11G - Reduce Transportable Tablespace Downtime using Cross Platform Incremental Backup (Doc ID 1389592.1)

In this Document

APPLIES TO:

Oracle Database Cloud Schema Service - Version N/A and later
Oracle Database Backup Service - Version N/A and later
Oracle Database Cloud Service - Version N/A and later
Oracle Database - Enterprise Edition - Version 10.2.0.3 and later
Oracle Database Exadata Express Cloud Service - Version N/A and later
Linux x86-64

PURPOSE

Consider using the new release of this procedure, version 4. This version has drastically simplified the steps and procedure. Before proceeding, review:
V4 Reduce Transportable Tablespace Downtime using Cross Platform Incremental Backup Note 2471245.1

No bug fixs or diagnostics will be conducted on this older version of XTTs. Unless absolutely necessary, only V4 of XTTs should be used.

When using Cross Platform Transportable Tablespaces (XTTS) to migrate data between systems that have different endian formats, the amount of downtime required can be substantial because it is directly proportional to the size of the data set being moved. However, combining XTTS with Cross Platform Incremental Backup can significantly reduce the amount of downtime required to move data between platforms.

Traditional Cross Platform Transportable Tablespaces

The high-level steps in a typical XTTS scenario are the following:

1. Make tablespaces in source database READ ONLY
2. Transfer datafiles to destination system
3. Convert datafiles to destination system endian format
4. Export metadata of objects in the tablespaces from source database using Data Pump
5. Import metadata of objects in the tablespaces into destination database using Data Pump
6. Make tablespaces in destination database READ WRITE

Because the data transported must be made read only at the very start of the procedure, the application that owns the data is effectively unavailable to users for the entire duration of the procedure. Due to the serial nature of the steps, the downtime required for this procedure is proportional to the amount of data. If data size is large, datafile transfer and convert times can be long, thus downtime can be long.

Reduce Downtime using Cross Platform Incremental Backup

To reduce the amount of downtime required for XTTS, Oracle has enhanced RMAN's ability to roll forward datafile copies using incremental backups, to work in a cross-platform scenario. By using a series of incremental backups, each smaller than the last, the data at the destination system can be brought almost current with the source system, before any downtime is required. The downtime required for datafile transfer and convert when combining XTTS with Cross Platform Incremental Backup is now proportional to the rate of data block changes in the source system.

The Cross Platform Incremental Backup feature does not affect the amount of time it takes to perform other actions for XTTS, such as metadata export and import. Hence, databases that have very large amounts of metadata (DDL) will see limited benefit from Cross Platform Incremental Backup since migration time is typically dominated by metadata operations, not datafile transfer and conversion.

Only those database objects that are physically located in the tablespaces that are being transported will be copied to the destination system. If you need for other objects to be transported, that are located in different tablespaces (such as, for example, pl/sql objects, sequences, etc., that are located in the SYSTEM tablespace), you can use data pump to copy those objects to the destination system.

The high-level steps using the cross platform incremental backup capability are the following:

1. Prepare phase (source data remains online)
    1. Transfer datafiles to destination system
    2. Convert datafiles, if necessary, to destination system endian format
2. Roll Forward phase (source data remains online - Repeat this phase as many times as necessary to catch destination datafile copies up to source database)
    1. Create incremental backup on source system
    2. Transfer incremental backup to destination system
    3. Convert incremental backup to destination system endian format and apply the backup to the destination datafile copies

NOTE: In Version 3, if a datafile is added to the tablespace OR a new tablespace name is added to the xtt.properties file, a warning and additional instructions will be required.

3. Transport phase (source data is READ ONLY)
    1. Make tablespaces in source database READ ONLY
    2. Repeat the Roll Forward phase one final time
        - This step makes destination datafile copies consistent with source database.
        - Time for this step is significantly shorter than traditional XTTS method when dealing with large data because the incremental backup size is smaller.
    3. Export metadata of objects in the tablespaces from source database using Data Pump
    4. Import metadata of objects in the tablespaces into destination database using Data Pump
    5. Make tablespaces in destination database READ WRITE

The purpose of this document is to provide an example of how to use this enhanced RMAN cross platform incremental backup capability to reduce downtime when transporting tablespaces across platforms.

SCOPE

Consider using the new release of this procedure, version 4. This version has drastically simplified the steps and procedure. Before proceeding, review:
V4 Reduce Transportable Tablespace Downtime using Cross Platform Incremental Backup Note 2471245.1

The source system may be any platform provided the prerequisites referenced and listed below for both platform and database are met.

If you are migrating from a little endian platform to Oracle Linux, then the migration method that should receive first consideration is Data Guard. See Note 413484.1 for details about heterogeneous platform support for Data Guard between your current little endian platform and Oracle Linux.

This method can also be used with 12c databases, however, for an alternative method for 12c see:

Note 2005729.1 12C - Reduce Transportable Tablespace Downtime using Cross Platform Incremental Backup.

NOTE: Neither method supports 12c multitenant databases. Enhancement bug 22570430 addresses this limitation.

DETAILS

Consider using the new release of this procedure, version 4. This version has drastically simplified the steps and procedure. Before proceeding, review:
V4 Reduce Transportable Tablespace Downtime using Cross Platform Incremental Backup Note 2471245.1

Overview

This document provides a procedural example of transporting two tablespaces called TS1 and TS2 from an Oracle Solaris SPARC system to an Oracle Exadata Database Machine running Oracle Linux, incorporating Oracle's Cross Platform Incremental Backup capability to reduce downtime.

After performing the Initial Setup phase, moving the data is performed in the following three phases:

Prepare phase
During the Prepare phase, datafile copies of the tablespaces to be transported are transferred to the destination system and converted. The application being migrated is fully accessible during the Prepare phase. The Prepare phase can be performed using RMAN backups or dbms_file_transfer. Refer to the Selecting the Prepare Phase Method section for details about choosing the Prepare phase method.

Roll Forward phase
During the Roll Forward phase, the datafile copies that were converted during the Prepare phase are rolled forward using incremental backups taken from the source database. By performing this phase multiple times, each successive incremental backup becomes smaller and faster to apply, allowing the data at the destination system to be brought almost current with the source system. The application being migrated is fully accessible during the Roll Forward phase.

Transport phase
During the Transport phase, the tablespaces being transported are put into READ ONLY mode, and a final incremental backup is taken from the source database and applied to the datafile copies on the destination system, making the destination datafile copies consistent with source database. Once the datafiles are consistent, the tablespaces are TTS-exported from the source database and TTS-imported into the destination database. Finally, the tablespaces are made READ WRITE for full access on the destination database. The application being migrated cannot receive any updates during the Transport phase.

Cross Platform Incremental Backup Supporting Scripts

The Cross Platform Incremental Backup core functionality is delivered in Oracle Database 11.2.0.4 and later. See the Requirements and Recommendations section for details. In addition, a set of supporting scripts in the file `rman-xttconvert_3.0.zip` are attached to this document that are used to manage the procedure required to perform XTTS with Cross Platform Incremental Backup. The two primary supporting scripts files are the following:

- Perl script `xttdriver.pl` - the script that is run to perform the main steps of the XTTS with Cross Platform Incremental Backup procedure.
- Parameter file `xtt.properties` - the file that contains your site-specific configuration.

Requirements and Recommendations

This section contains the following subsections:

- Prerequisites
- Selecting the Prepare Phase Method
- Destination Database 11.2.0.3 or Earlier Requires a Separate Incremental Convert Home and Instance

Prerequisites

The following prerequisites must be met before starting this procedure:

- The limitations and considerations for transportable tablespaces must still be followed. They are defined in the following manuals:
    - Oracle Database Administrator's Guide
    - Oracle Database Utilities
- In addition to the limitations and considerations for transportable tablespaces, the following conditions must be met:
    - The current version does NOT support Windows.
    - The source database must be running 10.2.0.3 or higher.
    - Cross platform is only possible with Enterprise Edition. This procedure cannot be used with Standard Edition.
    - The source database must have its COMPATIBLE parameter set to 10.2.0 or higher.
    - The source database's COMPATIBLE parameter must not be greater than the destination database's COMPATIBLE parameter.
    - The source database must be in ARCHIVELOG mode.
    - The destination database must be running 11.2.0.4 or higher.
    - Although preferred destination system is Linux (either 64-bit Oracle Linux or a certified version of RedHat Linux), this procedure can be used with other unix based operating systems.
    - The Oracle version of source must be lower or equal to destination. Therefore, this procedure can be used as an upgrade method. Restrictions to transportable tablespace will apply..
    - RMAN's default device type should be configured to DISK
    - RMAN on the source system must not have DEVICE TYPE DISK configured with COMPRESSED. If so, procedure may return: ORA-19994: cross-platform backup of compressed backups different endianess.
    - The set of tablespaces being moved must all be online, and contain no offline data files. Tablespaces must be READ WRITE. Tablespaces that are READ ONLY may be moved with the normal XTTS method. There is no need to incorporate Cross Platform Incremental Backups to move tablespaces that are always READ ONLY.
- All steps in this procedure are run as the oracle user that is a member of the OSDBA group. OS authentication is used to connect to both the source and destination databases.
- If the Prepare Phase method selected is dbms_file_transfer, then the destination database must be 11.2.0.4. See the Selecting the Prepare Phase Method section for details.
- If the Prepare Phase method selected is RMAN backup, then staging areas are required on both the source and destination systems. See the Selecting the Prepare Phase Method section for details.
- It is not supported to execute this procedure against a standby or snapshot standby databases.
- If the destination database version is 11.2.0.3 or lower, then a separate database home containing 11.2.0.4 running an 11.2.0.4 instance on the destination system is required to perform the incremental backup conversion. See the Destination Database 11.2.0.3 and Earlier Requires a Separate Incremental Convert Home and Instance section for details. If using ASM for 11.2.0.4 Convert Home, then ASM needs to be on 11.2.0.4, else error ORA-15295 (e.g. ORA-15295: ASM instance software version 11.2.0.3.0 less than client version 11.2.0.4.0) is raised.

Whole Database Migration

If Cross Platform Incremental Backups will be used to reduce downtime for a whole database migration, then the steps in this document can be combined with the XTTS guidance provided in the MAA paper Platform Migration Using Transportable Tablespaces: Oracle Database 11g.

This method can also be used with 12c databases, however, for an alternative method for 12c see:
Note 2005729.1 12C - Reduce Transportable Tablespace Downtime using Cross Platform Incremental Backup.

Selecting the Prepare Phase Method

During the Prepare phase, datafiles of the tablespaces to be transported are transferred to the destination system and converted by the xttdriver.pl script. There are two possible methods:

1. Using dbms_file_transfer (DFT) transfer (using xttdriver.pl -S and -G options)
2. Using Recovery Manager (RMAN) RMAN backup (using xttdriver.pl -p and -c options)

The dbms_file_transfer method uses the dbms_file_transfer.get_file() subprogram to transfer the datafiles from the source system to the target system over a database link. The dbms_file_transfer method has the following advantages over the RMAN method: 1) it does not require staging area space on either the source or destination system; 2) datafile conversion occurs automatically during transfer - there is not a separate conversion step. The dbms_file_transfer method requires the following:

- A destination database running 11.2.0.4. Note that an incremental convert home or instance do not participate in dbms_file_transfer file transfers.
- A database directory object in the source database from where the datafiles are copied.
- A database directory object in the destination database to where the datafiles are placed.
- A database link in the destination database referencing the source database.

The RMAN backup method runs RMAN on the source system to create backups on the source system of the datafiles to be transported. The backups files must then be manually transferred over the network to the destination system. On the destination system the datafiles are converted by RMAN, if necessary. The output of the RMAN conversion places the datafiles in their final location where they will be used by the destination database. In the original version of xttdriver.pl, this was the only method supported. The RMAN backup method requires the following:

- Staging areas are required on both the source and destination systems for the datafile copies created by RMAN. The staging areas are referenced in the xtt.properties file using the parameters dfcopydir and stageondest. The final destination where converted datafiles are placed is referenced in the xtt.properties file using the parameter storageondest. Refer to the Description of Parameters in Configuration File xtt.properties section for details and sizing guidelines.

Details of using each of these methods are provided in the instructions below. The recommended method is the dbms_file_transfer method.

Destination Database 11.2.0.3 or Earlier Requires a Separate Incremental Convert Home and Instance

The Cross Platform Incremental Backup core functionality (i.e. incremental backup conversion) is delivered in Oracle Database 11.2.0.4 and later. If the destination database version is 11.2.0.4 or later, then the destination database can perform this function. However, if the destination database version is 11.2.0.3 or earlier, then, for the purposes of performing incremental backup conversion, a separate 11.2.0.4 software home, called the incremental convert home, must be installed, and an instance, called the incremental convert instance, must be started in NOMOUNT state using that home. The incremental convert home and incremental convert instance are temporary and are used only during the migration.

Note that because the dbms_file_transfer Prepare Phase method requires destination database 11.2.0.4, which can be used to perform the incremental backup conversions function (as stated above), an incremental convert home and incremental convert instance are usually only applicable when the Prepare Phase method is RMAN backup.

For details about setting up a temporary incremental convert instance, see instructions in Phase 1.

Troubleshooting

To enable debug mode, either run xttdriver.pl with the -d flag, or set environment variable XTTDEBUG=1 before running xttdriver.pl. Debug mode enables additional screen output and causes all RMAN executions to be performed with the debug command line option.

Known Issues

1. If the source database contains nested IOTs with key compression, then the fix for Bug 14835322 must be installed in the destination database home (where the tablespace plug operation occurs).
2. If you wish to utilize block change tracking on the source database when incremental backups are created, then the fix for Bug 16850197 must be installed in the source database home.
3. If using ASM in both source and destination, see XTTS Creates Alias on Destination when Source and Destination use ASM (Note 2351123.1)
4. If the roll forward phase (xttdriver.pl -r) fails with the following errors, then verify RMAN DEVICE TYPE DISK is not configured COMPRESSED.

   ```
   Entering RollForward
   After applySetDataFile
   Done: applyDataFileTo
   Done: RestoreSetPiece
   DECLARE
   ```

```
*
ERROR at line 1:
ORA-19624: operation failed, retry possible
ORA-19870: error while restoring backup piece
/dbfs_direct/FS1/xtts/incrementals/xtts_incr_backup
ORA-19608: /dbfs_direct/FS1/xtts/incrementals/xtts_incr_backup is not a backup
piece
ORA-19837: invalid blocksize 0 in backup piece header
ORA-06512: at "SYS.X$DBMS_BACKUP_RESTORE", line 2338
ORA-06512: at line 40
```

5. This document can be referred as well for known issues : Note 17866999.8 & If the source contains cluster objects, then run "analyze cluster &cluster_name validate structure cascade" after XTTS has been completed in the target database and if it reports an ORA-1499 open the trace file and review if it has entries like:

   ```
   kdcchk: index points to block 0x01c034f2 slot 0x1 chain length is 256
   kdcchk: chain count wrong 0x01c034f2.1 chain is 1 index says 256
   last entry 0x01c034f2.1 blockcount = 1
   kdavls: kdcchk returns 3 when checking cluster dba 0x01c034a1 objn 90376
   ```

   Then to repair this inconsistency either:

   1. rebuild the cluster index.
   or
   2. Install fix bug 17866999 and run dbms_repair.repair_cluster_index_keycount

   If after repairing the inconsistency the "analyze cluster &cluster_name validate structure cascade" still reports issues then recreate the affected cluster which involves recreating its tables.

   Note that the fix of bug 17866999 is a workaround fix to repair the index cluster; it will not avoid the problem.Oracle did not find a valid fix for this situation so it will affect any rdbms versions.

6.

Transport Tablespaces with Reduced Downtime using Cross Platform Incremental Backup

The XTTS with Cross Platform Incremental Backups procedure is divided into the following four phases:

- Phase 1 - Initial Setup phase
- Phase 2 - Prepare phase
- Phase 3 - Roll Forward phase
- Phase 4 - Transport phase

Conventions Used in This Document

- All command examples use bash shell syntax.
- Commands prefaced by the shell prompt string `[oracle@source]$` indicate commands run as the oracle user on the source system.
- Commands prefaced by the shell prompt string `[oracle@dest]$` indicate commands run as the oracle user on the destination system.

Phase 1 - Initial Setup

Perform the following steps to configure the environment to use Cross Platform Incremental Backups:

Step 1.1 - Install the Destination Database Software and Create the Destination Database

Install the desired Oracle Database software on the destination system that will run the destination database. It is **highly recommended** to use Oracle Database 11.2.0.4 or later. Note that the dbms_file_transfer Prepare Phase method requires the destination database to be 11.2.0.4.

Identify (or create) a database on the destination system to transport the tablespace(s) into and create the schema users required for the tablespace transport.

> Per generic TTS requirement, ensure that the schema users required for the tablespace transport exist in the destination database.

Step 1.2 - If necessary, Configure the Incremental Convert Home and Instance

See the Destination Database 11.2.0.3 and Earlier Requires a Separate Incremental Convert Home and Instance section for details.

Skip this step if the destination database software version is 11.2.0.4 or later. Note that the dbms_file_transfer Prepare Phase method requires the destination database to be 11.2.0.4.

If the destination database is 11.2.0.3 or earlier, then you must configure a separate incremental convert instance by performing the following steps:

- Install a new 11.2.0.4 database home on the destination system. This is the incremental convert home.
- Using the incremental convert home startup an instance in the NOMOUNT state. This is the incremental convert instance. A database does not need to be created for the incremental convert instance. Only a running instance is required.

The following steps may be used to create an incremental convert instance named xtt running out of incremental convert home /u01/app/oracle/product/11.2.0.4/xtt_home:

```
[oracle@dest]$ export ORACLE_HOME=/u01/app/oracle/product/11.2.0.4/xtt_home

[oracle@dest]$ export ORACLE_SID=xtt

[oracle@dest]$ cat << EOF > $ORACLE_HOME/dbs/init$ORACLE_SID.ora
db_name=xtt
compatible=11.2.0.4.0
EOF

[oracle@dest]$ sqlplus / as sysdba
SQL> startup nomount
```

> If ASM storage is used for the xtt.properties parameter `backupondest` (described below), then the `COMPATIBLE` initialization parameter setting for this instance must be equal to or higher than the `rdbms.compatible` setting for the ASM disk group used.

Step 1.3 - Identify Tablespaces to be Transported

Identify the tablespace(s) in the source database that will be transported. Tablespaces TS1 and TS2 will be used in the examples in this document. As indicated above, the limitations and considerations for transportable tablespaces must still be followed.

Step 1.4 - If Using dbms_file_transfer Prepare Phase Method, then Configure Directory Objects and Database Links

Note that the dbms_file_transfer Prepare Phase method requires the destination database to be 11.2.0.4.

If using dbms_file_transfer as the Prepare Phase method, then three database objects must be created:

1. A database directory object in the source database from where the datafiles are copied
2. A database directory object in the destination database to where the datafiles are placed
3. A database link in the destination database referencing the source database

The source database directory object references the location where the datafiles in the source database currently reside. For example, to create directory object sourcedir that references datafiles in ASM location +DATA/prod/datafile, connect to the source database and run the following SQL command:

```
SQL@source> create directory sourcedir as '+DATA/prod/datafile';
```

The destination database directory object references the location where the datafiles will be placed on the destination system. This should be the final location where the datafils will reside when in use by the destination database. For example, to create directory object dstdir that will place transferred datafiles in ASM location +DATA/prod/datafile, connect to the destination database and run the following SQL command:

```
SQL@dest> create directory destdir as '+DATA/prod/datafile';
```

The database link is created in the destination database, referencing the source database. For example, to create a database link named ttslink, run the following SQL command:

```
SQL@dest> create public database link ttslink connect to system identified by <password>
using '<tns_to_source>';
```

Verify the database link can properly access the source system:

```
SQL@dest> select * from dual@ttslink;
```

Step 1.5 - Create Staging Areas

Create the staging areas on the source and destinations systems as defined by the following xtt.properties parameters: backupformat, backupondest.

Also, if using RMAN backups in the Prepare phase, create the staging areas on the source and destinations systems as defined by the following xtt.properties parameters: dfcopydir, stageondest.

Step 1.6 - Install xttconvert Scripts on the Source System

On the source system, as the oracle software owner, download and extract the supporting scripts attached as `rman-xttconvert_3.0.zip` to this document.

```
[oracle@source]$ pwd
/home/oracle/xtt

[oracle@source]$ unzip rman_xttconvert_v3.zip
Archive: rman_xttconvert_v3.zip
inflating: xtt.properties
inflating: xttcnvrtbkupdest.sql
inflating: xttdbopen.sql
inflating: xttdriver.pl
inflating: xttprep.tmpl
extracting: xttstartupnomount.sql
```

Step 1.7 - Configure xtt.properties on the Source System

Edit the `xtt.properties` file on the source system with your site-specific configuration.   For more information about the parameters in the `xtt.properties` file, refer to the Description of Parameters in Configuration File xtt.properties section in the Appendix below.

Step 1.8 - Copy xttconvert Scripts and xtt.properties to the Destination System

As the oracle software owner copy all xttconvert scripts and the modified `xtt.properties` file to the destination system.

```
[oracle@source]$ scp -r /home/oracle/xtt dest:/home/oracle/xtt
```

Step 1.9 - Set TMPDIR

In the shell environment on both source and destination systems, set environment variable TMPDIR to the location where the supporting scripts exist. Use this shell to run the Perl script `xttdriver.pl` as shown in the steps below. If TMPDIR is not set, output files are created in and input files are expected to be in /tmp.

```
[oracle@source]$ export TMPDIR=/home/oracle/xtt

[oracle@dest]$ export TMPDIR=/home/oracle/xtt
```

Phase 2 - Prepare Phase

During the Prepare phase, datafiles of the tablespaces to be transported are transferred to the destination system and converted by the xttdriver.pl script. There are two possible methods:

1. Phase 2A - dbms_file_transfer Method
2. Phase 2B - RMAN Backup Method

Select and use one of these methods based upon the information provided in the Requirements and Recommendations section above.

NOTE: For large number of files, using dbms_file_transfer has been found to be the fastest method for transferring datafiles to destination.

Phase 2A - Prepare Phase for dbms_file_transfer Method

Only use the steps in Phase 2A if the Prepare Phase method chosen is dbms_file_transfer and the setup instructions have been completed, particularly those in Step 1.4.

During this phase datafiles of the tablespaces to be transported are transferred directly from source system and placed on the destination system in their final location to be used by the destination database. If conversion is required, it is performed automatically during transfer. No separate conversion step is required. The steps in this phase are run only once. The data being transported is fully accessible in the source database during this phase.

Step 2A.1 - Run the Prepare Step on the Source System

On the source system, logged in as the oracle user with the environment (ORACLE_HOME and ORACLE_SID environment variables) pointing to the source database, run the prepare step as follows:

```
[oracle@source]$ $ORACLE_HOME/perl/bin/perl xttdriver.pl -S
```

The prepare step performs the following actions on the source system:

- Verifies the tablespaces are online, in READ WRITE mode, and do not contain offline datafiles.
- Creates the following files used later in this procedure:
    - xttnewdatafiles.txt
    - getfile.sql

The set of tablespaces being transported must all be online, contain no offline data files, and must be READ WRITE. The Prepare step will signal an error if one or more datafiles or tablespaces in your source database are offline or READ ONLY. If a tablespace is READ ONLY and will remain so throughout the procedure, then simply transport those tablespaces using the traditional cross platform transportable tablespace process. No incremental apply is needed for those files.

Step 2A.2 - Transfer the Datafiles to the Destination System

On the destination system, log in as the oracle user and set the environment (ORACLE_HOME and ORACLE_SID environment variables) to the destination database (it is invalid to attempt to use an incremental convert instance). Copy the `xttnewdatafiles.txt` and `getfile.sql` files created in step 2A.1 from the source system and run the -G get_file step as follows:

NOTE: This step copies all datafiles being transported from the source system to the destination system. The length of time for this step to complete is dependent on datafile size, and may be substantial.
 Use getfileparallel option for parallelism.

```
[oracle@dest]$ scp oracle@source:/home/oracle/xtt/xttnewdatafiles.txt
/home/oracle/xtt
[oracle@dest]$ scp oracle@source:/home/oracle/xtt/getfile.sql /home/oracle/xtt

# MUST set environment to destination database
[oracle@dest]$ $ORACLE_HOME/perl/bin/perl xttdriver.pl -G
```

When this step is complete, the datafiles being transported will reside in the final location where they will be used by the destination database. Note that endian conversion, if required, is performed automatically during this step.

Proceed to Phase 3 to create and apply incremental backups to the datafiles.

Phase 2B - Prepare Phase for RMAN Backup Method

Only use the steps in Phase 2B if the Prepare Phase method chosen is RMAN backup and the setup instructions have been completed, particularly those in Step 1.5.

During this phase datafile copies of the tablespaces to be transported are created on the source system, transferred to the destination system, converted, and placed in their final location to be used by the destination database. The steps in this phase are run only once. The data being transported is fully accessible in the source database during this phase.

Step 2B.1 - Run the Prepare Step on the Source System

On the source system, logged in as the oracle user with the environment (ORACLE_HOME and ORACLE_SID environment variables) pointing to the source database, run the prepare step as follows:

```
[oracle@source]$ $ORACLE_HOME/perl/bin/perl xttdriver.pl -p
```

The prepare step performs the following actions on the source system:

- Creates datafile copies of the tablespaces that will be transported in the location specified by the xtt.properties parameter `dfcopydir`.
- Verifies the tablespaces are online, in READ WRITE mode, and do not contain offline datafiles.
- Creates the following files used later in this procedure:
  - xttplan.txt
  - rmanconvert.cmd

The set of tablespaces being transported must all be online, contain no offline data files, and must be READ WRITE. The Prepare step will signal an error if one or more datafiles or tablespaces in your source database are offline or READ ONLY. If a tablespace is READ ONLY and will remain so throughout the procedure, then simply transport those tablespaces using the traditional cross platform transportable tablespace process. No incremental apply is needed for those files.

Step 2B.2 - Transfer Datafile Copies to the Destination System

On the destination system, logged in as the oracle user, transfer the datafile copies created in the previous step from the source system. Datafile copies on the source system are created in the location defined in xtt.properties parameter `dfcopydir`. The datafile copies must be placed in the location defined by xtt.properties parameter `stageondest`.

Any method of transferring the datafile copies from the source system to the destination system that results in a bit-for-bit copy is supported.

If the `dfcopydir` location on the source system and the `stageondest` location on the destination system refer to the same NFS storage location, then this step can be skipped since the datafile copies are already available in the expected location on the destination system.

In the example below, `scp`is used to transfer the copies created by the previous step from the source system to the destination system.

```
[oracle@dest]$ scp oracle@source:/stage_source/* /stage_dest
```

Note that due to current limitations with cross-endian support in DBMS_FILE_TRANSPORT and ASMCMD, you must use OS-level commands, such as SCP or FTP to transfer the copies from the source system to destination system.

Step 2B.3 - Convert the Datafile Copies on the Destination System

On the destination system, logged in as the oracle user with the environment (ORACLE_HOME and ORACLE_SID environment variables) pointing to the destination database, copy the `rmanconvert.cmd` file created in step 2B.1 from the source system and run the convert datafiles step as follows:

```
[oracle@dest]$ scp oracle@source:/home/oracle/xtt/rmanconvert.cmd /home/oracle/xtt

[oracle@dest]$ $ORACLE_HOME/perl/bin/perl xttdriver.pl -c
```

The convert datafiles step converts the datafiles copies in the `stageondest` location to the endian format of the destination system. The converted datafile copies are written in the location specified by the xtt.properties parameter `storageondest`. This is the final location where datafiles will be accessed when they are used by the destination database.

When this step is complete, the datafile copies in `stageondest` location are no longer needed and may be removed.

Phase 3 - Roll Forward Phase

During this phase an incremental backup is created from the source database, transferred to the destination system, converted to the destination system endian format, then applied to the converted destination datafile copies to roll them forward. This phase may be run multiple times. Each successive incremental backup should take less time than the prior incremental backup, and will bring the destination datafile copies more current with the source database. The data being transported is fully accessible during this phase.

Step 3.1 - Create an Incremental Backup of the Tablespaces being Transported on the Source System

On the source system, logged in as the oracle user with the environment (ORACLE_HOME and ORACLE_SID environment variables) pointing to the source database, run the create incremental step as follows:

```
[oracle@source]$ $ORACLE_HOME/perl/bin/perl xttdriver.pl -i
```

The create incremental step executes RMAN commands to generate incremental backups for all `tablespaces` listed in xtt.properties. It creates the following files used later in this procedure:

- `tsbkupmap.txt`
- `incrbackups.txt`

Step 3.2 - Transfer Incremental Backup to the Destination System

Transfer the incremental backup(s) created during the previous step to the `stageondest` location on the destination system. The list of incremental backup files to copy are found in the `incrbackups.txt` file on the source system.

```
[oracle@source]$ scp `cat incrbackups.txt` oracle@dest:/stage_dest
```

If the `backupformat` location on the source system and the `stageondest` location on the destination system refer to the same NFS storage location, then this step can be skipped since the incremental backups are already available in the expected location on the destination system.

Step 3.3 - Convert the Incremental Backup and Apply to the Datafile Copies on the Destination System

On the destination system, logged in as the oracle user with the environment (ORACLE_HOME and ORACLE_SID environment variables) pointing to the destination database, copy the `xttplan.txt` and `tsbkupmap.txt` files from the source system and run the rollforward datafiles step as follows:

```
[oracle@dest]$ scp oracle@source:/home/oracle/xtt/xttplan.txt /home/oracle/xtt
[oracle@dest]$ scp oracle@source:/home/oracle/xtt/tsbkupmap.txt /home/oracle/xtt

[oracle@dest]$ $ORACLE_HOME/perl/bin/perl xttdriver.pl -r
```

The rollforward datafiles step connects to the incremental convert instance as SYS, converts the incremental backups, then connects to the destination database and applies the incremental backups for each tablespace being transported.

Note:
1. You must copy the `xttplan.txt` and `tsbkupmap.txt` files each time that this step is executed, because their content is different each iteration.
2. Do NOT change, copy or make any changes to the `xttplan.txt.new` generated by the script.
3. The destination instance will be shutdown and restarted by this process.

Step 3.4 - Determine the FROM_SCN for the Next Incremental Backup

On the source system, logged in as the oracle user with the environment (ORACLE_HOME and ORACLE_SID environment variables) pointing to the source database, run the determine new FROM_SCN step as follows:

```
[oracle@source]$ $ORACLE_HOME/perl/bin/perl xttdriver.pl -s
```

The determine new FROM_SCN step calculates the next FROM_SCN, records it in the file xttplan.txt, then uses that SCN when the next incremental backup is created in step 3.1.

Step 3.5 - Repeat the Roll Forward Phase (Phase 3) or Move to the Transport Phase (Phase 4)

At this point there are two choices:

1. If you need to bring the files at the destination database closer in sync with the production system, then repeat the Roll Forward phase, starting with step 3.1.
2. If the files at the destination database are as close as desired to the source database, then proceed to the Transport phase.

NOTE: If a datafile is added to one a tablespace since last incremental backup and/or a new tablespace name is added to the xtt.properties, the following will appear:

Error:

The incremental backup was not taken as a datafile has been added to the tablespace:

Please Do the following:
----
1. Copy fixnewdf.txt from source to destination temp dir

2. Copy backups:
<backup list>
from <source location> to the <stage_dest> in destination

3. On Destination, run $ORACLE_HOME/perl/bin/perl xttdriver.pl --fixnewdf

4. Re-execute the incremental backup in source:
$ORACLE_HOME/perl/bin/perl xttdriver.pl --bkpincr

NOTE: Before running incremental backup, delete FAILED in source temp dir or
run xttdriver.pl with -L option:

$ORACLE_HOME/perl/bin/perl xttdriver.pl -L --bkpincr

These instructions must be followed exactly as listed. The next incremental backup will include the new datafile.

Phase 4 - Transport Phase

NOTE: Be sure the destination database has the necessary objects to allow the import to succeed. This includes pre-creating the owners of the tables in the tablespace being plugged in. See information on Transportable Tablespace and the guidance provided in the MAA paper Platform Migration Using Transportable Tablespaces: Oracle Database 11g.

During this phase the source data is made READ ONLY and the destination datafiles are made consistent with the source database by creating and applying a final incremental backup. After the destination datafiles are made consistent, the normal transportable tablespace steps are performed to export object metadata from the source database and import it into the destination database. The data being transported is accessible only in READ ONLY mode until the end of this phase.

Step 4.1 - Make Source Tablespaces READ ONLY in the Source Database

On the source system, logged in as the oracle user with the environment (ORACLE_HOME and ORACLE_SID environment variables) pointing to the source database, make the tablespaces being transported READ ONLY.

```
system@source/prod SQL> alter tablespace TS1 read only;

Tablespace altered.

system@source/prod SQL> alter tablespace TS2 read only;
```

```
Tablespace altered.
```

Step 4.2 - Create the Final Incremental Backup, Transfer, Convert, and Apply It to the Destination Datafiles

Repeat steps 3.1 through 3.3 one last time to create, transfer, convert, and apply the final incremental backup to the destination datafiles.

```
[oracle@source]$ $ORACLE_HOME/perl/bin/perl xttdriver.pl -i

[oracle@source]$ scp `cat incrbackups.txt` oracle@dest:/stage_dest

[oracle@source]$ scp xttplan.txt oracle@dest:/home/oracle/xtt
[oracle@source]$ scp tsbkupmap.txt oracle@dest:/home/oracle/xtt

[oracle@dest]$ $ORACLE_HOME/perl/bin/perl xttdriver.pl -r
```

Step 4.3 - Import Object Metadata into Destination Database

On the destination system, logged in as the oracle user with the environment (ORACLE_HOME and ORACLE_SID environment variables) pointing to the destination database, run the generate Data Pump TTS command step as follows:

```
[oracle@dest]$ $ORACLE_HOME/perl/bin/perl xttdriver.pl -e
```

The generate Data Pump TTS command step creates a sample Data Pump network_link transportable import command in the file `xttplugin.txt` with the transportable tablespaces parameters TRANSPORT_TABLESPACES and TRANSPORT_DATAFILES correctly set. Note that network_link mode initiates an import over a database link that refers to the source database. A separate export or dump file is not required. If you choose to perform the tablespace transport with this command, then you must edit the import command to replace import parameters DIRECTORY, LOGFILE, and NETWORK_LINK with site-specific values.

The following is an example network mode transportable import command:

```
[oracle@dest]$ impdp directory=DATA_PUMP_DIR logfile=tts_imp.log
network_link=ttslink \
transport_full_check=no \
transport_tablespaces=TS1,TS2 \
transport_datafiles='+DATA/prod/datafile/ts1.285.771686721', \
'+DATA/prod/datafile/ts2.286.771686723', \
'+DATA/prod/datafile/ts2.287.771686743'
```

After the object metadata being transported has been extracted from the source database, the tablespaces in the source database may be made READ WRITE again, if desired.

Database users that own objects being transported must exist in the destination database before performing the transportable import.

If you do not use network_link import, then perform the tablespace transport by running transportable mode Data Pump Export on the source database to export the object metadata being transported into a dump file, then transfer the dump file to the destination system, then run transportable mode Data Pump Import to import the object metadata into the destination database. Refer to the following manuals for details:

- Oracle Database Administrator's Guide
- Oracle Database Utilities

Step 4.4 - Make the Tablespace(s) READ WRITE in the Destination Database

The final step is to make the destination tablespace(s) READ WRITE in the destination database.

```
system@dest/prod SQL> alter tablespace TS1 read write;

Tablespace altered.

system@dest/prod SQL> alter tablespace TS2 read write;

Tablespace altered.
```

Step 4.5 - Validate the Transported Data

At this step, the transported data is READ ONLY in the destination database. Perform application specific validation to verify the transported data.

Also, run RMAN to check for physical and logical block corruption by running VALIDATE TABLESPACE as follows:

```
RMAN> validate tablespace TS1, TS2 check logical;
```

Phase 5 - Cleanup

If a separate incremental convert home and instance were created for the migration, then the instance may be shutdown and the software removed.

Files created by this process are no longer required and may now be removed. They include the following:

- `dfcopydir` location on the source system
- `backupformat` location on the source system
- `stageondest` location on the destination system
- `backupondest` location on the destination system
- $TMPDIR location in both destination and source systems

Appendix

Description of Perl Script xttdriver.pl Options

The following table describes the options available for the main supporting script xttdriver.pl.

Option	**Description**
-S prepare source for transfer	-S option is used <u>only</u> when Prepare phase method is dbms_file_transfer. Prepare step is run once on the source system during Phase 2A with the environment (ORACLE_HOME and ORACLE_SID) set to the source database. This step creates files xttnewdatafiles.txt and getfile.sql.
-G get datafiles from source	-G option is used <u>only</u> when Prepare phase method is dbms_file_transfer. Get datafiles step is run once on the destination system during Phase 2A with the environment (ORACLE_HOME and ORACLE_SID) set to the destination database. The -S option must be run beforehand and files xttnewdatafiles.txt and getfile.sql transferred to the destination system. This option connects to the destination database and runs script getfile.sql. getfile.sql invokes dbms_file_transfer.get_file() subprogram for each datafile to transfer it from the

	source database directory object (defined by parameter srcdir) to the destination database directory object (defined by parameter dstdir) over a database link (defined by parameter srclink).
-p prepare source for backup	-p option is used only when Prepare phase method is RMAN backup. Prepare step is run once on the source system during Phase 2B with the environment (ORACLE_HOME and ORACLE_SID) set to the source database. This step connects to the source database and runs the xttpreparesrc.sql script once for each tablespace to be transported, as configured in xtt.properties. xttpreparesrc.sql does the following: 1. Verifies the tablespace is online, in READ WRITE mode, and contains no offline datafiles. 2. Identifies the SCN that will be used for the first iteration of the incremental backup step and writes it into file $TMPDIR/xttplan.txt. 3. Creates the initial datafile copies on the destination system in the location specified by the parameter dfcopydir set in xtt.properties. These datafile copies must be transferred manually to the destination system. 4. Creates RMAN script $TMPDIR/rmanconvert.cmd that will be used to convert the datafile copies to the required endian format on the destination system.
-c convert datafiles	-c option is used only when Prepare phase method is RMAN backup. Convert datafiles step is run once on the destination system during Phase 2B with the environment (ORACLE_HOME and ORACLE_SID) set to the destination database. This step uses the rmanconvert.cmd file created in the Prepare step to convert the datafile copies to the proper endian format. Converted datafile copies are written on the destination system to the location specified by the parameter storageondest set in xtt.properties.
-i create incremental	Create incremental step is run one or more times on the source system with the environment (ORACLE_HOME and ORACLE_SID) set to the source database. This step reads the SCNs listed in $TMPDIR/xttplan.txt and generates an incremental backup that will be used to roll forward the datafile copies on the destination system.
-r rollforward datafiles	Rollforward datafiles step is run once for every incremental backup created with the environment (ORACLE_HOME and ORACLE_SID) set to the destination database. This step connects to the incremental convert instance using the parameters cnvinst_home and cnvinst_sid, converts the incremental backup pieces created by the Create Incremental step, then connects to the destination database and rolls forward the datafile copies by applying the incremental for each tablespace being transported.
-s determine new FROM_SCN	Determine new FROM_SCN step is run one or more times with the environment (ORACLE_HOME and ORACLE_SID) set to the source database. This step calculates the next FROM_SCN, records it in the file xttplan.txt, then uses that SCN when the next incremental backup is created in step 3.1. It reports the mapping of the new FROM_SCN to wall clock time to indicate how far behind the changes in the next incremental backup will be.
-e generate Data Pump TTS command	Generates Data Pump TTS command step which can be executed on the destination system. This command is executed on the source and the command file created can be transfered to the destination for execution. This step creates the template of a Data Pump Import command that uses a network_link to import metadata of objects that are in the tablespaces being transported. It should be executed on the source and the resulting file transferred to the destination.

-d debug	-d option enables debug mode for xttdriver.pl and RMAN commands it executes. Debug mode can also be enabled by setting environment variable XTTDEBUG=1.

Description of Parameters in Configuration File xtt.properties

The following table describes the parameters defined in the xtt.properties file that is used by xttdriver.pl.

Parameter	**Description**	**Example Setting**
tablespaces	Comma-separated list of tablespaces to transport from source database to destination database. Must be a single line, any subsequent lines will not be read.	tablespaces=TS1,TS2
platformid	Source database platform id, obtained from V$DATABASE.PLATFORM_ID.	platformid=2

srcdir	Directory object in the source database that defines where the source datafiles currently reside. Multiple locations can be used separated by ",". The srcdir to dstdir mapping can either be N:1 or N:N. i.e. there can be multiple source directories and the files will be written to a single destination directory, or files from a particular source directory can be written to a particular destination directory.
	
	This parameter is used only when Prepare phase method is dbms_file_transfer.
		srcdir=SOURCEDIR
		
		srcdir=SRC1,SRC2

dstdir	Directory object in the destination database that defines where the destination datafiles will be created. If multiple source directories are used (srcdir), then multiple destinations can be defined so a particular source directory is written to a particular destination directory.
	
	This parameter is used only when Prepare phase method is dbms_file_transfer.
		dstdir=DESTDIR
		
		dstdir=DST1,DST2

srclink	Database link in the destination database that refers to the source database. Datafiles will be transferred over this database link using dbms_file_transfer. This parameter is used <u>only</u> when Prepare phase method is dbms_file_transfer.	srclink=TTSLINK
dfcopydir		dfcopydir=/stage_source

	Location on the source system where datafile copies are created during the "-p prepare" step. This location must have sufficient free space to hold copies of all datafiles being transported. This location may be an NFS-mounted filesystem that is shared with the destination system, in which case it should reference the same NFS location as the stageondest parameter for the destination system. See Note 359515.1 for mount option guidelines. This parameter is used only when Prepare phase method is RMAN backup.	
backupformat	Location on the source system where incremental backups are created. This location must have sufficient free space to hold the incremental backups created for one iteration through the process documented above. This location may be an NFS-mounted filesystem that is shared with the destination system, in which case it should reference the same NFS location as the stageondest parameter for the destination system.	backupformat=/stage_source
stageondest	Location on the destination system where datafile copies are placed by the user when they are transferred manually from the source system. This location must have sufficient free space to hold copies of all datafiles being transported. This is also the location from where datafiles copies and incremental backups are read when they are converted in the "-c conversion of datafiles" and "-r roll forward datafiles" steps. This location may be a DBFS-mounted filesystem. This location may be an NFS-mounted filesystem that is shared with the source system, in which case it should reference the same NFS location as the dfcopydir and backupformat parameters for the source system. See Note 359515.1 for mount option guidelines.	stageondest=/stage_dest
storageondest	Location on the destination system	storageondest=+DATA - or -

	where the converted datafile copies will be written during the "-c conversion of datafiles" step. This location must have sufficient free space to permanently hold the datafiles that are transported. This is the final location of the datafiles where they will be used by the destination database. This parameter is used only when Prepare phase method is RMAN backup.	storageondest=/oradata/prod/%U
backupondest	Location on the destination system where converted incremental backups on the destination system will be written during the "-r roll forward datafiles" step. This location must have sufficient free space to hold the incremental backups created for one iteration through the process documented above. NOTE: If this is set to an ASM location then define properties asm_home and asm_sid below. If this is set to a file system location, then comment out asm_home and asm_sid parameters below.	backupondest=+RECO
cnvinst_home	Only set this parameter if a separate incremental convert home is in use. ORACLE_HOME of the incremental convert instance that runs on the destination system.	cnvinst_home=/u01/app/oracle/product/11.2.0.4/xtt_home
cnvinst_sid	Only set this parameter if a separate incremental convert home is in use. ORACLE_SID of the incremental convert instance that runs on the destination system.	cnvinst_sid=xtt
asm_home	ORACLE_HOME for the ASM instance that runs on the destination system. NOTE: If backupondest is set to a file system location, then comment out both asm_home and asm_sid.	asm_home=/u01/app/11.2.0.4/grid
asm_sid	ORACLE_SID for the ASM instance that runs on the destination system.	asm_sid=+ASM1
parallel	Defines the degree of parallelism set in the RMAN CONVERT command file rmanconvert.cmd. This file is created during the prepare step and used by RMAN in the convert datafiles step to	parallel=3

	convert the datafile copies on the destination system. If this parameter is unset, xttdriver.pl uses parallel=8. NOTE: RMAN parallelism used for the datafile copies created in the RMAN Backup prepare phase and the incremental backup created in the rollforward phase is controlled by the RMAN configuration on the source system. It is not controlled by this parameter.	
rollparallel	Defines the level of parallelism for the -r roll forward operation.	rollparallel=2
getfileparallel	Defines the level of parallelism for the -G operation Default value is 1. Maximum supported value is 8.	getfileparallel=4

Known Issue

Known Issues for Cross Platform Transportable Tablespaces XTTS Document 2311677.1

Change History

Change	**Date**
rman_xttconvert_v3.zip released - adds support for added datafiles	2017-Jun-06
rman-xttconvert_2.0.zip released - add support for multiple source and destination directories	2015-Apr-20
rman-xttconvert_1.4.2.zip released - add parallelism support for -G get file from source operation	2014-Nov-14
rman-xttconvert_1.4.zip released - remove staging area requirement, add parallel rollforward, eliminate conversion instance requirements when using 11.2.0.4.	2014-Feb-21
rman-xttconvert_1.3.zip released - improves handling of large databases with large number of datafiles.	2013-Apr-10

REFERENCES

NOTE:733824.1 - How To Recreate A Database Using TTS (Transportable TableSpace)

NOTE:25670970.8 - Bug 25670970 - ORA-600 [25027] or ORA-600 [kcfrbd_3] on Securefile Lob Segment after XTTS

BUG:17866999 - ORA-1499 FOR CLUSTER FOLLOWING RMAN CONVERT
NOTE:25890056.8 - Bug 25890056 - Fix for Bug 25670970 on 12.2 - ORA-600 [25027] on Securefile Lob Segment after XTTS
NOTE:2100369.1 - Cross Platform Incremental Failed with "Error in executing xttstartupnomount.sql"
NOTE:2141314.1 - Running Roll forward Phase of XTT Procedure in Note 1389592.1 Returns Error ORA-19848

Didn't find what you are looking for?