

ORACLE[®]

Patching and HA environments

March 2010

Philippe Brys
OSS



Patching

Agenda

- Common types of patches
- The opatch utility
- Prerequisites for a patch installation
- Windows patching
- Patching in RAC/CRS environments
- Patching in dataguard environments



Types of patches

- PSE or one-off patches
- MRL merges or Cumulative patches
- Bundles Patches: a collection of critical patches which should be applied to a patchset
 - CPU
 - CRS Bundle
 - PSU
- Recommended patches



Acronyms around patches

- PSE – Patch Set Exception. A one-off patch. That is, delivery of a bug fix or merge of bug fixes on a particular version/platform combination.
- OOB – One Off Backport. A request for a patch filed by Support
- BLR – Base Label Request. A bug fix done on top of a Release Label in preparation of providing a customer a patch.
- RFI – Request For Inclusion. A request to include fix in a future patchset/bundle
- MLR – Merge Label Request. A merge of bug fixes done on top of a Release Label. Needed when the source files for different fixes overlap.
- CPU – Critical Patch Updates (security flaws)
- PSU – Patch Set Updates



Patch Set Exception

- Fix one particular bug
- Normally require to change a subset of routines in Oracle code and relink that code
- Is delivered as it, i.e. only tested in limited development env
- it will fix the bug since the code is logically changed in function of the bug diagnostic
- We don't know whether it can have impact on other Oracle functions. It is however rare to have side effects due to modular design of Oracle software



MRL or Cumulative Patches

- Patch merging requires when two or more bugs change the same code routine
- Driven by customer requests, I.e. some customers use some functionality extensively and fall on bugs in the same area
- Around 2 to 10 patches per merge



Bundle Patches and PSU

- Bundles are merge of important patches for some Oracle functionality
 - CPU: security related napply patches,
 - CRS: patching CRS and RAC functions
 - Dataguard
 - Stream
- Snapshot of a Patch Set Codeline – Includes all fixes in the Patch Set up to the point of the snapshot of the chosen code boundary
- Driven by Oracle Development and tested
- PSU: Sum of Bundle Patches. Is always a superset of CPU and is intended to replace the CPU. New release every 3months
- Oracle Recommended patches are thus PSU or Bundles (see note:756671.1)



Opatch utility

- Utility to install/rollback/report the patches
 - Performs prechecks before applying a patch
 - Permits conflict detection (patch overlaps) and propose the most suitable way to handle them
 - Installed patches are registered in the runinstaller inventory
 - Can undo a patch (rollback)
 - Shipped with the Oracle software distribution (in `$ORACLE_HOME/Opatch`) or via dedicated patch numbers
- It is a JAVA/perl utility in 10.1 and JAVA only utility from 10.2 to 11.2
- Is using OS commands like `ar/fuser/make` (`unzip` in 10.1)
- Is integrated with Oracle Configuration Manager : tool to collect configuration information and link it to the customer CSI for support



Opatch Versions

- Note 224346.1 Opatch - Where Can I Find the Latest Version of Opatch? Via patch:6880880
- The Opatch shipped via the \$ORACLE_HOME/Opatch is never the latest version, e.g.
 - Opatch shipped with 10.1.0.5 patchset has version 10.0.0.53
 - Opatch shipped via bug 6880880 has version 1.0.0.0.62
- To install the latest version, unzip the opatch from bug:6880880 directly in \$ORACLE_HOME



Versions of Opatch

- For Oracle9i Release 2 (9.2.0.x) and Oracle10g Release 1 (10.1.0.x)
- The current OPatch version for these RDBMS releases is 1.0.0.0.63

- For Oracle10g Release 2 (10.2.0.x), select release "10.2.0.0.0"
- The current OPatch version for this RDBMS release is 10.2.0.4.9

- For Oracle11g Release 1 (11.1.0.x), select release "11.1.0.0.0"
- The current OPatch version for this RDBMS release is 11.1.0.6.7

- For Oracle11g Release 2 (11.2.0.x), select release "11.2.0.0.0"
- The current OPatch version for this RDBMS release is 11.2.0.1.1



Opatch documentation

- Documentation with patch updates stands in `$ORACLE_HOME/OPatch/docs` (FAQ, Users_Guide.txt, README.txt, bt(x).txt)
- Universal Installer and Opatch User's Guide (from 10.2 onwards)
- Metalink notes:
 - Note 293369.1 OPatch documentation list (for 10.1)
 - Note 334108.1 Opatch FAQ (10.2)



Opatch Commands

- Common to all version
 - ‘opatch apply’ or ‘opatch napply’
 - ‘opatch rollback’ or ‘opatch nrollback’
 - ‘opatch lsinventory’
 - ‘opatch query’
 - ‘opatch version’
- Additional command in 10.2 onwards: ‘opatch prereq’
- Additional command in 11.1 onwards: ‘opatch util’
- Additional command in 11.2 onwards: ‘opatch auto’



Common Opatch parameters

- Environment variables
 - OPATCH_DEBUG=true or not
 - OPATCH_REMOTE_SHELL=/usr/bin/scp
 - OPATCH_SKIP_VERIFY=true or not (10.2 onwards = OPatch.SKIP_VERIFY)
- Location of the runinstaller inventory via '-invPtrLoc <Path to oraInst.loc>'
- Location of the jdk/jre environment via '-jdk <LOC>' and '-jre <LOC>'



Opatch logging

- 10.1 version
 - All logging in `$ORACLE_HOME/.patch_storage/<patch ID>`
 - Logging `Apply_<patch>_<date&time>.log` and/or `Rollback_<patch>_<date&time>.log`
- 10.2 version and higher
 - Check the logfile in `$ORACLE_HOME/cfgtoollogs/opatch`
Contains `opatch_history.txt`
 - `$ORACLE_HOME/.patch_storage` with several directories:
backup, files, original_patch
 - `Opatch_<patch>_<date&time>.log`
- Installation lockfile stands in
`$ORACLE_HOME/.patch_storage/patch_locked`



Opatch lsinventory

- Set \$ORACLE_HOME and run:
 - 'opatch lsinventory' : show installed patches
 - 'opatch lsinventory -detail' : show detailed info
 - 'opatch lsinventory -all'
- When there are multiple ORACLE_HOME, you need to do it for all ORACLE_HOME to have a complete view of the installation since it only show:
 - The central inventory info (from /etc/oraInst.loc)
 - The oracle home inventory (from one of the \$ORACLE_HOME/inventory)



Opatch apply

- Lock the inventory via flagfile
\$ORACLE_HOME/.patch_storage/patch_locked
- Check inventory and patch installation (inventory.xml)
- Check for active processes (fuser check)
- Backing up pre comps.xml + files changed
- Apply patch + relink
- Update inventory
- Verify inventory and patch install
- Backup post comps.xml
- Release inventory lock



Opatch apply options

- All versions:
 - -local : install locally only. Don't install on all nodes
 - -silent : install without prompting
- Starting with 10.2
 - -no_relink: relink only once at the end
 - -no_sysmod : only register patch in inventory
 - -no_inventory: don't update the inventory
 - -report: just show the actions to be taken
 - -verbose: print info to screen
- Latest versions: auto flag for rac installations



Opatch napply

- Should be used when the zip containing the patch is made up of plenty of individual molecule bugs
- Main options are:
 - -skip_subset : skip patches that are subsets of installed patches
 - -skip_duplicate: skip fixes already installed



Some other interesting opatch commands

- `opatch$ rollback -id <IDs>`
- `opatch$ nrollback -id <IDs>`
- `Opatch query -all` : permit to view the full info about a patch
- `Opatch prereq checkconflictagainsthwithdetail -ph . -id <Ids>` : permit to precheck for conflicts between the installation and the patch to be applied
- `Opatch util verify -ph .` : verify whether a patch is really installed



Prerequisites before applying a patch

- Have a valid inventory => check with 'opatch lsinventory'
- use the latest opatch version of the software to be patched
- Check there is no conflict via 'opatch prereq checkconflictagainststohwithdetail' and check note:1061295.1 Patch Set Updates - Conflicts that Can Be Ignored
- Ssh or net use needs to work when software is installed on different nodes. check with cluvfy
cluvfy comp admprv -n all -o user_equiv -verbose
- Oracle home environment variable (ORACLE_HOME) must point to a valid Oracle home. The library path must be set correctly. Use the -oh option, too
- Backup your \$ORACLE_HOME + central inventory
- Look at the readme of the patch for any additional info



The inventory explained

- The inventory is updated via runinstaller or opatch
- Inventory type
 - Central inventory: pointed by /etc/oraInst.loc
 - Local inventory: in \$ORACLE_HOME/inventory
- Central inventory can be recreated. Local inventory can't
- Use runInstaller 10.2 or higher to correct central inventory problems



runInstaller execution options

- -silent : don't prompt for anything
- -invPtrLoc <central inventory location file, e.g. /etc/orainst.loc>
- -local : do everything locally only
- -ignoreSysPrereqs: don't check system requirements
- -attachHome : add an ORACLE_HOME reference from the inventory
- -detachHome : remove an ORACLE_HOME from the central inventory without deleting local inventory (set ORACLE_HOME)
- -removeAllPatches: remove all patches from the specified home



runInstaller execution options

- CRS=true : it concerns a CRS home
- -updateNodeList: update the nodelist
 - CLUSTER_NODES=rac1, rac2
 - LOCAL_NODE='rac1'
- ORACLE_HOME="`<OracleHome-Directory>`"
- ORACLE_HOME_NAME="`<OracleHome-Name>`"
- Trace parameters: -debug (11gR2) or
 - J-DTRACING.LEVEL=2
 - J-DTRACING.ENABLED=true



Inventory corrections examples

=> Add an ORACLE_HOME

```
./runInstaller -silent -attachHome -invPtrLoc <Inventory-  
Pointer-Location-File>  
\ORACLE_HOME="<OracleHome-Directory>"  
ORACLE_HOME_NAME="<OracleHome-Name>"
```

=> Remove an ORACLE_HOME

```
./runInstaller -silent -detachHome  
ORACLE_HOME="<OracleHome-Directory>"  
ORACLE_HOME_NAME="<OracleHome-Name>"
```



Inventory corrections examples

=> Update the nodelist

```
$ORACLE_HOME/oui/bin/runInstaller -updateNodeList  
ORACLE_HOME=/u01/app/oracle/10g  
CLUSTER_NODES={<node1>,<node2>}  
LOCAL_NODE={<node1>}
```

=> Recreate an inventory of a CRS_HOME

```
./runInstaller -silent -ignoreSysPrereqs -attachHome  
ORACLE_HOME="/u01/app/oracle/product/10.2.0/crs_1"  
ORACLE_HOME_NAME="OraCrs10g_home"  
LOCAL_NODE='rac 1' CLUSTER_NODES=rac1,rac2 CRS=true
```



Windows patching

- Mini Patch Bundles (see note:161549.1 Oracle Database Server and Networking Patches for Microsoft Platforms)
- Use of %ORACLE_HOME/Opatch/opatch.bat. The readme will still have some additional steps (like drivers copy, e.g. ocfs.sys)
- Runinstaller is %ORACLE_HOME%\oui\bin\setup.exe
- Use “SET” to do an ‘export/setenv’
- Central inventory in c:\program files\oracle\oraInventory



Windows patching

- File locking problems (see note:note:294350.1)
- Windows Services need to be shutdown before applying a patch (even the hidden ones like Orafenceservice) via service panel or commands like 'net stop OracleService<SID>'
- The most secure patch way:
 - 1. Disable all Oracle services and reboot the box before installing the patch
 - 2. Patch locally only



Patching in RAC/CRS environments

- CRS Bundle patches
- CRS cumulative patches
- Opatch embedded in pre/post scripts
- Rolling upgrades



CRS Patches and Bundles

- Delivered on a subset of platforms and test consists of 18 critical testing scenarios done on each platform.
- CRS patch comes in two parts:
 - one part must be applied to the CRS_HOME
 - The other part must be applied to the RDBMS home if it is the same version
- an ASM_HOME is considered as being a RDBMS_HOME
- Install first the CRS part, then the ASM part, then the RDBMS part



CRS/RAC specifics

- The RDBMS part has only a meaning when the CRS part is installed, i.e. The corrections are CRS fixes that can need a RDBMS part to be applied.
- The RDBMS part has no meaning when RAC is not installed
- RDBMS/ASM can be a lower version
- RDBMS part of CRS patches and bundles can not (and must not) be applied to the RDBMS home if the version is lower than the CRS
- All are rolling patches like bundles
- Note 363254.1 Applying one-off Oracle Clusterware patches in a mixed version home environment



CRS Cumulative Patches

- Quality Patches provided on top of Patch Sets and CRS Bundles.
- Something between the merge and the bundle (done that way because there is only one merge for the crs code)
- Contents:
 - Driven by Customer requests for critical fix one-offs
 - Cumulative, in that each patch contains the fixes of the previous patch.
 - Each incremental patch ideally has no more than three new fixes included.
 - Both platform-specific and generic fixes included.



Opatch embedded via pre/post scripts

- prerootpatch.sh - Detects CRS stack status and prompts the user and unlocks CRS home. To be run on \$CRS_HOME only
- prepatch.sh - Script run on \$CRS_HOME and \$RDBMS_HOME. This script harvests configuration values specified by the user at install time and saves the values in \$HOME/install/params.crs in a specific format to be later consumed by postpatch.sh.
- run the 'opatch apply'
- postpatch.sh for parsing CRS/RAC script files. This script change the installer specific variables in CRS scripts that are to be patched with the values stored in param.crs. Run on \$CRS_HOME and \$RDBMS_HOME



Opatch embedded via pre/post scripts

- postrootpatch.sh - Post patch script for starting CRS stack after a bug has been patched:
 - 1) Copies the CRS init script to init.d for init process
 - 2) Locks the CRS home, i.e. set it as root owned
 - 3) Starts the CRS stack

This is run during CRS installation and not during RAC/RDBMS part
- It is prone to errors and depend on a correct setup (advise: check params.crs before running opatch)



The “opatch auto” for CRS Bundles

- Permits to install the CRS Bundle with one command
 - install the patch on all potential homes
 - install everything on one node only (so it has to be run once per node in a rolling way)
 - Don't use when oracle_home is shared
- Commands are:
 - `./opatch auto <unzipped crs bundle patch Location>`
 - `./opatch auto -rollback <unzipped crs bundle patch Location>`
- Starting with Opatch version 10.2.0.4.7 and 11.1.0.6.7 (or higher)



RAC Rolling Patch Installations

- RAC Rolling patches are patches that can be installed without downtime, i.e. By patching one node/instance then the next node instance
- Not possible when shared homes are used
- A patch is a rolling patch when he can be installed in a rolling way
- note743126.1 How to Apply Oracle Windows Mini Patch Bundles in a Rolling Fashion
- The CRS Bundles, CPU and PSU patches are rolling patches.
- RDBMS Bundles are only rolling upgradable in case a previous CPU or BUNDLE was installed once, otherwise view recompilation requires a downtime
- The ASM RDBMS Bundle installations are always rolling upgradable



RAC Rolling Patch Installations

- One off and cumulative patches may or may not be rolling upgrades
- How to see if patch is rolling?
 - `opatch query -is_rolling (10.1)`
 - `opatch query -all`
 - check the `./etc/config/inventory` file



Patching dataguard environments

- Patching a physical dataguard environment
 - The normal way
 - Via a logical environment starting in 10.2
- Patching a logical dataguard environment
 - The normal way
 - Rolling upgrades starting in 10.1.0.3
- **Oracle® Data Guard Concepts and Administration
10g Release 2 (10.2) Appendix B Upgrading Databases in a
Data Guard Configuration**
- **11 Using SQL Apply to Upgrade the Oracle Database**



Physical dataguard environments

- Normal procedure is (downward compatibility model)
 - Stop the primary database
 - Stop the standby database
 - Apply patch on both \$ORACLE_HOME
 - Start the standby database in managed recovery
 - Start the primary database
 - Run the post installation steps on the primary e.g. catpatch.sql



Logical Dataguard environments

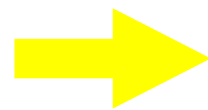
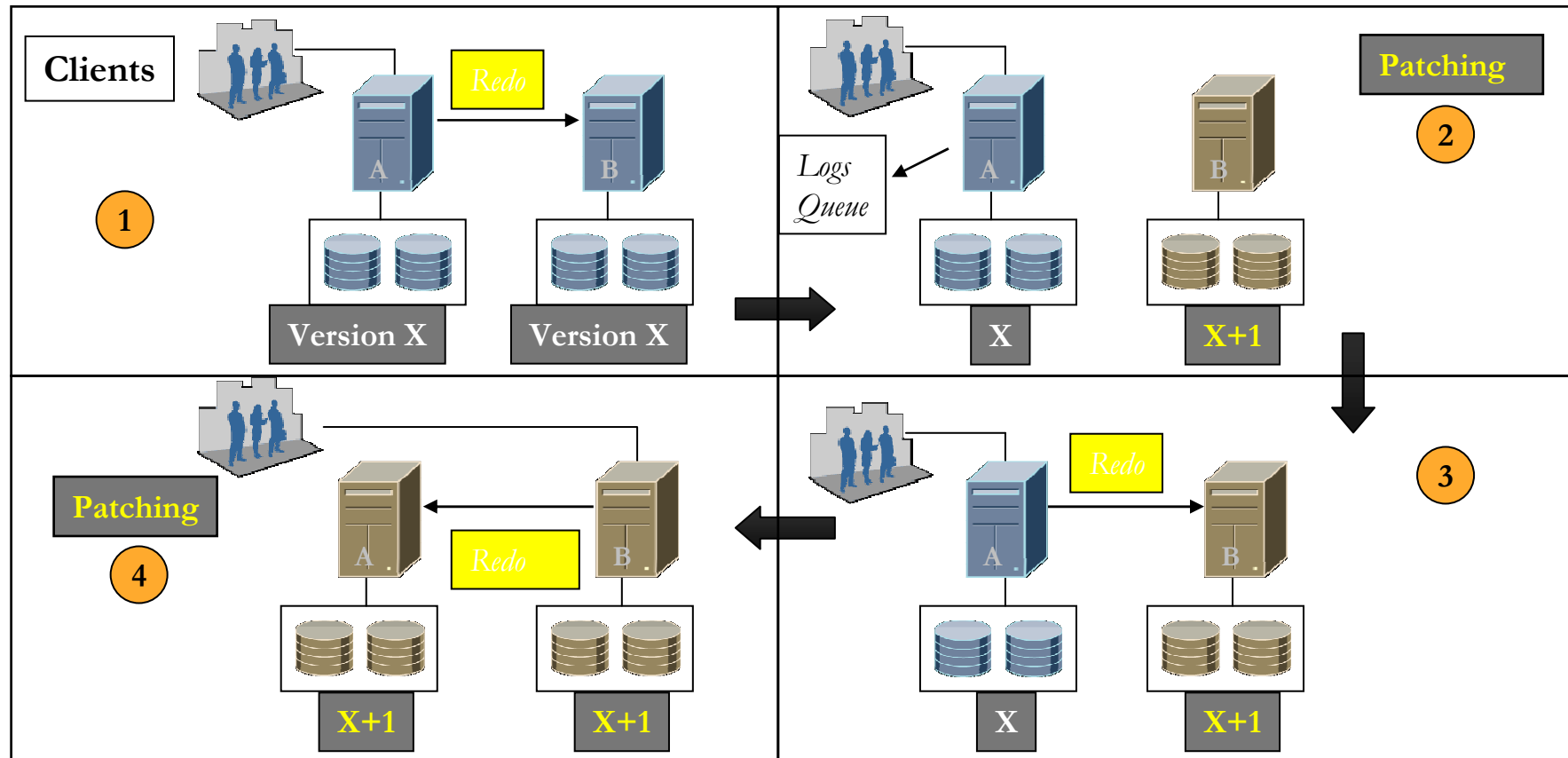
- Normal procedure is (downward compatibility model)
 - Stop the primary database
 - Stop the logical standby database once all redo have been applied
 - Sys/System only:
 - Apply patch on primary + post installation steps
 - Build new logminer dictionary on primary database
 - Apply patch, then issue 'ALTER DATABASE ACTIVATE LOGICAL STANDBY DATABASE' and do post installation steps
 - Copy logfile with logminer dictionary and start logical apply again



Logical Dataguard environments

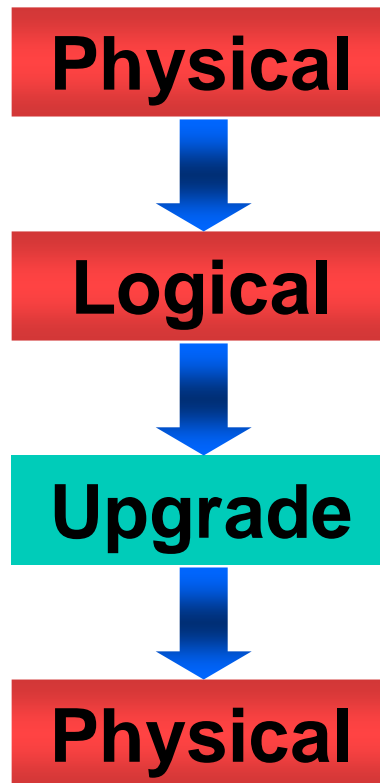
- Rolling upgrade procedure is (starting in 10.1.0.3)
 - Stop SQL Apply and upgrade the logical standby database
 - Restart SQL Apply
 - Monitor events on the upgraded standby database
 - Begin a switchover
 - Determine if unsupported objects were modified during the upgrade
 - Complete the switchover and activate user applications
 - Upgrade the former primary database
 - Start SQL Apply
 - Monitor events on the new logical standby database
 - Optionally, perform another switchover

Data Guard Rolling Upgrade



Rolling Upgrade

Rolling Database Upgrades Using Transient Logical Standby



- Logical standby allows rolling upgrades but has data type and performance limitations
- Temporarily convert physical standby to logical to perform a rolling database upgrade
- Performance and data type restriction of logical standby are limited to upgrade window
- No need for separate logical standby for upgrade
- Also possible in 10g with more manual steps



High Availability and patching

- HA environments should normally not be patched since tested before put in production
- There is a need to patch only when there is an undetected bug with no valid workaround, however the best is to prevently install the regular PSU's
- It is better to install PSU and bundles than mlr or oneoffs (tested versus non tested, more fixes installed at once, no 'customized' installations)
- Use environments that limits production downtime (RAC, dataguard) via the rolling patches
- Prepare a patch installation before the patching date



Q&A