ORACLE

# Platinum MAA: Oracle GoldenGate Microservices Architecture Integrated with Oracle Active Data Guard

October 2022, Version 2.1

## Purpose statement

This document describes the best practices for configuring Oracle GoldenGate Microservices Architecture, to work seamlessly with Oracle Data Guard, using Oracle Real Application Clusters (Oracle RAC), Oracle Clusterware, and Oracle Database File System (DBFS).

## Disclaimer

# Table of contents

## Introduction

To achieve the highest levels of availability and disaster recovery, resulting in zero or near-zero downtime for both unplanned outages and all planned maintenance activities, customers frequently use the combination of Oracle Exadata Engineered Systems (with built-in Oracle RAC clusters and HA best practices), Oracle Active Data Guard, and Oracle GoldenGate. Oracle RAC, Oracle Active Data Guard, and Oracle GoldenGate are essential components of Platinum MAA reference architectures as described in the Oracle MAA Reference Architectures and the *Oracle Database High Availability Overview and Best Practices* guide.

Platinum MAA provides the following benefits:

- RTO = zero or near-zero for all local failures using Exadata database platform that includes Oracle RAC and inherent full stack redundancy and failover capabilities
- RTO = zero or near-zero for disasters, such as database, cluster, or site failures, by redirecting the application to an active GoldenGate replica
- Zero downtime maintenance for software and hardware updates using Oracle RAC
- Zero downtime database upgrade or application upgrade by redirecting the application to an upgraded GoldenGate replica
- RPO = zero or near-zero depending on Data Guard protection mode setting, which dictates the redo transport (SYNC, FAR SYNC, or ASYNC)
- Fast resynchronization and zero or near-zero RPO between GoldenGate replicas after a disaster since the failed GoldenGate replica will be recovered quickly with Data Guard Fast-Start Failover, and the resynchronization between GoldenGate replicas can happen quickly. For SYNC transport, this leads to eventual zero data loss between the GoldenGate replicas and avoids too many cases of conflicts.

This paper does not cover these other essential areas that can complement the holistic solution:

- Application failover between distributed systems can be aided by Global Data Services. Application failover between clusters and zero data loss Data Guard switchover and failover can also be complemented by following Continuous Availability - Application Checklist for Continuous Service for MAA Solutions.
- GoldenGate conflict detection and resolution needs to be discussed and established based on application requirements. Refer to Using Oracle GoldenGate with Oracle Database documentation.
- Prerequisite for implementing the following best practices is configuring either Oracle GoldenGate Microservices Architecture with Oracle Real Application Clusters for on-premise or Oracle GoldenGate Microservices Architecture on Oracle Exadata Database Service Configuration Best Practices for cloud.

This technical brief describes the configuration best practices to enable Oracle GoldenGate Microservices replication using a database that is protected by a Data Guard standby, to transparently and seamlessly work following an Oracle Data Guard role transition, no matter which Data Guard protection mode is configured (Maximum Performance, Maximum Availability, or Maximum Protection).

There are a number of key software requirements that this technical brief is based on:

- Oracle Grid Infrastructure 19c or later

  Oracle Grid Infrastructure provides the necessary components needed to manage high availability for any business-critical applications. Using Oracle Clusterware (a component of Oracle Grid Infrastructure) network, database, and Oracle GoldenGate resources can be managed to provide availability in the event of a failure.

- Oracle Grid Infrastructure Agent version 10.2 or later

  The Oracle Grid Infrastructure Agent leverages the Oracle Grid Infrastructure components to provide integration between Oracle GoldenGate and its dependent resources, such as the database, network, and file system. The agent also integrates Oracle GoldenGate with Oracle Data Guard so that Oracle GoldenGate is restarted on the new primary database following a role transition.

- Oracle Database 19c or later

  Refer to My Oracle Support note 2193391.1 for a full list of recommended Oracle Database patches when using Oracle GoldenGate.

- Oracle GoldenGate Microservices version 21c or later

  Oracle GoldenGate 21c introduces unified build support so a single software installation supports capturing and applying replicated data to multiple major Oracle Database versions (11g Release 2 to 21c). This is possible because an Oracle GoldenGate installation includes the required Oracle Database client libraries without requiring a separate database `ORACLE_HOME` installation.

- Oracle DBFS to protect and replicate critical Oracle GoldenGate files

  The Oracle Database File System (DBFS) is the only MAA-validated and recommended file system for an Oracle Data Guard and Oracle GoldenGate configuration, because it allows the storage of the required Oracle GoldenGate files, such as the checkpoint and trail files, to be located inside the same database that is protected with Oracle Data Guard, ensuring consistency between the Oracle GoldenGate files and the database in a seamless fashion.

  .

## MAA Architecture Examples with Oracle GoldenGate and Oracle Active Data Guard

For some applications, an active-active HA database architecture is required to take advantage of the unplanned and planned maintenance advantages that come with Oracle GoldenGate. However, Oracle GoldenGate and Oracle RAC alone cannot guarantee zero data loss in the case of database, cluster, or data corruption failures. To protect against these types of failures, the MAA architectures illustrated in the following figures are suggested.

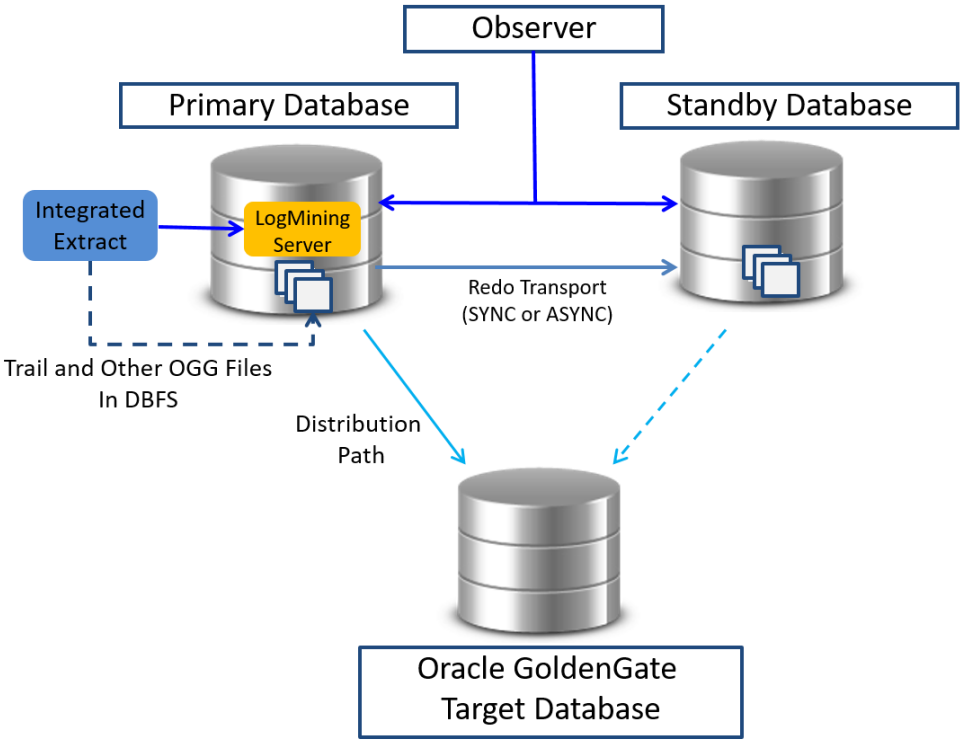


Figure 1: Example Architecture 1

In Figure 1, when executing a failover or switchover to the physical standby database, the Oracle GoldenGate Extract and Target databases work seamlessly after the Oracle Data Guard role transition. The Oracle Active Data Guard standby database is the primary failover target in case there are database or cluster failures. Oracle Active Data Guard is read-only and active, but all application transactions that contain DMLs or DDLs must be redirected to the primary database.

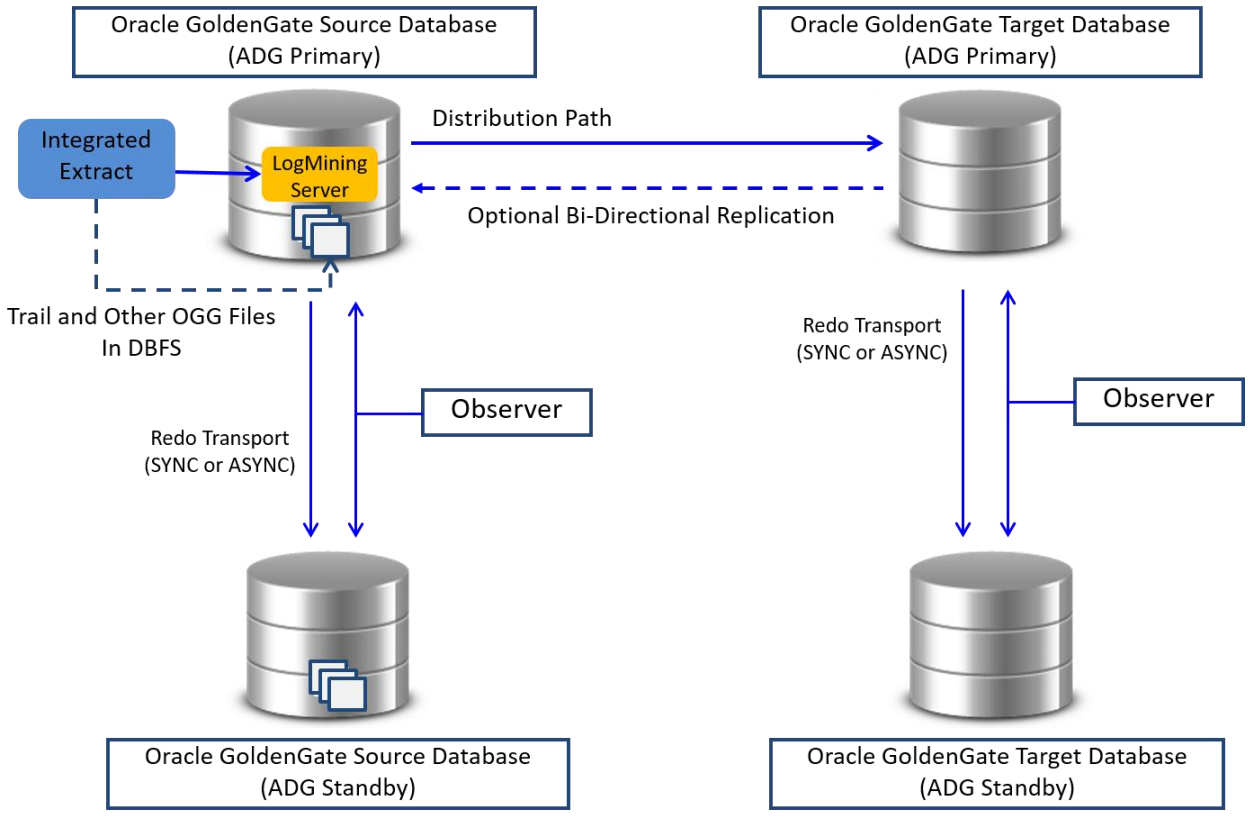


Figure 2: Example Architecture 2

In Figure 2, which is classic Platinum MAA, the two primary databases replicate data between each other using Oracle GoldenGate. The data replication can be uni-directional or bi-directional. When data loss cannot be tolerated, Oracle Active Data Guard fast-start failover configured in Maximum Availability or Maximum Protection mode (`SYNC` transport) is configured for each Oracle GoldenGate database. When fast-start failover is enabled, the Oracle Data Guard broker and its observer determine if a failover is necessary and initiate the failover to the specified standby database automatically, with no need for DBA intervention. After Oracle Data Guard failover or switchover, Oracle GoldenGate replication will continue to work seamlessly.

Figure 2 depicts Oracle GoldenGate being active on the primary database. The post-failover diagram in Figure 3 shows that Oracle GoldenGate has been re-enabled on the new primary database at the standby site.

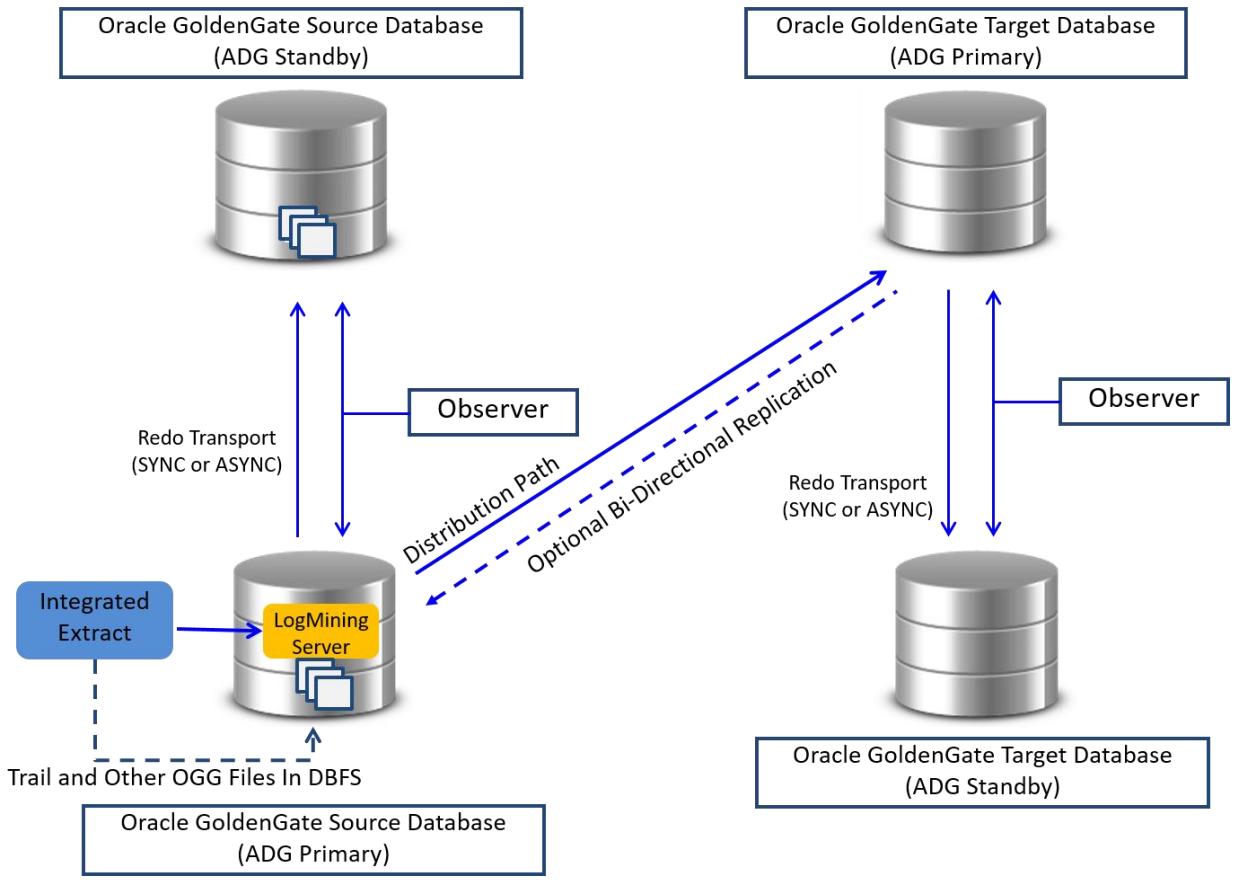


Figure 3: Post-Failover of the Oracle GoldenGate Source Database

Refer to Platinum MAA presentation for more detailed architectural and application flow before and after outages.

## Configuration Prerequisites

The configuration prerequisite for this paper is to have Oracle GoldenGate configured as detailed in the Oracle GoldenGate Microservices Architecture with Oracle Real Application Clusters Configuration Best Practices technical brief. The Database File System (DBFS) is required for critical Oracle GoldenGate files when integrating with Data Guard.

The Oracle Data Guard standby database should also be configured and operational before continuing. For further information on Oracle Data Guard refer to the Oracle Data Guard Concepts and Administration guide.

Once the prerequisites are met, the following configuration best practices should be implemented to ensure the seamless integration of GoldenGate Microservices with Oracle Data Guard, to ensure GoldenGate continues running after any Data Guard role transition.

## Configuration Best Practices

### Step 1: Configure the standby database for Oracle GoldenGate

The standby database initialization parameters should match those of the primary database, as specified in the Oracle GoldenGate Microservices Architecture with Oracle Real Application Clusters Configuration Best Practices technical brief. This includes the following parameters:

- `ENABLE_GOLDENGATE_REPLICATION=TRUE`
- For GoldenGate source databases, enable `FORCE LOGGING` mode and enable minimal supplemental logging.
- If a GoldenGate source database, or running integrated Replicat (parallel or non-parallel), configure the `STREAMS_POOL_SIZE`.

### Step 2: Modify the Primary Database Service

On the primary database server, modify the existing database service that was created as part of the original Oracle GoldenGate on Oracle RAC configuration. Set the service role to PRIMARY, so that the service will only be started when the database becomes the Data Guard primary database role after a role transition.

Modify the service using the following command, as the `oracle` user.

```
$ srvctl modify service -db <db_name> -service <service name> -role PRIMARY
```

For example:

```
$ srvctl modify service -db ggdb -service oggserv_pdb –role PRIMARY
```

If your database is part of a multitenant environment, remember to modify both the multitenant container database (CDB) and pluggable database (PDB) services.

### Step 3: Create the Standby Database Service

On the standby cluster, a database service is required for the standby database so that the Oracle Grid Infrastructure Agent will automatically start the Oracle GoldenGate deployment when the database is opened with the primary role.

When a source database is in a multitenant environment, a separate service is required for the root container database (CDB) and the pluggable database (PDB) that contains the schema being replicated. For a multitenant environment target database, a single service is required for the PDB.

Create the service using the following command, as the `oracle` user, the same way the service was created on the primary cluster. It is recommended that you use the same service name as was specified on the primary cluster. The service must be created as a singleton service, using the `–preferred` option, because the application Virtual IP address (VIP), DBFS, and Oracle GoldenGate will run on the cluster node where the service is running.

```
$ srvctl add service -db <db_name> -service <service name> -preferred <instance_1>
-available <instance_2, instance_3 etc.> -pdb <PDB name> -role PRIMARY
```

For example:

```
$ srvctl add service -db ggdb -service oggserv_pdb -preferred ggdb1 -available ggdb2
-pdb GGPDB01 -role PRIMARY
```

If the database is not in a multitenant environment, or the database is a target database for Oracle GoldenGate, omit the `-pdb` parameter.

Refer to the *Oracle Real Application Clusters Administration and Deployment Guide* for further details on creating a database service at

https://docs.oracle.com/en/database/oracle/oracle-database/21/racad/server-control-utility-reference.html#GUID-3ED4DBCE-A148-462B-8A79-534A3F0D6E7D

### Step 4: Configure DBFS on the Standby Cluster Nodes

The Database File System (DBFS) is the only recommended solution when configuring Oracle GoldenGate with Oracle Data Guard. The DBFS user, tablespace, and file system in the database was previously created in the primary database, as detailed in the Oracle GoldenGate Microservices Architecture with Oracle Real Application Clusters Configuration Best Practices technical brief. The remaining configuration steps are required on **all** nodes of the standby cluster where GoldenGate may run.

4.1 Install the required FUSE libraries, if they are not already installed, by following the instructions in My Oracle Support note 869822.1

4.2 Create the tnsnames.ora Oracle Net connection alias using the IPC protocol, similar to the one created on the primary cluster. For example:

```
dbfs =
    (DESCRIPTION =
      (ADDRESS = (PROTOCOL = IPC)(KEY=LISTENER))
      (CONNECT_DATA =
        (SERVICE_NAME = <NAME>)
       )
    )
```

4.3 Create the **same** mount point for DBFS that is used on the primary cluster.

It is important that the mount point is identical, because the physical location of the Oracle GoldenGate deployment is included within the deployment configuration files.

For example:

```
# mkdir /mnt/dbfs
```

4.4 Copy the `mount-dbfs.conf` and `mount-dbfs.sh` files from the primary cluster to the standby cluster nodes.

It is recommended that you place them in the same directory as the primary cluster.

Register the DBFS resource with Oracle Clusterware, using the following example command. If using Oracle Multitenant, make sure to use the service name for the same PDB that contains the DBFS repository as was created in the primary database.

```
DBNAME=ggdbs
DEPNAME=ora.$DBNAME.oggserv_pdb.svc
```

```
crsctl add resource $RESNAME \
  -type cluster_resource \
  -attr "ACTION_SCRIPT=$ACTION_SCRIPT, \
         CHECK_INTERVAL=30,RESTART_ATTEMPTS=10, \
         START_DEPENDENCIES='hard($DEPNAME)pullup($DEPNAME)',\
         STOP_DEPENDENCIES='hard($DEPNAME)',\
         SCRIPT_TIMEOUT=300"
```

### Step 5: Install Oracle GoldenGate Software

Install the Oracle GoldenGate software locally on all nodes in the standby cluster that will be part of the Oracle GoldenGate configuration. Make sure the installation directory is the **identical** on all nodes to match the primary cluster installation directory.

Download the Oracle GoldenGate 21c software, or later version, from Oracle Technology Network (OTN) at:

http://www.oracle.com/technetwork/middleware/goldengate/downloads/index.html

### Step 6: Create Oracle GoldenGate Deployment Directories

The Oracle GoldenGate Service Manager and deployment were already created on the primary cluster, but certain directories and symbolic links need to be configured on the standby cluster nodes. These directories and symbolic links were created on the primary cluster, as described in the Oracle GoldenGate Microservices Architecture with Oracle Real Application Clusters Configuration Best Practices technical brief.

Create the following directories and symbolic links on the all Oracle RAC nodes on the standby cluster.

6.1 If there are multiple GoldenGate Service Managers configured on the primary cluster, each with their own deployment, and individually registered with XAG, they must belong to separate OGG_HOME software installation directories.

The same directories and symbolic links for the `OGG_HOME` directories that were configured on primary cluster, must match on the standby cluster.

6.2 If the GoldenGate deployment was created with the Performance Metric Server enabled, the metric datastore home directory must be created on the standby Oracle RAC nodes.

For example, determine the datastore directory on the primary cluster nodes:

```
$ grep RepoDatastorePath <deployment directory>/var/log/pmsrvr.log|uniq
"RepoDatastorePath": "",
 "RepoDatastorePath": "/u01/oracle/goldengate/datastores/ggnorth",
```

Create the directory on all standby cluster nodes:

```
$ mkdir -p /u01/oracle/goldengate/datastores/ggnorth
```

6.3 If the database version is lower than Oracle Database Release 21c (21.3), create the Oracle GoldenGate deployment temp directory local storage to match the symbolic link created on the primary cluster.

For example, on the primary cluster:

```
$ ls -lrt <DBFS GoldenGate deployment home directory>/var/temp
lrwxrwxrwx  1 oracle oinstall 32 Aug 31 12:27 temp ->
/u01/oracle/goldengate/deployments/ggnorth/temp
```

Create the same directory on the standby cluster nodes:

```
$ mkdir -p /u01/oracle/goldengate/deployments/ggnorth/temp
```

## Step 7: Configure Standby NGINX Reverse Proxy

### 7.1 Install NGINX Reverse Proxy

If NGINX Reverse Proxy has not already been installed, follow the installation instructions at https://nginx.org/en/linux_packages.html.

As the root user, copy the Oracle GoldenGate deployment NGINX configuration files from a primary cluster node to a single standby node directory `/etc/nginx/conf.d`.

For example:

```
[root@dc2north01]# scp dc1north01:/etc/nginx/conf.d/ogg_north.conf /etc/nginx/conf.d
```

The standby cluster will need a different CA signed certificate due to using a different VIP name/address than the primary cluster. Contact your systems administrator to follow your corporate standards to create or obtain the server certificate before proceeding. A separate certificate is required for each VIP and Service Manager pair.

### 7.2 Install Server Certificates for NGINX

Install the server CA certificates and key files in the `/etc/nginx/ssl` directory, owned by root with file permissions 400 (-r--------):

```
# mkdir /etc/nginx/ssl
# chmod 400 /etc/nginx/ssl
```

For each reverse proxy configuration file copied from the primary cluster, set the correct filenames for the certificate and key file using the following example:

```
ssl_certificate /etc/nginx/ssl/gg-stby-vip1.pem;
ssl_certificate_key /etc/nginx/ssl/gg-stby-vip1.key;
```

When using CA signed certificates, the certificate named with the ssl_certificate NGINX parameter must include the root, intermediate, and CA signed certificates in a single file. The order is very important, otherwise NGINX fails to start and displays the error message

`(SSL: error:0B080074:x509 certificate routines: X509_check_private_key:key values mismatch).`

The root and intermediate certificates can be downloaded from the CA signed certificate provider.

The single file can be generated using the following example command:

```
# cat CA_signed_cert.crt intermediate.crt root.crt > gg-stby-vip1.pem
```

The ssl_certificate_key file is the key file generated when creating the Certificate Signing Request (CSR), which is required when requesting a CA signed certificate.

Change the `server_name` parameter in the reverse proxy configuration file copied from the primary cluster, setting to the correct VIP name. For example:

Before:

```
server_name             dc1north-vip1.example.com;
```

After:

```
server_name             dc2north-vip1.example.com;
```

### 7.3 Validate and Restart NGINX

Because the VIP will not be running on the standby cluster until the primary database service is running, there is a parameter that needs to be set in the `/etc/sysctl.conf` file.

As the `root` user, make the following modifications to /etc/sysctl.conf.

```
# vi /etc/sysctl.conf
```

Add the following parameter:

```
# allow processes to bind to the non-local address
net.ipv4.ip_nonlocal_bind = 1
```

Reload the modified configuration:

```
# sysctl -p /etc/sysctl.conf
```

Validate the NGINX configuration file to detect any errors in the configuration. If there are errors in the file, they will be reported by the following command.

```
# nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginxconf test is successful
```

Restart NGINX with the new configuration:

```
# systemctl restart nginx
```

Once the NGINX configuration is complete, copy the configuration file and certificates to matching directories on the other standby cluster nodes.

### 7.4 Create an NGINX Clusterware Resource

Oracle Clusterware needs to have control over starting the NGINX reverse proxy so that it can be started automatically before the GoldenGate deployments are started.

The NGINX resource is created with a dependency on the underlying network CRS resource, the name of which can be determined using the following command:

```
$ $GRID_HOME/bin/crsctl stat res -w "TYPE == ora.network.type"|grep NAME
NAME=ora.net1.network
```

As the `root` user, use the following example command to create a Clusterware resource to manage NGINX.

```
# $GRID_HOME/bin/crsctl add resource nginx -type generic_application -attr
"ACL='owner:root:rwx,pgrp:root:rwx,other::r--,group:oinstall:r-
x,user:oracle:rwx',EXECUTABLE_NAMES=nginx,START_PROGRAM='/bin/systemctl start -f
nginx',STOP_PROGRAM='/bin/systemctl stop -f nginx',CHECK_PROGRAMS='/bin/systemctl status
nginx' ,START_DEPENDENCIES='hard(ora.net1.network) pullup(ora.net1.network)',
STOP_DEPENDENCIES='hard(intermediate:ora.net1.network)', RESTART_ATTEMPTS=0,
HOSTING_MEMBERS='dc1north01,dc1north02', CARDINALITY=2"
```

The NGINX resource created in this example run on the named cluster nodes at the same time, specified by HOSTING_MEMBERS. This is recommended when multiple GoldenGate Service Manager deployments are configured, and they can independently move between cluster nodes.

Once the NGINX Clusterware resource is created, alter the GoldenGate XAG resources so that NGINX must be started before the GoldenGate deployments are started.

As the `root` user, modify the XAG resources using the following example commands.

Determine the current `--filesystems` parameter:

```
# agctl config goldengate GGNORTH | grep "File System"
File System resources needed: dbfsgg
```

## Step 8: Oracle Clusterware Configuration

### 8.1 Modify the primary cluster XAG GoldenGate instance

The Oracle Grid Infrastructure Standalone Agent (XAG) GoldenGate instance on the primary cluster must be modified as the `root` user, to identify that it is part of an Oracle Data Guard configuration using the following example command.

```
# agctl modify goldengate <instance_name> --dataguard_autostart yes
```

On the standby cluster, follow the instructions in the Oracle GoldenGate Microservices Architecture with Oracle Real Application Clusters Configuration Best Practices technical brief to do steps 8.2-8.4:

### 8.2 Install the XAG software on each standby cluster node

It is recommended that you install the XAG software into the same directory as the primary cluster.

### 8.3 Prepare for the XAG application VIP creation

It is assumed the VIP and VIP name will be different from that of the primary cluster, so the VIP address will need to be allocated by your systems administrator for the standby cluster.

### 8.4 Register Oracle GoldenGate Microservices with XAG

The parameters used to register Oracle GoldenGate Microservices with XAG are similar to those used when registering with the primary cluster.

To determine the current parameters in the primary cluster, use the following command:

```
$ agctl config goldengate <GoldenGate instance name>
```

For example:

```
$ agctl config goldengate GGNORTH

Instance name: GGNORTH
Application GoldenGate location is: /u01/oracle/goldengate/gg21c_MS
Goldengate MicroServices Architecture environment: yes
Goldengate Service Manager configuration directory:
/mnt/dbfs/goldengate/deployments/ggnorth_sm/etc/conf
Goldengate Service Manager var directory: /mnt/dbfs/goldengate//deployments/ggnorth_sm/var
Service Manager Port: 9100
Goldengate Administration User: oggadmin
Autostart on DataGuard role transition to PRIMARY: yes
Configured to run on Nodes: dc1north01,dc1north02
ORACLE_HOME location is: /u01/oracle/goldengate/gg21c_MS/lib/instantclient
Database Services needed: ora.ggdg.oggserv_cdb.svc,ora.ggdg.oggserv_pdb.svc
File System resources needed: dbfsgg,nginx
VIP name: gg_vip_prmy
```

In addition, the XAG parameter '`--filesystem_verify no`' must be specified to prevent XAG from checking the existence of the DBFS deployment directory when registering the GoldenGate instance. Without setting this parameter, the XAG registration will fail, because DBFS is not mounted on the standby cluster.

*NOTE: It is recommended to use the same GoldenGate instance name when registering GoldenGate with XAG as was used in the primary cluster.*

Example command to register GoldenGate with XAG on the standby cluster, as the `root` user:

```
# agctl add goldengate GGNORTH \
--gg_home /u01/oracle/goldengate/gg21c_MS \
--service_manager \
--config_home /mnt/dbfs/goldengate/deployments/ggnorth_sm/etc/conf \
--var_home /mnt/dbfs/goldengate/deployments/ggnorth_sm/var \
--port 9100 \
--oracle_home /u01/goldengate/gg21c_MS/lib/instantclient \
--adminuser oggadmin \
--user oracle \
--group oinstall \
--vip_name gg_vip_stby \
--filesystems dbfsgg,nginx \
--db_services ora.ggdgs.oggserv_cdb.svc,ora.ggdgs.oggserv_pdb.svc \
--use_local_services \
--nodes dc2north01,dc2north02 \
--filesystem_verify no \
--dataguard_autostart yes
```

Further information on the Oracle Grid Infrastructure Bundled Agent:

http://www.oracle.com/technetwork/database/database-technologies/clusterware/downloads/xag-agents-downloads-3636484.html

### Step 9: Create Oracle Net TNS Alias for Oracle GoldenGate Database Connections

The same TNS aliases created on the primary cluster for the primary database using the IPC protocol must be created with the **same** alias names on each node of the standby cluster, using the IPC communication protocol as specified in the Oracle GoldenGate Microservices Architecture with Oracle Real Application Clusters Configuration Best Practices technical brief.

The location of the tnsnames.ora used by the Oracle GoldenGate deployment **must** be identical on the standby cluster nodes as it is on the primary cluster. Use the following query REST API call to query the `TNS_ADMIN` location on the primary cluster.

```
$ curl -s -u <OGG admin username> https://<vip_name>/services/v2/deployments/<deployment_name>
-XGET|python -m json.tool|grep TNS_ADMIN -A1
```

You will be prompted to enter the Oracle GoldenGate Service Manager administrator user password.

For example:

```
$ curl -s -u oggadmin https://dc1north01-vip1/services/v2/deployments/ggnorth -XGET|python -m
json.tool|grep TNS_ADMIN -A1
                "name": "TNS_ADMIN",
                "value": "/u01/goldengate/network/admin"
```

Make sure the tnsnames.ora is located in this **same** directory on **all** standby cluster nodes.

Example TNS alias for the Goldengate database:

```
ggnorth_pdb =
    (DESCRIPTION =
      (SDU = 2097152)
      (ADDRESS = (PROTOCOL = IPC)(KEY=LISTENER))
      (CONNECT_DATA =
        (SERVICE_NAME = oggserv_pdb.us.oracle.com)
       )
    )
```

### Step 10: Configure Oracle GoldenGate Processes

In addition to the advice provided in the Oracle GoldenGate Microservices Architecture with Oracle Real Application Clusters Configuration Best Practices technical brief, follow the recommendations provided below for Extract, Distribution Paths, and Replicats.

#### Extract Configuration on the Primary Cluster

For GoldenGate Extract processes using Data Guard configurations that are using redo transport Maximum Performance or Maximum Availability modes, the following parameter must be added to the Extract process parameter file **on the primary cluster** to avoid losing transactions and resulting in logical data inconsistencies:

```
TRANLOGOPTIONS HANDLEDLFAILOVER
```

This parameter prevents Extract from extracting transaction data from redo that has not yet been applied to the Data Guard standby database. This is crucial to preventing Oracle GoldenGate from replicating data to a target database that does not exist in the source standby database.

If this parameter is not specified, after a data loss failover of the source database it is possible to have data in the target database that is not present in the source database, leading to logical data inconsistencies.

By default, after 60 seconds, a warning message will be written to the Extract report file when the Extract is stalled due to not being able to query the standby database applied SCN information. For example:

```
WARNING OGG-02721  Extract has been waiting for the standby database for 60 seconds.
```

The amount of time before the warning message is written to Extract report file can be adjusted using the Extract parameter `TRANLOGOPTIONS HANDLEDLFAILOVER STANDBY_WARNING`.

If the Extract is still not able to query the standby database applied SCN information after 30 minutes (default), the Extract process will abend, logging the following message in the Extract report file:

```
ERROR   OGG-02722  Extract abended waiting for 1,800 seconds for the standby database to be
accessible or caught up with the primary database.
```

If the standby database becomes available before the 30 default timeout expires, Extract continues mining data from

the source database and reports the following message to the report file:

```
INFO    OGG-02723  Extract resumed from stalled state and started processing LCRs.
```

The timeout value of 30 minutes can be adjusted using the Extract parameter `TRANLOGOPTIONS HANDLEDLFAILOVER STANDBY_ABEND <value>`, where value is the number of seconds the standby is unavailable before abending.

If the standby database will be unavailable for a prolonged duration, such as during a planned maintenance outage, and you wish Extract to continue extracting data from the primary database, remove the `TRANLOGOPTIONS HANDLEDLFAILOVER` parameter from the Extract parameter file and restart Extract (see example below in Figures 4 to 6). Remember to set the parameter after the standby becomes available.

*NOTE: If extracting from a primary database continues while the standby is unavailable, a data loss failover could result after the standby becomes available, and not all the primary redo was applied before a failover. The GoldenGate target database will contain data that does not exist in the source database.*

Refer to the *Oracle GoldenGate Reference Guide* for further information on the `TRANLOGOPTIONS HANDLEDLFAILOVER` parameters at

https://docs.oracle.com/en/middleware/goldengate/core/21.3/reference/reference-oracle-goldengate.pdf

If the Extract process has been assigned an auto restart profile, as documented in the Oracle GoldenGate Microservices Architecture with Oracle Real Application Clusters Configuration Best Practices technical brief, after a Data Guard role transition, the Extract process will automatically restart. Extract will continue to mine redo data from the new primary database, ignoring the current state of the new standby database, until a default 5 minute timeout period expires. After this time, if the standby is not available Extract will abend with the following errors:

```
INFO    OGG-25053  Timeout waiting for 300 seconds for standby database reinstatement. Now
enforcing HANDLEDLFAILOVER.
ERROR   OGG-06219  Unable to extract data from the Logmining server OGG$CAP_EXT1.
ERROR   OGG-02078  Extract encountered a fatal error in a processing thread and is abending.
```

Extract will continue to automatically restart, based on the GoldenGate Microservices auto restart profile, and failing due to reaching the `HANDLEDLFAILOVER` timeout, until the number retries is reached or the new standby database becomes available.

During the timeout period following a database role transition, the `HANDLEDLFAILOVER` parameter is automatically suspended, so data will be replicated to the Oracle GoldenGate replica database without consideration of the source standby database not being kept up to date. The timeout period for the standby database to start up before Extract abends can be adjusted using the Extract parameter `TRANLOGOPTIONS DLFAILOVER_TIMEOUT`.

It is recommended that you leave `DLFAILOVER_TIMEOUT` at the default of 5 minutes, to allow the old primary to convert to a standby. If the new standby database will be unavailable for an extended period of time or completely gone, then in order for Extract to start and remain running, you must remove the `HANDLEDLFAILOVER` parameter from the Extract parameter file. After removing the parameter, Extract no longer waits until redo has been applied to the standby database before extracting the data.

During the time it takes for the standby database to come back online and apply all the redo from the primary

database, there will be data divergence between it and the Oracle GoldenGate replica database. This will be resolved once the standby database is up to date. At which point, add the `HANDLEDLFAILOVER` parameter back into the integrated Extract process parameter file, and then stop and restart the Extract.

When Oracle Data Guard is configured with fast-start failover, such that the broker can automatically fail over to a standby database in the event of loss of the primary database, you must specify an additional integrated Extract parameter shown below.

```
TRANLOGOPTIONS FAILOVERTARGETDESTID n
```

This parameter identifies which standby database the Oracle GoldenGate Extract process must remain behind, with regards to not extracting redo data that has not yet been applied to the standby database.

To determine the correct value for `FAILOVERTARGETDESTID`, use the `LOG_ARCHIVE_DEST_N` parameter from the GoldenGate source database which is used for sending redo to the source standby database. For example, if `LOG_ARCHIVE_DEST_2` points to the standby database, then use a value of 2.

For example:

```
SQL> show parameters log_archive_dest
NAME                 TYPE        VALUE
-------------------- ----------- ------------------------------------------------------
log_archive_dest_1   string      location=USE_DB_RECOVERY_FILE_DEST,
                                 valid_for=(ALL_LOGFILES, ALL_ROLES)

log_archive_dest_2   string      service="ggnorths", SYNC AFFIRM delay=0
                                 optional compression=disable max_failure=0 reopen=300
                                 db_unique_name="GGNORTHS" net_timeout=30,
                                 valid_for=(online_logfile,all_roles)
```

In this example, the Extract parameter would be set to the following:

```
TRANLOGOPTIONS FAILOVERTARGETDESTID 2
```

To add the parameters to the Extract parameter file, use the Oracle GoldenGate Administration Server to select display the Extract details as shown in Figure 4.

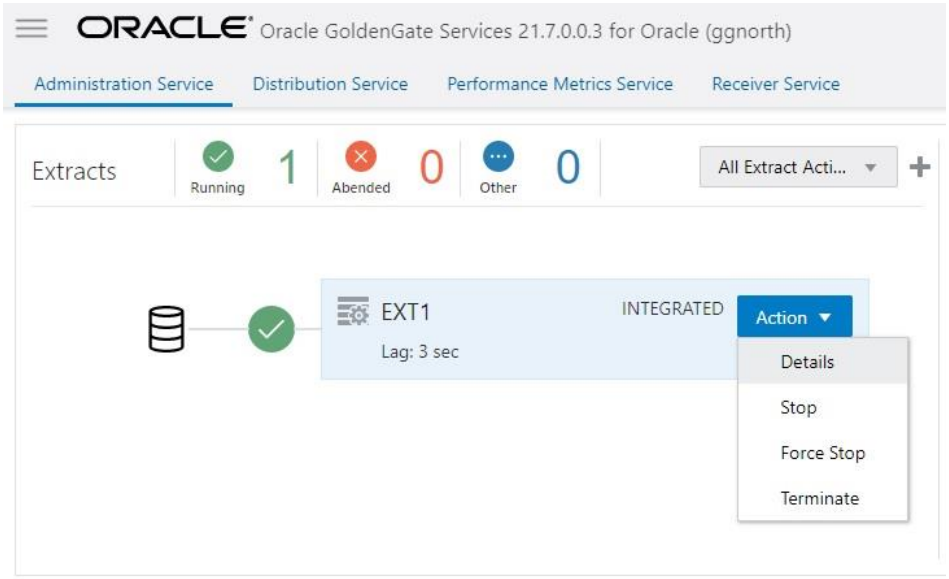


Figure 4: Viewing the Extract parameter file

Select the **Parameters** tab, and then select the pencil icon to edit the current parameter file, as shown in Figure 5.

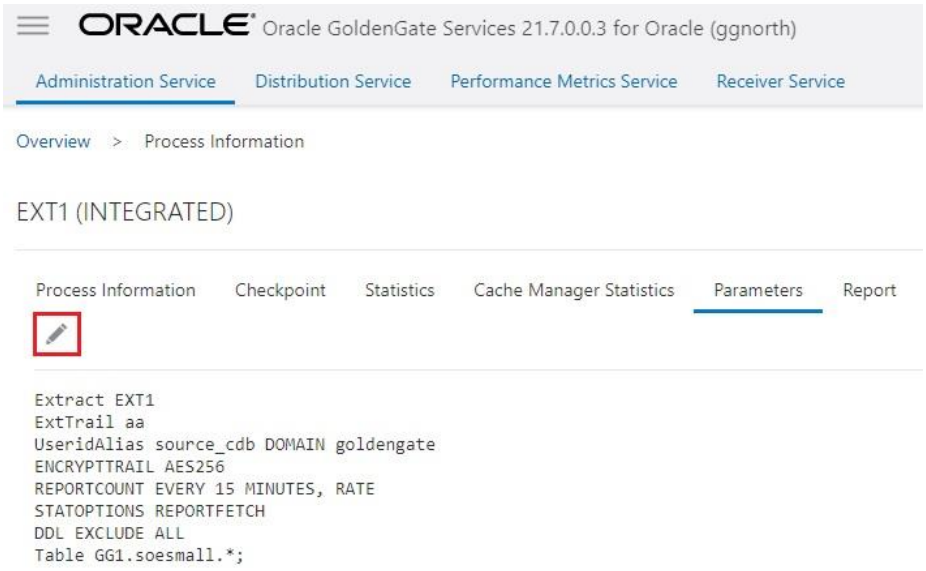


Figure 5: Editing the Extract parameter file

Add the `TRANLOGOPTIONS` parameters as shown in Figure 6, and select **Apply** to save the changes.

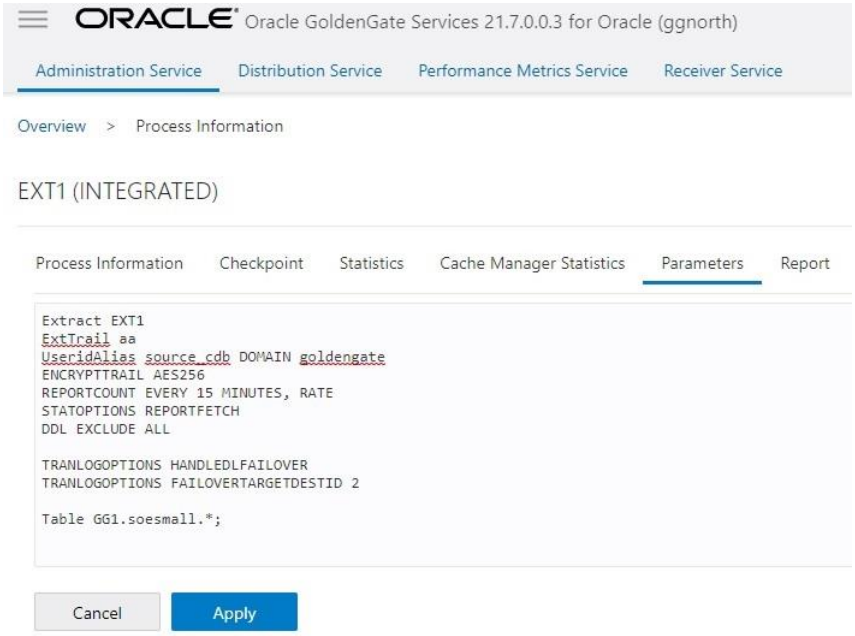

Figure 6: Adding the `TRANLOGOPTIONS` to the Extract parameter file

For the new parameters to take effect, the Extract process needs to be stopped and restarted, which can be done using the Administration Server.

Further information about the Extract `TRANLOGOPTIONS` parameters mentioned above, can be found in the *Reference for Oracle GoldenGate* at

https://docs.oracle.com/en/middleware/goldengate/core/21.3/reference/tranlogoptions.html#GUID-B6ADFEC9-10E6-456D-9477-088513E113AF

## Distribution Path Configuration on the Primary and Standby Cluster

When the target database of an Oracle GoldenGate environment, where the Receiver Server runs, is protected with Oracle Data Guard, there is an important consideration that must be given to any Distribution Paths that are sending trail files to the Receiver Server. When the Receiver Server moves to a different cluster after an Oracle Data Guard role transition, any distribution paths must be altered to reflect the new target cluster address.

You can automatically change the distribution paths using a database role transition trigger in the target database on the Receiver Server cluster.

If the primary and standby cluster VIPs use different root CA certificates, the standby certificate will need to be added to the source deployment Service Manager, as detailed in the Oracle GoldenGate Microservices Architecture with Oracle Real Application Clusters Configuration Best Practices technical brief.

Follow the instructions below to create a database role transition trigger to modify the distribution path target address when the receiver server moves between the primary and standby cluster, during target database Data Guard role transitions.

1. Create a shell script to modify the distribution paths.

   Appendix A contains an example shell script that can be used to modify a distribution path target address. Refer to the example script comments for setting appropriate variable values.

   The script should be placed in the same local directory on **all** Oracle RAC nodes of the primary **and** standby database clusters. Set the script file permissions to `6751`.

   For example:

```
$ chmod 6751 /u01/oracle/goldengate/scripts/change_path_target.sh
```

   The example shell script uses REST API calls to access the GoldenGate distribution path. In order to make the REST API calls secure, it is recommended that you include the GoldenGate deployment administrator user name and password in a configuration file (`access.cfg`), as shown here.

```
$ cat /u01/oracle/goldengate/scripts/access.cfg
user = "oggadmin:<password>"
```

   The `access.cfg` file is also referenced in the database role transition trigger below.

2. Create a DBMS_SCHEDULER job.

   Creating a DBMS_SCHEDULER job is required to run an operating system shell script from within PL/SQL. Create the scheduler job as a SYSDBA user in the root container database (CDB).

   For example:

```
SQL> exec dbms_scheduler.drop_job('gg_change_path_target');
SQL> exec dbms_scheduler.create_job(job_name=>'gg_change_path_target',
job_type=>'EXECUTABLE', number_of_arguments => 6,
job_action=>'/u01/oracle/goldengate/scripts/change_path_target.sh', enabled=>FALSE);
```

   To run an external job, you must set the `run_user` and `run_group` parameters in the `$ORACLE_HOME/rdbms/admin/externaljob.ora` file to the Oracle database operating system user and group.

   For example:

```
run_user = oracle
run_group = oinstall
```

   The `extrernaljob.ora` must be configured on **all** Oracle RAC nodes of the primary **and** standby database clusters.

3. Create the database role transition trigger.

   Create a role transition trigger on the GoldenGate target database that will fire when a standby database becomes

a primary database, changing the distribution path target address, using the following example.

```
CREATE OR REPLACE TRIGGER gg_change_path
AFTER db_role_change ON DATABASE
declare
  role varchar2(30);
  hostname varchar2(64);
begin
  select database_role into role from v$database;
  select host_name into hostname from v$instance;
  DBMS_SCHEDULER.SET_JOB_ARGUMENT_VALUE('gg_change_path_target',1,'<source primary cluster
VIP>');

DBMS_SCHEDULER.SET_JOB_ARGUMENT_VALUE('gg_change_path_target',2,'<source standby cluster
VIP>');
  DBMS_SCHEDULER.SET_JOB_ARGUMENT_VALUE('gg_change_path_target',4,'<dist. path name>');
  DBMS_SCHEDULER.SET_JOB_ARGUMENT_VALUE('gg_change_path_target',5,'<deployment name>');
  DBMS_SCHEDULER.SET_JOB_ARGUMENT_VALUE('gg_change_path_target',6, '<dir/access.cfg>');
  if role = 'PRIMARY' and hostname like '<primary target cluster name>%'
  then

    DBMS_SCHEDULER.SET_JOB_ARGUMENT_VALUE('gg_change_path_target',3,'<primary target
cluster VIP>:443');
  elsif role = 'PRIMARY'
  then
    DBMS_SCHEDULER.SET_JOB_ARGUMENT_VALUE('gg_change_path_target',3,'<standby target
cluster VIP>:443');
  end if;
  DBMS_SCHEDULER.RUN_JOB(job_name=>'gg_change_path_target');
end;
/
```

After creating the database trigger, switch the logfile on the primary database to ensure the code is propagated to the standby database using the following command:

```
SQL> alter system switch all logfile;
```

## Replicat Configuration on the Primary Cluster

As documented in the Oracle GoldenGate Microservices Architecture with Oracle Real Application Clusters Configuration Best Practices technical brief, a checkpoint table in the target database is required for all Oracle GoldenGate Replicat processes. There are no other configuration requirements for Replicat when configured with Oracle Data Guard.

## Conclusion

The combination and integration of Oracle GoldenGate Microservices and Oracle Data Guard enables customers to achieve a Platinum MAA service-level configuration that achieves zero or near zero downtime for all planned and unplanned outages. With the configuration and operational steps described in this paper, Oracle GoldenGate can be configured to work seamlessly with Oracle Data Guard after any zero data loss or data loss role transition. By using DBFS as the file system for the Oracle GoldenGate Microservices deployment files, Oracle GoldenGate Extract, Distribution Paths, and Replicat processes continue to stay synchronized with the database after a role transition.

## References

- Oracle GoldenGate Microservices Architecture with Oracle Real Application Clusters Configuration Best Practices
- Oracle GoldenGate 21c Documentation
- Oracle Database SecureFiles and Large Object Developer's Guide 19c (DBFS)
- Oracle Clusterware Administration and Deployment Guide 19c
- Oracle Data Guard Concepts and Administration 19c
- Oracle Maximum Availability Architecture Web site
  http://www.otn.oracle.com/goto/maa

## Appendix A - Example Distribution Path Target Change Script

The following example script can be used to change a source GoldenGate deployment distribution path target address to reflect the new location of the receiver server after a Data Guard role transition. This example assumes the source GoldenGate deployment is configured in an MAA architecture with Data Guard, such that the distribution server can relocate between a primary and standby cluster.

```
#!/bin/bash
# change_path_target.sh - changes the target host of a GG Distribution Path when the target
#                         moves between primary/standby clusters.
# Example usage:
# ./change_path_target.sh <primary source VIP>:443 <standby source VIP>:443 <path target VIP>
<path name> <deployment name> <credentials file>
SOURCE1=$1      # PRIMARY Distribution Server VIP
SOURCE2=$2      # STANDBY Distribution Server VIP
TARGET=$3       # Distribution path target VIP
DPATH=$4        # Distribution path name
DEP=$5          # Deployment name
ACCESS=$6       # access.cfg file containing the deployment credentials. Example contents:
                # user = "oggadmin:<password>"
CONNECT=0
#echo "#${i} - `date`:"
LOGFILE=/tmp/ogg_dpatch_change.txt
result=$(curl -si -K $ACCESS https://$SOURCE1/$DEP/distsrvr/services/v2/sources/$DPATH -X GET|
grep HTTP |  awk '{print $2}')
# Will return NULL of nginx not running, 502 if cannot contact server, 200 if contact to
server good, and others (404) for other bad reasons:
if [[ -z $result || $result -ne 200 ]]; then  # Managed to access the Distr Server
  echo "`date` - Couldn't contact Distribution Server at $SOURCE1 Deployment $DEP ****" >>
$LOGFILE
else # Try the other source host:
  echo "`date` - Got status of Distribution Server at $SOURCE1 Deployment $DEP ***"  >>
$LOGFILE
  SOURCE=$SOURCE1
  CONNECT=1
fi
if [ $CONNECT -eq 1 ]; then
# For secure NGINX patch destination (wss)
  PAYLOAD='{"target":{"uri":"wss://'${TARGET}'/services/ggnorth/v2/targets?trail=bb"}}'
  curl -s -K $ACCESS https://$SOURCE/$DEP/distsrvr/services/v2/sources/$DPATH -X PATCH --data
'{"status": "stopped"}'
# Set new target for path:
  curl -s -K $ACCESS https://$SOURCE/$DEP/distsrvr/services/v2/sources/$DPATH -X PATCH --data
"$PAYLOAD"
  echo "`date` - Set path $DPATH on $SOURCE deployment $DEP:" >> $LOGFILE
  curl -s -K $ACCESS https://$SOURCE/$DEP/distsrvr/services/v2/sources/$DPATH -X GET | python
-m json.tool | grep uri >> $LOGFILE
  curl -s -K $ACCESS https://$SOURCE/$DEP/distsrvr/services/v2/sources/$DPATH -X PATCH --data
'{"status": "running"}'
```

```
exit 0
else
  echo "`date` - ERROR: COULDN'T CHANGE DISTRIBUTION PATH ($DPATH) in Deployement $DEP at
$SOURCE! ***" >> $LOGFILE
fi
# If here, means we couldn't connect to either Distribution Servers
exit 1
```

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