ORACLE

Oracle Zero Downtime Migration – Logical Offline Migration to ADB-S on Oracle Database@Google Cloud

Technical Brief

November, 2024, Version [1.0]

Purpose statement

This document provides an overview of features and enhancements included in ZDM 21.5. It is intended solely to help you assess the business benefits of upgrading to ZDM 21.5 and planning for the implementation and upgrade of the product features described.

Disclaimer

This document in any form, software or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle software license and service agreement, which has been executed and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced or distributed to anyone outside Oracle without prior written consent of Oracle. This document is not part of your license agreement nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

This document is for informational purposes only and is intended solely to assist you in planning for the implementation and upgrade of the product features described. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, timing, and pricing of any features or functionality described in this document remains at the sole discretion of Oracle. Due to the nature of the product architecture, it may not be possible to safely include all features described in this document without risking significant destabilization of the code.

ORACLE

Table of contents

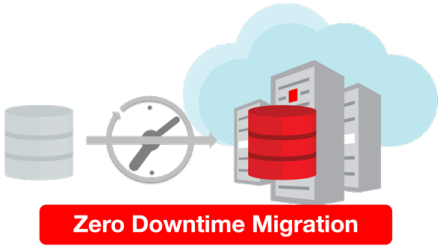


Figure 1. The Oracle Zero Downtime Migration Logo comprises a Database and a Clock with an arrow pointing to a Database deployed in the Cloud.

Purpose

Oracle customers are rapidly increasing their workload migration into the Oracle Cloud, Engineered Systems, and Oracle Database@Google Cloud. However, migrating workloads has been a source of challenges for many years. Migrating database workloads from one system to another or into the Cloud is easier said than done.

Based on years of experience migrating Oracle workloads, Oracle has developed Zero Downtime Migration (ZDM). ZDM is Oracle's premier solution for a simplified and automated migration experience, providing zero to negligible downtime for the production system depending on the migration scenario. ZDM allows you to migrate your on-premises Oracle Databases directly and seamlessly to and between Oracle Database@Azure, Oracle Database@ Google Cloud, Oracle Database@AWS, and any Oracle-owned infrastructure, including Exadata Database Machine On-Premises, Exadata Cloud at Customer, and Oracle Cloud Infrastructure. Oracle ZDM supports a wide range of Oracle Database versions and, as the name implies, ensures minimal to no production database impact during the migration.

ZDM follows Oracle Maximum Availability Architecture (MAA) principles[1] and incorporates products such as GoldenGate and Data Guard to ensure High Availability and an online migration workflow that leverages technologies such as the Recovery Manager, Data Pump, and Database Links.

This technical brief is a step-by-step guide for migrating your on-premises Oracle Databases to Oracle Autonomous Database Serverless (ADB-S) on Oracle Database@Google Cloud, with ZDM's Logical Offline workflow.

Oracle ZDM will run on a separate node and connect to Source and Target to perform the migration. This guide will cover all requirements for installing the Oracle ZDM Service Host, the Source Database, the Target Database recipient of the migration process, and the networking used. The migration process will be dissected and done in a step-by-step fashion. This guide will answer the most frequently asked questions regarding the product and the overall migration process.

For more information on Oracle Zero Downtime Migration, please visit ZDM's product website and Oracle Database@Google Cloud product website.[2]

[1] https://oracle.com/goto/maa
[2] https://www.oracle.com/goto/zdm
https://www.oracle.com/cloud/google/oracle-database-at-google-cloud/

ORACLE

Zero Downtime Migration

Oracle Zero Downtime Migration (ZDM) is the Oracle Maximum Availability Architecture (MAA)-recommended solution to migrate Oracle Databases to the Oracle Cloud. ZDM's inherent design keeps in mind the migration process as straightforward as possible to ensure the most negligible impact on production workloads. The Source Database to be migrated can be on-premises, deployed on Oracle Cloud Infrastructure, or a 3rd Party Cloud. The Target Database deployment can be on Oracle Database@Azure, Oracle Database@Google Cloud, Oracle Database@AWS, Database Cloud Service on Oracle Cloud Infrastructure (OCI) Virtual Machine, Exadata Cloud Service, Exadata Cloud at Customer, and Autonomous Database. ZDM automates the entire migration process, reducing the chance of human errors. ZDM leverages Oracle Database-integrated high availability (HA) technologies such as Oracle Data Guard and GoldenGate and follows all MAA best practices that ensure no significant downtime of production environments. Oracle ZDM supports both Physical and Logical Migration workflows. This technical brief covers a step-by-step guide for the Logical Offline Migration Workflow.

A standard Logical Offline migration will take the following steps:

1. Download and Configure ZDM.
2. ZDM starts database migration.
3. ZDM starts a data pump export job.
4. ZDM transfers dump files to the backup location.
5. ZDM starts a data pump import job and instantiates the target database.
6. ZDM switches over.
7. ZDM validates, cleans up, and finalizes the database migration.

Supported Configurations

Oracle ZDM supports Oracle Database versions 11.2.0.4, 12.1.0.2, 12.2.0.1, 18c, 19c, 21c, and 23ai. ZDM's physical migration workflow requires the Source and Target Databases to be in the same database release.

Oracle ZDM supports Source Oracle Databases hosted on Linux, Solaris, and AIX operating systems. Oracle ZDM supports single-instance databases, Oracle RAC One Node databases, or Oracle RAC databases as sources. Oracle ZDM supports Oracle Database Enterprise & Standard Edition as Source and Target Databases. ZDM's physical migration workflow supports only Source Databases hosted on Linux platforms.

Architecture

An architectural overview of the ZDM server, the source database on-premises, the target database on Oracle Autonomous Database Serverless (ADB-S) on Oracle Database@Google Cloud, and all networks and components required are described in the diagram below:

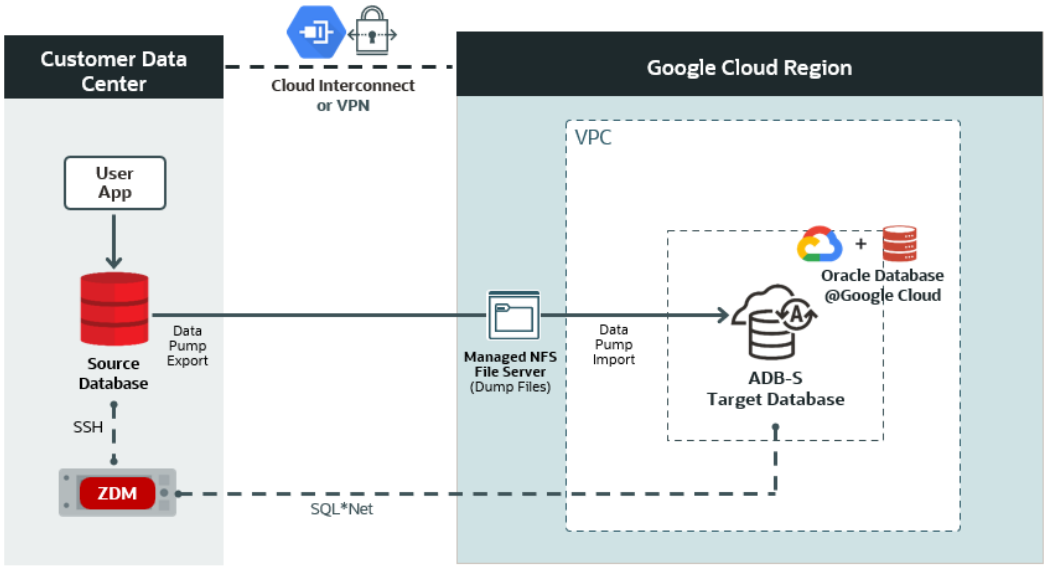


Figure 2. This is a High-Level Architectural overview showcasing the customer data center where the source database and ZDM's server reside. It also shows all connectivity to the target Oracle Autonomous Database Serverless (ADB-S) on Oracle Database@Google Cloud.

Zero Downtime Migration Service Host

Zero Downtime Migration Service Host Requirements

Oracle Zero Downtime Migration installation must take place on a separate host, which must fulfill the following requirements:

- Linux host running on Oracle 7, 8, or RHEL 8 (only these OS platforms/versions are supported).
- 100 GB of free storage space. This space is required for all the logs that ZDM will generate.
- A `zdm` group and a `zdmuser` as part of this group.
- The following packages must be installed:
  - `glibc-devel`
  - `expect`
  - `unzip`
  - `libaio`
  - `oraclelinux-developer-release-el7`
- All hostnames and IP addresses to be used must be present as entries in the `/etc/hosts` file.

For more information on the ZDM Service Host requirements and setting up ZDM on RHEL platforms, please refer to Oracle ZDM's product documentation, specifically "Setting Up Zero Downtime Migration Software" section[3].

For this step-by-step guide, the ZDM Service Host runs on-premises on an Oracle Linux Server 8.9. The host private IP is masked for this guide, but as an example we will use the fictional ***zz.dd.mm.hh*** and the hostname is ***zdmhost***.

[3] https://docs.oracle.com/en/database/oracle/zero-downtime-migration/index.html

Network and Connectivity

Google Cloud Region
A Google Cloud region is a geographical area that contains data centers and infrastructure for hosting resources. It is made up of zones that are isolated from each other within the region.

Google Cloud Project
A Google Cloud Project is required to use Google Workspace APIs and build Google Workspace add-ons or apps. A Cloud project forms the basis for creating, enabling, and using all Google Cloud services, including managing APIs, enabling billing, adding and removing collaborators, and managing permissions.

Google Virtual Private Cloud
Google Cloud Virtual Private Cloud (VPC) provides networking functionality to Compute Engine virtual machine (VM) instances, Google Kubernetes Engine (GKE) containers, database services, and serverless workloads. VPC provides global, scalable, and flexible networking for your cloud-based service.

Google Cloud Interconnect
Cloud Interconnect extends your on-premises network to the Google network through a highly available, low-latency connection. You can use Dedicated Interconnect to connect directly to Google or Partner Interconnect to connect to Google through a supported service provider.

Autonomous Database
Oracle Autonomous Database is a fully managed, preconfigured database environments that you can use for transaction processing and data warehousing workloads. You do not need to configure or manage any hardware, or install any software. Oracle Cloud Infrastructure handles creating the database and backing up, patching, upgrading, and tuning the database.

Source Database

The source database runs on-premises on an Oracle Linux Server 7.7 for this step-by-step guide. The host's private IP is masked for this guide, but as an example, we will use the fictional **aa.bb.sr.db** address, and the hostname is **onphost**.

The source Oracle database is a single-instance Enterprise Edition database version 19.22 with multitenant architecture. The Oracle SID is ***oradb***, and the PDB name is **orclpdb**.

Target Database

Oracle Database@ Google Cloud offers the following products:

- **Oracle Exadata Database Service on Dedicated Infrastructure (ExaDB-D)**
  - You can provision flexible Exadata systems that allow you to add database compute servers and storage servers to your system anytime after provisioning.

- **Oracle Autonomous Database Serverless (ADB-S)**
  - Autonomous Database provides an easy-to-use, fully autonomous database that scales elastically, delivers fast query performance, and requires no database administration.

Oracle Database@Google Cloud integrates Oracle Exadata Database Service, Oracle Real Application Clusters (Oracle RAC), and Oracle Data Guard technologies into the Google Cloud platform. The Oracle Database service runs on Oracle Cloud Infrastructure (OCI) and is co-located in Google's data centers. The service offers features and price parity with OCI.

Oracle Database@Google Cloud service offers the same low latency as other Google-native services and meets mission-critical workloads and cloud-native development needs. Users manage the service on the Google Cloud console and with Google Cloud automation tools. The service is deployed in Google Virtual Private Cloud (VPC). The service requires that users have a Google Cloud Project and an OCI tenancy.

For this step-by-step guide, the target platform is Oracle Autonomous Database Serverless (ADB-S) on Oracle Database@Google Cloud. ZDM requires configuring a placeholder database target environment before beginning the migration process.

Enhanced Security for Outbound Connections with Private Endpoints

Setting the ROUTE_OUTBOUND_CONNECTIONS database property to the value PRIVATE_ENDPOINT enforces that all outgoing connections to a target host are subject to and limited by the private endpoint's egress rules.

```
ALTER DATABASE PROPERTY SET ROUTE_OUTBOUND_CONNECTIONS = 'PRIVATE_ENDPOINT';
```

NFS File Share via Google Cloud Managed NFS File Server

ZDM Logical Offline migration workflow uses Oracle Data Pump export and import to migrate the data from the source to the target database. An NFS file share is provided through the Google Cloud-managed NFS File Server to store the Data Pump dump files.

The IP address of the NFS server is masked for this guide, but as an example, we will use the fictional aa.an.fs.pe address. The NFS path is **aa.an.fs.pe:/zdm_share**
The NFS share must be mounted on both the source database host and the target Autonomous Database.

To mount the NFS Share on the source database server:

As *root*:

```
mkdir -p /mnt/nfs3zdm
```

```
mount -o rw aa.an.fs.pe:/zdm_share /mnt/nfs3zdm
```

Make sure the *Oracle* user has access to the NFS mount

```
chown oracle:oinstall /mnt/nfs3zdm
```

As *oracle* user:

```
touch /mnt/nfs3zdm/test.txt
```

On the source PDB:

```
SQL> create directory DATA_PUMP_DIR_NFS as '/mnt/nfs3zdm';
```

Pre-Requisites

ZDM Service Host

On the ZDM host, as root, add the on-premises hostname and IP and the Autonomous Database Private Endpoint URL **sample.adb.us-region-1.oraclecloud.com** to the /etc/hosts file to be resolved to the Autonomous Database Private Endpoint IP **aa.dd.bb.ss**.

Access Network File System from Autonomous Database

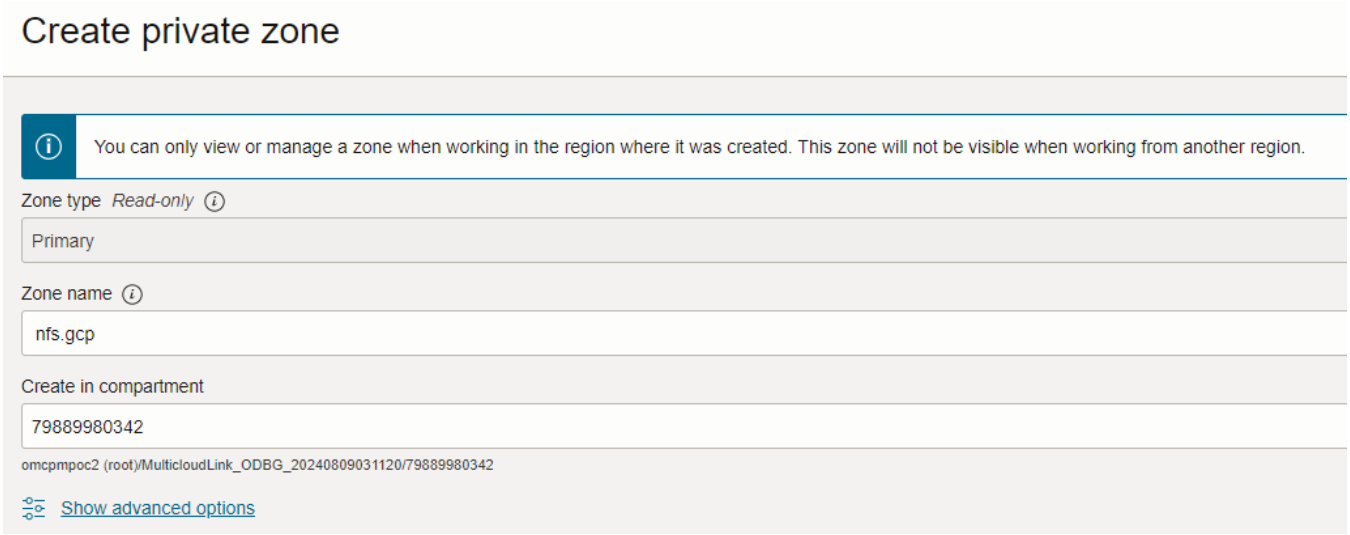
You can attach a Network File System to a directory location in your Autonomous Database[4]. This allows you to load data from Oracle Cloud Infrastructure File Storage in your Virtual Cloud Network (VCN), Google Cloud-managed NFS Server[5], or any other Network File System in on-premises data centers. Depending on the Network File System version you want to access, both **NFSv3** and **NFSv4** are supported.

Step 1: Add NFS Server Name to OCI DNS VCN Resolver

Bear in mind that if the OCI tenancy is a new tenancy created within the Oracle Database@Google Cloud provisioning process, the limits for OCI private DNS and A records might need to be increased. To increase the limits, open a Service Request with Oracle Support. A limit of at least three records is needed.

Follow these steps to create an A-record in OCI DNS to resolve the NFS server name:

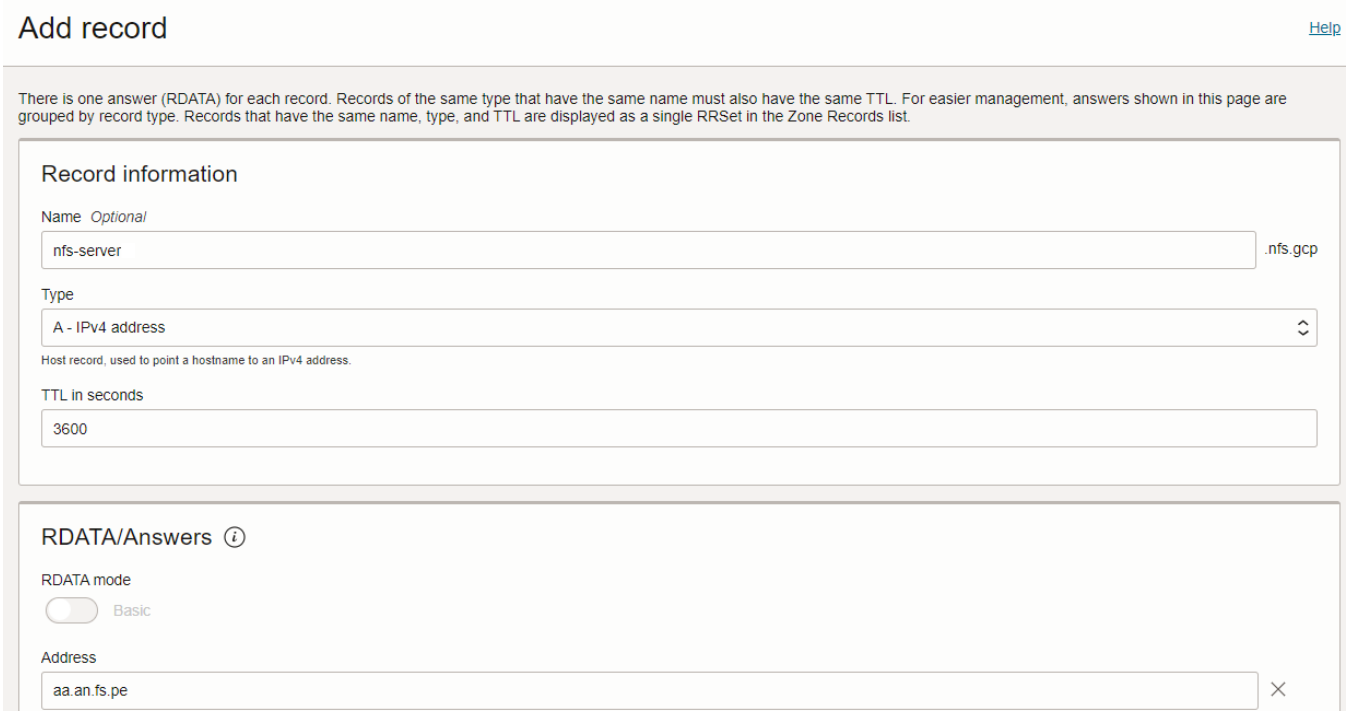
1. From the Oracle Autonomous Database details page in Google Cloud, click on the MANAGE IN OCI link.
2. From the Oracle Autonomous Database details page in OCI, click on the virtual cloud network in the Network section.
3. On the network details page, click on the DNS Resolver.
4. On the private resolver details page, click on the default private view.
5. Click the create zone button and create a new zone using the name of your choice, e.g., ***nfs.gcp***



[4] https://docs.oracle.com/en/cloud/paas/autonomous-database/serverless/adbsb/load-oci-file-storage.html#GUID-7C396A7A-D20A-40F7-99D7-50B85B9B18DC

[5] https://cloud.google.com/filestore/docs

6. Click on the newly created zone, manage records, and add a record with the name of your choice, e.g., ***nfs-server***. Replace aa.an.fs.pe with the actual IP address of the Google Cloud-managed NFS server.



7. Publish the changes.

Domain	Type	TTL	State	RDATA
nfs-server.nfs.gcp	A	3600	Unmodified	[redacted]
nfs.gcp	NS	86400	Unmodified	vcn-dns.oraclevcn.com.
nfs.gcp	SOA	86400	Unmodified	vcn-dns.oraclevcn.com. hostmaster.oracle.com.

8. Update the Network Security Group (NSG) in OCI to allow network traffic flow from the VPC where the NFS server resides.

On the target database:

Step 2: Add the NFS Server Name to the Access Control List (ACL)

```
SQL> exec DBMS_NETWORK_ACL_ADMIN.APPEND_HOST_ACE(host => 'nfs-server.nfs.gcp', ace =>
xs$ace_type(privilege_list => xs$name_list('connect', 'resolve'), principal_name =>
'ADMIN', principal_type => xs_acl.ptype_db));

PL/SQL procedure successfully completed.
```

Step 3: Create a Directory on the Autonomous Database

```
SQL> CREATE or replace DIRECTORY FSS_DIR AS 'fss';

Directory created.
```

Step 4: Attach NFS to Autonomous Database

Set the NFS version accordingly in the parameter "**params => JSON_OBJECT('nfs_version' value <value>)**".

```
SQL> BEGIN
DBMS_CLOUD_ADMIN.ATTACH_FILE_SYSTEM(
file_system_name => 'GCPNFS',
file_system_location => 'nfs-server.nfs.gcp:/zdm_share',
directory_name => 'FSS_DIR',
description => 'Attach GCP NFS',
params => JSON_OBJECT('nfs_version' value 3)
);
END;
/

PL/SQL procedure successfully completed.

SQL> SELECT object_name FROM DBMS_CLOUD.LIST_FILES('FSS_DIR');

OBJECT_NAME
--------------------------------------------------------------------------------
test.txt
```

Additional Configuration

SSH Key

ZDM connects via SSH to the Source Database servers; hence, an SSH key pair for the zdmuser is required. As zdmuser, run the following:

```
[zdmuser@zdmhost ~]$ mkdir ~/.ssh
[zdmuser@zdmhost ~]$ chmod 700 ~/.ssh
[zdmuser@zdmhost ~]$ /usr/bin/ssh-keygen -t rsa
Generating public/private rsa key pair.

Enter file in which to save the key (/home/zdmuser/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/zdmuser/.ssh/id_rsa.
Your public key has been saved in /home/zdmuser/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:keyfingerprintsample zdmuser@zdmhost
[zdmuser@zdmhost ~]$ cd ~/.ssh
[zdmuser@zdmhost .ssh]$ cat id_rsa.pub >> authorized_keys
[zdmuser@zdmhost .ssh]$ chmod 600 authorized_keys
```

You can find more information on ZDM Product's documentation section, "Generating a Private SSH Key Without a Passphrase."[6]

[6] https://docs.oracle.com/en/database/oracle/zero-downtime-migration/index.html

Before continuing with the migration environment setup, rename the id_rsa.pub file to <zdm_service_host_name>.ppk

On the ZDM Service Host.

```
[zdmuser@zdmhost .ssh]$ cd /home/zdmuser/.ssh

[zdmuser@zdmhost .ssh]$ mv id_rsa zdm.ppk
```

Authentication Token

The OCI user requires an Authentication Token, which can be created from the user's detail page. Click on the "Auth Tokens" option and the "Generate Token" button. ZDM uses the Auth Token during the migration; hence, it is of the utmost importance that it is securely copied and stored.

OCI CLI Command Line Tool

The Oracle Cloud Infrastructure command-line tool (OCI CLI) accesses OCI resources during the migration, among other tasks. To install the OCI CLI on the ZDM Service Host, as the zdmuser, run as follows:

```
[zdmuser@zdmhost ~]$ sudo yum install python36-oci-cli
```

API Signing Public Key and Configuration File

ZDM uses an API Signing Public Key to call REST APIs. First, you need to create the API Keys. Do so by accessing the terminal on the ZDM Service Host, and as the zdmuser, run the following:

```
[zdmuser@zdmhost ~]$ mkdir .oci
[zdmuser@zdmhost ~]$ cd .oci
[zdmuser@zdmhost ~]$ openssl genrsa -out /u01/app/zdmhome/.oci/oci_api_key.pem 2048
[zdmuser@zdmhost ~]$ openssl rsa -pubout -in /u01/app/zdmhome/.oci/oci_api_key.pem -out
/u01/app/zdmhome/.oci/oci_api_key_public.pem
[zdmuser@zdmhost ~]$ cat oci_api_key_public.pem
```

Copy the catted 'oci_api_key_public.pem' file and save it; you will need it in the next step. Include the "Begin Public Key" and "End Public Key" lines during the copy. Go to your Oracle Cloud OCI Dashboard, navigate to the top right, click on your user profile icon, and select the top option representing your user. Select API Keys and Add API Key. Paste the public OCI API key file you copied above and click Add Key.

You will see a configuration file preview. Copy its contents; you will use them to populate your configuration file in the following step.

As the zdmuser in the ZDM Service Host, create a configuration file in the command prompt; you can use vi/vim or any editor you prefer. In the empty file, paste the configuration file contents copied from above. Replace < path to your private key file > # TODO with the line above; once done, save the file and quit the editor:

```
/u01/app/zdmhome/.oci/oci_api_key.pem
```

Database Migration Step by Step with ZDM

Step 1: Prepare the Source Database Host On-Premises

Copy the SSH public key of the zdmuser from the ZDM host to the .ssh/authorized_keys file on the source database host for the user you want to use for login, in this case, ***onpuser***:

```
#on ZDM host as zdmuser
[zdmuser@zdmhost ~]$ cat .ssh/id_rsa.pub
#on the source database host as user onpuser
[onpuser@onphost ~]$ vi .ssh/authorized_keys
#insert the public key and save the changes
```

Step 2: Prepare the Source Database On-Premises

For offline logical migrations, for optimal Data Pump performance, it is recommended that you set STREAMS_POOL_SIZE to a minimum of 256MB-350MB, to have an initial pool allocated, otherwise you might see a significant delay during start-up.

```
SQL> alter system set streams_pool_size = 256M;
System altered.

SQL> grant DATAPUMP_EXP_FULL_DATABASE to system container=all;
Grant succeeded.
```

Step 3: Fill the response file

```
vi /home/zdmuser/logical_offline_adb_nfs/logical_offline_adb_nfs.rsp

# migration method
MIGRATION_METHOD=OFFLINE_LOGICAL
DATA_TRANSFER_MEDIUM=NFS

# data pump
DATAPUMPSETTINGS_JOBMODE=SCHEMA
DATAPUMPSETTINGS_METADATAREMAPS-1=type:REMAP_TABLESPACE,oldValue:USERS,newValue:DATA
INCLUDEOBJECTS-1=owner:HR
DATAPUMPSETTINGS_EXPORTDIRECTORYOBJECT_NAME=DATA_PUMP_DIR_NFS
DATAPUMPSETTINGS_IMPORTDIRECTORYOBJECT_NAME=FSS_DIR

# source db

SOURCEDATABASE_CONNECTIONDETAILS_HOST=onphost
SOURCEDATABASE_CONNECTIONDETAILS_PORT=1521
SOURCEDATABASE_CONNECTIONDETAILS_SERVICENAME=orclpdb
SOURCEDATABASE_ADMINUSERNAME=SYSTEM

# target db

TARGETDATABASE_OCID=ocid1.autonomousdatabase.oc1.iad.aaaa.bbb.cccc.ddddd
TARGETDATABASE_ADMINUSERNAME=ADMIN
```

```
# oci cli

OCIAUTHENTICATIONDETAILS_USERPRINCIPAL_USERID=ocid1.user.oc1..aaaa.bbb.ccccc.ddddd
OCIAUTHENTICATIONDETAILS_USERPRINCIPAL_TENANTID=ocid1.tenancy.oc1.aaa.bbbbb
OCIAUTHENTICATIONDETAILS_USERPRINCIPAL_FINGERPRINT=12:ac:34:cc:aa
OCIAUTHENTICATIONDETAILS_USERPRINCIPAL_PRIVATEKEYFILE=/home/zdmuser/.oci/oci_api_key.pem
OCIAUTHENTICATIONDETAILS_REGIONID=us-ashburn-1
```

Step 4: Perform a migration in evaluation mode

Execute the following command on the ZDM host as zdmuser to evaluate the migration. ZDM will check the source and target database configurations, but the migration will not be performed; it will just be an evaluation. On the ZDM host as zdmuser

```
$ZDMHOME/bin/zdmcli migrate database \
-rsp /home/zdmuser/logical_offline_adb_nfs/logical_offline_adb_nfs.rsp \
-sourcenode onphost \
-sourcesid oradb \
-srcauth zdmauth \
-srcarg1 user:onpuser \
-srcarg2 identity_file:/home/zdmuser/.ssh/id_rsa \
-srcarg3 sudo_location:/usr/bin/sudo \
-eval
```

Check the job status:

```
$ZDMHOME/bin/zdmcli query job -jobid 1
zdmhost.zdm: Audit ID: 185
Job ID: 1
User: zdmuser
Client: zdmhost
Job Type: "EVAL"
Scheduled job command: "zdmcli migrate database -rsp
/home/zdmuser/logical_offline_adb/logical_offline_adb.rsp -sourcenode onphost -sourcesid
oradb -srcauth zdmauth -srcarg1 user:onpuser -srcarg2
identity_file:/home/zdmuser/.ssh/id_rsa -srcarg3 sudo_location:/usr/bin/sudo -eval"
Scheduled job execution start time: 2024-10-18T11:00:52Z. Equivalent local time: 2024-
10-18 11:00:52
Current status: SUCCEEDED
Result file path: "/home/zdmuser/zdm/zdmbase/chkbase/scheduled/job-1-2024-10-18-
11:01:21.log"
Metrics file path: "/home/zdmuser/zdm/zdmbase/chkbase/scheduled/job-1-2024-10-18-
11:01:21.json"
Excluded objects file path: "/home/zdmuser/zdm/zdmbase/chkbase/scheduled/job-1-filtered-
objects-2024-10-18T11:05:34.879.json"
Job execution start time: 2024-10-18 11:01:21
Job execution end time: 2024-10-18 11:07:21
```

```
Job execution elapsed time: 2 minutes 33 seconds
ZDM_VALIDATE_TGT ..................... COMPLETED
ZDM_VALIDATE_SRC ..................... COMPLETED
ZDM_SETUP_SRC ........................ COMPLETED
ZDM_PRE_MIGRATION_ADVISOR ............ COMPLETED
ZDM_VALIDATE_DATAPUMP_SETTINGS_SRC .... COMPLETED
ZDM_VALIDATE_DATAPUMP_SETTINGS_TGT .... COMPLETED
ZDM_PREPARE_DATAPUMP_SRC .............. COMPLETED
ZDM_DATAPUMP_ESTIMATE_SRC ............. COMPLETED
ZDM_CLEANUP_SRC ...................... COMPLETED
```

Step 5: Migrate the Database

To initiate the actual migration, execute the same command for evaluation, but this time without the -eval parameter. On the ZDM host as zdmuser:

```
$ZDMHOME/bin/zdmcli migrate database \
-rsp /home/zdmuser/logical_offline_adb_nfs/logical_offline_adb_nfs.rsp \
-sourcenode onphost \
-sourcesid zdmdb \
-srcauth zdmauth \
-srcarg1 user:onpuser \
-srcarg2 identity_file:/home/zdmuser/.ssh/id_rsa \
-srcarg3 sudo_location:/usr/bin/sudo
```

Check the job status:

```
$ZDMHOME/bin/zdmcli query job -jobid 2
zdmhost.zdm: Audit ID: 186
Job ID: 2
User: zdmuser
Client: zdmhost
Job Type: "MIGRATE"
Scheduled job command: "zdmcli migrate database -rsp
/home/zdmuser/logical_offline_adb/logical_offline_adb.rsp -sourcenode onphost -sourcesid
oradb -srcauth zdmauth -srcarg1 user:onpuser -srcarg2
identity_file:/home/zdmuser/.ssh/id_rsa -srcarg3 sudo_location:/usr/bin/sudo"
Scheduled job execution start time: 2024-10-18T11:30:57Z. Equivalent local time: 2024-
10-18 11:30:57
Current status: SUCCEEDED
Result file path: "/home/zdmuser/zdm/zdmbase/chkbase/scheduled/job-2-2024-10-18-
11:31:21.log"
Metrics file path: "/home/zdmuser/zdm/zdmbase/chkbase/scheduled/job-2-2024-10-18-
11:31:21.json"
Excluded objects file path: "/home/zdmuser/zdm/zdmbase/chkbase/scheduled/job-2-filtered-
objects-2024-10-18T11:31:38.567.json"
Job execution start time: 2024-10-18 11:31:22
Job execution end time: 2024-10-18 11:35:58
```

```
Job execution elapsed time: 4 minutes 36 seconds
ZDM_VALIDATE_TGT ..................... COMPLETED
ZDM_VALIDATE_SRC ..................... COMPLETED
ZDM_SETUP_SRC ........................ COMPLETED
ZDM_PRE_MIGRATION_ADVISOR ............ COMPLETED
ZDM_VALIDATE_DATAPUMP_SETTINGS_SRC .... COMPLETED
ZDM_VALIDATE_DATAPUMP_SETTINGS_TGT .... COMPLETED
ZDM_PREPARE_DATAPUMP_SRC ............. COMPLETED
ZDM_DATAPUMP_ESTIMATE_SRC ............ COMPLETED
ZDM_PREPARE_DATAPUMP_TGT ............. COMPLETED
ZDM_DATAPUMP_EXPORT_SRC .............. COMPLETED
ZDM_TRANSFER_DUMPS_SRC ............... COMPLETED
ZDM_DATAPUMP_IMPORT_TGT .............. COMPLETED
ZDM_POST_DATAPUMP_SRC ................ COMPLETED
ZDM_POST_DATAPUMP_TGT ................ COMPLETED
ZDM_POST_ACTIONS ..................... COMPLETED
ZDM_CLEANUP_SRC ...................... COMPLETED
```

Troubleshooting Oracle ZDM & Other Resources

For Oracle ZDM log review:

- ZDM Server Logs:
  - Check - $ZDM_BASE/crsdata/<zdm_service_node>/rhp/rhpserver.log.0

- Check source node logs
  - - <oracle_base>/zdm/zdm_<src_db_name>_<job_id>/zdm/log

- Check target node logs.
  - - <oracle_base>/zdm/zdm_<tgt_db_name>_<job_id>/zdm/log

For all Oracle Support Service Requests related to Zero Downtime Migration, please be sure to follow the instructions in My Oracle Support Document:

- SRDC – Data Collection for Database Migration Using Zero Downtime Migration (ZDM) (DOC ID 2595205.1)
- https://support.oracle.com/epmos/faces/DocContentDisplay?id=2595205.1

ORACLE

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