tbs_name, enc_tbs_name)
        select table_name, dbms_metadata.get_ddl('TABLE',table_name) x,
        tablespace_name, tablespace_name||'_ENC' from user_tables;

SYSADM> update table_in_tbs set owner = 'SYSADM';
```

```
oracle@dbsec-psft:~/work
File Edit View Terminal Tabs Help
SYSADM> select tbs_name, table_ddl from tables_in_tbs where table_name='PS ACH LAYOUT_TBL';
FSAPP

CREATE TABLE "SYSADM"."PS ACH LAYOUT_TBL"
(
    "ACH_LAYOUT_CD" VARCHAR2(10) NOT NULL ENABLE,
    "DESCR" VARCHAR2(30) NOT NULL ENABLE,
    "DESCRSHORT" VARCHAR2(10) NOT NULL ENABLE,
    "LAST_EFT_FILE_ID" VARCHAR2(4) NOT NULL ENABLE,
    "EFT_FILE_NBR" VARCHAR2(2) NOT NULL ENABLE,
    "OUTPUT_FILENAME" VARCHAR2(30) NOT NULL ENABLE,
    "FILE_EXTENSION" VARCHAR2(4) NOT NULL ENABLE,
    "FEDRL_EFT_ACH_IND" VARCHAR2(1) NOT NULL ENABLE
) SEGMENT CREATION IMMEDIATE
PCTFREE 10 PCTUSED 80 INITTRANS 1 MAXTRANS 255 NOCOMPRESS LOGGING
STORAGE(INITIAL 40960 NEXT 106496 MINEXTENTS 1 MAXEXTENTS 2147483645
PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL DEFAULT FLASH_CACHE DEF
FAULT CELL_FLASH_CACHE DEFAULT)
TABLESPACE "FSAPP" ;
```

Figure 3: Original DDL for application tables

Verify that all application tables can be re-defined online

Some tables cannot be online redefined. The procedure **check_redef** (source code in ‘01-check_redef_procedure.sql’, accompanying this document in the downloaded zip file) will test all of them and store the result in ‘tables_in_tbs.can_redef’:

```
SYSADM> create or replace procedure check_redef AS
  2  CURSOR tables_c IS
  3    SELECT table_name from tables_in_tbs FOR UPDATE;
  4  BEGIN
  5    FOR table_rec in tables_c LOOP
  6      BEGIN
  7        DBMS_REDEFINITION.CAN_REDEF_TABLE('SYSADM',table_rec.table_name,
  8          dbms_redefinition.cons_use_rowid);
  9        UPDATE tables_in_tbs SET can_redef ='YES'
 10          WHERE CURRENT OF tables_c;
 11      EXCEPTION
 12        WHEN OTHERS THEN
 13        UPDATE tables_in_tbs set can_redef ='NO'
 14          WHERE CURRENT OF tables_c;
 15      END;
 16    END LOOP;
 17  end check_redef;
 18  /
```

Next, execute the procedure; it will run for approximately 5 minutes, depending on the number of tables to examine:

```
SYSADM> execute check_redef;
```

Verify that no table tested negative:

```
SYSADM> select tbs_name, table_name from
  tables_in_tbs where can_redef = 'NO' order by tbs_name;
```

Extract table names and DDL commands (Part II)

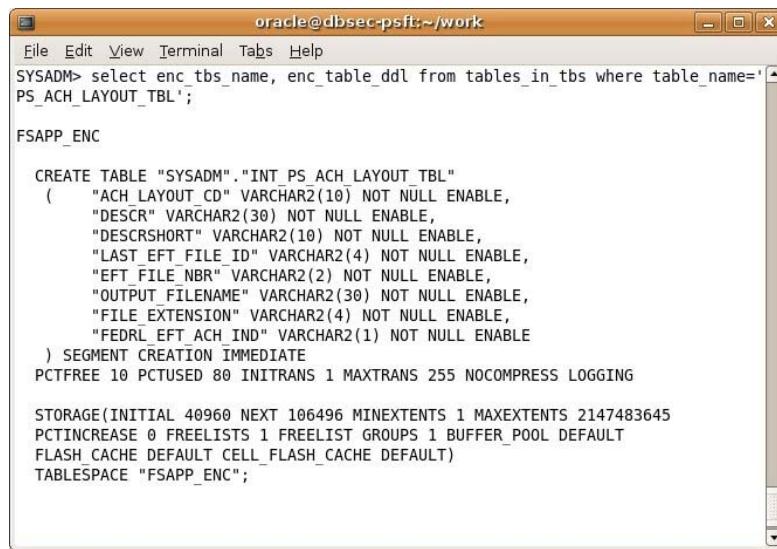
The names of the tablespaces in ‘tables_in_tbs.enc_tbs_name’ already have been appended with ‘_ENC’, but the table-DDL still contains the original names for both tablespaces and tables. The target tablespace names in the table-DDL need to end with ‘_ENC’, and the names of the new (interim) tables in the encrypted tablespaces begin with ‘INT_’. Change them with:

```
SYSADM> update tables_in_tbs set enc_table_ddl =
  replace (table_ddl,'" ;','_ENC"');
```

```
SYSADM> update tables_in_tbs set enc_table_ddl =
    replace (enc_table_ddl,'SYSADM'.'','SYSADM'."INT_');

SYSADM> update tables_in_tbs set enc_table_ddl =
    replace (enc_table_ddl,'DEFAULT FLASH',
    'DEFAULT'||CHR(10)||'FLASH');
```

Inserting ‘CHR(10)’ avoids a line break in the word ‘DEFAULT’ later when the information is exported to a SQL script.



The screenshot shows an Oracle terminal window titled "oracle@dbsec-psft:~/work". The command entered is:

```
SYSADM> select enc_tbs_name, enc_table_ddl from tables_in_tbs where table_name='PS_ACH_LAYOUT_TBL';

FSAPP_ENC

CREATE TABLE "SYSADM"."INT_PS_ACH_LAYOUT_TBL"
( "ACH_LAYOUT_CD" VARCHAR2(10) NOT NULL ENABLE,
  "DESCR" VARCHAR2(30) NOT NULL ENABLE,
  "DESCRSHORT" VARCHAR2(10) NOT NULL ENABLE,
  "LAST_EFT_FILE_ID" VARCHAR2(4) NOT NULL ENABLE,
  "EFT_FILE_NBR" VARCHAR2(2) NOT NULL ENABLE,
  "OUTPUT_FILENAME" VARCHAR2(30) NOT NULL ENABLE,
  "FILE_EXTENSION" VARCHAR2(4) NOT NULL ENABLE,
  "FEDRL_EFT_ACH_IND" VARCHAR2(1) NOT NULL ENABLE
) SEGMENT CREATION IMMEDIATE
PCTFREE 10 PCTUSED 80 INITTRANS 1 MAXTRANS 255 NOCOMPRESS LOGGING
STORAGE(INITIAL 40960 NEXT 106496 MINEXTENTS 1 MAXEXTENTS 2147483645
PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL DEFAULT
FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
TABLESPACE "FSAPP_ENC";
```

Figure 4: Modified DDL to create interim tables in encrypted tablespaces

In tables with LOB or CLOB columns, the command to append the tablespace name with ‘_ENC’ does not find the original tablespace name; hence, the names are not changed. In these cases, the word ‘TABLESPACE’ appears more than once, and the tables that include LOB or CLOB columns can be found with:

```
SYSADM> update tables_in_tbs set has_lob = 'YES' where
    dbms_lob.instr(enc_table_ddl,'TABLESPACE',1,2) > 0;

SYSADM> create bitmap index has_lob_bidx on tables_in_tbs(has_lob)
    tablespace "PSINDEX";

SYSADM> create unique index table_name_idx on tables_in_tbs(table_name)
    tablespace "PSINDEX";
```

The procedure ‘replace_CT_with_ENC’ (source code in ‘02-replace_ENC.sql’) reads the names of tables with LOB or CLOB columns, and replaces the original tablespace names with the name of the new encrypted tablespace:

```
SQL> create or replace procedure replace_CT_with_ENC as
  2  cursor tbs_c is
  3  select distinct tbs_name from tables_in_tbs where has_lob='YES';
  4  cursor tab_c(p_tbs varchar2) is
  5  select table_name from tables_in_tbs where
  6  tbs_name = p_tbs and has_lob = 'YES';
  7  BEGIN
  8    FOR tbs_rec in tbs_c LOOP
  9      BEGIN
 10        FOR tab_rec in tab_c(tbs_rec.tbs_name) LOOP
 11          BEGIN
 12            update tables_in_tbs set enc_table_ddl =
 13              replace(enc_table_ddl,'TABLESPACE "'||tbs_rec.tbs_name||'"',
 14              'TABLESPACE "'||tbs_rec.tbs_name||'_ENC"') where
 15              table_name = tab_rec.table_name;
 16          END;
 17        END LOOP;
 18      END;
 19    END LOOP;
 20  END replace_CT_with_ENC;
 21  /
```

Preparing Indexes for Online Table Redefinition

Online Table Redefinition is designed to automatically move and register indexes with their underlying base tables. In Oracle PeopleSoft, indexes are stored in a separated tablespace ‘PSINDEX’, not in the same tablespace as their base tables. The automatic movement of dependent objects has to exclude indexes; they need to be migrated manually, instead:

```
SYSADM> set long 1000000;
SYSADM> set heading off;
SYSADM> set feedback off;
SYSADM> set echo off;
SYSADM> set pages 100;
SYSADM> set trimspool on;
SYSADM> set linesize 2500;

SYSADM> execute dbms_metadata.set_transform_param
          (dbms_metadata.session_transform,'SQLTERMINATOR',true);

SYSADM> column index_ddl format a2500 word_wrapped;
```

```
SYSADM> insert into index_log (index_name, int_index_name, index_ddl,
  table_name, int_table_name) select
  index_name, 'INT_'||index_name,
  dbms_metadata.get_ddl('INDEX', index_name, 'SYSADM'),
  table_name, 'INT_'||table_name from user_indexes where
  tablespace_name = 'PSINDEX';

SYSADM> create unique index index_name_idx on index_log(index_name)
  tablespace "PSINDEX";

SYSADM> insert into index_log (index_name, int_index_name, index_ddl,
  table_name, int_table_name) values
  ('INDEX_NAME_IDX', 'INT_INDEX_NAME_IDX',
  dbms_metadata.get_ddl('INDEX', 'INDEX_NAME_IDX', 'SYSADM'),
  'INDEX_LOG', 'INT_INDEX_LOG');

SYSADM> update index_log i set i.tbs_name =
  (select t.tbs_name from tables_in_tbs t
  where t.table_name = i.table_name);

SYSADM> update index_log set enc_tbs_name = tbs_name||'_ENC';

SYSADM> update index_log set owner ='SYSADM';
```

Next, the DDL for the indexes needs to be changed so they are re-created in new encrypted ‘PSINDEX’ tablespace:

Change index names and table names to begin with ‘INT_’:

```
SYSADM> update index_log set int_index_ddl =
  replace (index_ddl,'"SYSADM".'', '"SYSADM"."INT_');
```

Rename tablespace names to end with ‘_ENC’:

```
SYSADM> update index_log set int_index_ddl =
  replace (int_index_ddl,'' ;','_ENC';');
```

Insert ‘new line’ before interim table name to avoid line breaks within interim table name:

```
SYSADM> update index_log set int_index_ddl =
  replace (int_index_ddl, 'ON "SYSADM"' , 'ON'||CHR(10)||'"SYSADM"');
```

Insert ‘new line’ after each column name to avoid line breaks in column names:

```
SYSADM> update index_log set int_index_ddl =
  replace (int_index_ddl,'", "','", '|CHR(10)||"');
```

Avoid line breaks within the word ‘DEFAULT’:

```
SYSADM> update index_log set int_index_ddl =
    replace (int_index_ddl,
    'DEFAULT FLASH', 'DEFAULT' || CHR(10) || ' FLASH');
```

Some indexes are not stored in the ‘PSINDEX’ tablespace, but in the same tablespace as their base tables. These are automatically generated indexes. Due to the fact that the tables in the encrypted tablespaces are created with the exact same DDL that was used for the original tables, these indexes will be automatically re-generated in the encrypted tablespaces.

Part Two: Generate tablespace migration scripts

The following procedures will generate scripts that will allow migrating one original tablespace to an encrypted tablespace at a time, reducing the amount of necessary free disk space to the size of the largest tablespace, plus the ‘PSINDEX’ tablespace.

In order to be able to create and write to files on the Operating System, a DIRECTORY needs to be defined, with WRITE privileges for the ‘SYSADM’ user:

```
SYSADM> conn sys as sysdba
SYS> create DIRECTORY work as '/home/oracle/work';
SYS> grant write on DIRECTORY work to SYSADM;
SYS> conn SYSADM
```

A procedure ‘create_tbs_scripts’ (source code: ‘03-create_enc_tbs_procedure.sql’) creates a master SQL script (‘work/create_enc_tbs_script.sql’) that contains all the commands to create one script per tablespace; these individual scripts will create a new encrypted tablespace and all tables in that tablespace.

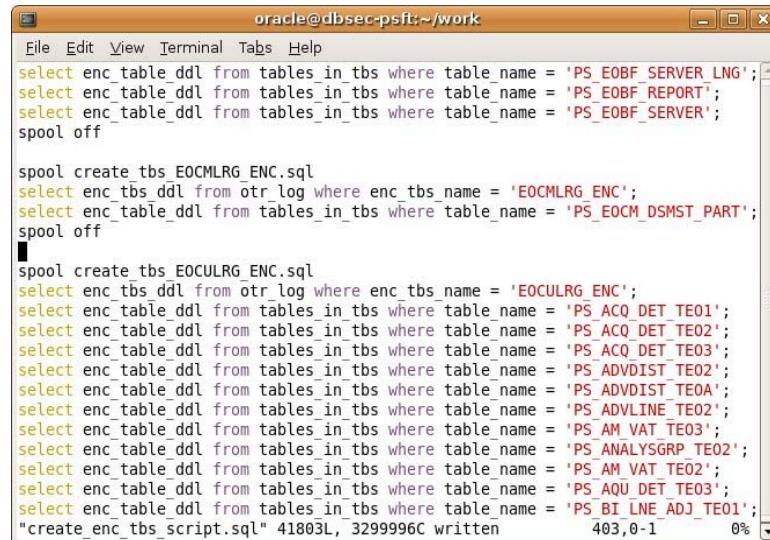
```
SYSADM> create or replace procedure create_tbs_scripts as
 2  cursor tbs_c is
 3  select enc_tbs_name from otr_log;
 4  cursor tab_c(p_tbs varchar2) is
 5    select table_name from tables_in_tbs where enc_tbs_name = p_tbs;
 6  fhnd utl_file.file_type;
 7  BEGIN
 8    fhnd := utl_file.fopen ('WORK', 'create_enc_tbs_script.sql', 'a');
 9    utl_file.put_line(fhnd, 'set timing off;', true);
10    utl_file.put_line(fhnd, 'set long 100000;', true);
11    utl_file.put_line(fhnd, 'set heading off;', true);
12    utl_file.put_line(fhnd, 'set feedback off;', true);
13    utl_file.put_line(fhnd, 'set echo off;', true);
14    utl_file.put_line(fhnd, 'set pages 100;', true);
15    utl_file.put_line(fhnd, 'set trimspool on;', true);
```

```

15  utl_file.put_line(fhnd, 'set linesize 2500;', true);
16  FOR tbs_rec IN tbs_c LOOP
17    BEGIN
18      utl_file.put_line (fhnd,
19      'spool ./work/01-create_tbs_||tbs_rec.enc_tbs_name||.sql',
20      true);
21      utl_file.put_line (fhnd,
22      'select enc_tbs_ddl from otr_log where enc_tbs_name =
23      ''||tbs_rec.enc_tbs_name||''; ', true);
24      FOR tab_rec IN tab_c(tbs_rec.enc_tbs_name) LOOP
25        BEGIN
26          utl_file.put_line (fhnd,
27          'select enc_table_ddl from tables_in_tbs where table_name =
28          '''||tab_rec.table_name||'''; ', true);
29      END;
30    END LOOP;
31    utl_file.put_line (fhnd, 'spool off', true);
32  END;
33  END LOOP;
34  utl_file fclose(fhnd);
35 END create_tbs_scripts;
36 /

```

Excerpt from the resulting master script ('work/create_enc_tbs_script.sql') generated by this procedure:



The screenshot shows a terminal window titled "oracle@dbsec-psft:~/work". The window displays a series of SQL commands. It starts with several "select" statements for tables like 'PS_EOBF_SERVER_LNG', 'PS_EOBF_REPORT', and 'PS_EOBF_SERVER'. These are followed by a "spool off" command. Then, it begins a new section starting with "spool create_tbs_EOCMLRG_ENC.sql". This section contains multiple "select" statements for tables such as 'PS.ACQ_DET_TE01', 'PS.ACQ_DET_TE02', 'PS.ACQ_DET_TE03', 'PS.ADVIST_TE02', 'PS.ADVLINE_TE02', 'PS.AM_VAT_TE03', 'PS.ANALYSGRP_TE02', and 'PS.AM_VAT_TE02'. Finally, the script ends with the command "create_enc_tbs_script.sql" and some statistics: "41803L, 3299996C written 403,0-1 0%".

```

oracle@dbsec-psft:~/work
File Edit View Terminal Tabs Help
select enc_table_ddl from tables_in_tbs where table_name = 'PS_EOBF_SERVER_LNG';
select enc_table_ddl from tables_in_tbs where table_name = 'PS_EOBF_REPORT';
select enc_table_ddl from tables_in_tbs where table_name = 'PS_EOBF_SERVER';
spool off

spool create_tbs_EOCMLRG_ENC.sql
select enc_tbs_ddl from otr_log where enc_tbs_name = 'EOCMLRG_ENC';
select enc_table_ddl from tables_in_tbs where table_name = 'PS_EOCM_DSMST_PART';
spool off
spool create_tbs_EOCULRG_ENC.sql
select enc_tbs_ddl from otr_log where enc_tbs_name = 'EOCULRG_ENC';
select enc_table_ddl from tables_in_tbs where table_name = 'PS.ACQ_DET_TE01';
select enc_table_ddl from tables_in_tbs where table_name = 'PS.ACQ_DET_TE02';
select enc_table_ddl from tables_in_tbs where table_name = 'PS.ACQ_DET_TE03';
select enc_table_ddl from tables_in_tbs where table_name = 'PS.ADVIST_TE02';
select enc_table_ddl from tables_in_tbs where table_name = 'PS.ADVLINE_TE02';
select enc_table_ddl from tables_in_tbs where table_name = 'PS.AM_VAT_TE03';
select enc_table_ddl from tables_in_tbs where table_name = 'PS.ANALYSGRP_TE02';
select enc_table_ddl from tables_in_tbs where table_name = 'PS.AM_VAT_TE02';
select enc_table_ddl from tables_in_tbs where table_name = 'PS.AQU_DET_TE03';
select enc_table_ddl from tables_in_tbs where table_name = 'PS.BI_LNE_ADJ_TE01';
"create_enc_tbs_script.sql" 41803L, 3299996C written 403,0-1 0%

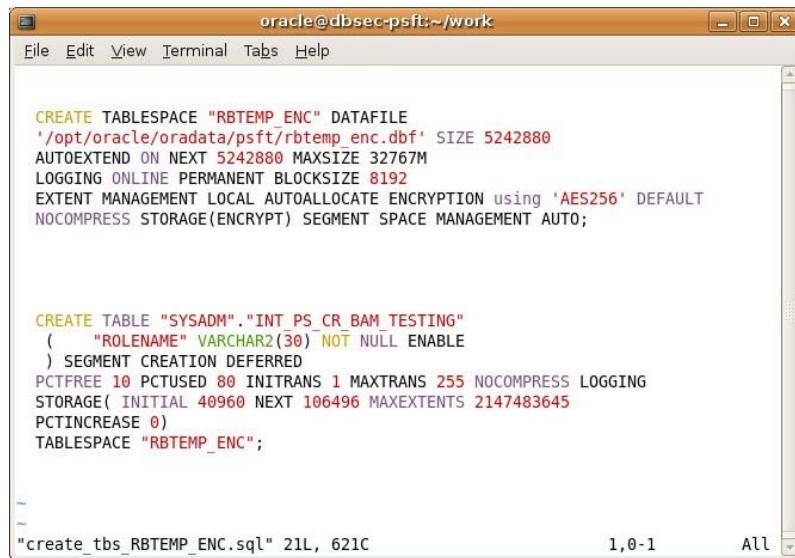
```

Figure 5: Master script which generates scripts for each new tablespace

Run this script with:

```
SYSADM> @./work/create_enc_tbs_script.sql
```

The image below shows the content of one of the generated files:



The screenshot shows a terminal window titled "oracle@dbsec-psft:~/work". The window contains the following SQL code:

```

CREATE TABLESPACE "RBTEMP_ENC" DATAFILE
'/opt/oracle/oradata/psft/rbtemp enc.dbf' SIZE 5242880
AUTOEXTEND ON NEXT 5242880 MAXSIZE 32767M
LOGGING ONLINE PERMANENT BLOCKSIZE 8192
EXTENT MANAGEMENT LOCAL AUTOALLOCATE ENCRYPTION using 'AES256' DEFAULT
NOCOMPRESS STORAGE(ENCRYPT) SEGMENT SPACE MANAGEMENT AUTO;

CREATE TABLE "SYSADM"."INT_PS_CR_BAM_TESTING"
(
    "ROLENAME" VARCHAR2(30) NOT NULL ENABLE
) SEGMENT CREATION DEFERRED
PCTFREE 10 PCTUSED 80 INITTRANS 1 MAXTRANS 255 NOCOMPRESS LOGGING
STORAGE( INITIAL 40960 NEXT 106496 MAXEXTENTS 2147483645
PCTINCREASE 0)
TABLESPACE "RBTEMP_ENC";

```

The status bar at the bottom of the terminal window indicates "create_tbs_RBTEMP_ENC.sql" has 21L, 621C lines, and the cursor is at position 1,0-1.

Figure 6: Scripts to generate one tablespace with all corresponding application tables

The scripts to generate encrypted counterparts of all application tablespaces, and all interim tables in these tablespaces, are now ready.

Start the Online Table Redefinition process

The next step is to create one script per tablespace that starts the Online Table Redefinition process for all tables in that tablespace.

The procedure 'create_start_redef_tbs_scripts' (source code in '04-create_start_redef_procedure.sql'):

```
SYSADM> create or replace procedure create_start_redef_tbs_scripts as
 2  cursor tbs_c is
 3  select enc_tbs_name from otr_log;
 4  cursor tab_c(p_tbs varchar2) is
 5  select table_name from tables_in_tbs where enc_tbs_name = p_tbs;
 6  fhnd utl_file.file_type;
 7  BEGIN
 8    FOR tbs_rec in tbs_c LOOP
 9      BEGIN
10        fhnd := utl_file.fopen
11        ('WORK', '02-start_redef_'||tbs_rec.enc_tbs_name||'.sql', 'a');
12        FOR tab_rec in tab_c(tbs_rec.enc_tbs_name) LOOP
13          BEGIN
14            utl_file.put_line (fhnd,
15            'execute dbms_redefinition.start_redef_table('''SYSADM''',
16            '''||tab_rec.table_name||'''',
17            
```

```
    '' INT_''||tab_rec.table_name||'', NULL,
    DBMS_REDEFINITION.CONS_USE_ROWID)', true);
14    END;
15    END LOOP;
16    utl_file fclose(fhnd);
17    END;
18    END LOOP;
19 END create_start_redef_tbs_scripts;
20 /
```

The screen shot below shows the content of one of the scripts generated by this procedure:

Figure 7: Script to start Online Table Redefinition of all application tables in current tablespace

Create indexes in encrypted tablespaces

All indexes are stored in ‘PSINDEX’, not in the tablespaces that hold their base tables. Right after starting the Online Table Redefinition process for each tablespace, all indexes belonging to the tables in that tablespace need to be created and stored in ‘PSINDEX_ENC’.

The procedure ‘**create_index_scripts**’ (source code in ‘05-create_indexes.sql’) creates a master script that generates scripts that build the indexes on tables in the currently processed tablespace:

```
SYSADM> create or replace procedure create_index_scripts as
  2  cursor tbs_c is
  3  select enc_tbs_name from otr_log;
  4  cursor int_c(p_tbs varchar2) is
    select index_name from index_log where enc_tbs_name = p_tbs;
  5  fhnd utl_file.file_type;
  6  BEGIN
```

```
7  fhnd := utl_file.fopen('WORK','create_int_index_script.sql', 'a');
8  utl_file.put_line(fhnd, 'set timing off;', true);
9  utl_file.put_line(fhnd, 'set long 100000;', true);
10 utl_file.put_line(fhnd, 'set heading off;', true);
11 utl_file.put_line(fhnd, 'set feedback off;', true);
12 utl_file.put_line(fhnd, 'set echo off;', true);
13 utl_file.put_line(fhnd, 'set pages 100;', true);
14 utl_file.put_line(fhnd, 'set trimspool on;', true);
15 utl_file.put_line(fhnd, 'set linesize 2500;', true);
16 FOR tbs_rec in tbs_c LOOP
17 BEGIN
18     utl_file.put_line (fhnd,
19                         'spool /work/03-create_index_for_' || tbs_rec.enc_tbs_name || '.sql',
20                         true);
21     utl_file.put_line (fhnd,
22                         'select enc_tbs_ddl from otr_log where enc_tbs_name =
23                         ''PSINDEX_ENC'';', true);
24     utl_file.put_line (fhnd,
25                         'select int_index_ddl from index_log where enc_tbs_ddl =
26                         '''||tbs_rec.enc_tbs_name||'''', true)
27     utl_file.put_line(fhnd, 'spool off', true);
28 END;
29 END LOOP;
30 utl_file.fclose(fhnd);
31 END create_index_scripts;
32 /
```

Run the generated script ‘create_int_index_script.sql’ with
SYSADM> @./work/create_int_index_script.sql to generate the scripts that will re-create the indexes in the encrypted ‘PSINDEX_ENC’ tablespace.

Sometimes, a blank line interrupts the list of columns in the ‘create table’ or the ‘create index’ commands; when these SQL commands are executed by the database, they will cause an error. To remove all blank lines in all generated scripts with, exit from SQL*Plus and:

```
$ cd /home/oracle/work
$ find . -type f -exec sed -i '/^\$/d' {} \;
```

As an alternative, if ‘sed’ is not available, use ‘perl’:

```
$ find . -type f -exec perl -pi -e '/\s^$/d' {} \;
```

Change back to the directory where you unpacked the original scripts and this document and log back into ‘SQL*Plus’:

```
$ cd ..
$ sqlplus sysadm
```

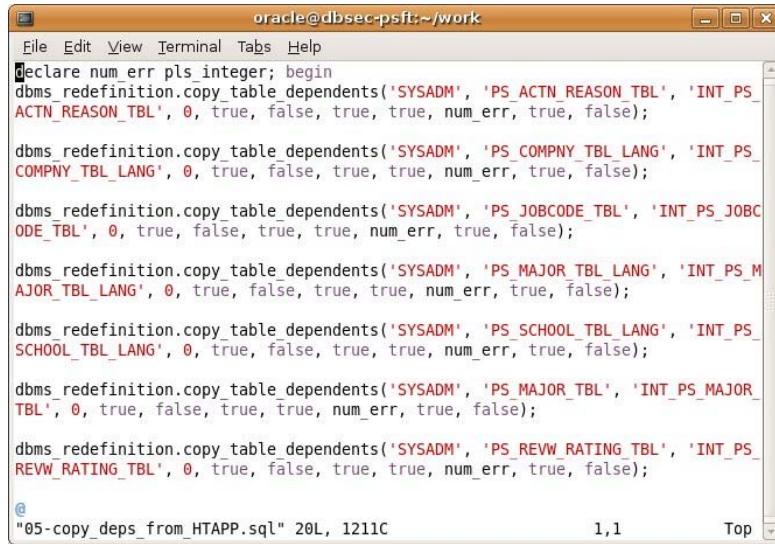
Copy and automatically register dependent objects

One of the biggest conveniences of Online Table Redefinition is the ability to automatically copy all dependent objects from the source table to the interim table, including un-registering the objects in the source table and registering them with the target table. For each table, error messages are written to the view ‘dba_redefinition_errors’.

The procedure **create_copy_deps_scripts** (source code in 06-create_copy_deps_procedure.sql) generates the scripts to copy the dependent objects of all tables in the currently processed tablespace.

```
SYSADM> create or replace procedure create_copy_deps_scripts as
 2  cursor tbs_c is
 3    select enc_tbs_name from otr_log;
 4  cursor tab_c(p_tbs varchar2) is
 5    select table_name from tables_in_tbs where enc_tbs_name = p_tbs;
 6    fhnd      utl_file.file_type;
 7    gen_code   varchar2(4000);
 8    num_err    pls_integer;
 9  BEGIN
10    FOR tbs_rec in tbs_c LOOP
11      fhnd := utl_file.fopen
12        ('WORK', '04-copy_deps_in_||tbs_rec.enc_tbs_name||.sql', 'a');
13      gen_code := 'declare num_err pls_integer; begin';
14      utl_file.put_line(fhnd, gen_code, true);
15      FOR tab_rec in tab_c(tbs_rec.enc_tbs_name) LOOP
16        gen_code := 'dbms_redefinition.copy_table_dependents
17          (''SYSADM'', ''||tab_rec.table_name||'',
18           ''INT_''||tab_rec.table_name||'', 0, true, false, true, true,
19           num_err, true, false)';
20        utl_file.put_line(fhnd, gen_code, true);
21        utl_file.put_line(fhnd, '', true);
22      END LOOP;
23      utl_file.fclose(fhnd);
24    END LOOP;
25  END create_copy_deps_scripts;
26  /
```

The screen shot below shows the content of one of the scripts generated by this procedure:



```

oracle@dbsec-psft:~/work
File Edit View Terminal Tabs Help
declare num_err pls_integer; begin
  dbms_redefinition.copy_table_dependents('SYSADM', 'PS_ACTN_REASON_TBL', 'INT_PS_
ACTN_REASON_TBL', 0, true, false, true, true, num_err, true, false);

  dbms_redefinition.copy_table_dependents('SYSADM', 'PS_COMPNY_TBL_LANG', 'INT_PS_
COMPNY_TBL_LANG', 0, true, false, true, true, num_err, true, false);

  dbms_redefinition.copy_table_dependents('SYSADM', 'PS_JOBCODE_TBL', 'INT_PS_JOB_
CODE_TBL', 0, true, false, true, true, num_err, true, false);

  dbms_redefinition.copy_table_dependents('SYSADM', 'PS_MAJOR_TBL_LANG', 'INT_PS_M_
AJOR_TBL_LANG', 0, true, false, true, true, num_err, true, false);

  dbms_redefinition.copy_table_dependents('SYSADM', 'PS_SCHOOL_TBL_LANG', 'INT_PS_
SCHOOL_TBL_LANG', 0, true, false, true, true, num_err, true, false);

  dbms_redefinition.copy_table_dependents('SYSADM', 'PS_MAJOR_TBL', 'INT_PS_MAJOR_
_TBL', 0, true, false, true, true, num_err, true, false);

  dbms_redefinition.copy_table_dependents('SYSADM', 'PS_REVW_RATING_TBL', 'INT_PS_
REVW_RATING_TBL', 0, true, false, true, true, num_err, true, false);
@  "05-copy_deps_from_HTAPP.sql" 20L, 1211C          1,1      Top

```

Figure 8: Copy dependent objects of all tables in current tablespace

Synchronize tables and finalize Online Table Redefinition

During Online Table Redefinition, the source tables remain fully accessible for all DML (insert, update, delete). That means that a row in a source table can change after it has been copied over to the interim table in the encrypted tablespace. Synchronizing the data in the interim table with the changes to the source table will shorten the time the tables need to be locked during the following ‘*finish redefinition*’ process.

The procedure ‘`create_sync_finish_scripts`’ (source code in `07-create_sync_finish_procedure.sql`) creates one script for each tablespace that will synchronize an interim table with the changes to the original table, and finish the redefinition process in the immediately following step. The advantage of this approach is that the brief moment when the tables are locked while the table names are swapped (and the interim tables in the encrypted tablespaces becomes your current tables) is reduced to the minimum:

```

SYSADM> create or replace procedure create_sync_finish_scripts as
  2  cursor tbs_c is
  3  select enc_tbs_name from otr_log;
  4  cursor tab_c(p_tbs varchar2) is
  5  select table_name from tables_in_tbs where enc_tbs_name=p_tbs;
  6  fhnd utl_file.file_type;
  7  BEGIN
  8  FOR tbs_rec in tbs_c LOOP
  9    BEGIN
 10      fhnd := utl_file.fopen
 11      ('WORK','05-sync_finish_' || tbs_rec.enc_tbs_name || '.sql','a');
 12      FOR tab_rec in tab_c(tbs_rec.enc_tbs_name) LOOP

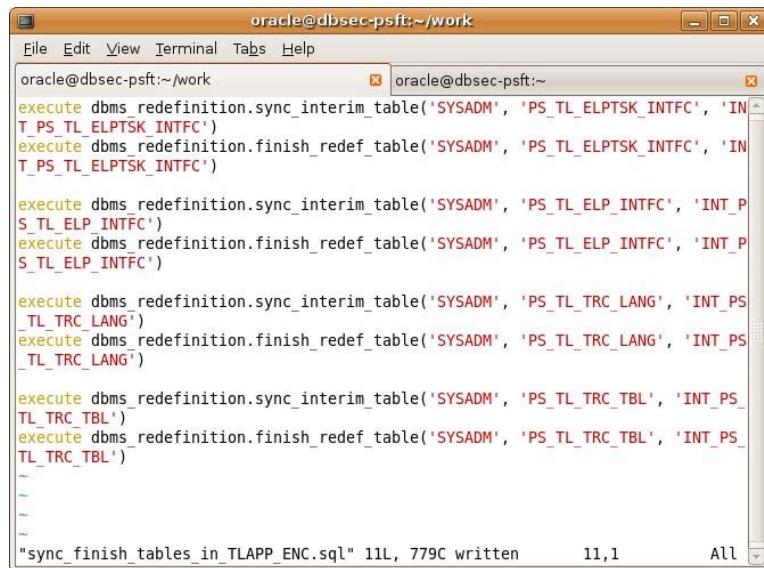
```

```

12      BEGIN
13          utl_file.put_line (fhnd,
14          'execute dbms_redefinition.sync_interim_table
15          (''SYSADM'', ''||tab_rec.table_name||'',
16          ''INT_''||tab_rec.table_name||''), true);
17          utl_file.put_line (fhnd,
18          'execute dbms_redefinition.finish_redef_table
19          (''SYSADM'', ''||tab_rec.table_name||'',
20          ''INT_''||tab_rec.table_name||''), true);
21      END;
22  END LOOP;
23  utl_file.fclose(fhnd);
24
25  END;
26
27  END LOOP;
28
29 END create_sync_finish_scripts;
30 /

```

This procedure creates a script for each tablespace to synchronize the tables and immediately ‘finish’ the Online Table Redefinition process by swapping the name of the original application tables with the name of the interim table in the encrypted tablespace:



The screenshot shows a terminal window titled "oracle@dbsec-psft:~/work". It contains several lines of Oracle PL/SQL code. The code uses the DBMS_REDEFINITION package to synchronize tables between the 'SYSADM' tablespace and an interim tablespace named 'PS_TL_ELPTSK_INTFC'. It performs this operation for four different table names: 'PS_TL_ELP', 'PS_TL_TRA', 'PS_TL_TRC_LANG', and 'PS_TL_TRC_TBL'. For each table, it first executes 'sync_interim_table' and then 'finish_redef_table'. The code is enclosed in a loop, indicated by the 'END LOOP;' statements at lines 22 and 27.

```

oracle@dbsec-psft:~/work
File Edit View Terminal Tabs Help
oracle@dbsec-psft:~/work
execute dbms_redefinition.sync_interim_table('SYSADM', 'PS_TL_ELPTSK_INTFC', 'IN
T_PS_TL_ELPTSK_INTFC')
execute dbms_redefinition.finish_redef_table('SYSADM', 'PS_TL_ELPTSK_INTFC', 'IN
T_PS_TL_ELPTSK_INTFC')

execute dbms_redefinition.sync_interim_table('SYSADM', 'PS_TL_ELP_INTFC', 'INT_P
S_TL_ELP_INTFC')
execute dbms_redefinition.finish_redef_table('SYSADM', 'PS_TL_ELP_INTFC', 'INT_P
S_TL_ELP_INTFC')

execute dbms_redefinition.sync_interim_table('SYSADM', 'PS_TL_TRA_LANG', 'INT_P
S_TL_TRA_LANG')
execute dbms_redefinition.finish_redef_table('SYSADM', 'PS_TL_TRA_LANG', 'INT_P
S_TL_TRA_LANG')

execute dbms_redefinition.sync_interim_table('SYSADM', 'PS_TL_TRC_TBL', 'INT_P
S_TL_TRC_TBL')
execute dbms_redefinition.finish_redef_table('SYSADM', 'PS_TL_TRC_TBL', 'INT_P
S_TL_TRC_TBL')

```

Figure 9: Synchronize tables and immediately finish redefinition process

Part Three: Migrate tablespaces

Migrate the PSINDEX tablespace

The scripts that are needed for the migration are stored in the ‘WORK’ directory. The first table that needs to be migrated is ‘PSINDEX’, because the encrypted version is needed when the indexes of the first application tablespace are migrated. Note that **both** ‘PSINDEX’ and ‘PSINDEX_ENC’ need to exist until the last application tablespace is migrated. The ‘PSINDEX’ tablespace can be very large, and the syntax to build the ‘PSINDEX_ENC’ tablespace contains the same file size. Verify if it is feasible to reduce the initial file size of the ‘PSINDEX_ENC’ tablespace. The syntax allows the tablespace to expand over time as needed, until the maximum file size is reached.

Make sure the Oracle Wallet (or HSM) is open to make the master encryption key available to the database, and run:

```
SYSADM> @./work/01-create_tbs_PSINDEX_ENC.sql
```

Migrate the PSDEFAULT tablespace

The tablespace ‘PSDEFAULT’ contains the tables that are used for this migration project, and a table ‘PSDBOWNER’, which is not owned by ‘SYSADM’, but ‘PS’. This table is crucial for your application and must be migrated manually. Also, ‘PSDEFAULT’ is the default tablespace for the users ‘SYSADM’ and ‘PEOPLE’.

Execute the following steps to successfully migrate ‘PSDEFAULT’:

```
SYSADM> @./work/01-create_tbs_PSDEFAULT_ENC.sql
SYSADM> @./work/02-start_redef_PSDEFAULT_ENC.sql
SYSADM> @./work/03-create_index_for_PSDEFAULT_ENC.sql
SYSADM> @./work/04-reg_indexes_in_PSDEFAULT_ENC.sql
SYSADM> @./work/05-copy_deps_from_PSDEFAULT.sql
SYSADM> @./work/06-sync_finish_tables_in_PSDEFAULT_ENC.sql
SYSADM> alter tablespace PSDEFAULT rename to PSDEFAULT_backup;
SYSADM> alter tablespace PSDEFAULT_ENC rename to PSDEFAULT;
SYSADM> conn sys as sysdba
SYS> grant connect, resource, dba to PS identified by PS;
SYS> conn PS/PS;
PS> drop index PS_PSDBOWNER;
PS> drop public synonym PSDBOWNER;
PS> alter table PSDBOWNER move tablespace PSDEFAULT;
```

```
PS> create unique index PS_PSDBOWNER on PSDBOWNER(DBNAME) tablespace  
      PSDEFAULT  
PS> create public synonym PSDBOWNER for PSDBOWNER;  
PS> grant select on PSDBOWNER to PUBLIC;  
PS> conn sys as sysdba  
SYS> revoke connect, resource, dba from PS;  
SYS> alter user SYSADM default tablespace PSDFAULT;  
SYS> alter user PEOPLE default tablespace PSDFAULT;  
SYS> conn sysadm  
SYSADM> drop tablespace PSDEFAULT_backup including contents and  
      datafiles;
```

Migrate your application tablespaces

For the other tablespaces, locate six scripts in the WORK directory and execute them following their numbering scheme; for example:

```
SYSADM> @./work/01-create_tbs_PYAPP_ENC.sql
```

This creates the new encrypted tablespace with otherwise unmodified characteristics, and all interim tables that belong to this tablespace.

```
SYSADM> @./work/02-start_redef_PYAPP_ENC.sql
```

This script starts the Online Table Redefinition process by copying the content from the active source tables to the interim tables in the encrypted tablespace.

```
SYSADM> @./work/03-create_index_for_PYAPP_ENC.sql
```

This script generates the indexes for the tables in PYAPP_ENC in PSINDEX_ENC.

```
SYSADM> @./work/04-reg_indexes_in_PYAPP_ENC.sql
```

Here, the indexes are registered with the new tables in the encrypted tablespace

```
SYSADM> @./work/05-copy_deps_from_PYAPP.sql
```

This script copies remaining dependent objects for all tables in the current tablespace, and automatically registers them with the interim tables.

```
SYSADM> @./work/06-sync_finish_tables_in_PYAPP_ENC.sql
```

The last script synchronizes the interim table with recent changes to the source table; as soon as the synchronization process for the currently processed table is complete, the ‘finish_redef’ process starts, which swaps the table names. From now on, the tables in the encrypted tablespace are the active tables; the original tables in the clear-text tablespace are no longer used.

Once it has been verified that the application continues to work as expected, the clear text tablespace can be dropped:

```
SYSADM> drop tablespace PYAPP including contents and datafiles;
```

Verify and attest

One way to verify and attest is to use SQL*Developer's Reports:

TSName	Algorithm	Filename
RSFAPP	AES256	/opt/oracle/oradata/rsft/rsfapp_enc.dbf
B2APP	AES256	/opt/oracle/oradata/rsft/b2app_enc.dbf
PTAUDIT	AES256	/opt/oracle/oradata/rsft/ptaudit_enc.dbf
PYAPP	AES256	/opt/oracle/oradata/rsft/pyapp_enc.dbf
FAAPP	AES256	/opt/oracle/oradata/rsft/faapp_enc.dbf
PSIMGR	AES256	/opt/oracle/oradata/rsft/psimgr_enc.dbf
RBTEMP	AES256	/opt/oracle/oradata/rsft/rbtemp_enc.dbf
RBAAPP	AES256	/opt/oracle/oradata/rsft/rbaapp_enc.dbf
TRAPP	AES256	/opt/oracle/oradata/rsft/trapp_enc.dbf
TLAPP	AES256	/opt/oracle/oradata/rsft/tlapp_enc.dbf

Figure 10: SQL*Developer shows all encrypted tablespaces

Owner	Table Name	Algorithm
SYSADM	INT_PS_BANK_ACCHRT_TMP	AES256
SYSADM	INT_PS_BANK_ACCT14_LNG	AES256
SYSADM	INT_PS_BANK_ACCT_CF_DR	AES256
SYSADM	INT_PS_BAX_HEADER_TR	AES256
SYSADM	INT_PS_BANK_ACCT_CNTCT	AES256
SYSADM	INT_PS_BANK_ACCT_CPTY	AES256
SYSADM	INT_PS_BANK_ACCT_CURR	AES256
SYSADM	INT_PS_BANK_ACCT_DEFN	AES256
SYSADM	INT_PS_BANK_ACCT_LNG2	AES256
SYSADM	INT_PS_BANK_ACCT_RECON	AES256
SYSADM	INT_PS_BANK_ACCT_VDT	AES256
SYSADM	INT_PS_BANK_ACCTINT_LNG	AES256

Figure 11: SQL*Developer lists all application tables in encrypted tablespaces

Appendix A: Using a Hardware Security Module (HSM)

When the version of the underlying database is 11.1.0.7 (regardless if new installation or migration from Oracle Database 10gR2 with or without encrypted application table columns using TDE column encryption), do **not** set or re-key the master encryption key before changing the appropriate entry in `sqlnet.ora` from ‘METHOD = FILE’ to ‘METHOD = HSM’. Otherwise, encryption master keys for TDE column encryption and TDE tablespace encryption will be generated in the Oracle Wallet; the master key for TDE tablespace encryption **cannot** be migrated from wallet to HSM. These restrictions do not apply to Oracle Database 11g Release 2.

Detailed upgrade procedures are available in the Transparent Data Encryption Best Practices document, available from the Oracle Technology Network.

If you are running Oracle Database 11gR1, it is mandatory to install the following patches prior to attempting to use an HSM:

<http://updates.oracle.com/download/8211698.html> and
<http://updates.oracle.com/download/7563307.html>.



TDE Tablespace Encryption
Implementation Guide for Oracle Peoplesoft
February 2010
Author: Peter A. Wahl
Contributing Authors: Chao Liang, James Spiller

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
oracle.com



Oracle is committed to developing practices and products that help protect the environment

Copyright © 2010, Oracle and/or its affiliates. All rights reserved. This document is provided for information purposes only and the contents hereof are subject to change without notice. This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.