ORACLE

Disaster Recovery Using Cross-Region Backups

Low-Cost Disaster Recovery Solution Using Object Storage's Cross-Region Replication in Oracle Public Cloud

10/06/2022

Table of contents

Purpose

This paper provides guidelines and best practices for configuring a Low-Cost Disaster Recovery solution using Object Storage Service cross-region replication feature. Detailed steps covering the Disaster Recovery of the database in a remote region are also included. This paper is relevant for all Oracle databases residing in Oracle Public Cloud using Oracle Exadata Database Service on Dedicated Infrastructure (ExaDB-D).

Overview

Oracle's Object Storage Service cross-region replication provides an easy to implement and inexpensive replication feature, allowing for automatic replication of your local region's Exadata Database Service Database backups. If a local region disaster occurs, the replicated copy of your local backups provides the ability to restore your database to a new or existing Exadata Database Service (ExaDB-D) in the remote region with minimum data loss or Recovery Point Objective (RPO). As database and archive backups are written to the local region's Object Storage Service (OSS) bucket, they are replicated almost immediately to the cross-region OSS bucket. MAA testing observed less than a 5 minute lag, although the lag may vary from region to region due to latency and bandwidth. So, the overall RPO is essentially a few minutes higher than the last successful archive log backup, which is set by default to be every 30 minutes.

ExaDB-D supports the use of a customer-defined Object Storage Service (OSS) bucket for backups. A customer-created and managed OSS bucket for an ExaDB-D database is only applicable for user-configured backups. User-configured backups leverage the dbaascli utility for configuration and management of the backup. User-configured backups cannot be enabled, configured, or displayed using the Cloud Console, SDK, or the REST API endpoints. Although user-configured backups are set up independently of the console, they are fully supported using the dbaascli utility. User-configured backups follow MAA Backup and Recovery best practices, which are fully integrated into the dbaascli utility.

The backup retention window of the source database is automatically maintained on the remote bucket. As incremental backup pieces are obsoleted and deleted by RMAN, OSS will automatically delete the corresponding piece in the remote region. The amount of storage required in the remote region mirrors the amount of storage consumed by the backups of the source database.

Recovery Time Objective (RTO) is influenced by the steps required on the remote region at the time of the disaster. RTO can be reduced by leveraging an existing ExaDB-D that is maintained and patched to the same version as the source. The disaster recovery steps defined within this document leverage RMAN's restore and recover commands and are parallelized across all ExaDB-D nodes. Parallelization of channels across the cluster can reduce RTO by lowering the time needed to complete the restore and recover phase of the Disaster Recovery plan. The following MAA collateral provides channel performance and throughput recommendations: ExaDB-D Database Backup and Restore with Object Storage Performance Observations

OSS provides Console, CLI, SDKs, and REST API access to several replication metrics. However, these metrics do not provide a real-time estimate of the object count and sync times. The last replicated sync time provides the most beneficial value for determining RPO. **Any successful backup completed before the replicated sync time is guaranteed to be replicated to the remote region**. See the Monitoring Replication section within this document for more details,

If your database's RTO and RPO requirements are satisfied, using the OSS Cross-Region Replication Disaster Recovery Solution described within this technical brief is a viable choice. For RPO < 10 seconds and RTO of seconds to few minutes, consider Cloud Data Guard implementation. Also refer to Oracle Maximum Availability Architecture in Exadata Cloud Database Systems

Security Requirements

Databases created on OCI automatically enable Transparent Data Encryption (TDE) to ensure data is encrypted at rest. Backups generated using dbaascli use RMAN encryption/compression to ensure TDE datafiles are decrypted, compressed, and then re-encrypted before being transported to OSS. ExaDB-D supports both Oracle Managed keys

and Customer Manager keys for storing the encryption keys. Note that this technical brief is only validated with Oracle Managed keys (file-based Oracle wallets). The dbaascli utility automatically backs up the **ewallet.p12** on a daily basis at part of the incremental level0/1 backup. Note, dbaascli does not include the **cwallet.sso** as part of the wallet backup workflow.

Disaster Recovery Overview

This technical brief is divided into four areas:

1. Initial configuration of Oracle databases for cross-region OSS replication
2. Monitoring cross-region OSS replication and potential data loss
3. Restore and recovery of the database in a disaster scenario due to loss of primary region
4. Configuring backups for the newly restored database

Initial Configuration for Cross-Region Replication

This section describes the necessary steps to configure a database for user-configured backups and OSS replication.

1. Enable OSS bucket replication *BEFORE* you do the backup configuration.

   If you enabled replication *AFTER* backup configuration, it will be necessary to take a new Level 0 backup. This is required because **enabling replication will not replicate already existing backups**, only new backups. See Appendix B for details.

2. Set up the OSS buckets and replication, See OSS documentation for information about how to create the OSS buckets and enabling replication.
3. Configure the database for backups. Follow the steps in the user-configured backups section of the ExaDB-D documentation.
4. Gather information needed for Disaster Recovery.

   While you perform the above configurations, gather the information listed in the following tables to use in the sections that follow, and update as necessary. Having this information updated and available will lower the RTO if a disaster occurs.

   Note: In some documentation the word CONTAINER is used to indicate an OSS Bucket.

Table 1.

OSS DATA	LOCAL REGION	REMOTE REGION
REGION NAME	<LOCAL_REGION_NAME>	<REPL_REGION_NAME>
COMPARTMENT	<LOCAL_REGION_COMPARTMENT>	<REPL_REGION_COMPARTMENT>
SWIFT_URL	<LOCAL_REGION_SWIFT_URL>	<REPL_REGION_SWIFT_URL>
OPC_CONTAINER	<LOCAL_REGION_BUCKET>	<REPL_REGION_BUCKET>
OSS_USER	<LOCAL_REGION_OSS_USERNAME>	<REPL_REGION_OSS_USERNAME>
OSS_PASSWORD	<LOCAL_REGION_OSS_PASSWORD>	<REPL_REGION_OSS_PASSWORD>

Gather this information from the source database:

database name	<DB_NAME>
database unique name	<DB_UNIQUE_NAME>
hostname node 1	<SOURCE_HOSTNAME_NODE1>
database id	<DBID>
autologin wallet password	<EWALLET_PASSWORD>

Monitoring Cross-Region Replication

Replication status and progress can be monitored either on the local region's Console or using other supported access methods described in OSS replication documentation

The source bucket's Replication Policy section on the Console provides the following information:

- **Last Replicated**

  Indicates the last time when all objects in the source bucket are known to be successfully replicated to the destination. Any Backups inserted to the source bucket before Last Replicated are known to be successfully replicated.
  Note: This value is refreshed with a frequency of 5-6 min.

- **Replication Status**

  The replication status of the policy.

  The correct status is Active. If the status is not Active review the console for more details.

Disaster Recovery Steps after Primary Region Failure

The Source database in this section is the database configured in the initial configuration above.

The Target is the new database on the replicated backups of the source database.

Step 1: Create a new database (target) using Cloud Console

- Create the database with the same <DB_NAME> and <DB_UNIQUE_NAME> as the source database.
- Use the value PDBTEMP for the database's PDB. This PDB is deleted as part of the Post Restore Task.
- Select the same database version used by the source database. The Oracle Home directory can be different.
- TDE must be Oracle Managed. The configuration exemplified in this document does not support Customer Managed keys.
- Do not configure backup for the database. Enabling backups is covered in step h of the Post-database recovery section.
- Use the same admin password as the source database.
  This is not mandatory; the steps in the post duplication tasks cover password changes post cloning
- Regardless of the number of PDBs in the source database, create only one PDB, called PDBTEMP.
  After the recovery is completed, the console automatically discovers all the PDBs.
- To create the database, refer to Using the Console to Create a Database for instructions.

Step 2: Install the cloud backup module

Download and install opc_installer

Download opc_installer.zip from https://www.oracle.com/database/technologies/oracle-cloud-backup-downloads.html and copy to a temporary location on node 1.

As the oracle user, create a directory to copy and extract the opc_installer.zip file:

```
$ mkdir /home/oracle/tmp<DB_NAME>/down_opc_installer
$ cd /home/oracle/tmp<DB_NAME>/down_opc_installer
$ unzip opc_installer.zip
```

Run opc_install.jar

Use the directory structure as shown in the following example below for the -libDir, -configfile and –walletDir options. These options leverage the Oracle Automatic Storage Management Cluster File System (Oracle ACFS), allowing all nodes file access. Failure to places these files on ACFS will cause other nodes in the cluster to fail during restore.

```
$ cd /home/oracle/tmp<DB_NAME>/down_opc_installer/opc_installer/opc_installer
$ java -jar opc_install.jar -host <REPL_REGION_SWIFT_URL> -opcid '<REPL_REGION_OSS_USERNAME>' -container
'<REPL_REGION_BUCKET>' -opcPass '<REPL_REGION_OSS_PASSWORD>' -libDir
/var/opt/oracle/dbaas_acfs/<DB_NAME>/opc -configfile
/var/opt/oracle/dbaas_acfs/<DB_NAME>/opc/opc<DB_NAME>.ora -walletDir
/acfs01/dbaas_acfs/<DB_NAME>/opc/opc_wallet
```

Step 3: Prepare target database for restore

Preparing for restore consists of:

Take note of the following init.ora parameters for the database created in Step 1 above. These values will be needed later to update the restored init.ora parameter in Step 6.b below.

remote_listener	show parameter remote_listener
db_domain	show parameters db_domain
cluster_interconnects Note: Each instance will have a different value for cluster_interconnects	select name, inst_id, value from gv$parameter where name='cluster_interconnects';

Remove the files for the cloud database that was created in step 1.
Create the following SQL script build_delete_ASM_files.sql as oracle user

```
$ cat /home/oracle/tmp<DB_NAME>/build_delete_ASM_files.sql
set head off
set echo off
set term off
set feed off
set linesize 999
set pagesize 0
set trimspool on
column value for a120
column name for a100
column HANDLE for a100
spool /tmp/delete_ASM_files.sh
select 'asmcmd rm '||name from v$datafile union all select 'asmcmd rm '||name from v$tempfile union all
select 'asmcmd rm '||member from v$logfile union all select 'asmcmd rm '||name from v$archived_log where
name is not null union all select 'asmcmd rm '||name from  v$controlfile union all select 'asmcmd rm
'||ltrim(UPPER(value),'TO ') from  V$RMAN_CONFIGURATION where NAME='SNAPSHOT CONTROLFILE NAME' union all
select 'asmcmd rm '||handle from v$backup_piece where DEVICE_TYPE='DISK' and handle is not null union all
select 'asmcmd rm '||NAME FROM V$FLASHBACK_DATABASE_LOGFILE union all select 'asmcmd rm '||value from
v$parameter where NAME='spfile';
select 'asmcmd ls -lt ' || value || '/' ||(select DB_UNIQUE_NAME from v$database) from v$parameter where
name='db_create_file_dest';
select 'asmcmd ls -lt ' || value || '/' ||(select DB_UNIQUE_NAME from v$database) from v$parameter where
name='db_recovery_file_dest';
spool off;
exit
```

As oracle user, in SQL*Plus, run the script build_delete_ASM_files.sql.
The SQL generates a shell script, /tmp/delete_ASM_files.sh

```
$ sqlplus / as sysdba
SQL> @ build_delete_ASM_files.sql
```

Change permissions on the file /tmp/delete_ASM_files.sh.

```
# chmod 774 /tmp/delete_ASM_files.sh
```

Stop the database with the abort option

```
$ srvctl stop database -d <DB_UNIQUE_NAME> -o abort
```

Remove the data files, redo logs, archive logs, and control files by running the script /tmp/delete_ASM_files.sh

As grid user execute the script /tmp/delete_ASM_files.sh.

```
$(grid) /tmp/delete_ASM_files.sh > /tmp/delete_ASM_files.log
```

Verify the output of the script execution on the screen or in the log file.

Step 4: Restore TDE wallets and Oracle Net Configuration

Restoration of the TDE wallet consists of:

- Locating and downloading the most recent backup of the source database configuration files: ohcfgfiles_<YYYYMMDD_HHMI>.tar.gz
- Replacing target's current TDE wallet with the TDE wallet from the source database
- Creating cwallet.sso

Locate the most recent version of the file ohcfgfiles_<YYYYMMDD_HHMI>.tar.gz in the bucket. This file contains the ewallet.p12, tnsname.ora, and sqlnet.ora of the source database. These files are backed up as part of the daily incremental backup on the source database. Locate the file with the most recent date/time.

Three methods can be used to locate and download the file from the replicated OSS bucket. You can use one of the methods described below.

Create the tmp staging directory and cd into it.

```
$ mkdir /home/oracle/tmp<DB_NAME>/OHCFGFILES
$ cd /home/oracle/tmp<DB_NAME>/OHCFGFILES
```

Method1: Using the CP

From Console go to **Object Storage & Archive Storage** > Bucket Details for the Replicated Bucket.
Display objects on section **<SOURCE_HOSTNAME_NODE1>-<DBID>-dbaastools-regular** and find the most recent object with name ohcfgfiles_<YYYYMMDD_HHMI>.tar.gz

Use download feature from the kebab menu (three vertical dots ⋮) The kebab menu is located to the right of the file.

Method 2: Using oci-cli

Use the *os object list* command. Use the following options to the command:

bucket-name	<REPL_REGION_BUCKET>
prefix	<SOURCE_HOSTNAME_NODE1>-<DBID>-dbaastools-regular/ohcfgfiles
output	Table

```
List object using oci-cli
```

```
oci os object list --bucket-name <REPL_REGION_BUCKET> --prefix <SOURCE_HOSTNAME_NODE1>-<DBID>-dbaastools-regular/ohcfgfiles --output table
```

Determine the file with the latest <YYYYMMDD_HHMI> from list generated above and download

```
Download the specific file:
oci os object get --bucket-name <REPL_REGION_BUCKET> --file ohcfgfiles_<YYYYMMDD_HHMI>.tar.gz "--name <SOURCE_HOSTNAME_NODE1>-<DBID>-dbaastools-regular/ohcfgfiles_<YYYYMMDD_HHMI>.tar.gz"
```

Method3: Using curl

```
Using curl:
curl -u '<REPL_REGION_OSS_USERNAME>:<REPL_REGION_OSS_PASSWORD>'
<REPL_REGION_SWIFT_URL>/<REPL_REGION_BUCKET>?prefix=<SOURCE_HOSTNAME_NODE1> | python -m json.tool | grep -i ohc
```

Determine the file with the latest <YYYYMMDD_HHMI> from list generated above and download

```
Download the specific file: Use the "name:" value listed in output above for <NAME>
curl -u '<REPL_REGION_OSS_USERNAME>:<REPL_REGION_OSS_PASSWORD>' -X GET
<REPL_REGION_SWIFT_URL>/<REPL_REGION_BUCKET>/<NAME> -o ohcfgfiles_<YYYYMMDD_HHMI>.tar.gz
```

Extract the tar.gz file downloaded

As oracle user, copy the file ewallet.p12 from the extract of the ohcfgfiles_<YYYYMMDD_HHMI>.tar.gz file.

```
Depending on the version: The wallet_root/tde directory is tde_wallet
$ cd /home/oracle/tmp<DB_NAME>/OHCFGFILES
$ tar -xvf ohcfgfiles_<YYYYMMDD_HHMI>.tar.gz
$ rm -rf /var/opt/oracle/dbaas_acfs/<DB_NAME>/wallet_root/tde/*
$ cp /home/oracle/tmp<DB_NAME>/OHCFGFILES/var/opt/oracle/dbaas_acfs/<DB_NAME>/wallet_root/tde/ewallet.p12 /var/opt/oracle/dbaas_acfs/<DB_NAME>/wallet_root/tde/
```

Create the autologin cwallet.sso file.

You must supply the autologin password as oracle user

```
Create cwallet
$ orapki wallet create -wallet /var/opt/oracle/dbaas_acfs/<DB_NAME>/wallet_root/tde -pwd "<EWALLET_PASSWORD>" -auto_login
```

Step 5: Verify tnsnames.ora and sqlnet.ora

The ohcfgfiles__<YYYYMMDD_HHMI>.tar.gz also contains the tnsnames.ora and sqlnet.ora files from the source database. These files can help create the service tnsnames endpoints.

The location for these files will be:
/home/oracle/tmp<DB_NAME>/OHCFGFILES/<SOURE_ORACLE_HOME>/network/admin/<DB_NAME>

Compare the source tnsnames.ora and sqlnet.ora files with the existing ones for the target database

If necessary, you can modify or add information to the existing files, for example you can add customer services to connect the application to other databases.

Step 6: Restore and recover the target database

Do the following steps to restore and recover the target database from the source backups.

a) Restore the spfile as pfile from autobackup (RMAN).

b) Shut down / Start up nomount the database with the pfile (SQL*Plus)

c) Restore the control file from autobackup (RMAN)

d) Update the database parameter values in the pfile, for example: control_files, remote_listener (SQL*Plus)

e) Create the spfile from the pfile and update the database cluster resource (SQL*Plus)

f) Catalog the backup pieces

g) Validate the Restore (RMAN)

h) Restore/Recover the database to the latest

i) Open the database with the resetlogs option (SQL*Plus)

a) Restore spfile to pfile from autobackup.

```
$ rman target /
RMAN> startup force nomount;
RMAN> set dbid=<DBID>;
RMAN> run
{
allocate channel ch1 device type 'SBT_TAPE' parms
"SBT_LIBRARY=/var/opt/oracle/dbaas_acfs/<DB_NAME>/opc/libopc.so,
ENV=(OPC_PFILE=/var/opt/oracle/dbaas_acfs/<DB_NAME>/opc/opc<DB_NAME>.ora)";
RESTORE SPFILE TO PFILE '<ORACLE_HOME>/dbs/init<DB_NAME>.ora' FROM AUTOBACKUP;
}
```

b) Shutdown the database, update the pfile's database parameters and startup nomount

Shutdown database

```
$ sqlplus / as sysdba
SQL> shutdown;
```

Update the pfile. The following pfile parameters must be updated, use the values captured in Step 3.

db_domain	Change to the current client subnet
remote_listener	Change to the current SCAN hostname
cluster_interconnects	Note: each instance will have a different set of values

Edit the restored pfile at <ORACLE_HOME>/dbs/init<DB_NAME>.ora and update the required parameters listed above.

Startup nomount using the pfile

```
$ sqlplus / as sysdba
SQL> startup nomount pfile='<ORACLE_HOME>/dbs/init<DB_NAME>.ora';
```

c) Restore the control file from autobackup

The control file will be restored to the ASM diskgroup indicated in the pfile.
Take note of the control file name restored; it will be used in the next step

```
$ rman target /
RMAN> set dbid=<DBID>;
RMAN> run
{
```

```
allocate channel ch1 device type 'SBT_TAPE' parms
"SBT_LIBRARY=/var/opt/oracle/dbaas_acfs/<DB_NAME>/opc/libopc.so,
ENV=(OPC_PFILE=/var/opt/oracle/dbaas_acfs/<DB_NAME>/opc/opc<DB_NAME>.ora)";
RESTORE CONTROLFILE FROM AUTOBACKUP;
}
```

d) Update the pfile's database parameters with new control file value

The following pfile parameters must be updated:

control_file	Use the name generated from the restore control file command above

Edit the restored pfile at <ORACLE_HOME>/dbs/init<DB_NAME>.ora and update the required parameters listed above.

e) Create the spfile and update the database cluster resource with the spfile directory/name: <SPFILE_VAL>.

```
Create spfile and shutdown database
$ sqlplus / as sysdba
SQL> create spfile from pfile='<ORACLE_HOME>/dbs/init<DB_NAME>.ora';
Shutdown database
SQL> shutdown immediate
```

To determine the value of <SPFILE_VAL>, use *asmcmd ls -lt* to locate the file linked from +<DATA_DG>/spfile<DB_NAME>1.ora

The spfile format looks like:
+<DATA_DG>/<DB_UNIQUE_NAME>/PARAMETERFILE/spfile.<num>.<inc>

Run as grid user:

```
# asmcmd ls -lt +<DATA_DG>/spfile<DB_NAME>1.ora
```

Modify the database cluster by setting the spfile to the location determined above. Once modified, start the database in mount mode

```
Modify database cluster resources
$ srvctl modify database -d <DB_UNIQUE_NAME> -spfile <SPFILE_VAL>
Mount database
$ srvctl start database -d <DB_UNIQUE_NAME> -o mount
```

f) Disable block change tracking and flashback database

```
# sqlplus / as sysdba
SQL> alter database disable block change tracking;
SQL> alter database flashback off;
```

g) Catalog the backup pieces

Catalog any pieces backed up in the OSS

Using an editor, create the following script, catalog_pieces.sh, as oracle user, set the executable permission, and run the script, passing in the proper arguments.

```
#@/bin/bash
if (($# != 5)); then
  echo "usage: $0 odbsrmt_dir_location oss_url oss_username oss_password oss_bucket"
  exit
fi
dir_loc=$1
[[ -d ${dir_loc} ]] || { echo "${dir_loc} is not a directory or does not exist"; exit; }
oss_url=$2; oss_username=$3;oss_password=$4;oss_bucket=$5
[[ -f ${dir_loc}/odbsrmt.py  ]] || { echo "odbsrmt.py not in ${dir_loc}"; exit; }
sqlplus -s / as sysdba << EOT  > /dev/null
exit;
EOT
if [[ $? -ne 0 ]] ; then
  echo "unable to run sqlplus"
  exit
fi
rm -f ${dir_loc}/rep_report
python2 ${dir_loc}/odbsrmt.py --mode report --ocitype swift --host ${oss_url} --dir ${dir_loc} --credential
${oss_username}/${oss_password} --container ${oss_bucket} --dbid 0 --thread 32 --format text --forcename
rep_report || { echo "odbsrmt.py failed from ${dir_loc}"; exit; }
sed -i 'N;s/\n/ /' ${dir_loc}/rep_report || { echo "sed failed against ${dir_loc}"; exit; }
rm -f ${dir_loc}/ctl_info.dat
sqlplus -s / as sysdba << EOT > /dev/null
set head off
set echo off
set term off
set feed off
set linesize 999
set pagesize 0
set trimspool on
spool ${dir_loc}/ctl_info.dat
select handle from v\$backup_piece where handle like 'c-%' and handle not like 'c-%-dbaastools' order by 1;
spool off;
exit;
EOT
if [[ $? -ne 0 ]] ; then
       echo "unable to run sqlplus to gather controlfile details"
       exit
fi
ctl_line=$( tail -2 ${dir_loc}/ctl_info.dat | head -1 )
ctl_line=$(echo $ctl_line)
IFS=$'\n' read -r -d '' -a cat_array < <( tail -n +2  ${dir_loc}/rep_report |  sort -k6,7 | awk
"/$ctl_line/"',EOF' | awk '{print $1}' )
len=${#cat_array[@]}
rm -f  ${dir_loc}/catlist.rman
printf '%s' "catalog device type 'sbt_tape' backuppiece " > ${dir_loc}/catlist.rman
for catalog_piece in ${cat_array[@]:1:$len - 2}; do
       printf '%s\n' "'${catalog_piece}'," >> ${dir_loc}/catlist.rman
done
printf '%s\n' "'${cat_array[$len-1] }';" >> ${dir_loc}/catlist.rman
```

Set permission as oracle user:

```
$chmod 755 /home/oracle/tmp<DB_NAME>/catalog_pieces.sh
```

Parameters to use in the catalog_pieces.sh script

Parameter position	Use
1	/var/opt/oracle/dbaas_acfs/<DB_NAME>/opc
2	<REPL_REGION_SWIFT_URL>
3	<REPL_REGION_OSS_USERNAME>
4	<REPL_REGION_OSS_PASSWORD>
5	<REPL_REGION_BUCKET>

Run the script with the required parameters as oracle user:
Note: Depending on the number of objects in the replicated bucket, this script can run for several minutes.

```
$ /home/oracle/tmp<DB_NAME>/catalog_pieces.sh <p1> <p2> <p3> <p4> <p5>
```

Run the rman script catlist.rman generated as oracle user

```
$ rman target /
RMAN> @<dir_loc>/catlist.rman
```

h) Validate database restore

See Appendix A if point in time recovery (PITR).

```
$ rman target /
RMAN> run
{
allocate channel stb1 device type 'SBT_TAPE' parms
"SBT_LIBRARY=/var/opt/oracle/dbaas_acfs/<DB_NAME>/opc/libopc.so,
ENV=(OPC_PFILE=/var/opt/oracle/dbaas_acfs/<DB_NAME>/opc/opc<DB_NAME>.ora) ";
RESTORE DATABASE PREVIEW VALIDATE HEADER;
}
```

i) Restore and recover the target database

Restore and Recover database is done by spreading RMAN channels among all Oracle RAC instances: This can be achieved under the following conditions:

- RMAN target connection is done through the TNS alias using the SCAN
- RMAN SBT channels are allocated among all the cluster instances through the SCAN

```
rman target sys/<SYS_PASSWORD>@<DB_UNIQUE_NAME>
run
{
allocate channel sbt1 device type 'SBT_TAPE' parms
"SBT_LIBRARY=/var/opt/oracle/dbaas_acfs/<DB_NAME>/opc/libopc.so,
ENV=(OPC_PFILE=/var/opt/oracle/dbaas_acfs/<DB_NAME>/opc/opc<DB_NAME>.ora)";
```

```
allocate channel sbt2 device type 'SBT_TAPE' parms
"SBT_LIBRARY=/var/opt/oracle/dbaas_acfs/<DB_NAME>/opc/libopc.so,
ENV=(OPC_PFILE=/var/opt/oracle/dbaas_acfs/<DB_NAME>/opc/opc<DB_NAME>.ora)";
allocate channel sbt<n> device type 'SBT_TAPE' parms
"SBT_LIBRARY=/var/opt/oracle/dbaas_acfs/<DB_NAME>/opc/libopc.so,
ENV=(OPC_PFILE=/var/opt/oracle/dbaas_acfs/<DB_NAME>/opc/opc<DB_NAME>.ora)";
RESTORE DATABASE;
RECOVER DATABASE UNTIL AVAILABLE REDO;
release channel sbt1;
release channel sbt2;
release channel sbt<n>;
}
```

j) Open the database with resetlogs (SQLPLUS)

a. Shut down database (SRVCTL)

b. Start up MOUNT (SQL*Plus)

c. Open database with RESETLOGS

Only one instance can be started for opening the database in RESETLOG.

```
$ srvctl stop database -d <DB_UNIQUE_NAME>
$ sqlplus / as sysdba
SQL> startup mount;
SQL> alter database open resetlogs;
SQL> exit
$ start database -d <DB_UNIQUE_NAME>
```

Step 7: Post-database recovery steps

a) Perform the following post wallet tasks on the database

TDE autologin password (Mandatory) (dbaascli)

Run the following to ensure the TDE password needed by the automation is synchronized across its metadata.

```
dbaascli tde changepassword --dbname <DB_NAME>
```

Provide the TDE autologin password for the production system:
Enter the same password for each prompt: old, new, and re-enter.

Optional: You can use the same command to change the TDE password for the TDE autologin wallet.

TDE masterkey rotation (optional) (dbaascli)

Use the following command to change the master key

```
dbaascli tde rotate masterkey --dbname <DB_NAME>
```

Validating the TDE wallet (mandatory) (dbaascli)

Run the following command to validate the wallet

```
dbaascli tde status --dbname <DB_NAME>
```

b) Check and add temporary TEMP tablespace files

After opening the restored database, check existing temporary tablespaces and add temporary files needed.

c) Verify PDBs status

PDBs should be mounted after starting database services. Mount them manually if necessary.

```
$ sqlplus / as sysdba
SQL> select INST_ID,CON_ID,NAME,OPEN_MODE from gv$pdbs order by INST_ID,CON_ID;
SQL> alter pluggable database all OPEN instances=all;
```

d) PDB Metadata in Control Plane

For the control plane to display the proper PDB information, remove the initial PDB created: PDBTEMP.

The console will show PDBTEMP in a failed state. Using the console, select the Delete option from the kebab menu (three vertical dots ⋮) to the right of the PDB.

Typically, the Control Plane Metadata is synchronized several minutes after the database is restored. However, due to the synchronization process, the delay could be several hours.

e) PDB Services (Check resources in Cluster)

On the target database check for PDBs and PDBs services and adjust the status of the PDBs.

On the database, services and PDBs can be checked by running:

```
SQL> SELECT NAME, PDB FROM V$SERVICES ORDER BY PDB,NAME;
```

On the cluster, database and PDBs services resources can be checked by running:

```
$ srvctl config service database -d <DB_UNIQUE_NAME>
```

Check if the database Cluster Resources has services for each PDB: <PDB_NAME>.paas.oracle.com
If necessary, add any PDB service by running:

```
Add service
$ srvctl add service -db <DB_UNIQUE_NAME> -service <PDB_NAME>.<domain> -pdb <PDB_NAME> -role PRIMARY -
preferred <instance1,instance2,.. , instanceN>
```

Then start the new PDB service by running:

```
$ srvctl start service -db <DB_UNIQUE_NAME>
$ srvctl status service -db <DB_UNIQUE_NAME>
```

f) Enable Block Change Tracking (BCT)

Check if BCT status on the restored database is enabled. If not, enable it.

To check BCT status on the database

```
SQL> select * from v$block_change_tracking;
```

To enable BCT on database:

```
SQL> alter database enable block change tracking;
```

g) Enable Flashback Database

Check if FLASHBACK status on the restored database is enabled. If disabled, enable it.

To check FLASHBACK database:

```
SQL> SELECT FLASHBACK_ON FROM V$DATABASE
```

To enable FLASHBACK database:

```
$srvctl stop database -d <DB_UNIQUE_NAME> -o immediate
$sqlplus / as sysdba
SQL> startup mount
SQL> alter database flashback on;
SQL> shutdown immediate;
$srvctl start database -d <DB_UNIQUE_NAME>
```

h) Configure Backups for Newly Restored Database

Configuring the target database to back up to the same OSS bucket location allows the newly restored database to use the previous backups. This is possible because the restored database has the same DBID as the source database.

To configure the database to backup, the bucket must first be enabled for read/write.

To configure backups:
1) Enable the <REPL_OSS_BUCKET> as read/write by following the stop replication section of the OSS doc.

2) Follow the steps in the user-configured backups section of the ExaDB-D documentation, use the values for <REMOTE_REGION_*>. See table 1. in the Initial Configuration for Cross-Region Replication section of this guide.

Conclusion

If you have one database, or you are consolidating a lot of databases into one or multiple ExaDB-D systems, you can leverage Oracle's Object Storage cross-region replication for a low cost disaster recovery solution with minimal data loss or Recovery Point Objective (RPO).

Appendix A: Point in time recovery to timestamp

a) Point in time recovery

TIMESTAMP: <TIMESTAMP_VAL>
If you need to recover the database to a point in time, provide the appropriate TIMESTAMP_VAL.

Note that the <TIMESTAMP_VAL> value supplied is relative to source database's time zone. For example, if the source database region is EST, and the replicated region is PST, the timestamp value for PITR on the replicated region would be in EST.

b) PITR Option: Validate restore database Preview Validate Header

TIMESTAMP format:

```
$ rman target /
RMAN> run
{
set until time "to_date('<TIMESTAMP_VAL>','YYYY-MM-DD HH24:MI:SS')";
allocate channel sbt1 device type 'SBT_TAPE' parms
"SBT_LIBRARY=/var/opt/oracle/dbaas_acfs/<DB_NAME>/opc/libopc.so,
ENV=(OPC_PFILE=/var/opt/oracle/dbaas_acfs/<DB_NAME>/opc/opc<DB_NAME>.ora)";
RESTORE DATABASE PREVIEW VALIDATE HEADER;
}
```

c) PITR Option: Restore and recover target database

If Restore database preview finishes successfully, then restore and recover the database using the same <TIMESTAMP_VAL>.

Restore and Recover database is done by spreading RMAN channels among all Oracle RAC instances: This can be achieved under the following conditions:

- RMAN target connection is done through the TNS alias using SCAN
- RMAN SBT channels are allocated among all of the cluster instances through SCAN

```
If Restore database preview finishes successfully then restore and recover the database using the same
<TIMESTAMP_VAL>.
$ rman target /
RMAN> run
{
set until time "to_date('<TIMESTAMP_VAL>','YYYY-MM-DD HH24:MI:SS')";
allocate channel sbt1 device type 'SBT_TAPE' parms
"SBT_LIBRARY=/var/opt/oracle/dbaas_acfs/<DB_NAME>/opc/libopc.so,
ENV=(OPC_PFILE=/var/opt/oracle/dbaas_acfs/<DB_NAME>/opc/opc<DB_NAME>.ora)";
allocate channel sbt2 device type 'SBT_TAPE' parms
"SBT_LIBRARY=/var/opt/oracle/dbaas_acfs/<DB_NAME>/opc/libopc.so,
ENV=(OPC_PFILE=/var/opt/oracle/dbaas_acfs/<DB_NAME>/opc/opc<DB_NAME>.ora)";
allocate channel sbt<n> device type 'SBT_TAPE' parms
"SBT_LIBRARY=/var/opt/oracle/dbaas_acfs/<DB_NAME>/opc/libopc.so,
ENV=(OPC_PFILE=/var/opt/oracle/dbaas_acfs/<DB_NAME>/opc/opc<DB_NAME>.ora)";
RESTORE DATABASE;
RECOVER DATABASE;
release channel sbt1;
release channel sbt2;
release channel sbt<n>;
}
```

Continue with Step 6.i) Open database with resetlogs.

```
rman target sys/<SYS_PASSWORD>@<DB_UNIQUE_NAME>
run {
set until time "to_date('<TIMESTAMP_VAL>','YYYY-MM-DD HH24:MI:SS')";
allocate channel sbt1 device type 'SBT_TAPE' parms
"SBT_LIBRARY=/var/opt/oracle/dbaas_acfs/<DB_NAME>/opc/libopc.so,
ENV=(OPC_PFILE=/var/opt/oracle/dbaas_acfs/<DB_NAME>/opc/opc<DB_NAME>.ora)";
allocate channel sbt2 device type 'SBT_TAPE' parms
"SBT_LIBRARY=/var/opt/oracle/dbaas_acfs/<DB_NAME>/opc/libopc.so,
ENV=(OPC_PFILE=/var/opt/oracle/dbaas_acfs/<DB_NAME>/opc/opc<DB_NAME>.ora)";
allocate channel sbt<n> device type 'SBT_TAPE' parms
"SBT_LIBRARY=/var/opt/oracle/dbaas_acfs/<DB_NAME>/opc/libopc.so,
ENV=(OPC_PFILE=/var/opt/oracle/dbaas_acfs/<DB_NAME>/opc/opc<DB_NAME>.ora)";
RESTORE DATABASE;
RECOVER DATABASE UNTIL AVAILABLE REDO;
release channel sbt1;
release channel sbt2;
release channel sbt<n>;
}
```

Appendix B: Complete Level 0

If replication is enabled **AFTER the local region is backing up**: NONE of the existing backup pieces will be replicated; however, any backup taken after enabling replication will be replicated. Because of this behavior, a COMPLETE Level 0 Database Backup is required. This backup MUST include ALL read-only tablespaces.

Disabling backup optimization, taking a level 0 with dbaascli, and then enabling backup optimization will satisfy the above requirements.

You cannot restore from the replicated bucket before the date/time of enabling replication and issuing a complete level 0.

As oracle user:

```
rman target /
RMAN> CONFIGURE BACKUP OPTIMIZATION off;
exit
```

As root user: run the following command

```
dbaascli database backup --dbname <value> --start --level 0
```

As oracle user

```
rman target /
RMAN> CONFIGURE BACKUP OPTIMIZATION on;
exit
```

Connect with us

Call +**1.800.ORACLE1** or visit **oracle.com**. Outside North America, find your local office at: **oracle.com/contact**.

blogs.oracle.com facebook.com/oracle twitter.com/oracle