

Oracle GoldenGate on Oracle
Exadata Database Machine
Configuration

*Oracle Maximum Availability Architecture White Paper
May 2011*

Maximum Availability Architecture

Oracle Best Practices For High Availability

ORACLE

Executive Overview	2
Configuration Overview	3
Oracle GoldenGate.....	3
Oracle Exadata Database Machine	3
Oracle Database File System	4
Oracle Clusterware.....	4
Migrating to Oracle Exadata Database Machine.....	5
Configuration Best Practices	5
Step 1: Set Up DBFS on Oracle Exadata Database Machine	5
Step 2: Configure GoldenGate and Database Parameters	7
Step 3: Install Oracle GoldenGate	9
Step 4: Set Up Checkpoint Files and Trail Files in DBFS.....	9
Step 5: Set Up Discard and Page Files on the Local File System	12
Step 6: Configure Replicat Commit Behavior.....	12
Step 7: Configure Autostart of Extract, Data Pump and Replicat Processes	12
Step 8: Oracle Clusterware Configuration.....	13
Appendix: Example Agent Script	17
References.....	21

Executive Overview

The strategic integration of Oracle Exadata Database Machine and Oracle Maximum Availability Architecture (MAA) best practices (Exadata MAA) provides the best and most comprehensive Oracle Database availability solution.

This white paper describes best practices for configuring Oracle GoldenGate to work with Oracle Exadata Database Machine and Exadata storage. Oracle GoldenGate is instrumental for many reasons, including the following:

- To migrate to an Oracle Exadata Database Machine, incurring minimal downtime
- As part of an application architecture that requires Oracle Exadata Database Machine plus the flexible availability features provided by GoldenGate, such as active-active database for data distribution and continuous availability, and zero or minimal downtime during planned outages for system migrations, upgrades, and maintenance
- To implement a near real-time data warehouse or consolidated database on Oracle Exadata Database Machine, sourced from various, possibly heterogeneous source databases, populated by Oracle GoldenGate
- To capture from an OLTP application running on Oracle Exadata Database Machine to support further downstream consumption such as SOA type integration

This paper focuses on configuring Oracle GoldenGate to run on Oracle Exadata Database Machine. Oracle Exadata Database Machine can act as the source database, as the target database, or in some cases as both source and target databases for GoldenGate processing.

In addition, this paper covers the GoldenGate regular mode of continuously extracting logical changes from either online redo log files or archived redo log files.

Configuration Overview

This section introduces you to Oracle GoldenGate, Oracle Exadata Database Machine, and Oracle Database File System (DBFS). For more information about these features, see the [References](#) section at the end of this white paper.

Oracle GoldenGate

Oracle GoldenGate provides real-time, log-based change data capture, and delivery between heterogeneous systems. Using this technology, it enables a cost-effective and low-impact real-time data integration and continuous availability solution.

Oracle GoldenGate moves committed transactions with transaction integrity and minimal overhead on your existing infrastructure. The architecture supports multiple data replication topologies such as one-to-many, many-to-many, cascading, and bidirectional. Its wide variety of use cases includes real-time business intelligence; query offloading; zero-downtime upgrades and migrations; and active-active databases for data distribution, data synchronization, and high availability. Figure 1 shows the Oracle GoldenGate architecture.

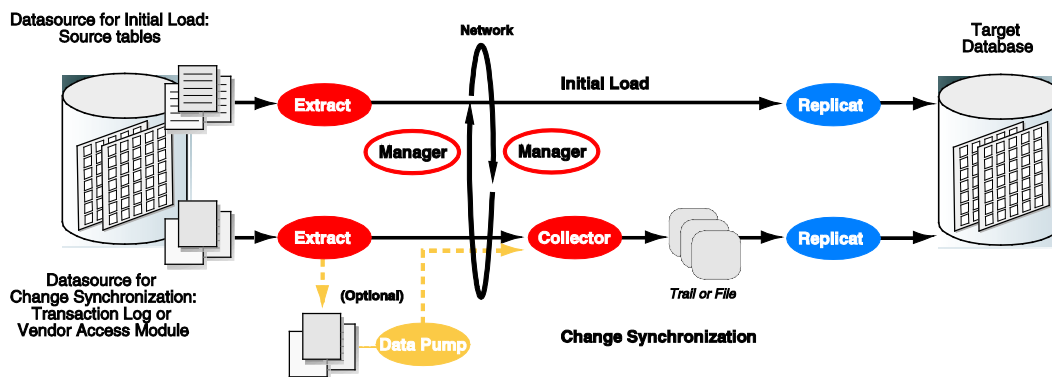


Figure 1. Oracle GoldenGate Architecture

Oracle Exadata Database Machine

Oracle Exadata Database Machine is an easy to deploy, out-of-the-box solution for hosting the Oracle Database for all applications while delivering the highest levels of performance available.

Oracle Exadata Database Machine is a “grid in a box” composed of database servers, Oracle Exadata Storage Servers (Exadata), an InfiniBand fabric for storage networking and all the other components required for hosting an Oracle Database. Oracle Exadata Storage Server is a storage product optimized for use with Oracle Database applications and is the storage building block of Oracle Exadata Database Machine. Exadata delivers outstanding I/O and SQL processing

performance for online transaction processing (OLTP), data warehousing (DW) and consolidation of mixed workloads. Extreme performance is delivered for all types of database applications by leveraging a massively parallel grid architecture using Oracle Real Application Clusters (Oracle RAC), Exadata storage, Exadata Smart Flash Cache, high-speed InfiniBand connectivity, and compression technology

Oracle Database File System

The Oracle Database File System (DBFS) creates a file system interface to files stored in the database. DBFS is similar to NFS in that it provides a shared network file system that looks like a local file system. Because the data is stored in the database, the file system inherits all the HA/DR capabilities provided by the database.

With DBFS, the server is the Oracle Database. Files are stored as SecureFiles LOBs. PL/SQL procedures implement file system access primitives such as create, open, read, write, and list directory. The implementation of the file system in the database is called the DBFS SecureFiles Store. The DBFS SecureFiles Store allows users to create file systems that can be mounted by clients. Each file system has its own dedicated tables that hold the file system content.

Oracle Clusterware

Oracle Clusterware enables servers to communicate with each other, so that they appear to function as a collective unit. This combination of servers is commonly known as a cluster. Although the servers are standalone servers, each server has additional processes that communicate with other servers. In this way the separate servers appear as if they are one system to applications and end users.

Oracle Clusterware provides the infrastructure necessary to run Oracle Real Application Clusters (Oracle RAC). Oracle Clusterware also manages [resources](#), such as virtual IP (VIP) addresses, databases, listeners, services, and so on.

There are APIs to register an application and instruct Oracle Clusterware regarding the way an application is managed in a clustered environment. You use the APIs to register the Oracle GoldenGate Manager process as an application managed through Oracle Clusterware. The Manager process should then be configured to automatically start or restart other GoldenGate processes.

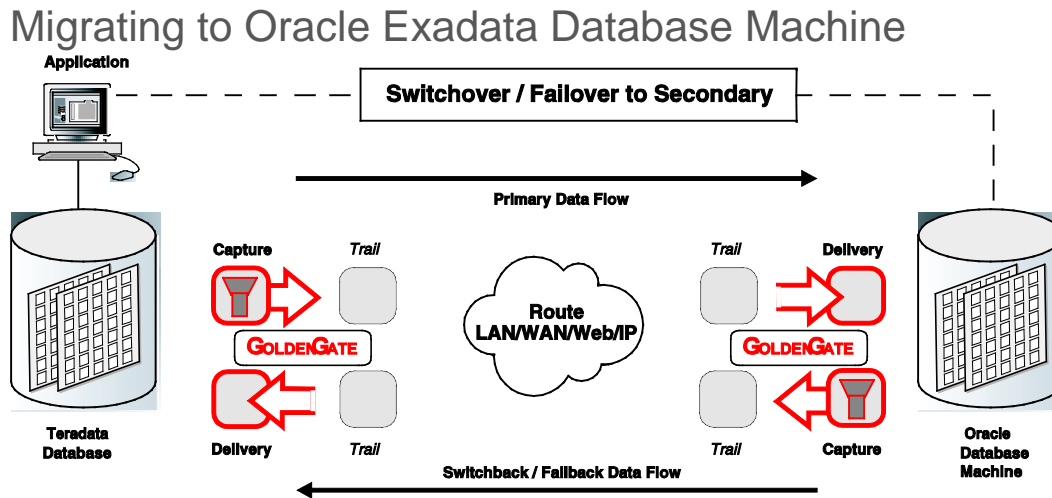


Figure 2. Migrating Oracle GoldenGate to Oracle Exadata Database Machine

Oracle GoldenGate supports an active-passive bidirectional configuration, where GoldenGate replicates data from an active primary database to a full replica database on a live standby system that is ready for failover during planned and unplanned outages. This provides the ability to migrate to Oracle Exadata Database Machine allowing the new system to work in tandem until testing is completed and a switchover planned.

This paper includes instructions for configuring a target system on Oracle Exadata Database Machine that will act as the standby database shown in Figure 2.

Configuration Best Practices

Step 1: Set Up DBFS on Oracle Exadata Database Machine

When setting up the configuration, the best practice is to store the GoldenGate trail files and checkpoint files in DBFS to provide recoverability and failover capabilities in the event of a system failure.

Using DBFS is fundamental to the continuing availability of the checkpoint and trail files in the event of a node failure. Ensuring the availability of the checkpoint files is essential to ensure that, after a failure occurs, the Extract process can continue mining from the last known archived redo log file position and Replicat processes can start applying from the same trail file position before a failure occurred. Using DBFS allows one of the surviving database instances to be the source of an Extract process or destination for the Replicat processes.

See My Oracle Support note [1054431.1](https://support.oracle.com/epos1/docs?id=1054431.1) for configuring DBFS on the Oracle Exadata Database Machine. DBFS should be mounted using `fstab`.

For better performance of trail file and checkpoint file I/O operations it is recommended to change the storage parameters of the LOB segment used by DBFS:

```
-- Connect to the DBFS database
SQL> connect system/<passwd>@<dbfs_tns_alias>

-- View current LOB storage:
SQL> SELECT table_name, segment_name, logging
       FROM dba_lobs WHERE tablespace_name='<dbfs tablespace name>';

-- More than likely it will be something like this:
--
-- TABLE_NAME          SEGMENT_NAME          LOGGING CACHE
-- -----
-- T_GOLDENGATE         LOB_SFSS$_FST_73     YES      NO

-- Alter the LOB segment:
SQL> ALTER TABLE DBFS.<TABLE_NAME> MODIFY LOB (FILEDATA)
       (CACHE LOGGING);

-- View the new LOB storage:
SQL> SELECT table_name, segment_name, logging
       FROM dba_lobs WHERE tablespace_name='<dbfs tablespace name>';

-- TABLE_NAME          SEGMENT_NAME          LOGGING CACHE
-- -----
-- T_GOLDENGATE         LOB_SFSS$_FST_73     YES      YES
```

This paper focuses on using DBFS for the shared file system. However, alternatively, you could place the GoldenGate trail files and checkpoint files on the database machine local file system or an NFS mount point hosted by a server outside of the database machine. Using the local file system provides limited bandwidth with a single drive and offers no high availability capabilities. NFS can be used as a suitable alternative to DBFS for storing the trail files and checkpoint files to allow sharing between the source and target GoldenGate hosts.

Starting with GoldenGate version 11.1.1 a new Bounded Recovery feature was added that guarantees an efficient recovery after Extract stops for any reason, planned or unplanned, no matter how many open (uncommitted) transactions there were at the time that Extract stopped, nor how old they were. Bounded Recovery sets an upper boundary for the maximum amount of time that it would take for Extract to recover to the point where it stopped and then resume normal processing. The Bounded Recovery checkpoint files should be placed on a shared file system such that, in an event of a failover when there are open long running transactions Extract can use Bounded Recovery to reduce the time to taken to perform recovery. At this time, DBFS is not supported for a Bounded Recovery checkpoint file location, so using something like NFS is suggested. It is possible to store the checkpoint files on local file system, and when Extract performs recovery after a node failure, the standard checkpoint mechanism will be used until

new local Bounded Recovery checkpoint files are subsequently created. This will only be noticeable if there are long running transactions at the time of the failure.

For more information on Bounded Recovery refer to the Oracle GoldenGate Reference Guide:

http://download.oracle.com/docs/cd/E18101_01/doc.1111/e17791.pdf

Note: If you are using a GoldenGate Data Pump process to transfer the trail files from a source host on the database machine using DBFS, then contact Oracle Support to obtain the fix to Bug 10146318. This bug fix improves trail file creation performance on DBFS by the GoldenGate server/collector process.

Step 2: Configure GoldenGate and Database Parameters

When running GoldenGate in regular mode, there is no need to explicitly set additional database parameters. However, consider setting up the following options:

- Use the default Oracle Automatic Storage Manager (Oracle ASM) naming convention for the archived redo log files.
- Configure the GoldenGate Extract parameter for the new Oracle ASM log read API.

GoldenGate release 11.1.1 introduces a new method of reading log files stored in Oracle ASM. This new method uses the database server to access the redo and archived redo log files, instead of connecting directly to the Oracle ASM instance. The database must contain the libraries with the API modules. The libraries are currently included with Oracle Database release 10.2.0.5 and 11.2.0.2. Contact Oracle Support to request a backport for Bug 9397368 to obtain the libraries for releases (after release 10.2.0.5).

To successfully mine the Oracle archived redo log files located on the storage cells that are managed by Oracle ASM, configure the GoldenGate Extract parameter as follows:

1. Set the `TRANLOGOPTIONS` parameter to specify use of the new log read API. For example:

```
TRANLOGOPTIONS DBLOGREADER
```

- Configure the GoldenGate Extract parameter for the old Oracle ASM log read API.

It is also recommended to configure GoldenGate with the old log read API (below). This way, if the source database is unavailable, the Extract process can continue to mine the log files using the old API. Comment the Extract parameters to disable them until they are needed.

To set the Oracle ASM parameters with the old log read API:

1. Set the `TRANLOGOPTIONS` parameter to specify the Oracle ASM logon information in the Extract parameter file. For example:

```
TRANLOGOPTIONS ASMUSER sys@ASM, ASMPASSWORD g32adrwe, ENCRYPTKEY
DEFAULT
```

Comment this parameter to disable the old log read API until it is needed.

2. If a password is not already in use by the Oracle ASM instance, create it using `orapwd` and set the `REMOTE_LOGIN_PASSWORDFILE` initialization parameter to `SHARED` or `EXCLUSIVE`.

A password file is required for connection to the Oracle ASM instance by GoldenGate.

3. Configure the `LISTENER.ORA` and `TNSNAMES.ORA` files for remote connections to the ASM instance to work, as shown in the following examples:

Example listener.ora:

```
SID_LIST_LISTENER =
  (SID_LIST =
    (SID_DESC =
      (SID_NAME = PLSExtProc)
      (ORACLE_HOME = /app/oracle/grid)
      (PROGRAM = extproc)
    )
    (SID_DESC =
      (ORACLE_HOME = /app/oracle/grid)
      (SID_NAME = +ASM1)
    )
  )

LISTENER =
  (DESCRIPTION_LIST =
    (DESCRIPTION =
      (ADDRESS=(PROTOCOL=TCP)(HOST=ggtest)(PORT=1521))
      (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC0))
    )
  )
```

Example TNSNAMES.ORA

```
ASM = (DESCRIPTION =
  (ADDRESS = (PROTOCOL = TCP)(HOST = ggtest)(PORT
= 1521))
  (CONNECT_DATA =
    (SERVER = DEDICATED)
    (SERVICE_NAME = ASM)
    (INSTANCE_NAME = +ASM1)
  )
)
```

)

Step 3: Install Oracle GoldenGate

1. Download the GoldenGate software from Oracle Technology Network (OTN) at:
<http://www.oracle.com/technetwork/middleware/goldengate/downloads/index.html>
2. Install GoldenGate locally on the primary source and target nodes in the Oracle RAC configuration. Make sure the installation directory is the same on all nodes.
3. Once you have successfully configured GoldenGate on the primary source and/or target nodes, shut down Extract/Replicat and copy the entire GoldenGate home directory to the other source and target nodes.
4. Follow the generic installation instructions for the source and target machine installations available in Chapter 2: “Installing GoldenGate” at:
http://download.oracle.com/docs/cd/E18101_01/doc.1111/e17799.pdf

Step 4: Set Up Checkpoint Files and Trail Files in DBFS

1. Set up checkpoint files

Checkpoint files contain the current read and write positions of the Extract and Replicat processes. Checkpoints provide fault tolerance by preventing the loss of data should the system, the network, or a GoldenGate process need to be restarted.

Placing the checkpoint files on the local file system will not provide high availability in the event of a database node failure. A checkpoint table can be used to record Replicat checkpoint information to provide an alternative method of fault tolerance.

To store the checkpoint files on DBFS, the best practice is to create a symbolic link from the GoldenGate home directory to a directory in DBFS. For example:

```
# Ensure the DBFS file system is already mounted
# In this example, the DBFS mount point is /mnt/dbfs
% mkdir /mnt/dbfs/goldengate/dirchk
% cd /GoldenGate/v11_1_1_0
% rm -rf dirchk
% ln -s /mnt/dbfs/goldengate/dirchk dirchk
```

Note: GoldenGate uses file locking on the checkpoint files to determine if the Extract or Replicat processes are already running. This would normally prevent the process from being started a second time on another Oracle RAC node that has access to the checkpoint files. DBFS does not support this method of file locking. Mounting DBFS on a single Oracle RAC node prevents access to the checkpoint files from other nodes. This will in turn prevent the extract or replicat from being started concurrently on multiple nodes.

2. Set up trail files.

Trail files contain the data extracted from the archived redo log files. The trail files are automatically generated by the Extract process.

Store the trail files on DBFS. By mounting the same DBFS directory on both the source and target databases, much like an NFS mount, the Replicat process can read from the same trails created by the Extract process. This removes the need for the GoldenGate Data Pump if both the source and target databases run in the same Oracle Exadata Database Machine.

If the source and target databases are not part of the same Exadata Database Machine, then use a GoldenGate Data Pump to transfer the trail files between the hosts. Be sure to use DBFS to provide fault tolerance for the trail files.

To configure GoldenGate trail files on DBFS for the source database:

1. Create a DBFS directory:

```
# Ensure DBFS file system is already mounted
# In this example, the DBFS mount point is /mnt/dbfs
% mkdir /mnt/dbfs/goldengate/dirdat
```

2. Set the EXTTRAIL Extract parameter:

```
EXTTRAIL /mnt/dbfs/goldengate/dirdat/aa
```

3. After creating the Extract, use the same EXTTRAIL parameter value to add the local trail:

```
% ggsci
GGSCI (ggtest.oracle.com) 1> ADD EXTTRAIL
/mnt/dbfs/goldengate/dirdat/aa, EXTRACT ext_db, Megabytes
500
```

Further instructions about creating the Extract are available in the *Oracle GoldenGate Administration Guide* at

http://download.oracle.com/docs/cd/E18101_01/doc.1111/e17341.pdf

To configure GoldenGate trail files on DBFS for the target database:

1. Make sure the DBFS directory is already created on the source database
2. Set the EXTTRAIL Replicat parameter, as follows:

```
EXTTRAIL /mnt/dbfs/goldengate/dirdat/aa
```

3. When adding the Replicat, use the same EXTTRAIL parameter value:

```
% ggsci
```

```
GGSCI (ggtest.oracle.com) 1> ADD REPLICAT rep_db1, EXTTRAIL  
/mnt/dbfs/goldengate/dirdat/aa
```

Do not place trail files on the local file system because it will lengthen restart times in the event of a node failure, reducing availability.

To configure Data Pump between a source and target database outside the same Exadata Database Machine:

1. Make sure Extract and Replicat are configured
2. Set the RMTHOST Data Pump parameter to the IP or hostname that will be used for connecting to the target. In Step 7 below, the Application Virtual IP address is created with Cluster Ready Services (CRS) so that a single IP address can be moved between compute nodes, so that Data Pump can continue to connect to the target host when it moves from a failed node to a surviving node:

```
RMTHOST gg_dbmachine, MGRPORT 8901
```

3. Set the RMTTRAIL Data Pump parameter to the trail file location on the target host:

```
RMTTRAIL /mnt/dbfs/goldengate/dirdat/aa
```

4. Create a Data Pump process using the local trail file location on the source host:

```
% ggsci  
GGSCI (ggtest.oracle.com) 1> ADD EXTRACT dpump_1,  
EXTTRAILSOURCE /mnt/dbfs/goldengate/dirdat/aa
```

5. Use the ADD RMTTRAIL command to specify the remote trail file location on the target host:

```
% ggsci
```

```
GGSCI (ggtest.oracle.com) 1> ADD RMTTRAIL  
/mnt/dbfs/goldengate/dirdat/aa EXTRACT dpump_1, MEGABYTES 500
```

Further instructions about creating the Data Pump process are available in the *Oracle GoldenGate Administration Guide* at

http://download.oracle.com/docs/cd/E18101_01/doc.1111/e17341.pdf

Step 5: Set Up Discard and Page Files on the Local File System

Discard files are used by GoldenGate to record any operations that failed. The discard file is most useful for the Replicat process to help resolve data errors, such as an invalid column mapping.

Because GoldenGate only replicates transactions that are committed, the capture component stores the operations of each transaction in a virtual-memory pool known as a cache until it receives a commit or rollback for that transaction. When the cache becomes full, transactions can be paged out to disk. By default, the files are stored in the `dirtmp` sub-directory of the GoldenGate installation directory.

Page files cannot be stored in DBFS because it is a memory mapped file, a file type that is not currently supported by DBFS. Store both discard and page files on the database server local file system within the GoldenGate installation directory structure.

For more details, see `CACHEMGR` in the *Oracle GoldenGate Reference Guide* at

http://download.oracle.com/docs/cd/E18101_01/doc.1111/e17791.pdf

Step 6: Configure Replicat Commit Behavior

If using GoldenGate version 11.1.1 or lower it is recommended to set the Replicat commit behavior to `COMMIT NOWAIT`. The Replicat processes will no longer wait at each commit when applying transactions, increasing throughput performance. This should only be considered when using a checkpoint table due to protection of recovery data during a checkpoint.

Set the Replicat parameter file to `COMMIT NOWAIT` as follows:

```
SQLEXEC "ALTER SESSION SET COMMIT_WRITE='NOWAIT'";
```

Step 7: Configure Autostart of Extract, Data Pump and Replicat Processes

Configure the Extract, Data Pump (if used) and Replicat processes to automatically start when the Manager process is started. Add the following parameter to the Manager parameter file:

```
AUTOSTART ER *
```

Step 8: Oracle Clusterware Configuration

The following step-by-step procedure shows how to instruct Oracle Clusterware to start GoldenGate, check whether it is running, and stop it. It provides an example shell script to carry out these functions, including registering the application with Oracle Clusterware and managing switchover and failover between the Oracle RAC nodes.

1. Create an Application VIP

If the source system is outside of Oracle Exadata Database Machine and it uses GoldenGate Data Pump to transfer the trail file data to the target database machine, an application VIP is required to ensure the remote data pumps can communicate with the target database machine, regardless of which node is hosting GoldenGate.

An application virtual internet protocol address (VIP) is a cluster resource that Oracle Clusterware manages. The VIP is assigned to a cluster node and will be migrated to another node in the event of a node failure. This allows GoldenGate data pump to continue transferring data to the newly assigned target node.

If both the Extract and Replicat processes are running within the same database machine and Data Pump is not used, then there is no need to create the Application VIP and you can skip to [step 2 to create an agent script](#). Otherwise, perform the following steps:

- a) To create the application VIP, run the following as the `root` user:

```
$GRID_HOME/bin/appvipcfg create -network=1 \
    -ip=10.1.41.93 \
    -vipname=ggatevip \
    -user=root
```

In the example:

- `$GRID_HOME` is the Oracle home in which Oracle 11g Release 2 Grid infrastructure components have been installed (for example: `/u01/app/grid`).
- `network` is the network number that you want to use. With Oracle Clusterware release 11.2.0.1, you can find the network number using the following command:

```
crsctl stat res -p |grep -ie .network -ie subnet |grep -ie name -ie subnet
```

Consider the following sample output:

```
NAME=ora.net1.network
USR_ORA_SUBNET=10.1.41.0
```

`net1` in `NAME=ora.net1.network` indicates this is network 1, and the second line indicates the subnet on which the VIP will be created.

- o `ip` is the IP address provided by your system administrator for the new Application VIP. This IP address must be in the same subnet as determined above.
 - o `Ggatevip` is the name of the application VIP that you will create.
- b) Run the following command to give the Oracle Database installation owner permission to start the VIP:

```
$GRID_HOME/bin/crsctl setperm resource ggatevip -u user:oracle:r-x
```

- c) As the Oracle Database installation owner, start the VIP resource:

```
$GRID_HOME/bin/crsctl start resource ggatevip
```

- d) To validate whether the VIP is running and on which node it is running, execute:

```
$GRID_HOME/bin/crsctl status resource ggatevip
```

See the Oracle Clusterware documentation for further details about creating an Application VIP:

http://download.oracle.com/docs/cd/E11882_01/rac.112/e10717/crschp.htm#BGBHIBEE

2. Create an agent script

Oracle Clusterware runs resource-specific commands through an entity called an *agent*. The agent script:

- Must be able to accept five parameter values: `start`, `stop`, `check`, `clean` and `abort`.
- Must be stored in the same location on all nodes. In this example, it is stored in the Grid Infrastructure (`$GRID_HOME`) `ORACLE_HOME/crs/script` directory.
- Must be owned by the Oracle user and have execute permissions.
- Must be accessible at the same location on every node in the cluster.
- Must include environment variable settings for `ORACLE_HOME`, `ORACLE_SID`, `PATH`, `TNS_ADMIN` and `LD_LIBRARY_PATH` so that CRS will be able to find the correct program executables and Oracle Net configuration. If the correct `sqlnet.ora`, `tnsnames.ora` or `dbfs_client` executable cannot be found when CRS tries to mount DBFS it will fail.

See the [Appendix](#) for an example agent script that mounts and unmounts a DBFS file system upon startup and failover, and starts and stops the GoldenGate Manager, Extract, Data Pump and Replicat processes.

It is important to manually test the agent script to start and stop the GoldenGate processes as well as mounting and dismounting DBFS before moving onto the next step.

3. Register a resource in Oracle Clusterware

Register Oracle GoldenGate as a resource in Oracle Clusterware using the `crsctl` utility.

When using DBFS to store the GoldenGate trail and checkpoint files, there is a start dependency on the DBFS database. It is recommended to include the DBFS instance as the start dependency for the new Oracle Clusterware resource.

- a. Use the Oracle Grid infrastructure user (`oracle`, in this example) to execute the following:

```
$GRID_HOME/bin/crsctl add resource ggateapp_ext \  
    -type cluster_resource \  
    -attr  
"ACTION_SCRIPT=/u01/app/11.2.0/grid/crs/script/11gr2_gg_action.scr,  
CHECK_INTERVAL=30, START_DEPENDENCIES='hard(ggappvip,ora.dbfs.db)  
pullup(ggappvip)' STOP_DEPENDENCIES='hard(ggappvip)'  
SCRIPT_TIMEOUT=300"
```

If an Application VIP is not used, issue the following command:

```
$GRID_HOME/bin/crsctl add resource ggateapp_ext \  
    -type cluster_resource \  
    -attr  
"ACTION_SCRIPT=/u01/app/11.2.0/grid/crs/script/11gr2_gg_action.scr,  
CHECK_INTERVAL=30, START_DEPENDENCIES='hard(ora.dbfs.db)'  
SCRIPT_TIMEOUT=300"
```

- b. To determine the name of the DBFS resource for the start dependency:

```
crsctl status resource | grep -i dbfs
```

This paper assumes a single Oracle Exadata Database Machine is used for either a source (Extract) or target (Replicat) host. If the database machine is split into separate clusters such that the source and target run within the same database machine, then make sure the Extract and Replicat is restricted to the designated clustered nodes:

```
$GRID_HOME/bin/crsctl add resource ggateapp_ext \  
    -type cluster_resource \  
    -attr  
"ACTION_SCRIPT=/u01/app/11.2.0/grid/crs/script/11gr2_gg_action.scr,  
CHECK_INTERVAL=30, START_DEPENDENCIES='hard(ora.dbfs.db)'  
HOSTING_MEMBERS='testbox03 testbox04' PLACEMENT='restricted'  
SCRIPT_TIMEOUT=300"
```

Be sure to add the appropriate start and stop dependencies if an Application VIP is used.

For more information about the `crsctl add resource` command and its options, see the [Oracle Clusterware Administration and Deployment Guide](#) at

http://download.oracle.com/docs/cd/E11882_01/rac.112/e10717/toc.htm

4. Start the resource

Once the resource has been added, you should always use Oracle Clusterware to start Oracle GoldenGate. Login as the Oracle Grid infrastructure user (`oracle`) and execute the following:

```
$GRID_HOME/bin/crsctl start resource ggateapp_ext
```

To check the status of the application, enter the command:

```
$GRID_HOME/bin/crsctl status resource ggateapp_ext
```

For example:

```
[oracle@testbox04]$ crsctl status resource ggateapp_ext
NAME=ggateapp_ext
TYPE=cluster_resource
TARGET=ONLINE
STATE=ONLINE on testbox04
```

5. Manage the application

To relocate Oracle GoldenGate onto a different cluster node, use the '`$GRID_HOME/bin/crsctl relocate resource`' API and include the `force` option. This command can be run on any node in the cluster as the Grid Infrastructure user (`oracle`).

For example:

```
[oracle@testbox04]$ crsctl relocate resource ggateapp_ext -f
CRS-2673: Attempting to stop 'ggateapp_ext' on 'testbox04'
CRS-2677: Stop of 'ggateapp_ext' on 'testbox04' succeeded
CRS-2672: Attempting to start 'ggateapp_ext' on 'testbox03'
CRS-2676: Start of 'ggateapp_ext' on 'testbox03' succeeded
```

To stop the Oracle GoldenGate resource, enter the following command:

```
$GRID_HOME/bin/crsctl stop resource ggateapp_ext
```

6. CRS Cleanup

To remove GoldenGate from Oracle Clusterware management, perform the following tasks:

- a) Stop GoldenGate (login as the Oracle Grid infrastructure (`oracle`) user):

```
$GRID_HOME/bin/crsctl stop resource ggateapp_ext
```

- b) As the `root` user, delete the application `ggateapp`:

```
$GRID_HOME/bin/crsctl delete resource ggateapp_ext
```

- c) If no longer needed, delete the agent action script: `llgr2_gg_action.scr`.

This does not delete the GoldenGate or DBFS configuration.

Note: Ensure the GoldenGate software was installed in the same directory on all nodes that may run the processes after a failure. Also ensure that the Manager, Extract, Data Pump, and Replicat parameter files are up to date on all nodes.

Recommendations When Deploying on Oracle RAC

When GoldenGate is configured in an Oracle RAC environment, follow these recommendations:

- Ensure the DBFS database has instances on all the database nodes involved in the Oracle RAC configuration.

This action provides access to GoldenGate if it is restarted after a node failure.

- Ensure the DBFS file system is mountable on all database nodes in the Oracle RAC configuration.

To prevent the Extract or Replicat processes from being started on multiple nodes concurrently, mount the file system only on the node where GoldenGate is running. Use the same mount point names on all the nodes to ensure seamless failover.

Appendix: Example Agent Script

The following example agent script mounts and unmounts a DBFS file system upon startup and failover, as well as starting and stopping the GoldenGate Manager, Extract and Replicat processes. The agent script accepts the parameter values: `start`, `stop`, `check`, `clean` and `abort`.

```
#!/bin/bash
#llgr2_gg_action.scr

# Edit the following environment variables:
export ORACLE_HOME=/u01/app/oracle/product/11.2.0.2/streams_0820
export ORACLE_SID=STRMSB1
GGS_HOME=/home/oracle/goldengate/latest
export PATH=$ORACLE_HOME/bin:$PATH
export TNS_ADMIN=$ORACLE_HOME/network/admin

#Include the GoldenGate home in the library path to start GGSCI
export LD_LIBRARY_PATH=${ORACLE_HOME}/lib:${LD_LIBRARY_PATH}:${GGS_HOME}

# Edit this to indicate the DBFS mount point
DBFS_MOUNT_POINT=/dbfs_direct

# Edit this to indicate the file system mounted in the DBFS mount point
DBFS_FILE_SYSTEM=/dbfs_direct/goldengate
```

```

# Edit for correct Extract name if running this script on the source:
EXTRACT=EXT

# OR edit for current Replicat names if running script on the target:
REPLICAT=REP

#specify delay after start before checking for successful start
start_delay_secs=5

#check_process validates that Manager/Extract/Replicat process is running
#at PID that GoldenGate specifies.
check_process () {
PROCESS=$1
  if [ ${PROCESS} = mgr ]
  then
    PFILE=MGR.pcm
  elif [ ${PROCESS} = ext ]
  then
    PFILE=${EXTRACT}*.pce
  else
    PFILE=${REPLICAT}*.pcr
  fi
  if ( [ -f "${GGS_HOME}/dirpcs/${PFILE}" ] )
  then
    pid=`cut -f8 "${GGS_HOME}/dirpcs/${PFILE}"`
    if [ ${pid} = `ps -e |grep ${pid} |grep ${PROCESS} |cut -d " " -f2` ]
    then
      #process is running on the PID <96> exit success
      exit 0
    else
      if [ ${pid} = `ps -e |grep ${pid} |grep ${PROCESS} |cut -d " " -f1`
]
      then
        #process is running on the PID <96> exit success
        exit 0
      else
        #process is not running on the PID
        exit 1
      fi
    fi
  else

```

```
#process is not running because there is no PID file
exit 1
fi
}

#call_ggsci is a generic routine that executes a ggsci command
call_ggsci () {
    ggsci_command=$1
    ggsci_output=`${GGS_HOME}/ggsci << EOF
                ${ggsci_command}
                exit
                EOF`
}

#mount_dbfs will mount the DBFS file system if it has not yet been
#mounted
mount_dbfs () {
    if ( [ ! -d ${DBFS_FILE_SYSTEM} ] )
    then
        #this assumes the DBFS mount point was added to fstab
        #will not mount automatically upon reboot because fuse does not
        #support this; use Oracle wallet for automatic DBFS database login
        mount ${DBFS_MOUNT_POINT}
    fi
}

#unmount_dbfs will unmount the DBFS file system
unmount_dbfs () {
    if ( [ -d ${DBFS_FILE_SYSTEM} ] )
    then
        fusermount -u ${DBFS_MOUNT_POINT}
    fi
}

stop_everything () {
    # Before starting, make sure everything is shutdown and process files are
    removed
    #attempt a clean stop for all non-manager processes
    call_ggsci 'stop er *'

    #ensure everything is stopped
    call_ggsci 'stop er *!'
```

```
#in case there are lingering processes
call_ggsci 'kill er *'

#stop Manager without (y/n) confirmation
call_ggsci 'stop manager!'

#Remove the process files:
rm -f $GGS_HOME/dirpcs/MGR.pcm
rm -f $GGS_HOME/dirpcs/*.pce
rm -f $GGS_HOME/dirpcs/*.pcr
}
case $1 in
'start')
#mount the DBFS file system if it is not yet mounted
mount_dbfs

# stop all GG processes and remove process files
stop_everything
sleep ${start_delay_secs}

#Now can start everything...
#start Manager
call_ggsci 'start manager'

#there is a small delay between issuing the start manager command
#and the process being spawned on the OS <96> wait before checking
sleep ${start_delay_secs}

#start Extract or Replicats
call_ggsci 'start er *'

#check whether Manager is running and exit accordingly
check_process mgr
sleep ${start_delay_secs}

#Check whether Extract is running
check_process ext
;;
'stop')
# stop all GG processes and remove process files
```

```
stop_everything

#unmount DBFS
unmount_dbfs

#exit success
exit 0
;;
'check')
  check_process mgr
  check_process ext
  check_process rep
  ;;
'clean')
  # stop all GG processes and remove process files
  stop_everything

#unmount DBFS
unmount_dbfs

#exit success
exit 0
;;
'abort')
  # stop all GG processes and remove process files
  stop_everything

#unmount DBFS
unmount_dbfs

#exit success
exit 0
;;
esac
```

References

- [Oracle GoldenGate Administration Guide version 11.1.1](#)
- [Oracle GoldenGate Oracle Installation and Setup Guide version 11.1.1](#)
- [Oracle GoldenGate Reference Guide version 11.1.1](#)

- [Oracle Database SecureFiles and Large Object Developer's Guide \(DBFS\)](#)
- [Oracle Clusterware Administration and Deployment Guide](#)
- Oracle Maximum Availability Architecture Web site
<http://www.otn.oracle.com/goto/maa>



Oracle GoldenGate on Oracle Database
Machine Configuration
May 2011
Author: Stephan Haisley
Contributing Authors: Mark Van de Wiel,
MAA team

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

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



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

Copyright © 2011, Oracle and/or its affiliates. All rights reserved. This document is provided for information purposes only and the contents hereof are subject to change without notice. This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission.

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