Rolling Patch - OPatch Support for RAC (Doc ID 244241.1)

APPLIES TO:

Oracle Database Cloud Service - Version N/A and later
Oracle Database - Enterprise Edition
Oracle Database Cloud Schema Service - Version N/A and later
Oracle Database Exadata Express Cloud Service - Version N/A and later
Gen 1 Exadata Cloud at Customer (Oracle Exadata Database Cloud Machine) - Version N/A and later
Information in this document applies to any platform.

PURPOSE

This note is to describe the current support of **OPatch** for Real Application Clusters. Before reading opatch RAC Support, you should be familiar with single-instance OPatch processing.

For more information about OPatch, please refer to Note 189489.1 - Oracle Data Server Interim Patch Installation (OPatch)

For more information about Oracle Clusterware (CRS) Rolling Upgrades, please refer to Note 338706.1

Also, refer to: note 1339140.1 - FAQ: OPatch/Patch Questions/Issues for Oracle Clusterware (Grid Infrastructure or CRS) and RAC Environments

NOTE: The information in this document pre-dates the existence of 'opatchauto' and is therefore only relevant to versions 12.1.0.X and lower when applying patches using a manual (non-opatchauto) methodology.

SCOPE

This document is intended for DBAs, System Administrators and Oracle Support Engineers who are going to apply Oracle Interim Patche(s) on RAC environment.

DETAILS

1 - RAC Patching methods

OPatch supports 3 different patch methods on a RAC environment:

- Patching RAC as a single instance (All-Node Patch)

  In this mode, OPatch applies the patch to the local node first, then propagates the patch to all the other nodes, and finally updates the inventory. All instances must be down during the whole patching process.

- Patching RAC using a minimum down-time strategy (Min. Downtime Patch)

  In this mode, OPatch patches the local node, asks users for a sub-set of nodes, which will be the first subset of nodes to be patched. After the initial subset of nodes are patched, Opatch propagates the patch to the other nodes and finally updates the inventory. The downtime would happen between the shutdown of the second subset of nodes and the startup of the initial subset of nodes patched.

- Patching RAC using a rolling strategy - No down time (Rolling Patch)

  With this method, there is no downtime. Each node would be patched and brought up while all the other nodes are up and running, resulting in no disruption of the system.

Rolling patching strategy incur no downtime, however, some rolling patches may incur downtime due to post-installation steps, i.e. running sql scripts to patch the actual database. Please refer to patch readme to find out whether post-installation steps requires downtime or not.

2 - Flow diagrams

- All-Node Patch

  . Shutdown all Oracle instances on all nodes
  . Apply the patch to the RAC home on all nodes
  . Bring all instances up

- Minimum downtime

  . Shutdown all the Oracle instances on node 1
  . Apply the patch to the RAC home on node 1
  . Shutdown all the Oracle instances on node 2
  . Apply the patch to the RAC home on node 2
  . Shutdown all the Oracle instances on node 3
  . At this point, instances on nodes 1 and 2 can be brought up
  . Apply the patch to the RAC home on node 3
  . Startup all the Oracle instances on node 3

- Rolling patch (no downtime)

  . Shutdown all the Oracle instances on node 1
  . Apply the patch to the RAC home on node 1
  . Start all the Oracle instances on node 1
  . Shutdown all the Oracle instances on node 2
  . Apply the patch to the RAC home on node 2
  . Start all the Oracle instances on node 2
  . Shutdown all the Oracle instances on node 3
  . Apply the patch to the RAC home on node 3
  . Start all the Oracle instances on node 3

3 - How does OPatch select which method to use?

To be eligible as a rolling patch, the patch needs to meet certain criterias, which are determined by Oracle developers. In order to be applied in a "rolling fashion", the patch must be designated as a "rolling updatable patch" or simply "rolling patch".

The algorithm used to decide which method is going to be used is the following:

```
If (users specify minimize_downtime)
      patching mechanism = Min. Downtime
else if (patch is a rolling patch)
      patching mechanism = Rolling
    else
         patching mechanism = All-Node
```

4 - Availability of rolling patches

When patches are released, they have a tag as "rolling" or "not rolling" patch. While most patches can be applied in a rolling fashion, some patches can not be applied in this fashion. Patches that could potentially be installed on rolling fashion include:

. Patches that do not affect the contents of the database.
. Patches that are not related to the RAC internode communication infrastructure.
. Patches that change procedural logic and do not modify common header definitions of kernel modules. This includes client side patches that only affect utilities like export, import, sql*plus, sql*loader, etc.

Only individual patches -- not patch sets -- will be “rollable”. It should also be noted that a merge patch of a “rolling patch” and an ordinary patch will not be a “rolling patch”.

From 9.2.0.4 onwards, all patches released will be marked as a "rolling" or "not rolling patch", based on defined set of rules. Patches previously released are packaged as "not rolling".

Because the set of rules currently defined are very conservative, patches released as "not rolling patches", either before and after 9.2.0.4, may be eligible to be re-released as "rolling patches", after analysis from Oracle Development.

If you plan to apply a patch that is marked as "not rolling" and want to check if is possible to take advantage of the rolling patch strategy, please contact Oracle Support.

5 - How to determine if a patch is a "rolling patch" or not?

As database user execute the following:

- 9i or 10gR1: opatch query -is_rolling

- 10gR2: opatch query -all [unzipped patch location] | grep rolling

- 10gR2 on Windows: opatch query -all [unzipped patch location] | findstr rolling

- Later 10gR2 or 11g: opatch query -is_rolling_patch [unzipped patch location]

The command may not work if unzipped patch location has more than one patch sub-directory, example output while checking CPU patches:

```
Failed to load the patch object.  Possible causes are:
  The specified path is not an interim Patch shiphome
  Meta-data files are missing from the patch area
  Patch location = /home/oracle/stage/8836308
  Details = Input metadata files are missing.

Patch Location "/home/oracle/stage/8836308" doesn't point to a valid patch area.

OPatch failed with error code 75
```

Please refer to patch readme to find out whether the patch is rolling patch or not.

6 - Current Limitations

- Patching with Shared File System

  Currently OPatch treats Shared File System, like CFS, as a single-instance patch. It means that OPatch will blindly patch files under a given ORACLE_HOME knowing that other nodes will pick up the changes via the Shared File System. Unfortunately, this means that OPatch cannot take advantage of a rolling patch on a Shared File System environment; all nodes must be down throughout the patching process.

- Patching one node at time

  The Opatch strategies discussed above (All-Node, Min. Down-Time, and Rolling) presumes that all nodes will be patched within 1 opatch command. Additionally, each node can be patched individually, at different times, using the "-local" key word, which will patch only the local node.

7. Default opatch behavior in cluster environment

In cluster environment, when applying/rolling back a patch, by default opatch will perform on all nodes unless option "-local" is specified.

However, this is changed in opatch 11.2.0.3.16, 12.2.0.1.9 onward; with the change, by default opatch will work only on local node.

REFERENCES

Didn't find what you are looking for?