This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are “commercial computer software” or “commercial technical data” pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

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

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.
Contents

Preface ......................................................................................................................................................... v
  Audience ........................................................................................................................................................ v
  Documentation Accessibility ......................................................................................................................... v
  Related Documents ..................................................................................................................................... v
  Conventions .................................................................................................................................................. v

1 Introduction to OPatch and Patching
  1.1 Overview of the Patch Process ........................................................................................................... 1-2
  1.1.1 Obtaining the Patches You Need .................................................................................................... 1-2
  1.1.2 Applying the Patch to the Desired Targets .................................................................................... 1-5
    1.1.2.1 Manual Patching ...................................................................................................................... 1-6
    1.1.2.2 Configuration Patching ........................................................................................................... 1-6
  1.2 Patching with Enterprise Manager ..................................................................................................... 1-6
  1.3 Who Should Use OPatch? ................................................................................................................... 1-7
  1.4 What’s Covered in this Guide ............................................................................................................. 1-7
  1.5 OPatch Integration with Other Oracle Software ................................................................................ 1-7
  1.6 How to Access the OPatch Utilities .................................................................................................. 1-8

2 Binary Patching Using OPatch
  2.1 Obtain the Latest OPatch Utility ........................................................................................................ 2-1
  2.2 Using OPatch ....................................................................................................................................... 2-1
    2.2.1 Patching Workflow ......................................................................................................................... 2-2
      2.2.1.1 Setting the ORACLE_HOME Environment Variable ......................................................... 2-2
      2.2.1.2 Determining What is Installed On Your System ................................................................... 2-2
      2.2.1.3 Ensuring Patch Application Prerequisites are Met ............................................................... 2-2
      2.2.1.4 Applying a Patch ..................................................................................................................... 2-3
      2.2.1.5 Running Post-apply Checks ................................................................................................... 2-3
  2.3 Applying a Patch Set Update (PSU) .................................................................................................... 2-3
    2.3.1 Applying a Single Patch ............................................................................................................... 2-3
    2.3.2 Apply Multiple Patches Using a Text File .................................................................................... 2-3
  2.4 Patch Conflict Detection and Resolution .......................................................................................... 2-4

3 Multi-Node Patch Orchestration Using OPatchauto
  3.1 Remote Host Execution using SSH .................................................................................................... 3-1
  3.2 Oracle Wallet for Credential Input ..................................................................................................... 3-2
3.3 Patch Orchestration Concepts ................................................................. 3-2
3.3.1 Phases and Sessions ........................................................................... 3-2
3.3.2 Patch Plans ....................................................................................... 3-3
3.3.3 Configuration Descriptions ............................................................... 3-4
3.3.4 Troubleshoot the Application of a Patch ........................................... 3-4

4 OPatchauto -binary
4.1 Supported Patch Types ................................................................. 4-1
4.2 Operations Performed by OPatchauto -binary ................................. 4-1
4.3 Analyze Mode of Operation .............................................................. 4-1
4.4 Target Types in a SystemPatch ......................................................... 4-2
4.5 Syntax and Commands ..................................................................... 4-2

5 Troubleshooting OPatch
5.1 Debugging: Enable Logging and Tracing ........................................ 5-1
5.2 References ....................................................................................... 5-2
5.3 Products and Patch Types Not Supported by OPatch ...................... 5-2

A OPatch Syntax and Commands
A.1 OPatch Syntax ................................................................................. A-1
A.2 apply .............................................................................................. A-2
A.3 compare ....................................................................................... A-4
A.4 lsinventory ................................................................................ A-5
A.5 lspatches .................................................................................... A-6
A.6 napply ........................................................................................ A-7
A.7 nrollback ................................................................................ A-10
A.8 rollback .................................................................................... A-12
A.9 query ........................................................................................ A-14
A.10 version .................................................................................... A-15
A.11 prereq .................................................................................... A-16
A.12 util ........................................................................................ A-19

B OPatchauto Syntax and Commands
B.1 OPatchauto Commands ............................................................... B-1
B.1.1 apply ........................................................................................ B-1
B.1.2 resume ................................................................................... B-3
B.1.3 rollback ................................................................................ B-4
B.1.4 help ..................................................................................... B-5
B.1.5 version ................................................................................ B-6
B.2 Wallet Creation Commands ......................................................... B-6

Glossary

Index
The *OPatch User’s Guide* provides information about Oracle’s patching solutions to help ensure your Oracle products stay current and secure.

- **Audience**
- **Documentation Accessibility**
- **Related Documents**
- **Conventions**

**Audience**

This document is intended for administrators who want to set up and manage Enterprise Manager security.

**Documentation Accessibility**

For information about Oracle’s commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc

**Access to Oracle Support**

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

**Related Documents**

For the latest releases of these and other Oracle documentation, check the Oracle Technology Network at:

http://www.oracle.com/technetwork/documentation/index.html#em

Oracle Enterprise Manager also provides extensive Online Help. Click **Help** at the top of any Enterprise Manager page to display the online help window.

**Conventions**

The following text conventions are used in this document:
<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>boldface</strong></td>
<td>Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.</td>
</tr>
<tr>
<td><em>italic</em></td>
<td>Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.</td>
</tr>
<tr>
<td><strong>monospace</strong></td>
<td>Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.</td>
</tr>
</tbody>
</table>
OPatch consists of patching utilities that help ensure your Oracle software stays current and secure. The utilities are:

- **OPatch**: A Java-based utility that enables the application and rollback of patches to Oracle software.

- **OPatchauto**: A patch orchestration tool that generates patching instructions specific to your target configuration and then uses OPatch to perform the patching operations without user intervention. Specifically, OPatchauto can:
  1. Perform pre-patch checks.
  2. Apply the patch
  3. Start and stop the running servers.
  4. Perform post-patch checks.
  5. Roll back patches when patch deinstallation is required.

- **OPatchauto -binary**: A patch application tool that applies a single patch on a selected Oracle home. OPatchauto -binary patches only one Oracle home per session.

These utilities provide you with the flexibility to analyze, troubleshoot, and patch an individual GI (Grid Infrastructure)/RAC (Real Application Cluster) home environments.

For large-scale IT environments, patching individual product (e.g., Fusion Middleware) homes may not be practical since patching large numbers of targets manually is both monotonous and error prone. To maintain and deploy Oracle patches across many targets across your organization, you can use Enterprise Manager Cloud Control’s patch automation capability. For more information about Enterprise Manager’s patch management solution, see the "Patching Software Deployments" in the Oracle® Enterprise Manager Lifecycle Management Administrator’s Guide.

This chapter covers the following introductory and overview topics:

- **Overview of the Patch Process**
- **OPatch Integration with Other Oracle Software**
- **Who Should Use OPatch?**
- **How to Access the OPatch Utilities**

The patch process is not always straightforward as there are numerous factors that determine which software patches you need and how these patches should be applied. For example, the types of Oracle software installed on each target, software versions, or platforms on which the software is running are just a few.
1.1 Overview of the Patch Process

Regardless of your environment’s patching requirements, the basic patching methodology is the same. The normal patching workflow can be broken down into the following nine steps shown in the following figure.

Figure 1–1  Patch Process Overview - Process Flow

1.1.1 Obtaining the Patches You Need

As shown in Figure 1–1, "Patch Process Overview - Process Flow", the first step is to determine what patches you need. You may find out about required patches from blogs, Oracle Technology Network (OTN), Service Requests, Knowledge Articles, Oracle documentation, or any number of other sources. However, the single source of truth for patching is the Oracle Support Web site—My Oracle Support (MOS).
https://support.oracle.com

From here, you have access to interactive support tools and information that simplify searching for and obtaining the requisite patches for your Oracle environment. You can find complete documentation about MOS at the following location:

http://docs.oracle.com/cd/E25290_01/index.htm

Figure 1–2  My Oracle Support Main Page

My Oracle Support contains many features and capabilities that are grouped under tabs across the top of the application. Of primary interest is the Patches and Updates tab shown in the following figure.

Figure 1–3  MOS Patches and Updates
From this page, you can search for the desired patch based on a specific configuration. One particularly useful search feature is the Recommended Patch Advisor. The Recommended Patch Advisor lets you find recommended and mandatory patches for standalone products, product combinations, or products for a product stack. For example, using the Recommended Patch Advisor, you search for patches for the following product:

- **Product:** Oracle Database
- **Release:** 11.2.0.2.0
- **Platform:** Linux x86-64

This search returns the following results:

```
Product: Oracle Database
Release: 11.2.0.2.0
Platform: Linux x86-64
```

By clicking on patch 14727315 (PSU) you are taken to the patch page where you can view bugs resolved by this patch, related Knowledge Articles, or view a generic patch README.
From this page, you also complete **Step 2** of the patching workflow—Download the patch to your local system. The following list summarizes sources from which you can obtain patches.

- **Oracle Support Services**: If you are working directly with an Oracle Support engineer, you may be provided with a diagnostic patch or an interim patch.
- **My Oracle Support**: As part of your regular patch maintenance schedule, you can obtain all patches from My Oracle Support:
  
  https://support.oracle.com

  Once you log in, click the **Patches & Updates** tab to begin your patch search. My Oracle Support offers several patch download options and automated tools to help you keep current with patches. See the Patches & Updates Web-based help for more information:
  
  http://docs.oracle.com/cd/E25290_01/doc.60/e25224/patchesupdates.htm#CJAGJJGI

- **Oracle Technology Network**: Some Oracle software may be distributed through the Oracle Technology Network:
  

**Types of Oracle Patches**

Oracle regularly makes patches available to upgrade features, enhance security, or fix problems with supported software. The major types of patches are:

- **Interim patches** - contain a single bug fix or a collection of bug fixes provided as required
- **Interim patches for security bug fixes** - contain customer-specific security bug fixes
- **Diagnostic patches** - intended to help diagnose or verify a fix or a collection of bug fixes
- **Bundle Patch Updates (BPUs)** - a cumulative collection of fixes for a specific product or component
- **Patch Set Updates (PSUs)** - a cumulative collection of high impact, low risk, and proven fixes for a specific product or component and Security Patch Updates
- **Security Patch Updates (SPU)** - a cumulative collection of security bug fixes. SPU were formerly known as Critical Patch Updates (CPU).
- **System Patch** - contains several sub-patches in a format that can be used by OPatchauto.
- **Merge Label Request (MLR)** - a *merge* of two or more fixes. MLR creation requires a *label* for the new set of merged code and a Patch Set Exception

### 1.1.2 Applying the Patch to the Desired Targets

Now that you have the requisite patch, you determine which targets in your environment need to be patched (Step 3 in the patching workflow) and then apply the patch to each target (Step 4). Step 4 is where the OPatch utilities come into play. See **Figure 1–1, "Patch Process Overview - Process Flow"** to view the complete patching workflow.
Note: Ensure that you have the latest version of OPatch. For more information, see "Obtain the Latest OPatch Utility" on page 2-1.

1.1.2.1 Manual Patching

Using OPatch, you follow the generic instructions in the patch README. You can view the patch README bundled with the patch or directly from the MOS page for the patch in question.

You are required to read the linked support documentation and fill in the details of your specific configuration before you can implement any of the commands or add them to custom install scripts. Although this method is laborious, it provides you with a great deal of diagnostic capability and control if patch conflicts arise. See Chapter 2, "Binary Patching Using OPatch."

1.1.2.2 Configuration Patching

OPatchauto performs end-to-end configuration patching. Configuration patching is the process of patching a target based on its configuration. By incorporating the site configuration information into the patch process, OPatchauto is able to simplify patching tasks by automating most of the steps.


1.2 Patching with Enterprise Manager

As useful as the OPatch utilities are, by themselves, they are limited in their ability to apply patches to large numbers of targets because they patch one GI/RAC home at a time. This could be challenging and time consuming in large, heterogeneous IT environments.

In order to handle large-scale patching, Oracle provides a new patch management solution that integrates OPatch with Enterprise Manager Cloud Control 12c: Enterprise Manager’s tight integration with My Oracle Support (MOS) allows you to view patch recommendations, search patches, and roll out patches from a single user interface. In addition, Enterprise Manager’s advanced Patch Plan feature provides you with a complete, end-to-end orchestration of the patching workflow. Automating the selection of deployment procedures and analysis of patch conflicts greatly reduces manual effort required to patch complex IT environments.

Enterprise Manager integrates both OPatch and My Oracle Support for downloading and applying patches. See the following documentation for information:

- See "Part VII: Patch Management" in the Oracle® Enterprise Manager Lifecycle Management Administrator’s Guide:
  http://docs.oracle.com/cd/E24628_01/em.121/e27046/part_patching.htm#BGBJFJBG

- See the "Patching Enterprise Manager" chapter in the Oracle® Enterprise Manager Cloud Control Administrator’s Guide:
  http://docs.oracle.com/cd/E24628_01/doc.121/e24473/patching.htm#CHDEBABC

- Automation of patch conflict resolution and deployment through EM 12c.
  (Webcast)
  https://support.oracle.com/epmos/faces/DocumentDisplay?_afrLoop=441420602917&id=1533881.1&_afrWindowMode=0&_=adf.ctrl-state=9b136mq35_4
1.3 Who Should Use OPatch?

While OPatch is integrated with many Oracle product installations, you may still find that you need to use the OPatch directly. The core OPatch tool is used directly by admins as part of manual patching. OPatchauto invokes OPatch, so an understanding of core OPatch is useful. OPatch is also useful for conflict detection and resolution as well as troubleshooting.

**Note:** Before performing any patch task, *always* read the patch README file for any special patching instructions.

You can use the OPatch utilities if your administrative tasks require you to:

- Report on installed products and patches.
- Apply one or more patches.
- Roll back the application of one or more patches.
- Detect conflicts among incoming patches and between it and previous patches that have been applied. OPatch suggests the best options to resolve a conflict.

1.4 What’s Covered in this Guide

This document describes how to use these patching utilities and covers the following topics:

**Note:** Recommendation of what tool to use when is stated in the README of the given patch. Always start with the README.

- Chapter 2, "Binary Patching Using OPatch" - Describes the basic functions of the core opatch tool, that applies patches to an Oracle Home.
- Chapter 3, "Multi-Node Patch Orchestration Using OPatchauto" - Recommended for administrators who wish to apply GI-RAC or Exadata patches to GI node in one shot via an orchestration tool.
- Chapter 5, "Troubleshooting OPatch.”
- Appendix A, "OPatch Syntax and Commands"
- Appendix B, "OPatchauto Syntax and Commands"

With these patching tools, you can design and implement a patch plan based on the configuration of your Oracle products.

**Note:** Before patching any Oracle product, always check the product documentation for patching instructions.

1.5 OPatch Integration with Other Oracle Software

In addition to Enterprise Manager, many Oracle software products have integrated the OPatch utilities to provide for a seamless and efficient patching task. Depending on the application, the call to the OPatch utility may be transparent, and all patching activity is maintained within the respective application.
These applications listed below have integrated OPatch into their respective environments. Always check the user documentation for any patching instructions before applying a patch.

**Fusion Middleware/Fusion Applications**
Other Oracle products, such as Fusion Middleware and Fusion Applications, integrate OPatch and may require different interaction to apply a particular patch. Refer to the following documentation:

- For patching Fusion Middleware:
  - Section 2.3 “OPatch in a Fusion Middleware Environment” in the *Oracle® Fusion Middleware Patching Guide*:
    - [http://docs.oracle.com/cd/E23943_01/doc.1111/e16793/opatch.htm#PATCH159](http://docs.oracle.com/cd/E23943_01/doc.1111/e16793/opatch.htm#PATCH159)
  - Section 3 Applying the Latest Oracle Fusion Middleware Patch Set in the *Oracle® Fusion Middleware Patching Guide*:
    - [http://docs.oracle.com/cd/E23943_01/doc.1111/e16793/patch_set_installer.htm](http://docs.oracle.com/cd/E23943_01/doc.1111/e16793/patch_set_installer.htm)

- For patching Fusion Applications:
  - Section 3 Using Oracle Fusion Applications Patch Manager in the *Oracle® Fusion Applications Patching Guide*:
    - [http://docs.oracle.com/cd/E29505_01/fusionapps.1111/e16602/applypatches.htm#CIHBDCBE](http://docs.oracle.com/cd/E29505_01/fusionapps.1111/e16602/applypatches.htm#CIHBDCBE)

### 1.6 How to Access the OPatch Utilities
With OPatch integrated in many Oracle products, the utility is automatically installed when you install the respective product (for example, Enterprise Manager). The patching tools are installed in the following directories:

- **OPatch** - `$ORACLE_HOME/OPatch/opatch`
- **OPatchauto** - `$ORACLE_HOME/OPatch/OPatchauto`

See Appendix A, "OPatch Syntax and Commands," and Appendix B, "OPatchauto Syntax and Commands" for a complete list of commands and options supported by OPatch utilities.
Binary Patching Using OPatch

OPatch is a utility that allows you to apply and/or roll back interim patches to Oracle’s software. The manual process of applying a patch is called *binary patching*. For bits patching, you can use the OPatch utility to:

- Obtain the Latest OPatch Utility
- Using OPatch
- Applying a Patch Set Update (PSU)
- Patch Conflict Detection and Resolution

**Note:** Always refer to the patch README for any special instructions before you apply a patch.

Oracle recommends that you back up the $ORACLE_HOME$ before any patch operation. You can back up the $ORACLE_HOME$ using your preferred method. You can use any method, such as `zip`, `cp -r`, `tar`, and `cpio` to compress the $ORACLE_HOME$.

### 2.1 Obtain the Latest OPatch Utility

You should use the version of OPatch that supports the $ORACLE_HOME$ release. For example, if you are patching a 12.0.1 Oracle Home, then use OPatch version 12.0.1. If the Oracle Home is version 11.2, then use OPatch version 11.2.

Oracle recommends that you use the latest released OPatch for 12.1 releases, which is available for download from My Oracle Support (patch 6880880). Select the ARU link for the 12.1.0.1.0 release.

[https://updates.oracle.com/download/6880880.html](https://updates.oracle.com/download/6880880.html)

You can also view the support document "OPatch - Where Can I Find the Latest Version of OPatch?" (Doc ID 224346.1). This document contains a link to the instructional video "Downloading the OPatch Tool from MOS."

### 2.2 Using OPatch

Before you use the OPatch command and available options, you must check that OPatch prerequisites have been fulfilled.
2.2.1 Patching Workflow

Using the OPatch utility to patch your product (e.g., Fusion Middleware) home typically consists of the following steps:

1. Setting the ORACLE_HOME Environment Variable
2. Determining What is Installed On Your System
3. Ensuring Patch Application Prerequisites are Met
4. Applying a Patch
5. Running Post-apply Checks

2.2.1.1 Setting the ORACLE_HOME Environment Variable

OPatch verifies whether the product home is present. You must ensure that the ORACLE_HOME environment variable is set to the product home of the product you are trying to patch. Check your respective vendor documentation for the details on how to set the environment variable.

**Note:** Oracle Universal Installer binaries and inventories must be present in the home to be patched.

Other environment variables used include:

- **OPATCH_DEBUG** — Boolean setting that specifies the amount of logging OPatch should perform.
- **PATH** — Product home path information.

**Note:** Adding $ORACLE_HOME/OPatch to the $PATH makes it more convenient to execute the OPatch commands from any directory.

2.2.1.2 Determining What is Installed On Your System

The next step is to determine what is already installed on your system. For this, you use the `opatch lsinventory` command with either the `patch` or `patch_id` options.

**Example 2–1 lsinventory Command**

```bash
opatch lsinventory
```

For more information about this command, see "lsinventory" on page A-5.

2.2.1.3 Ensuring Patch Application Prerequisites are Met

After you have determined that your system configuration is appropriate for the patches you wish to apply, it is advisable to view the operations OPatch will perform before performing the patch application to help determine whether all system prerequisites are met before applying the patch. For this, you use the OPatch `-report` option to print all patch application actions OPatch will perform without actually executing the actions.

**Example 2–2 report Option**

```bash
opatch apply -report
```
For more information about the -report option, see "OPatch Syntax and Commands" on page A-1.

2.2.1.4 Applying a Patch
Once you have determined that the patch can be applied to your system successfully, you can now use OPatch to apply the patch. For this, you use the OPatch apply command.

Example 2–3 apply Command

```
opatch apply /tmp/patch/12345678
```

For more information about this command, see "apply" on page A-2.

2.2.1.5 Running Post-apply Checks
After you have applied the patch to your system, you should perform a final check to ensure all patches have been successfully applied. For this, you again use the OPatch lsinventory command with either the patch or patch_id options. For more information about this OPatch command, see "lsinventory" on page A-5.

2.3 Applying a Patch Set Update (PSU)
Once you have verified the prerequisite checks, use OPatch to apply a patch:

- Applying a Single Patch
- Apply Multiple Patches Using a Text File

The OPatch utility is located in the $Oracle_Home/OPatch directory. You can run it with various commands and options. See Appendix A, "OPatch Syntax and Commands," for a complete list of command options available with OPatch.

2.3.1 Applying a Single Patch
You apply a single patch following the generic patching workflow discussed earlier:

1. Obtain the Latest OPatch Utility
2. Setting the ORACLE_HOME Environment Variable
3. Determining What is Installed On Your System
4. Ensuring Patch Application Prerequisites are Met
5. Applying a Patch
6. Running Post-apply Checks

Once you have downloaded the patch, you can apply it (step 5) using the following command:

```
# opatch apply <patch directory location>/<patch ID>
```

For example:

```
# opatch apply /tmp/patch/12345678
```

2.3.2 Apply Multiple Patches Using a Text File
You can create a text file containing the location of all patches you need to apply. Use OPatch to reference the file and apply the patches:
1. Create the text file of the patch location. The entry should have each patch location on a separate line:

   vi patches.txt
   /tmp/patchlocation1/12345678
   /tmp/patchlocation2/12365478
   /scratch/patchlocation3/32165487

   Save your changes.

2. Apply the patches with OPatch:

   # opatch napply - -phBaseFile <location of text file>

   For example:

   #opatch napply - -phBaseFile /tmp/patches/patches.txt

### 2.4 Patch Conflict Detection and Resolution

OPatch detects and reports any conflicts encountered when applying a patch with a previously applied patch. The patch application fails in case of conflicts. You can use the -force option of OPatch to override this failure. If you specify -force, the installer firsts rolls back any conflicting patches and then proceeds with the installation of the desired patch. Using -force means that you will likely lose some bug fixes, that are causing the conflict, in exchange for getting the incoming patch to be applied.

You may experience a bug conflict and might want to remove the conflicting patch. This process is known as patch rollback. During patch installation, OPatch saves copies of all the files that were replaced by the new patch before the new versions of these files are loaded, and stores them in $ORACLE_HOME/.patch_storage. These saved files are called rollback files and are key to making patch rollback possible. When you roll back a patch, these rollback files are restored to the system. If you have a complete understanding of the patch rollback process, you should only override the default behavior by using the -force flag. To roll back a patch, execute the following command:

   $ OPatch/opatch rollback -id <Patch_ID>

A patch conflict occurs when multiple fixes in different patches touch the same files but have not been tested together as a single entity. OPatch and OPatchauto help you avoid these conflicts by identifying these conditions. When patch conflicts occur and you are unable to resolve them using documented support procedures, MOS then becomes the go-to source for technical assistance. Conflict resolution may entail filing a Service Request and obtaining a Merge Label Request (MLR) patch to overcome a patching issue. Once a solution has been found, you use OPatch to apply the fixed patch.
OPatchauto is Oracle’s strategic tool for binary and configuration patching. For the supported environments, OPatchauto sequences and executes all required steps, on all nodes, for comprehensive patch application. Because OPatchauto can patch full systems in one invocation, it removes the burdens of:

- The physical effort of going host to host and executing commands
- The mental effort of remembering the sequence of commands across the nodes in your system

Your product’s patching documentation (Database, Fusion Middleware, Enterprise Manager Cloud Control) explains how to use OPatchauto to patch your specific product. This book augments those guides, by providing deeper conceptual and reference material for OPatchauto, in a product independent manner.

**Note:** Beginning with the Fusion Middleware 12.2.1 release, OPatchauto has evolved in capability from a single node to multi-node tool. Other products will have multi-node support; once included, this document will be updated with the new link.

This chapter covers the following topics:

- Remote Host Execution using SSH
- Oracle Wallet for Credential Input
- Patch Orchestration Concepts

### 3.1 Remote Host Execution using SSH

OPatchauto executes across all hosts in the associated system, using SSH as its remote execution mechanism for requisite commands, such as `opatch apply`. Of course, the user must supply the SSH credentials to permit this execution. This can be done in three ways:

- By supplying, via `-host`, the username and password for a single remote host
- By supplying a wallet with the SSH credentials for one or more hosts (so required for multi-host patching).

**Tip:** For understanding the basic principles of SSH equivalence, try the Web search “generating SSH keys” or “SSH-keygen” to understand how SSH manages the keys that enable SSH authentication.
Remote Host Patching on Windows

OPatchauto only supports SSH on Windows servers if OpenSSH is set up as part of Cygwin. See "Installing Cygwin and Starting the SSH Daemon" in the Oracle® Enterprise Manager Cloud Control Basic Installation Guide.

3.2 Oracle Wallet for Credential Input

OPatchauto accepts credentials, in oracle wallet format, for accessing run-time entities, such as databases and admin servers. You input a wallet on the command line; if you do not supply one, and OPatchauto needs one, it will prompt you for one on the command line. Successful usage depends on the user possessing both the wallet and the wallet password.

For convenience, you can simply create and configure a wallet for patching purposes using the commands `configWallet.sh` or `configWallet.cmd`.

These convenience tools in turn build on the underlying Oracle Wallet `mkstore` command, which allows the user to create, modify, list, and delete credentials. For more information on `mkstore`, see "Managing Password Store Credentials" in the Database Security Guide.

The command line syntax of "username&hostname:type" is expected for each credential. The SSH credentials should be defined for the each host in your cluster. The WLS credential only needs to be defined for the Admin Server host.

The tool will prompt you to enter/confirm the password for each credential. For the SSH credentials provide the SSH password of your current user. For the WLS credential provide the same value as PASSWORD from the common.sh.

Here is an example invocation for wallet creation for an Fusion Middleware domain for the hosts outlined below.

```bash
./config-wallet.sh -create "${USER}&${HOSTNAME}:SSH"
"${USER}&${HOST1}:SSH" "${USER}&${HOST2}:SSH"
"${USERNAME}&${HOSTNAME}:wls"
```

3.3 Patch Orchestration Concepts

Applying a patch involves an orchestrated series of steps. As OPatchauto’s name indicates, the customer does not need to understand these steps. They can just apply the patch.

However, the following underlying orchestration concepts are necessary for the customer to learn when the following are true:

- The patching operation has failed and they need to trouble-shoot.
- They see advantage to interleaving their own commands into the patching sequence in order to take advantage of their production system’s downtime window. In this case, understanding phases is required.

3.3.1 Phases and Sessions

The conceptual related steps of patching operation is called a phase. Executing all phases leads to a completed patching operation on the target; skipping a phase means the patch is not correctly applied. For example, the phase/sub-phase of applying the bits to an Oracle Home is called offline:binary-patching in OPatchauto.
Each invocation of OPatchauto apply generates a new session, whether you are executing all phases in one go (default) or just a sub-phase (advanced).

Phase input to the command is both optional and an advanced feature. However, if the customer wishes to interleave commands between the phases, they will need to invoke OPatchauto multiple times, specifying the specific phase in the correct sequence so that all phases are executed. Phases are composed of sub-phases. The user may also invoke the tool at the sub-phase granularity. The –help option lists the available phases and sub-phases.

Phases are idem-potent; you may execute them repeatedly. However, the tool will not inform you if you do not follow the correct sequence of invocations. (See ER 21553825.)

The high-level phases follow. They can be specified as (a) book-keeping, (b), life-cycle operations, of (c) configuration change operations

- **Init**: Book-keeping operation to initialize internal state needed for correction patching.
- **Shutdown**: Life-cycle operation that brings down run-time entities to permit patching.
- **Offline**: Configuration change operation to apply the patch content with the system down. Bits application happens here, for instance. So the opatch patch will be recorded to the homes OUI inventory in this phase.
- **Start-up**: Life-cycle operation that brings the shutdown entities back up again.
- **Online**: Configuration change operation apply patch content that requires that the systems be up. If these configuration changes have a system inventory, they will also be recorded to that system’s inventory at this point.
- **Finalize**: Book-keeping operation to record that patch operation is complete.

In the product’s documentation, you might see sub-phases that include “prepare” and “binary/product” variants. Prepare means “ready materials but do not make a configuration change.” Binary operations only change Oracle Homes. Product operations change the configured system, such as domain configurations or database dictionaries.

The specific content of the patch determines precisely what specific Oracle Home and configuration changes occur. Most Fusion Middleware patches, for example, only include offline content changes. But, of course, some include configuration changes as well.

In general, the session is an implicit parameter, set internally to the last session. It is visible to the user in the logs, communicated as a session ID, but there is no requirement for it to be supplied by the user. As a convenience for specifying the rollback parameters, you can specify the session ID. In this case, OPatchauto knows to query that session for the patch you wish to rollback.

### 3.3.2 Patch Plans

Patch plans describe, independent of a specific product instance’s topology, the sequence of steps to execute in order to deploy the patch. Patch plans are life-cycle programs, developed by Oracle life-cycle management experts, specifically for the given product being patched.

They are optional and advanced inputs. However, internally, OPatchauto always selects a patch plan to guide its execution. For example, internally, OPatchauto apply and OPatchauto rollback automatically select different patch plans implicitly.
Users will input patch plans to OPatchauto when executing more complex life-cycle operation, such as Zero Downtime Patching. The product documentation for these life-cycle operations will list the names of valid patch plans.

3.3.3 Configuration Descriptions

OPatchauto needs to know the configuration (topology) of the system it is to patch. It must either discover programmatically or be given by the user the topology of said system. Whether OPatchauto uses self-discovery or user discovery depends on the underlying product capabilities.

Database patching uses self-discovery; Fusion Middleware requires user discovery. So, for Fusion Middleware, patching multi-node via OPatchauto requires the user to supply manually a topology file. For convenience, patching single-node, single domain Fusion Middleware, whether local or remote, allows the user to supply a domain, via `-instance`, as an alternative configuration description.

Therefore, to use and benefit from multi-node patching, you must author the topology file using the Fusion Middleware Composer tool, installed under `oracle_common/bin`.

For more information about Fusion Middleware Composer, see the section on "Building A Topology Using Fusion Middleware Composer" in the Oracle® Fusion Middleware Patching with OPatch guide.

The provided topology provides to OPatchauto information about domains, clusters, `oracle_homes`, node managers, hosts, and credential aliases (to credentials in the accompanying wallet). OPatchauto uses this data to fully automate the patch application.

3.3.4 Troubleshoot the Application of a Patch

You can find the OPatchauto log files in the following location:

`$ORACLE_HOME/cfgtoollogs/OPatchauto`

For more information on troubleshooting OPatch, see Chapter 5, "Troubleshooting OPatch."

For both OPatch and OPatchauto, the log file location is also displayed in the console output.
OPatchauto -binary is a tool that applies a single patch on a selected Oracle home. It can patch only one Oracle home per session. The utility internally performs the prerequisite checks before applying the bits.

Operational Characteristics

- OPatchauto -binary assumes that the targets have been already shutdown.
- OPatchauto -binary does not perform any other operation apart from applying the binary bits to the Oracle home.
- OPatchauto -binary uses OPatch commands internally to perform the operations.

4.1 Supported Patch Types

OPatchauto -binary can apply only one patch per session. The selected patch can be any of the following types:

- OneOff patch
- Composite patch
- SystemPatch (containing multiple patches)

Note that while OPatchauto can apply all of the aforementioned patch types, OPatch cannot apply system patches. (OPatch can apply one off and composite patches.) So system patch application is the distinctive value for opatchauto -binary.

4.2 Operations Performed by OPatchauto -binary

OPatchauto -binary performs the following operations on the selected Oracle home:

1. Collects the ordered list of patches to be applied during the session. The list may contain more than one patch in case of Composite and System patches.
2. Runs the prerequisite checks on the patches.
3. Applies the patches.

4.3 Analyze Mode of Operation

When running in analyze mode, OPatchauto -binary runs the prerequisite checks on the patches but does not apply the bits. To run OPatchauto -binary in analyze mode, use the -analyze command line option. See "Syntax and Commands" on page 4-2 for more information.
4.4 Target Types in a SystemPatch

A SystemPatch contains a list of patches inside it and, for each of the patches, a list of applicable target types is defined. By default, `OPatchauto -binary` applies all the patches inside the SystemPatch to the selected Oracle home. You can specify a particular target type to have `OPatchauto -binary` apply only the applicable patches to the selected home.

4.5 Syntax and Commands

The command line syntax for the `OPatchauto -binary` utility is as follows.

```
OPatchauto [ apply | rollback ] <patch_location> -binary -oh <oracle_home path>
[-target_type <target_type>]
[-analyze]
[-invPtrLoc <oraInst.loc> ]
[-jre <JRE Location> ]
```

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;patch_location&gt;</td>
<td>Path to the location of the Patch. In case of a SystemPatch, this path should contain the <code>bundle.xml</code> file.</td>
</tr>
<tr>
<td>-oh</td>
<td>Path to the Oracle home to be patched.</td>
</tr>
<tr>
<td>-target_type</td>
<td>This specifies the type of the Oracle home. These are predefined values for every target type.</td>
</tr>
<tr>
<td>-analyze</td>
<td>Run only the prerequisite checks and report the feasibility of the patch session.</td>
</tr>
<tr>
<td>-invPtrLoc</td>
<td>Used to locate the <code>oraInst.loc</code> file when the installation used the <code>-invPtrLoc</code> option. This should be the path to the <code>oraInst.loc</code> file.</td>
</tr>
<tr>
<td>-jre</td>
<td>This option uses JRE (Java) from the specified location instead of the default location under the Oracle home.</td>
</tr>
</tbody>
</table>

Example 4–1  OPatchauto -binary Usage

```
OPatchauto apply /scratch/patches/12345678 -binary -target_type cluster -oh <GI_HOME>
```
This chapter describes common problems with patching and troubleshooting with OPatch. The following sections are discussed:

- Debugging: Enable Logging and Tracing
- References
- Products and Patch Types Not Supported by OPatch

5.1 Debugging: Enable Logging and Tracing

Logging and tracing is a common aid for debugging. OPatch maintains logs for all apply, rollback, and lsinventory operations. The log files are located at the following directory:

<ORACLE_HOME>/cfgtoollogs/opatch

Each log file is tagged with the timestamp of the operation. Log files are named as opatch_<date mm-dd-yyyy>_<time hh-mm-ss>.log.

Note: A new log file is created each time OPatch is executed.

For example, if a log file is created on May 17th, 2013 at 11.55 PM, it will be named as follows:

opatch_05-17-2013_23-55-00.log

Note: You can set OPatch to debug mode by setting the environment variable OPATCH_DEBUG to TRUE.

OPatch also maintains an index of the commands executed with OPatch and the log files associated with it in the opatch_history.txt file located in the <ORACLE_HOME>/cfgtoollogs/opatch directory. A sample of the history.txt file is as follows:

```
Date & Time : Tue Apr 26 23:00:55 PDT 2013
Oracle Home : /private/oracle/product/11.2.0/db_1/
OPatch Ver. : 12.1.0.1.2
Current Dir : /scratch/oui/OPatch
Command : lsinventory
Log File : /private/oracle/product/11.2.0/db_1/cfgtoollogs/opatch/opatch-2013_Apr_26_23-00-55-PDT_Tue.log
```
5.2 References

This section to contain a list of other documentation that may provide support. The main link will point to the "Master Note for OPatch" (Doc ID 293369.1) in MOS.

Patch Set FAQ: See Patchset FAQ (Doc ID 552777.1) for a complete summary of patchset FAQs. This document is available in My Oracle Support.

Information Center: Patching and Maintaining Oracle Database Server/Client Installations (Doc ID 1351428.1)

5.3 Products and Patch Types Not Supported by OPatch

Oracle now offers a wide range of products, including hardware and operating systems. However, patches required by some products are not currently supported by OPatch.

Note: If the Oracle product you install (such as Oracle Database or Fusion Middleware) creates an Oracle home directory, then OPatch is provided as part of that installation.

Typically, the types of Oracle products not supported by OPatch include:

- Hardware (for example, firmware updates for Sun servers).
- Operating system (for example, kernel patch updates for Oracle Linux or Oracle Solaris.)
- Java (for example, patch updates for JRE, JDK)
- Software products that do not create an Oracle home directory (for example, Oracle OpenOffice)
OPatch Syntax and Commands

This appendix provides a summary of the syntax and command options to use for the opatch command. Use these command options to develop your own patch plan.

A.1 OPatch Syntax

The OPatch utility is located in the $Oracle_Home/OPatch directory. You can run it with various commands and options. The following string shows the syntax for the OPatch utility:

```bash
<Path_to_OPatch>/opatch [-help] [-r[report]] [command] [-option]
```

where:
- `[-help]` displays the help message for the opatch command.
- `[-report]` prints the actions without executing.
- `[command]` is one of the OPatch commands described in Table A–1.
- `[-option]` is one of the OPatch command options. See each command listed below for a summary of available options.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>apply</td>
<td>Installs or applies a patch.</td>
</tr>
<tr>
<td>compare</td>
<td>Compare two files generated by the opatch lsinventory -xml command.</td>
</tr>
<tr>
<td>lsinventory</td>
<td>Lists what is currently installed on the system.</td>
</tr>
<tr>
<td>lspatches</td>
<td>Prints a summary of all installed patches.</td>
</tr>
<tr>
<td>napply</td>
<td>Installs n number of patches (hence napply).</td>
</tr>
<tr>
<td>nrollback</td>
<td>Rolls back patches from several product (e.g., Fusion Middleware) homes at the same time.</td>
</tr>
<tr>
<td>query</td>
<td>Queries a given patch for specific details.</td>
</tr>
<tr>
<td>rollback</td>
<td>Removes a patch.</td>
</tr>
<tr>
<td>version</td>
<td>Prints the current version of the patch tool.</td>
</tr>
<tr>
<td>prereq</td>
<td>Runs patching prerequisite checks on an ORACLE_HOME..</td>
</tr>
<tr>
<td>util</td>
<td>Invokes specified utilities on an GI/RAC home.</td>
</tr>
</tbody>
</table>

To view additional information for any command, use the following command:
If using Perl, use the following command:

```
perl opatch.pl command -help
```

To show the full syntax of the -help option, enter `opatch -h` to view the following display:

```
Usage: opatch [ -help ] [ -report ] [ command ]

command := apply
     compare
     lsinventory
     lspatches
     napply
     nrollback
     rollback
     query
     version
     prereq
     util

<global_arguments> := -help       Displays the help message for the command.
     -report     Print the actions without executing.

example:
  'opatch -help'
  'opatch -help -fmw'
  'opatch apply -help'
  'opatch compare -help'
  'opatch lsinventory -help'
  'opatch lspatches -help'
  'opatch napply -help'
  'opatch nrollback -help'
  'opatch rollback -help'
  'opatch prereq -help'
  'opatch util -help'

A.2 apply

The apply command applies an interim patch to an ORACLE_HOME from the current directory. The patch location can specified using the parameter patch_location. This command does not support System Patch.

**Syntax**

Use following syntax for this command:

```
patch apply  [-connectString  <List of connect strings>]
     [ -delay <value> ] [ -force ] [ -force_conflict ]
     [ -init <parameters for the init script in escaped double quotes> ] [ -opatch_init_end ]
     [ -invPtrLoc <Path to oraInst.loc> ]
     [ -jre <LOC> ] [ -local ] [ -local_node <Local node name> ]
     [ -minimize_downtime ] [ -no_bug_superset ] [ -no_inventory ]
     [ -no_relink ] [ -no_sysmod ] [ -ocmrf <Response file location> ]
     [ -oh <ORACLE_HOME> ]
     [ -post <parameters for the post script in escaped quotes> ] [ -opatch_post_end ]
```
apply

[-pre <parameters for the pre script
  in escaped double quotes> [-opatch_pre_end] ]
[-profile_mask <Name of profile>]
[-property_file <Path to property file>]
[-ptlConnect <portal connect string>]
[-ptlPassword <portal password>]
[-ptlSchema <portal schema>]
[-remote_nodes <List of remote nodes (node1,node2)>]
[-retry <value ] [-runSql ]
[silent ] [-sqlScript <path of the sql file>] [-verbose ]
[ <Patch Location> ]

Options
Table A–2 describes the options available for the apply command.

Table A–2  apply Command Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>delay</td>
<td>Specifies how many seconds to wait before attempting to lock the inventory in the case of a previous failure. You can use this option only if you specify the -retry option.</td>
</tr>
<tr>
<td>force</td>
<td>If a conflict exist which prevents the patch from being applied, the -force flag can be used to apply the patch. OPatch will remove all the conflicting patches before applying the current patch. In case of conflict among the patches to be applied, the non-conflicting patches will be applied.</td>
</tr>
<tr>
<td>force_conflict</td>
<td>If a conflict exists which prevents the patch from being applied, the -force_conflict flag can be used to apply the patch. OPatch will remove all the conflicting patches before applying the current patch. This will override the ‘silent’ behavior for conflicts and hence is meaningful only when used with the ‘silent’ option.</td>
</tr>
<tr>
<td>invPtrLoc</td>
<td>Specifies the location of the oraInst.loc file. This option is needed when the -invPtrLoc argument was used during installation. Oracle recommends using the default Central Inventory for a platform.</td>
</tr>
<tr>
<td>jre</td>
<td>Instructs OPatch to use JRE (Java) from the specified location instead of the default location under the GI/RAC home directory.</td>
</tr>
<tr>
<td>local</td>
<td>Specifies that the OPatch utility should patch the local node and update the inventory of the local node. It does not propagate the patch or inventory update to other nodes. You can use this option on Oracle Real Application Clusters environments and non-clustered environments. If an entire cluster is shut down before patching, you can use this argument for non-rolling patches.</td>
</tr>
<tr>
<td>local_node</td>
<td>Tells OPatch the local node for this cluster. You can use this option on Oracle Real Application Clusters environments.</td>
</tr>
<tr>
<td>minimize_downtime</td>
<td>Specifies the order of nodes that OPatch should patch. This option only applies to Oracle Real Application Clusters environments. You cannot use it with the -local option or a rolling patch.</td>
</tr>
<tr>
<td>no_bug_superset</td>
<td>Specifies to error out if the current patch bugs-to-fix is a superset (or same set) as an installed patch bugs-fixed in the GI/RAC home directory.</td>
</tr>
</tbody>
</table>
The compare command allows you to compare the bugs that have been fixed between two product (e.g., Fusion Middleware) homes. This command allows for comparison between two files generated by the `opatch lsinventory -xml` command. Currently, this command only accepts two files as input.

**Syntax**

```bash
opatch compare  [<file1> <file2>]
```
A.4 lsinventory

The lsinventory command lists the inventory for a particular GI/RAC home, or displays all installations that can be found. This command does not have any required options.

Syntax

Use the following syntax for this command:

```
```

```
[-ptlConnect <portal connect string>] [-ptlPassword <portal password>] [-ptlSchema <portal schema>]
[-property_file <path to property file>] [-retry <value>] [-translation_patch] [-xml <xmlFile>]
```

The following sections provide examples for the detail, bugs_fixed, and patch desc options. See Table A–3 for descriptions of the command options.

Options

Table A–3 describes the options available for the lsinventory command.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>all</td>
<td>Reports the name and installation directory for each GI/RAC home directory found.</td>
</tr>
<tr>
<td>bugs_fixed</td>
<td>Reports bugs fixed by installed patches in a tabular format. Besides the bugs fixed, the report also displays the installed patches, installed times, and bug descriptions. The fixed bugs are sorted per installed patch. Default display is patches in descending order based on installed time and ascending order of bugs within each patch. You can use 'asc' (or) 'desc' with this option to enforce sort order on bugs within each patch. You can use this option with the patch or patch option to obtain sort orders with installed patches.</td>
</tr>
<tr>
<td>delay</td>
<td>If you specify retry, this option tells OPatch how many seconds it should wait before attempting to lock the inventory again in case of a previous failure.</td>
</tr>
<tr>
<td>detail</td>
<td>Reports the installed products and other details. You cannot use this option with the -all option.</td>
</tr>
<tr>
<td>group_by_date</td>
<td>Specifies that OPatch should group all installed patches by the date they were installed in the GI/RAC home.</td>
</tr>
<tr>
<td>invPtrLoc</td>
<td>Specifies the location of the oraInst.loc file. You need this option if you used the invPtrLoc option during the installation. Oracle recommends using the default Central Inventory for a platform.</td>
</tr>
<tr>
<td>jre</td>
<td>Specifies the location of a particular JRE (Java) to use instead of the default location under the GI/RAC home directory.</td>
</tr>
</tbody>
</table>
The `lspatches` command prints a summary of all installed patches.

**Syntax**

Use the following syntax for the `lspatches` command:

```
opatch lspatches [-bugs] [-id <patch ID> ] 
[-invPtrLoc <Path to oraInst.loc> ] [-jre <LOC> ] 
[-oh <ORACLE_HOME> ] 
[-qfile <file path> ] [-required ] [-verify] <PATCH_ID or PATCH_LOCATION>
```

**Options**

`lspatches` command options are listed in Table A–4.

### Table A–4  `lspatches` Command Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bugs</td>
<td>Prints out bugs in addition to the summary</td>
</tr>
<tr>
<td>id</td>
<td>This option specifies the patch number. It must be registered in the GI/RAC home inventory. It can be any numeric sequence or combined with language. Example: 11111, 11111/zh_CN. It cannot support multiple patch IDs.</td>
</tr>
<tr>
<td>invPtrLoc</td>
<td>Used to locate the oraInst.loc file. When the installation uses the invPtrLoc flag, the value should indicate the path to oraInst.loc file</td>
</tr>
<tr>
<td>jre</td>
<td>This option tells OPatch to use JRE (java) from the specified location instead of the default location under GI/RAC home.</td>
</tr>
<tr>
<td>oh</td>
<td>The GI/RAC home to work on. This takes precedence over the environment variable ORACLE_HOME.</td>
</tr>
</tbody>
</table>
This command applies patches to several product (e.g., Fusion Middleware) homes at the same time. This command does not support System Patches.

**Syntax**

Use the following syntax for the *napply* command:

```
opatch napply [patch_location] [-id comma-separated list of patch IDs]
  [ -all_nodes ]
  [-connectString <List of connect strings>]
  [-delay <value>] [ -force ] [ -force_conflict ]
  [-idFile <path of the file that has list of patch IDs ]
  [-init <parameters for the init script in escaped double quotes>] [-opatch_init_end ]
  [-invPtrLoc <Path to oraInst.loc>] [ -jre <LOC>] [ -local ]
  [ -local_node <Local node name> ]
  [ -minimize_downtime ] [ -no_bug_superset ]
  [ -no_inventory ] [ -no_relink ]
  [ -no_sysmod ] [ -ocmrf <Response file location> ]
  [ -oh <ORACLE_HOME> ]
  [ -phBaseDir <Path to the directory that contains list of patch directories>] [
  [ -phBaseFile <Path to the file containing the location of the patches to be applied>] ]
  [-post <parameters for the post script in escaped double quotes>] [-opatch_post_end ]
  [-pre <parameters for the pre script in escaped double quotes>] [-opatch_pre_end ]
  [ -profile_mask <Name of profile>] [
  [ -property_file <Path to property file> ]
  [ -ptlConnect <portal connect string>] ]
  [-ptlPassword <portal password>]
```

**Example:**

```
opatch napply admin/rdbms/catcpu.sql
```

### Table A-4 (Cont.) `lspatches` Command Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>qfile</td>
<td>Specifies the relative path to GI/RAC home of the file to determine the latest patch that touches this file. Example: On Linux: admin/rdbms/catcpu.sql On Windows: admin\rdbms\catcpu.sql OPatch can tell which latest patch touches the file catcpu.sql in the GI/RAC home.</td>
</tr>
<tr>
<td>required</td>
<td>This option will print key metadata only. This includes the following metadata: required components, prereq patches, executables to shutdown and support platforms. This option should be accompanied by either option <code>-id &lt;PATCH_ID&gt;</code> or <code>&lt;PATCH_LOCATION&gt;</code>.</td>
</tr>
<tr>
<td>verify</td>
<td>This option verifies whether or not the specified patch ID or patch location is registered in the GI/RAC home inventory. In addition, this option validates all patch files in the GI/RAC home. This option should be accompanied by either option <code>-id &lt;PATCH_ID&gt;</code> or <code>&lt;PATCH_LOCATION&gt;</code>. This option doesn't support System Patch. Example: opatch lspatches -id 111 -verify opatch lspatches /scratch/test/111 -verify</td>
</tr>
</tbody>
</table>
napply

[-ptlSchema <portal schema>]
[ -remote_nodes <List of remote nodes (node1,node2)> ]
[-retry <value> ] [-runSql] [-silent ]
[-skip_subset]
[-skip_duplicate]
[-sqlScript <path of the sql file>]
[-verbose ]

Options

Table A–5 lists the options available for this command.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>delay</td>
<td>Specifies how many seconds to wait before attempting to lock the inventory again for a previous failure. You can use this option only if you specify the retry option.</td>
</tr>
<tr>
<td>force</td>
<td>If a conflict exist which prevents the patch from being applied, the -force flag can be used to apply the patch. OPatch will remove all the conflicting patches before applying the current patch. In case of conflict among the patches to be applied, the non-conflicting patches will be applied.</td>
</tr>
<tr>
<td>force_conflict</td>
<td>If a conflict exist which prevents the patch from being applied, the -force_conflict flag can be used to apply the patch. OPatch will remove all the conflicting patches before applying the current patch. This will override the 'silent' behavior for conflicts and hence is meaningful only when used with 'silent' option.</td>
</tr>
<tr>
<td>id</td>
<td>Use the 'lsinventory' option to display all patch ids. Each one-off patch is indicated by its id. A comma separated list of patches can be given to select the patches to be applied. For translation patches, the patch id should be of the format &lt;Patch ID&gt;/&lt;Language code&gt;.</td>
</tr>
<tr>
<td>idFile</td>
<td>The input to be given is a file location that contains a list of apply patch ids separated by commas or white spaces. This option cannot to be in conjunction with 'id' option. For translation patches, the patch id should be of the format &lt;Patch ID&gt;/&lt;Language code&gt;.</td>
</tr>
<tr>
<td>invPtrLoc</td>
<td>Specifies the location of the oraInst.loc file. The invPtrLoc option is needed when this option is used during installation. Oracle recommends the use of the default Central Inventory for a platform.</td>
</tr>
<tr>
<td>jre</td>
<td>Instructs OPatch to use JRE (Java) from the specified location instead of the default location under the GI/RAC home directory. You cannot specify the jdk and jre options together.</td>
</tr>
<tr>
<td>local</td>
<td>Specifies that OPatch should patch the local node and update the inventory of the local node. It does not propagate the patch or inventory update to other nodes. You can use this option on Oracle Real Application Clusters environments and non-clustered environments. If an entire cluster is shut down before patching, you can use this option for non-rolling patches.</td>
</tr>
<tr>
<td>no_bug_superset</td>
<td>Specifies to error out if the current patch’s bugs-to-fix is a superset (or same set) of an installed patch’s bugs-fixed in the GI/RAC home directory.</td>
</tr>
</tbody>
</table>
This option does not perform any make operations. You can use it during multiple patch applications and to perform the linking step only once. OPatch does not keep track of the make operations it did not perform. You need to make sure to execute OPatch without this option at the end for compilation.

Give OPatch the absolute path to the OCM response file to be used for OCM configuration. -silent must be used in conjunction with -ocmrf if GI/RAC home does not have OCM installed and configured.

Specifies the GI/RAC home directory to use instead of the default. This takes precedence over the environment variable ORACLE_HOME.

Marks the end of the post option. You use this option with the post option. If you do not use this option, everything after post until the end of the command is passed into post.

Marks the end of the pre options. You use this option with the pre option. If you do not use this option, everything after pre until the end of the command is passed into pre.

Used to specify a directory containing patch directories (or) zip files.

If you do not specify <patch_location>, use this option to point OPatch to a file containing a list of patches to be n-applied. Each line in the file points to a location of a patch.

Specifies the parameters to be passed to the post script. This script is executed after the patch is applied. You need to enclose the values for this option in double-quotes.

Specifies the parameters to be passed to the pre script. This script is executed before the patch is applied. You need to enclose the values for this option in double-quotes.

If the patch to be applied specifies WLS patch/patchset as prerequisites, OPatch will read the WLS default patch profile. To have OPatch read non-default patch profile, specify the patch profile name with this option.

Specifies the user-defined property file for OPatch to use. The path to the property file should be absolute. This property file takes precedence over the one that OPatch supplies.

Prints the action to the screen without executing it.

Tells OPatch how many times it should retry when there is an inventory lock failure.

Suppresses user interaction, and defaults any answers to "yes."

Skips patches to be applied that are duplicates of other patches installed in the GI/RAC home. Two patches are duplicates if they fix the same set of bugs.

Skips patches to be applied that are subsets of other patches installed in the GI/RAC home. One patch is a subset of another patch if the former fixes a subset of bugs fixed by the latter.

For example, if you used napply yesterday for patch A that fixed bugs 1 and 2, you use napply today with this option for patch B that fixes bug 1 and patch C that fixes bugs 1, 2, and 3. Then subset patch A is skipped, and patch C then becomes a superset of patch A.
Examples:

'opatch napply <patch_location>' to apply all patches under <patch_location> directory

'opatch napply <patch_location> -id 1,2,3' to apply patches 1, 2, and 3 which are present under <patch_location> directory

'opatch napply <patch_location> -skip_subset -skip_duplicate' to apply all patches under <patch_location> directory. OPatch will skip duplicate patches and subset patches (patches under <patch_location> that are subsets of patches installed in the Oracle Home)

'opatch napply <patch_location> -id 1,2,3 -skip_subset -skip_duplicate' to apply patches 1, 2, and 3 which are under <patch_location> directory. OPatch will skip duplicate patches and subset patches (patches under <patch_location> that are subsets of patches installed in the Oracle Home)

'opatch napply <patch_location> -idfile /tmp/list.txt' where list.txt contains a list of patch IDs to be applied. The list should be separated by a space or comma. For example: 1 2 3

'opatch napply <patch_location> -id 1/fr,2/de' to apply patches 1 (french patch), 1 (german patch) which are present in the <patch_location> directory

A.7 nrollback

The Nrollback command rolls back patches from several product (e.g., Fusion Middleware) homes at the same time.

Syntax

Use the following syntax for this command:

```
opatch nrollback -id <comma-separated list of patch IDs>  
[ -all_nodes ]  
[-connectString <List of connect strings>]  
[-delay <value>] -id <Comma separated list of patch IDs>  
[ -idFile <file location containing a list of rollback IDs separated by commas or white spaces>]  
[ -init <parameters for the init script in escaped double quotes> [-opatch_init_end] ]  
[ -invPtrLoc <Path to oraInst.loc> ]  
[ -jre <LOC> ] [-local]  
[ -local_node <Local node name>]  
[-no_inventory] [-no_relink] [-no_sysmod]  
[-oh <ORACLE_HOME>]  
[-post <parameters for the post script in escaped double quotes>[-opatch_post_end] ]  
[-pre <parameters for the pre script in escaped double quotes>[-opatch_pre_end] ]
```

[optional] (-property_file <Path to property file>)
[optional] (-ptlConnect <portal connect string>)
[optional] (-ptlPassword <portal password>)
[optional] (-ptlSchema <portal schema>)
[optional] (-remote_nodes <List of remote nodes (node1,node2)>)
[optional] (-retry <value>) [optional] [-runSql] [optional] [-silent]
[optional] (-sqlScript <path of the sql file>)
[optional] [-verbose]

Options
Table A–6 lists the options available for this command.

Table A–6  nrollback Command Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>delay</td>
<td>If you use the retry option with the rollback command, specifies how many seconds OPatch should wait before attempting to lock the inventory again if a previous failure occurs.</td>
</tr>
<tr>
<td>id</td>
<td>Indicates the patch to be rolled back. Use the lsinventory option to display all patch identifiers. Each one-off patch is indicated by its ID. To successfully roll back a patch, you must provide the patch identifier.</td>
</tr>
<tr>
<td>idFile</td>
<td>Use 'lsinventory' option to display all patch ids. Each one-off patch is indicated by its id. To rollback a patch, the id for that patch must be supplied. The input to be given is a file location that contains a list of rollback patch ids separated by commas or white spaces. For translation patches, the patch id should be of the format &lt;Patch ID&gt;/&lt;Language code&gt;. This option cannot to be in conjunction with 'id' option.</td>
</tr>
<tr>
<td>invPtrLoc</td>
<td>Specifies the location of the oraInst.loc file. You need to use this option if you used the invPtrLoc option during installation. Oracle recommends the use of the default Central Inventory for a platform.</td>
</tr>
<tr>
<td>jre</td>
<td>Specifies the location of a particular JRE (Java) for OPatch to use instead of the default location under the GI/RAC home directory.</td>
</tr>
<tr>
<td>local</td>
<td>Specifies that OPatch roll back the local node, then updates the inventory of the local node. It does not propagate the patch or inventory update to other nodes. You can use this option on Oracle Real Application Clusters environments and non-clustered environments. If an entire cluster is shut down before patching, you can use this option for non-rolling patches.</td>
</tr>
<tr>
<td>no_relink</td>
<td>This option does not perform any make operation in the patch. You can use this option during multiple patch removals and to perform the compilation step only once.</td>
</tr>
<tr>
<td>oh</td>
<td>Specifies the GI/RAC home directory to use instead of the default directory. This takes precedence over the ORACLE_HOME environment variable.</td>
</tr>
<tr>
<td>opatch_post_end</td>
<td>Marks the end of the post options. Use this option with the post option. If you do not use this option, everything after post until the end of the command is passed into post.</td>
</tr>
<tr>
<td>opatch_pre_end</td>
<td>Marks the end of the pre options. Use this option with the pre option. If you do not use this option, everything after pre until the end of the command is passed into pre.</td>
</tr>
</tbody>
</table>
A.8 rollback

The rollback command removes an existing one-off patch from the appropriate GI/RAC home directory indicated by the reference ID. The following syntax is used for this command:

Syntax

```
opatch rollback -id <ID> [-connectString <List of connect strings>]
[<delay <value>]]
[-init <parameters for the init script in escaped double quotes> [-opatch_init_end]]
[-invPtrLoc <Path to oraInst.loc>]
[-jre <LOC>] [-local]
[-local_node <Local node name>] [-no_inventory]
[-no_relink] [-no_sysmod]
[-oh <ORACLE_HOME>] [-ph <Patch Location>]
[-post <parameters for the post script in escaped double quotes> [-opatch_post_end]]
[-pre <parameters for the pre script in escaped double quotes> [-opatch_pre_end]]
[-property_file <path to property file>]
[-ptlConnect <portal connect string>]
```

Table A–6 (Cont.) nrollback Command Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>post</td>
<td>This option is used to pass parameters to the post script. This script is executed after removal of the patch. The value for this option have to be enclosed in double quotes. The parameters will be common parameters which will be passed to post scripts of all patches being rolled back. This option should be ended by option 'opatch_post_end'.</td>
</tr>
<tr>
<td>pre</td>
<td>This option is used to pass parameters to the pre script. This script is executed before removal of the patch. The value for this option have to be enclosed in double quotes. The parameters will be common parameters which will be passed to pre scripts of all patches being rolled back. This option should be ended by option 'opatch_pre_end'.</td>
</tr>
<tr>
<td>property_file</td>
<td>Specifies the user-defined property file for OPatch to use. The path to the property file should be absolute. This property file takes precedence over the one that OPatch supplies.</td>
</tr>
<tr>
<td>report</td>
<td>Prints the actions to the screen without executing them.</td>
</tr>
<tr>
<td>retry</td>
<td>Instructs OPatch how many times it should retry when there is an inventory lock failure.</td>
</tr>
<tr>
<td>silent</td>
<td>Suppressed user interaction, and defaults any yes/no questions to &quot;yes&quot;. An Oracle Real Application Clusters setup does not support this option.</td>
</tr>
<tr>
<td>verbose</td>
<td>Prints additional OPatch output to the screen as well as to the log file.</td>
</tr>
</tbody>
</table>

Examples

'opatch nrollback -id 1,2,3' to roll back patches 1, 2, and 3 that have been installed in the Oracle Home.

'opatch nrollback -id 1/fr,2/de to rollback patches 1 with language 'fr', 2 with language 'de' that have been installed in the Oracle Home.
rollback

[-ptlPassword <portal password>]
[-ptlSchema <portal schema>]
[-remote_nodes <List of remote nodes (node1,node2)>]
[-retry <value>] [-runSql] [-silent]
[-sqlScript <path of the sql file>] [-verbose]
[all_subpatches]

**Options**

Table A–7 describes the options available for the `rollback` command.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>delay</td>
<td>If you use the <code>retry</code> option with the <code>rollback</code> command, specifies how many seconds OPatch should wait before attempting to lock the inventory again if a previous failure occurs.</td>
</tr>
<tr>
<td>id</td>
<td>Indicates the patch to be rolled back. Use the <code>lsinventory</code> option to display all patch identifiers. Each one-off patch is indicated by its ID. To successfully roll back a patch, you must provide the patch identifier.</td>
</tr>
<tr>
<td>invPtrLoc</td>
<td>Specifies the location of the <code>oraInst.loc</code> file. You need to use this option if you used the <code>invPtrLoc</code> option during installation. Oracle recommends the use of the default Central Inventory for a platform.</td>
</tr>
<tr>
<td>jre</td>
<td>Specifies the location of a particular JRE (Java) for OPatch to use instead of the default location under the GI/RAC home directory.</td>
</tr>
<tr>
<td>local</td>
<td>Specifies that OPatch roll back the local node, then updates the inventory of the local node. It does not propagate the patch or inventory update to other nodes.</td>
</tr>
<tr>
<td>local_node</td>
<td>Specifies to OPatch that this is the local node for the cluster to be used for rollback. You can use this option for Oracle Real Application Clusters environments.</td>
</tr>
<tr>
<td>no_relink</td>
<td>Do not perform the make operations in the patch. This option can be used during multiple patch removals and perform the compilation step only once.</td>
</tr>
<tr>
<td>oh</td>
<td>Specifies the GI/RAC home directory to use instead of the default directory. This takes precedence over the <code>ORACLE_HOME</code> environment variable.</td>
</tr>
<tr>
<td>opatch_post_end</td>
<td>Marks the end of the <code>post</code> options. Use this option with the <code>post</code> option. If you do not use this option, everything after <code>post</code> until the end of the command is passed into <code>post</code>.</td>
</tr>
<tr>
<td>opatch_pre_end</td>
<td>Marks the end of the <code>pre</code> options. Use this option with the <code>pre</code> option. If you do not use this option, everything after <code>pre</code> until the end of the command is passed into <code>pre</code>.</td>
</tr>
<tr>
<td>ph</td>
<td>Specifies the valid patch directory area. Rollback uses the command types found in the patch directory to identify which commands are used for the current operating system.</td>
</tr>
<tr>
<td>post</td>
<td>Specifies the parameters to be passed inside the <code>post</code> script. This script executes after the patch is removed. You must enclose the value of this option in double-quotes.</td>
</tr>
</tbody>
</table>
The `query` command queries a specific patch for specific details. It provides information about the patch and the system being patched.

**Syntax**

Use the following syntax for this command:

```
opatch query [-all] [-is_auto_patch] [-is_translatable_patch]
[-get_base_bugs] [-get_component] [-get_os] [-get_date]
[-get_patch_language] [-get_patch_type] [-get_patching_model]
[-get_product_family] [-has_sql] [-is_online_patch]
[-is_rolling_patch] [-is_system_patch] [-jre <LOC>] [-oh <LOC>]
[ <patch_location> ]
```

**Options**

Table A–8 lists the options available for the `query` command.

**Table A–8 query Command Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>all</td>
<td>Retrieves all information about a patch. This is equivalent to setting all available options.</td>
</tr>
<tr>
<td>is_auto_patch</td>
<td>This option says 'true' if the patch is auto-enabled, 'false' otherwise. This command doesn't support System Patch.</td>
</tr>
<tr>
<td>is_system_patch</td>
<td>This option says 'true' if the patch is a System Patch, 'false' otherwise.</td>
</tr>
<tr>
<td>is_translatable_patch</td>
<td>This option says 'true' if the patch is translatable, 'false' otherwise. This option doesn't support System Patch.</td>
</tr>
<tr>
<td>get_base_bug</td>
<td>Retrieves bugs fixed by the patch.</td>
</tr>
<tr>
<td>get_component</td>
<td>Retrieves components the patch affects.</td>
</tr>
</tbody>
</table>
The `version` command shows the current version number of the OPatch utility.

**Syntax**

The following syntax is used for this command:

```
opatch version [-all] [-invPtrLoc <Path to oraInst.loc>] 
                [-jre <LOC>] [-oh <ORACLE_HOME>]
                [-v2c <5-digit version> -oui_loc <Custom OUI Location>
                -ph <Patch Location> -ohs <list of Oracle Homes
                separated by commas]
                [-help] [-h]
```

**Table A–9 version Command Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>all</td>
<td>This option displays versions of OPatch for all product (e.g., Fusion Middleware) homes registered in the Central Inventory.</td>
</tr>
<tr>
<td>invPtrLoc</td>
<td>Used to locate the oraInst.loc file. When the installation used the invPtrLoc flag. This should be the path to the oraInst.loc file.</td>
</tr>
<tr>
<td>jre</td>
<td>This option tells OPatch to use JRE (java) from the specified location instead of the default location under GI/RAC home.</td>
</tr>
<tr>
<td>oh</td>
<td>The GI/RAC home to work on. This takes precedence over the environment variable ORACLE_HOME</td>
</tr>
<tr>
<td>v2c</td>
<td>The standard 5-digit version to compare. If this option is specified with an valid version which made by no more than 5 numbers separated by '.', those product (e.g., Fusion Middleware) homes with valid version will be break up to two parts, one is those product (e.g., Fusion Middleware) homes which have opatch version higher or equal to the value of this option, and the other is those with lower version</td>
</tr>
</tbody>
</table>
This operation runs the prerequisite checks on an ORACLE_HOME. This command does not support System Patches.

**Syntax**

```
opatch prereq <command> [-id <Comma separated list of patch IDs>] [-invPtrLoc <Path to oraInst.loc>] [-jre <LOC>] [-local_node <Local node name>] [-oh <ORACLE_HOME>] [-ph <Path to the single patch location>] [-phBaseDir <Path to the dir containing all patches>] [-phBaseFile <Path to the file containing the location of the patches to be applied>] [-property_file <Path to property file>] [-remote_nodes <List of remote nodes (node1,node2)>] [-sid <Comma separated list of database SIDs>] [-connectString <List of connect strings>]
```

**Commands**

The `prereq` command executes commands that check for the prerequisite conditions shown in the table.

**Table A–10  `prereq` Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CheckActiveFilesAndExecutables</td>
<td>Check if there are any file(s) that are active, which are touched by the patch to be applied or rolled back.</td>
</tr>
<tr>
<td>CheckActiveServices</td>
<td>Check for the services that are active. Note: Applicable for Windows platforms only.</td>
</tr>
<tr>
<td>CheckApplicable</td>
<td>Check for the presence of the required components in the ORACLE_HOME and check if all the actions of the given patch(es) are applicable.</td>
</tr>
<tr>
<td>CheckApplicableProduct</td>
<td>Check if the patch is applicable for the given GI/RAC home. If the patch is marked for stand-alone homes, then it can not be applied on normal OUI-based home and vice versa. Also, a patch can be marked as a hybrid patch, where it is applicable for both homes.</td>
</tr>
<tr>
<td>CheckCentralInventoryForOH</td>
<td>Check if the given ORACLE_HOME is registered in the central inventory specified by the oraInst.loc file.</td>
</tr>
<tr>
<td>CheckCentralInventoryForRWSession</td>
<td>Check if a RW (read-write) session can be created for the given central inventory.</td>
</tr>
</tbody>
</table>
### Table A–10  (Cont.) `prereq` Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CheckCentralInventoryLocation</td>
<td>Validate the Central Inventory location. Check if it has the correct directory structure and has the inventory.xml with read permissions.</td>
</tr>
<tr>
<td>CheckComponents</td>
<td>Check for the presence of the required components in the ORACLE_HOME.</td>
</tr>
<tr>
<td>CheckConflictAgainstOH</td>
<td>Check if there are any conflicts between the patch(es) to be applied and the patch(es) in the OH.</td>
</tr>
<tr>
<td>CheckConflictAgainstOHWithDetail</td>
<td>Check if there are any conflicts between the patch(es) to be applied and the patch(es) in the OH, by giving out the detailed information about the conflicts/supersets.</td>
</tr>
<tr>
<td>CheckConflictAmongPatches</td>
<td>Check if there are any conflicts among the patch(es) to be applied.</td>
</tr>
<tr>
<td>CheckConflictAmongPatchesWithDetail</td>
<td>Check if there are any conflicts among the patch(es) to be applied, by giving out the detailed information about the conflicts/supersets.</td>
</tr>
<tr>
<td>CheckFileVersions</td>
<td>Check if the copy actions of Fusion Applications patch(es) have at least one or more file version(s) greater than the version(s) installed in the GI/RAC home.</td>
</tr>
<tr>
<td>CheckFusionAppsCompatible</td>
<td>Check if OUI for the GI/RAC home supports patching of Fusion applications.</td>
</tr>
<tr>
<td>CheckForIdenticalPatchInOracleHome</td>
<td>Check if the given list of patch(es) are identical with respect to the patch(es) installed in the GI/RAC home.</td>
</tr>
<tr>
<td>CheckForInputValues</td>
<td>Check if the input values provided to OPatch are enough for OPatch to proceed further.</td>
</tr>
<tr>
<td>CheckForNoOpPatches</td>
<td>Check if any of the patch(es) provided by the user are no-op patches. A no-op patch cannot be applied to the GI/RAC home and can be skipped. This prerequisite will fail for no-op. patches.</td>
</tr>
<tr>
<td>CheckIfOHLockedForPatching</td>
<td>Check if the ORACLE_HOME is locked for patching by any previous unsuccessful OPatch Session.</td>
</tr>
<tr>
<td>CheckInstalledOneOffs</td>
<td>Check if all the patches provided by the user to rollback are present in the given GI/RAC home.</td>
</tr>
<tr>
<td>CheckMinimumOPatchVersion</td>
<td>Check if all the patches provided by the user satisfy the requirement of minimum OPatch version for the OPatch currently being used.</td>
</tr>
<tr>
<td>CheckOneOffSuperset</td>
<td>Check if the given input Fusion Applications patch list are all candidates for one-off (or) singleton supersets. This prereq does not do any checks among the input patch list.</td>
</tr>
<tr>
<td>CheckOracleHome</td>
<td>Check if the given ORACLE_HOME is valid. Check if it has the inventory files with proper permissions.</td>
</tr>
<tr>
<td>CheckOraInstLocation</td>
<td>Check if the oraInst.loc file is proper and has the read permissions.</td>
</tr>
<tr>
<td>CheckOUILocation</td>
<td>Check the ORACLE_HOME for the presence of OUI.</td>
</tr>
<tr>
<td>CheckOUIVersionCompatible</td>
<td>Check if the OUI in the ORACLE_HOME is compatible for the OPatch.</td>
</tr>
<tr>
<td>CheckPatchApplicableOnCurrentPlatform</td>
<td>Check if the given patch(es) is applicable on the current platform.</td>
</tr>
<tr>
<td>CheckPatchApplyDependents</td>
<td>Check if all the patch(es) required by the patch(es) currently being installed is present in the GI/RAC home or not.</td>
</tr>
</tbody>
</table>
CheckPatchRollbackDependents: Check if there are any patch(es) in the GI/RAC home that are depending on the patch(es) being currently rolled back.

CheckPatchShipHome: Check if the given patch to be applied has the proper structure and has the correct permissions for all the files.

CheckRemoteCommandInvocable: Check if commands can be invoked on the remote machines.

CheckRemoteCopyAndRemove: Check if files can be copied to and removed from the remote machines.

CheckRequiredLibs: Check if all the required OUI libraries are present in the given ORACLE_HOME.

CheckRollbackable: Check if the given patch(es) can be rolled back from the ORACLE_HOME.

CheckSystemCommandAvailable: Check if all the commands required for applying or rolling back the given patch are present in the system.

CheckSystemSpace: Check if enough system space is available for the patch(es) to be applied.

CheckUserAdminPrivilege: Check if the user is 'root'. Note: OPatch should not be invoked by 'root', if so then this check fails.

CheckPatchingModel: Check if the patching model of all incoming patch(es) is compatible with that of the GI/RAC home.

**Table A-10 (Cont.) prereq Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>connectString</td>
<td>This option can be used to specify the list of database instances and remote nodes. The value for this option is specified as per the following syntax &quot;SID:User:Passwd:Node&quot;. The SID is a must, others can be ignored, OPatch takes default values for it. Example: oracle:dba:dba:mymachine,oracle1::: NOTE: If the system is not part of RAC setup, then to specify just the local node, provide the node name as empty string. This option cannot be used along with 'sid' option.</td>
</tr>
<tr>
<td>id</td>
<td>This option can be used to specify the patch IDs of all the patches that are to be rolled back from the given GI/RAC home.</td>
</tr>
<tr>
<td>invPtrLoc</td>
<td>Used to locate the oraInst.loc file. Needed when the installation used the -invPtrLoc flag. This should be the path to the oraInst.loc file.</td>
</tr>
<tr>
<td>jre</td>
<td>This option tells OPatch to use JRE (java) from the specified location instead of the default location under GI/RAC home.</td>
</tr>
<tr>
<td>local_node</td>
<td>This option can be used to specify to OPatch the local node name to be used for RAC mode application of the patch.</td>
</tr>
<tr>
<td>oh</td>
<td>The GI/RAC home to work on. This takes precedence over the environment variable ORACLE_HOME.</td>
</tr>
<tr>
<td>ph</td>
<td>This option can be used to specify the path to the patch location. Example: /tmp/101010</td>
</tr>
</tbody>
</table>
The `util` command invokes the chosen utilities on an ORACLE_HOME. This command does not support System Patches.

**Syntax**

```bash
opatch util [ -help ] [ COMMAND ]
```

Run `opatch util [ COMMAND ] -help` to get help on a specific command.

**Table A–12 `util` Commands**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CheckMinimumOpatchVersion</td>
<td>Check if a patch is compatible with the given OPatch version.</td>
</tr>
<tr>
<td>CheckComponents</td>
<td>Check if the given patch is suitable for the product (e.g., Fusion Middleware) homes registered in the Central Inventory by components check.</td>
</tr>
<tr>
<td>Cleanup</td>
<td>Remove the backup for restore area of the given patch or for all the patches.</td>
</tr>
<tr>
<td>DisableOnlinePatch</td>
<td>Disable and remove the specified online patch(es) on the given database instances.</td>
</tr>
<tr>
<td>EnableOnlinePatch</td>
<td>Install and enable the specified online patch(es) on the given database instances.</td>
</tr>
<tr>
<td>GetPatchLevel</td>
<td>Return the patching level on Local Grid Home.</td>
</tr>
<tr>
<td>InstallOCM</td>
<td>Install and configure OCM.</td>
</tr>
<tr>
<td>LoadXML</td>
<td>Prompt for path/name of the XML file, then check if the XML is correct.</td>
</tr>
<tr>
<td>SaveConfigurationSnapshot</td>
<td>Save configuration snapshot of current GI home to specified file. ORACLE_HOME shall point to GI home. Default snapshot file is ORACLE_HOME/cfgtoollogs/opatch/sysconfig/configData.txt</td>
</tr>
<tr>
<td>UpdateOpatchVersion</td>
<td>Update the version of OPatch in the inventory of GI/RAC home.</td>
</tr>
<tr>
<td>Verify</td>
<td>Using the defined ORACLE_HOME and given patch location via -ph, the program will check to make sure the patch was applied to the ORACLE_HOME. Example: ‘opatch util verify -ph /tmp/patchLoc’</td>
</tr>
</tbody>
</table>

**Table A–11 (Cont.) prereq Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>phBaseDir</td>
<td>This option can be used to specify the path to the base directory where all the patches to be applied are kept. Note: The directory should contain only non-duplicate patches in zipped or unzipped format.</td>
</tr>
<tr>
<td>phBaseFile</td>
<td>This option can be used to specify complete path to the file containing the location of the patches to be applied.</td>
</tr>
<tr>
<td>property_file</td>
<td>The user defined property file for OPatch to use. The path to the property file should be absolute. This property file takes precedence over the one that is supplied with OPatch.</td>
</tr>
<tr>
<td>sid</td>
<td>This option can be used to specify the SIDs of the database instances. This option can be used only for local system operations.</td>
</tr>
</tbody>
</table>
util
B

OPatchauto Syntax and Commands

This appendix provides a comprehensive listing and description of all OPatchauto commands.

IMPORTANT: OPatchauto commands must be run from the GI Home.

B.1 OPatchauto Commands

The OPatchauto commands are run from the product home out of the standard OPatch directory.

Example:

$PRODUCT_HOME/OPatch/OPatchauto apply <PATH_TO_PATCH_DIRECTORY>

where <PATH_TO_PATCH_DIRECTORY> is the full path to local staging area where you have downloaded your patches.

OPatchauto consists of three primary commands:

- apply
- resume
- rollback

And two global arguments:

- -help
- -version

B.1.1 apply

Apply a System Patch to a product home. User specified the patch location or the current directory will be taken as the patch location.

Important: OPatchauto must be run from the product home as a root user.

Syntax

OPatchauto apply [ <patch-location> ]
[ -phBaseDir <patch.base.directory> ]
[ -oh <home> ]
[ -phases <phases> ]
[ -log <log> ]
[ -logLevel <log_priority> ]
[ -binary ]
[ -analyze ]
[ -instance <instance> ]
[ -invPtrLoc <inventory.pointer.location> ]
[ -host <host> ]
[ -username <username> ]
[ -password <password> ]
[ -plan <patch.plan> ]
[ -wallet <wallet> ]
[ -walletPassword <wallet.password> ]
[ -topology <topology> ]]

Parameters

patch-location: The patch location.

Options

The following table describes the options available for the apply command.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>phBaseDir</td>
<td>The location of base patch directory.</td>
</tr>
<tr>
<td>oh</td>
<td>The location of the oracle home.</td>
</tr>
<tr>
<td>phases</td>
<td>The lifecycle phases and goals to execute.</td>
</tr>
<tr>
<td></td>
<td>Supported values:</td>
</tr>
<tr>
<td></td>
<td>init phase:</td>
</tr>
<tr>
<td></td>
<td>init:init</td>
</tr>
<tr>
<td></td>
<td>shutdown phase:</td>
</tr>
<tr>
<td></td>
<td>shutdown:prepare-shutdown</td>
</tr>
<tr>
<td></td>
<td>shutdown:shutdown</td>
</tr>
<tr>
<td></td>
<td>offline phase:</td>
</tr>
<tr>
<td></td>
<td>offline:prepare</td>
</tr>
<tr>
<td></td>
<td>offline:binary-patching</td>
</tr>
<tr>
<td></td>
<td>offline:product-patching</td>
</tr>
<tr>
<td></td>
<td>offline:finalize</td>
</tr>
<tr>
<td></td>
<td>startup phase:</td>
</tr>
<tr>
<td></td>
<td>startup:startup</td>
</tr>
<tr>
<td></td>
<td>startup:finalize</td>
</tr>
<tr>
<td></td>
<td>online phase:</td>
</tr>
<tr>
<td></td>
<td>online:product-patching</td>
</tr>
<tr>
<td></td>
<td>online:finalize</td>
</tr>
<tr>
<td></td>
<td>finalize phase:</td>
</tr>
<tr>
<td></td>
<td>finalize:finalize</td>
</tr>
<tr>
<td>log</td>
<td>The log location.</td>
</tr>
<tr>
<td>logLevel</td>
<td>The log level (defaults to &quot;INFO&quot;).</td>
</tr>
<tr>
<td>binary</td>
<td>Forces execution of -phases offline:binary-patching.</td>
</tr>
<tr>
<td>analyze</td>
<td>If analysis should be performed and no actual patching.</td>
</tr>
<tr>
<td>instance</td>
<td>The instance location.</td>
</tr>
</tbody>
</table>
Table B–1 (Cont.) apply Command Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>invPtrLoc</td>
<td>The central inventory pointer file location.</td>
</tr>
<tr>
<td>host</td>
<td>The remote host.</td>
</tr>
<tr>
<td>username</td>
<td>The remote host username.</td>
</tr>
<tr>
<td>password</td>
<td>The remote host password.</td>
</tr>
<tr>
<td>plan</td>
<td>The patch plan to execute.</td>
</tr>
<tr>
<td>wallet</td>
<td>The location of the wallet file.</td>
</tr>
<tr>
<td>walletPassword</td>
<td>The wallet password.</td>
</tr>
<tr>
<td>topology</td>
<td>The location of the topology file.</td>
</tr>
</tbody>
</table>

Notes:
If OPatchauto apply is run and encounters an individual patch within a patch set that cannot be installed, that patch will be skipped and OPatchauto will continue with the installation of the next patch in the sequence.

If OPatchauto apply is run and encounters an individual patch that is identical (same patch ID and Unique Patch Identifier (UPI)) to a patch already installed in the product home, OPatchauto perform the following based on specific patch conditions:

- If the individual patch was created later than the product home patch, OPatchauto installs the individual patch.
- If the individual patch’s creation date is the same as the product home patch, OPatchauto will skip installing the individual patch.
- If the individual patch was created before the product home patch, an error will be generated.

This analyze option simulates an OPatchauto apply session by running all prerequisite checks, when possible, without making changes to the system (either bits or configurations). Because the analyze command does not modify the system, it will perform the following checks:

- Run SQL sync in analyze mode.
- Validate all pre and post processing steps making sure the command is present and executable.

B.1.2 resume

This operation resumes a previous patching session.

Important: OPatchauto must be run from the GI Home as a root user.

Syntax

OPatchauto resume [ -log <log> ] [ -logLevel <log_priority> ]
[ -password <password> ]
[ -walletPassword <wallet.password> ]
[ -session <session> ]

Options

The following table describes the options available for the resume command.
**Table B–2 resume Command Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>log</td>
<td>The log location.</td>
</tr>
<tr>
<td>logLevel</td>
<td>The log level (defaults to ‘INFO’).</td>
</tr>
<tr>
<td>password</td>
<td>The remote host password.</td>
</tr>
<tr>
<td>walletPassword</td>
<td>The wallet password.</td>
</tr>
<tr>
<td>session</td>
<td>The session id.</td>
</tr>
</tbody>
</table>

### B.1.3 rollback

This operation rolls back a patch.

**Important:** OPatchauto must be run from the product home as a root user.

**Syntax**

```
OPatchauto rollback [ <patch-location> ]
    [ -phBaseDir <patch.base.directory> ]
    [ -oh <home> ] [ -phases <phases> ] [ -log <log> ]
    [ -logLevel <log_priority> ] [ -analyze ]
    [ -instance <instance> ]
    [ -invPtrLoc <inventory.pointer.location> ]
    [ -host <host> ] [ -username <username> ]
    [ -password <password> ] [ -plan <patch.plan> ]
    [ -id <id> ] [ -wallet <wallet> ]
    [ -walletPassword <wallet.password> ]
    [ -topology <topology> ]
```

**Parameters**

- **patch-location**: The patch location.

**Options**

The following table describes the options available for the `rollback` command.

**Table B–3 rollback Command Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>phBaseDir</td>
<td>The location of base patch directory.</td>
</tr>
<tr>
<td>oh</td>
<td>The location of the oracle home.</td>
</tr>
</tbody>
</table>
### B.1.4 help

You can view online help for any command by specifying the `-help` or `-h` option.

For example:

OPatchauto -help
OPatchauto -version
OPatchauto apply -help
OPatchauto resume -help

---

**Table B–3 (Cont.) rollback Command Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>phases</td>
<td>The lifecycle phases and goals to execute.</td>
</tr>
<tr>
<td></td>
<td>Supported values:</td>
</tr>
<tr>
<td></td>
<td>- init phase:</td>
</tr>
<tr>
<td></td>
<td>-init:init</td>
</tr>
<tr>
<td></td>
<td>- shutdown phase:</td>
</tr>
<tr>
<td></td>
<td>shutdown:prepare-shutdown</td>
</tr>
<tr>
<td></td>
<td>shutdown:shutdown</td>
</tr>
<tr>
<td></td>
<td>- offline phase:</td>
</tr>
<tr>
<td></td>
<td>offline:prepare</td>
</tr>
<tr>
<td></td>
<td>offline:binary-patching</td>
</tr>
<tr>
<td></td>
<td>offline:product-patching</td>
</tr>
<tr>
<td></td>
<td>offline:finalize</td>
</tr>
<tr>
<td></td>
<td>- startup phase:</td>
</tr>
<tr>
<td></td>
<td>startup:startup</td>
</tr>
<tr>
<td></td>
<td>startup:finalize</td>
</tr>
<tr>
<td></td>
<td>- online phase:</td>
</tr>
<tr>
<td></td>
<td>online:product-patching</td>
</tr>
<tr>
<td></td>
<td>online:finalize</td>
</tr>
<tr>
<td></td>
<td>- finalize phase:</td>
</tr>
<tr>
<td></td>
<td>finalize:finalize</td>
</tr>
<tr>
<td>log</td>
<td>The log location.</td>
</tr>
<tr>
<td>logLevel</td>
<td>The log level (defaults to &quot;INFO&quot;).</td>
</tr>
<tr>
<td>analyze</td>
<td>If analysis should be performed and no actual patching.</td>
</tr>
<tr>
<td>instance</td>
<td>The instance location.</td>
</tr>
<tr>
<td>invPtrLoc</td>
<td>The central inventory pointer file location.</td>
</tr>
<tr>
<td>host</td>
<td>The remote host.</td>
</tr>
<tr>
<td>username</td>
<td>The remote host username.</td>
</tr>
<tr>
<td>password</td>
<td>The remote host password.</td>
</tr>
<tr>
<td>plan</td>
<td>The patch plan to execute.</td>
</tr>
<tr>
<td>id</td>
<td>The patch id.</td>
</tr>
<tr>
<td>wallet</td>
<td>The location of the wallet file.</td>
</tr>
<tr>
<td>walletPassword</td>
<td>The wallet password.</td>
</tr>
<tr>
<td>topology</td>
<td>The location of the topology file.</td>
</tr>
</tbody>
</table>
Wallet Creation Commands

OPatchauto rollback -help

B.1.5 version

Print the version of the OPatch utility, dependent OPlan version, and the osysmodel version.

**Important:** OPatchauto must be run from the GI Home as a root user.

**Syntax**

```bash
<GI_HOME>/OPatch/OPatchauto version [-invPtrLoc <Path to oraInst.loc>]
[-jre <LOC>]
[-oh <ORACLE_HOME>]
[-oui_loc <Custom OUI Location>]
[-help] [-h]
```

**Options**

The following table describes the options available for the `version` command.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-invPtrLoc</td>
<td>The <code>invPtrLoc</code> option is used to locate the Central Inventory Pointer File (<code>oraInst.loc</code>). Input for this option is the path to the <code>oraInst.loc</code> file.</td>
</tr>
<tr>
<td>-jre</td>
<td>This <code>jre</code> option instructs OPatchauto to use the JRE (java) from the specified location instead of the default location under GI/RAC home.</td>
</tr>
<tr>
<td>-oh</td>
<td>The <code>oh</code> option specifies the GI/RAC home to work on. This takes precedence over the environment variable ORACLE_HOME.</td>
</tr>
<tr>
<td>-oui_loc</td>
<td>The <code>oui_loc</code> option displays the custom OUI location from which OPatch runs.</td>
</tr>
</tbody>
</table>

B.2 Wallet Creation Commands

**Via configWallet.sh or configWallet.cmd**

```bash
./config-wallet.sh -create "${USER}&${HOSTNAME}:ssh" "${USER}&${HOST1}:ssh"
"${USER}&${HOST2}:ssh" "${USERNAME}&${HOSTNAME}:wls"
```

**Via mkstore**

```bash
mkstore -wrl "/u01/app/oracle/wallet" -create
mkstore -wrl "/u01/app/oracle/wallet" -listCredential
mkstore -wrl /scratch/aime/fmw/walletdir -createUserCredential oracle WEBLOGIC welcome1
```

For more information about mkstore, see the chapter "Creating and Managing Oracle Wallet" in the Oracle® Fusion Middleware Administering JDBC Data Sources for Oracle WebLogic Server guide.
Glossary

**Bundle Patch**
A bundle patch is a cumulative collection of fixes for a specific product or component. A patch of this type is released as needed depending on the product's requirements. You may also know a bundle patch as: maintenance pack, service pack, MLRs, cumulative patch, or update release.

**Diagnostic Patch**
A diagnostic patch is designed to help diagnose or verify a fix or collection of bug fixes. You may also know a diagnostic patch as: test patch, Fix Verification Binary (FVB) or e-fix.

**Interim Patch**
An interim patch provides a single bug fix, a collection of bug fixes, or a customer-specific security fix. They generally address specific bugs for a particular customer, and generally should not be applied unless instructed by Oracle Support to do so. You may also know an interim patch as: security one-off, exception release, x-fix, PSE, MLR, or hotfix.

**MLR**
Merge Label Request. A bundle of patches fixing several bugs.

**Patch Set**
The main way in which Oracle provides bug fixes in between releases. Oracle bundles a number of fixes, test them thoroughly together, and package them together for easy download and installation. They generally do not include new functionality and do not require a new certification. All of the fixes in the patch set have been tested and are certified to work with each other.

**Patch Set Update**
A collection of proactive, stabilizing cumulative patches for a particular product version (base release or patch set). PSUs are cumulative and include all of the security fixes from SPU patches (formerly known as CPU), plus additional fixes.

**Security Patch Update**
A security patch update is a cumulative collection of security-related bug fixes. Generally, security patch updates are release regularly. The security patch update was previously known as Critical Patch Update or CPU.

**Singleton**
A patch with one bug fix.
Patch
A patch is a piece of code/software designed to fix problems with the existing code/software. This includes fixing security vulnerabilities and other bugs, and improving the usability or performance.

Patch Conflict
If a patch makes different changes to the same section of code that another OPatch modifies, then these two patches conflict, and only one of them can be installed (unless a merge or overlay patch is available).

Superset Patch
If a particular patch to be applied contains all of the fixes included in an already installed patch, plus additional fixes, then the patch with more fixes is a superset patch, and there is no conflict.

Combination Conflict
If a patch to be installed conflicts with more than one already installed patch, this is considered a combination conflict. In this case, OPatch will remove all conflicting patches then apply only the new patch.

Critical Patch Update
Critical Patch Updates (CPUs) are the primary means of releasing security fixes for Oracle products. CPUs are cumulative with respect to prior CPUs and generally contain only security fixes.

Merge Patch
A merge patch is one where multiple conflicting patches are combined into one integrated patch.

Overlay Patch
When an interim patch conflicts with a PSU, patch conflict resolution is achieved by providing a new patch that coexists with (and requires) the PSU patch. The new patch overlays the PSU, and the PSU is a prerequisite for the overlay patch.

Shared/Non-shared (GI or RAC) Home
In a shared GI/RAC home, all nodes in the cluster use the same physical copy of the software. This simplifies configuration and management of many database operations because there is a single Home location rather than separate Homes on each node.

When a GI Home or RAC Home is shared, individual nodes within the GI or RAC environments share a single file system and utilize a cluster file system such as Oracle Cluster File System 2 (OCFS2), in addition to sharing the same Home. Although this configuration is more disk space-efficient, the process of patching becomes a bit more complicated as the different nodes are utilizing the same resources/disk space.

Note: GI shared home installations can be patched only in non-rolling mode.

In a non-shared GI/RAC home, sometimes referred to as Private GI/RAC home, each node in the cluster maintains a complete copy of the Oracle software tree on local storage. This is the most common way in which Oracle Grid Infrastructure and Real Application Clusters are installed.
Index

B
- BPU, 1-5
- Bundle Patch Updates, 1-5

C
- Configuration Patching, 1-6

D
- Diagnostic patches, 1-5

E
- Enterprise Manager, patching with, 1-6

I
- Interim patches, 1-5
- interim patches for security bug fixes, 1-5

L
- Logging and Tracing, 5-1
- Logging and tracing, 5-1

M
- Manual Patching, 1-6
- Merge Label Request, 1-5
- MLR, 1-5
- Multiple Patches, applying, 2-3
- My Oracle Support, 1-2

O
- OPatch, 1-1
  - compare, A-4
  - lsinventory, A-5
  - lspatches, A-6
  - napply, A-7
  - nrollback, A-10
  - prereq, A-16
  - query, A-14
  - rollback, A-12
  - util, A-19
  - version, A-15
- OPatch Syntax, A-1
- OPatch Utilities, accessing, 1-8
- OPatchauto, 1-1
  - apply, B-1
  - resume, B-3
  - rollback, B-4
  - version, B-6
- OPatchautoBinary, 4-1
  - analyze mode, 4-1
  - supported patch types, 4-1
  - syntax, 4-2
- SystemPatch, 4-2
- Oracle Patches, types, 1-5
- Oracle Support Services, 1-5
- ORACLE_HOME, environment variable, 2-2

P
- Patch Application Prerequisites, 2-2
- Patch Conflict, 2-4
- Patch Conflict Detection and Resolution, 2-4
- Patch Process, 1-2
- Patch Set Updates, 1-5
- Patch Types, not supported, 5-2
- patches, Diagnostic, 1-5
- patches, Interim, 1-5
- Patches, obtaining, 1-2
- PSU, 1-5

S
- Security Patch Updates, 1-5
- Single Patch, applying, 2-3
- SPU, 1-5
- System Patch
  - patch, system, 1-5

T
- troubleshooting, 5-1