Oracle® Exalogic Elastic Cloud

ExaPatch User's Guide

Release 1.2

E50725-06

October 2015

This document describes how to install, configure, and use ExaPatch, a utility that simplifies the process of patching and upgrading the Exalogic infrastructure (ILOM, firmware, and software).

This document contains the following sections:

1 Preparing to Use ExaPatch

This section describes the tasks that must be completed before using ExaPatch to patch and upgrade the Exalogic infrastructure:

1.1 Installing ExaPatch

ExaPatch is a part of the Exalogic Lifecycle Toolkit (ELLC). For instructions about installing ExaPatch, see the ELLC My Oracle Support document ID 1912063.1.

After following the instructions in the My Oracle Support document, ExaPatch can be run from `/exalogic-lctools/bin/exapatch`.

1.2 Creating the Rack Configuration File

ExaPatch requires a rack configuration file containing the IP addresses and login name for each patchable component on the rack. For each component, the configuration file also includes a directive that specifies how ExaPatch authenticates itself to the component.

- For Exalogic physical and virtual configurations, generate the rack configuration file from the output of the ExaDiscover utility as described in Section 1.2.1.
- If required, you can create the rack configuration file by using a template, as described in Section 1.2.2.

1.2.1 Generating the Rack Configuration File from the Output of ExaDiscover

To generate the rack configuration file from the output of the ExaDiscover utility, which is bundled with ExaPatch, do the following:

1. Navigate to the `exadiscover` library directory in the installation directory of ExaPatch:

   ```
   # cd /exalogic-lctools/lib/exapatch/1.2.x/exadiscover
   ```

2. Generate a list of the components on the rack by running ExaDiscover from the first compute node, as follows:

   ```
   # ./exadiscover.sh -h address_of_compute_node1 > /tmp/exadiscover_output.out
   ```

   In this command, `address_of_compute_node1` is the host name or IP address of the **first** compute node in your Exalogic machine or the compute node on which Exalogic Configuration Utility was run.

   Note: When running ExaDiscover, you can specify the `root` user password for the compute node by using the `-p` *`password`* option.

3. Generate the rack configuration file from the output of ExaDiscover, by running the `createrackconfig.sh` script, which is available in the `/exalogic-lctools/lib/exapatch/1.2.x/scripts` folder, as follows:

   ```
   # cd /exalogic-lctools/lib/exapatch/1.2.x/scripts
   # ./createrackconfig.sh /tmp/exadiscover_output.out >
   /exalogic-lcdata/inventory/rack_configuration.py
   ```

   In this command, `rack_configuration.py` is the rack configuration file.

   Note: Oracle recommends creating a rack configuration file called `rack_configuration.py` in the `/exalogic-lcdata/inventory/` directory. By default, ExaPatch searches `/exalogic-lcdata/inventory/` for a rack configuration file called `rack_configuration.py`.

   The rack configuration file that is generated based on the output of ExaDiscover contains, by default, the `IPoIB-admin` IP addresses of the components on the rack.

4. If the IP addresses of the ZFS appliance heads in the `getZfsHeadHostInfo()` entry are the IPoIB-admin addresses, manually edit the `rack_configuration.py` file and set the two IP addresses to the `eth-admin` addresses of the ZFS appliance heads.

For a sample of the rack configuration file, see Section 1.4.

1.2.2 Creating the Rack Configuration File by Using a Template

You can create the rack configuration file from a template by running the `createrackconfig.sh` script as follows:

Syntax:

```
createrackconfig.sh -p|-v -c number_of_compute_nodes >
/exalogic-lcdata/inventory/rack_configuration.py
```

Table 1 createrackconfig.sh Options

Option	Purpose
`-c` *number*	Specifies the number of compute nodes on the Exalogic machine
`-p`	Specifies that the rack is in a physical configuration
`-v`	Specifies that the rack is in a virtual configuration

Example:

```
createrackconfig.sh -p -c 4 > /exalogic-lcdata/inventory/rack_configuration.py
```

Note: Oracle recommends creating a rack configuration file called `rack_configuration.py` in the `/exalogic-lcdata/inventory/` directory. By default, ExaPatch searches `/exalogic-lcdata/inventory/` for a rack configuration file called `rack_configuration.py`.

The rack configuration file will be of the format:

```
class RackConfig:
      def getComputeNodesHostInfo(self):
      return [
          ['FIXME', 'root', DefaultPassword]
      ]
```

You must edit the rack configuration file, and replace each `FIXME` entry with the IP address of the host, as shown in the sample rack configuration file in Section 1.4.

For the ZFS appliance heads, ensure that you use the `eth-admin` IP addresses.

Note: Do not edit the formatting of the rack configuration file.

Ensure that all the patchable components on the rack are listed in the configuration file. Ensure that the ILOM entries match the equivalent compute-node entries—that is, the first compute-node ILOM in the list must be physically connected to the first compute node in the list, and so on.

1.3 Specifying How ExaPatch Should Log In to Exalogic Components

You can configure how ExaPatch obtains the credentials for logging in to the components on an Exalogic rack in the following ways:

- Section 1.3.1, "Prompt for the Passwords at Runtime"
- Section 1.3.2, "Use Factory-Default Passwords"
- Section 1.3.3, "Use Password-Less SSH"

1.3.1 Prompt for the Passwords at Runtime

You can configure ExaPatch to prompt for the passwords at runtime in the following ways:

- To configure ExaPatch to prompt once for the password of all the physical components and once for the password of all the Exalogic Control vServers, use the following directives:

  - For physical components: `PromptRackPassword`
  - For Exalogic Control vServers (applicable only for Exalogic virtual configurations): `PromptECVServerPassword`

  ExaPatch prompts only once for each password even if the directive is repeated elsewhere in the file; so use these directives only if all the physical components have the same password and all the Exalogic Control vServers have the same password.

  Example (for physical components, compute nodes in this case):

  ```
  def getComputeNodesHostInfo(self):
      return [
          ['10.10.10.1', 'root', PromptRackPassword]
      ]
  ```

  Example (for Exalogic Control vServers):

  ```
  def getOvmmHostInfo(self):
      return [
          ['20.20.20.20', 'root', PromptECVServerPassword]
      ]
  ```

- Prompt for passwords for each component *group*:

  To configure ExaPatch to prompt the user once for the password for each component group (compute nodes, NM2-GW switches, vServers, and so on), use the `PromptGroupPassword` directive.

  Example:

  ```
  def getComputeNodesHostInfo(self):
      return [
          ['10.10.10.1', 'root', PromptGroupPassword]
          ['10.10.10.2', 'root', PromptGroupPassword]
      ]
  ```

- Prompt for the password of each component

  To configure ExaPatch to prompt individually for the password of each component, use the `PromptPassword` directive.

  Example:

  ```
  def getComputeNodesHostInfo(self):
  ```

```
return [
    ['10.10.10.1', 'root', PromptPassword]
    ['10.10.10.2', 'root', PromptPassword]
]
```

Note: For security reasons, the passwords that you enter at the prompts are not stored. The password prompts are displayed each time you run ExaPatch.

1.3.2 Use Factory-Default Passwords

If the credentials for the components on the Exalogic machine are at the factory default values, specify the `DefaultPassword` directive in the rack configuration file, as shown in the following example:

```
['host', 'root', DefaultPassword]
```

1.3.3 Use Password-Less SSH

For components that have been configured for password-less SSH access, specify the `PasswordlessSSH` directive as shown in the following example:

```
['host', 'root', PasswordlessSSH]
```

If you have specified the `PasswordlessSSH` directive, ExaPatch does not display password prompts for accessing the components on the rack.

Note: If a passphrase was specified while generating the SSH keys, ExaPatch cannot log in to the component, and an error message will be displayed. In such cases, you can configure ExaPatch to prompt for the password, as described in Section 1.3.1, "Prompt for the Passwords at Runtime."

1.4 Sample Rack Configuration File

The following is a sample rack configuration file for an Exalogic virtual configuration.

```
class RackConfig:

  def getComputeNodesHostInfo(self):
    return [
      ['10.10.54.171', 'root', DefaultPassword],
      ['10.10.54.172', 'root', DefaultPassword],
      ['10.10.54.173', 'root', DefaultPassword],
      ['10.10.54.174', 'root', DefaultPassword],
  ]
  def getComputeNodeIlomHostInfo(self):
    return [
      ['10.10.54.181', 'root', DefaultPassword],
      ['10.10.54.182', 'root', DefaultPassword],
      ['10.10.54.183', 'root', DefaultPassword],
      ['10.10.54.184', 'root', DefaultPassword],
  ]
  def getZfsHeadHostInfo(self):
    return [
      ['10.10.54.179', 'root', DefaultPassword],
      ['10.10.54.180', 'root', DefaultPassword],
  ]
  def getZfsHeadIlomHostInfo(self):
    return [
      ['10.10.54.189', 'root', DefaultPassword],
      ['10.10.54.190', 'root', DefaultPassword],
  ]
  def getNM2_gwIbSwitchHostInfo(self):
    return [
      ['10.10.54.192', 'root', DefaultPassword],
      ['10.10.54.193', 'root', DefaultPassword],
  ]
  def getNM2_36pIbSwitchHostInfo(self):
    return [
  ]
  def getPduHostInfo(self):
    return [
      ['10.10.54.194', 'admin', DefaultPassword],
      ['10.10.54.195', 'admin', DefaultPassword],
  ]
  def getOvmmHostInfo(self):
    return [
      '192.168.20.11', 'root', DefaultPassword
  ]
  def getEcEmocPcHostInfo(self):
    return [
      ['192.168.20.13', 'root', DefaultPassword],
      ['192.168.20.14', 'root', DefaultPassword],
  ]
  def getOVMMServiceCredentials(self):
    return [
      'admin', DefaultPassword
  ]

# class definitions for password values
class PromptPassword:
      pass
class PromptGroupPassword:
      pass
class PromptRackPassword:
      pass
class PromptECVServerPassword:
      pass
class PasswordlessSSH:
      pass
class DefaultPassword:
      pass
```

Note: The `getOvmmHostInfo()`, `getEcEmocPcHostInfo()`, and `getOVMMServiceCredentials()` function and `PromptECVServerPassword` class definitions toward the end of the rack configuration file are available in the case of Exalogic *virtual* configurations only.

1.5 Location for Running ExaPatch

The location from which you can run ExaPatch varies depending on the component that you want to patch as follows:

- Patching InfiniBand switches; and the ZFS storage head ILOMs, software, and workflows: Run ExaPatch on the Exalogic Control vServer (recommended in the case of an Exalogic virtual configuration) or any compute node
- Patching compute nodes or compute node ILOMs: Run ExaPatch on any compute node that is not being patched.
- Patching Exalogic Control vServers: Run ExaPatch on any compute node.
- Patching guest vServers: Run ExaPatch on the Exalogic Control vServer.

Note: If you run ExaPatch on a compute node, ExaPatch cannot access the ILOM of that compute node.

1.6 ExaPatch Command Syntax and Options

The following is the general syntax of the ExaPatch command:

```
# /exalogic-lctools/bin/exapatch [options]
```

Table 2 describes the options of the ExaPatch command.

Table 2 ExaPatch Command Options

Option	**Description**
`-a` *`action`* [*`component`*] or `--action=`*`action`* [*`component`*]	The action to be performed. Specify one of the following values for the `action` argument: ▪ `baseVersion`: Display the current version of a component as listed in the rack history file. If you created a rack history file using the `-a refreshHistory` option, you can obtain the version information of rack components, even if they are powered down. ▪ `checkAuthentication`: Check whether ExaPatch can successfully log in to the components on the rack by using the password directives defined in the rack configuration file. ▪ `ecvserversshutdown`: Stop the Exalogic Control vServers in the following sequence: Proxy Controller 2, Proxy Controller 1, Exalogic Control. ▪ `ecvserversstartup`: Start the Exalogic Control vServers in the following sequence: Exalogic Control, Proxy Controller 1, Proxy Controller 2. ▪ `getHistory`: Display the rack history information from the rack history xml file you generated using the `-a refreshHistory` option for the specified components. ▪ `getVersion`: Check the version number of each patchable component. ▪ `listComponents`: View the components that are included in the rack configuration file. ▪ `listVMs`: Obtain a list of the vServers running on specified compute nodes. ▪ `patch`: Apply patches to the specified components. ▪ `postPatchCheck`: Check whether the patch was applied successfully. ▪ `prePatchCheck`: Check whether the specified component is ready for patching. ▪ `refreshHistory`: Create a rack history file. The rack history file stores the patching-related activities performed for each component. ▪ `restartovsagent`: Restart the Oracle VM Server agent on all Oracle VM Servers. When Oracle VM Manager cannot communicate with an Oracle VM Server, use this action. ▪ `runExtension`: Run the specified ExaPatch plug-in, a Python script included in the PSU, upgrade kit, or one-off patch. The plug-in or script must be specified after the `runExtension` parameter. ▪ `startemoc`: Start the EM Ops Center services in the following order: Database, Oracle VM Manager, Exalogic Control, Proxy Controller 1, Proxy Controller 2. ▪ `stopemoc`: Stop the EM Ops Center services in the following order: Proxy Controller 2, Proxy Controller 1, Exalogic Control, Oracle VM Manager, Database. ▪ `zfsSwapActive`: Change the active storage controller of the ZFS storage appliance.

Table 2 (Cont.) ExaPatch Command Options

Option	Description
	The `component` argument of the `--action` option specifies the group of components for which the specified action should be performed. It is relevant for the `patch`, `prePatchCheck`, `postPatchCheck`, and `getHistory` actions. Specify one of the following values. ■ `cn`: Compute nodes ■ `cn_ilom`: Compute node ILOMs ■ `ecservices`: Exalogic Control components ■ `ectemplates`: Exalogic Control templates ■ `ecu_config`: Exalogic Configuration Utility files. Of the previously mentioned actions, the `getHistory` action is not applicable to this component. ■ `nm2-gw`: NM2-GW switches ■ `nm2-36p`: NM2-36P switches ■ `vserver`: Guest vServers ■ `zfs_ilom`: ZFS ILOMs ■ `zfs_software`: ZFS software and workflows
`--actionshelp`	Displays a list of all ExaPatch actions and their descriptions.
`--concurrency=`*`number`*	The maximum number of tasks (`checkAuthentication`, `getVersion`, `patch`, `postPatchCheck`, or `prePatchCheck`) that ExaPatch should perform in parallel. By default, ExaPatch performs up to 20 tasks in parallel. You can use this option when you want to patch multiple components of a particular type (compute nodes or guest vServers) in parallel, *but* you want ExaPatch to perform the patching in batches of a defined size rather than for all components at once. To ensure that the host running ExaPatch is not overloaded, by default the number of concurrent tasks is set to 20.

`-d` *`file`*
or
`--patchDefinitionFile=`*`file`*
	The full path and name of the `exapatch_descriptor.py` file that is included in the PSU, upgrade kit, or one-off patch.
	This option is relevant for the `patch`, `prePatchCheck`, and `postPatchCheck` actions.
	If you do not specify this option, Exapatch looks for a file named `exapatch_descriptor.py` in the directory specified by the `-p` option, or in the current directory, if the `-p` option was not specified.

`-f`
or
`--force`
	Forces ExaPatch to apply (or re-apply) the patch while ignoring failures from pre-patching and current version checks. Use this option with caution.
	You can use this option to re-apply a patch when, for example, any files have been modified or deleted on a patched component.

Table 2 (Cont.) ExaPatch Command Options

Option	Description
`-h` *host* or `--host=`*host*	The host name or IP address of the component (say, a particular compute node) for which the specified action should be performed. The host name or IP address must match the entry in the rack configuration file you created except for `vserver` components because guest vServers are not listed in the rack configuration file. This option is relevant for the `cn`, `cn_ilom`, `vserver` components and `patch`, `listVMs`, `getHistory`, `baseVersion` actions. You can specify multiple `-h` options, one for each component. **Note:** If you do not specify the `-h` option when patching compute nodes or compute node ILOMs, ExaPatch patches **all** compute nodes or compute node ILOMs, except for the compute node on which ExaPatch is running.
`--help`	Displays a list of the supported options with a short description for each.
`-l` *file* or `--logFile=`*file*	The full path and name of the log file for ExaPatch. If you do not specify this option, ExaPatch creates a log file in the `/var/log` directory, and displays the name of the log file in the first line of the ExaPatch output on the console, as in the following example: `Logging to file /var/log/exapatch_20130613123509.log`
`-p` *dir* or `--path=`*dir*	The full path to the directory in the extracted PSU, upgrade kit, or one-off patch that contains the `exapatch_descriptor.py` file. This option is relevant when patching compute nodes, vServers, Exalogic Control services and Exalogic Control templates. It is also relevant for the `prePatchCheck` and `postPatchCheck` commands. If you do not specify this option, ExaPatch looks for the extracted PSU, upgrade kit, or one-off patch bundle in the current directory. **Note**: For the `runExtension` action, you must use the `-p` option to specify the full path to the required plug-in.
`-r` *file* or `--rackConfigFile=`*file*	The full path and name of the rack configuration file. This option is relevant for all actions and components, excluding the rack history and ECU related commands. If you do not specify this option, ExaPatch looks for a file named `rack_configuration.py` in the `/exalogic-lcdata/inventory/` directory.
`--rackHistoryFile=`*file*	The full path and name of the rack history file. This option is relevant for rack history related commands and the baseVersion command. If you do not specify this option, ExaPatch looks for a file named `rack_history.xml` in the `/exalogic-lcdata/inventory/` directory.

Table 2 (Cont.) ExaPatch Command Options

Option	Description

`-u` *`URI`*
 	
or
 	
`--uri=`*`URI`*
	The FTP or HTTP URI location of the patch.
	
	This option is relevant for the `patch`, `prePatchCheck`, and `postPatchCheck` actions, and is required when the PSU, upgrade kit, or one-off patch is installed in a location other than the `common/exalogic-lcdata` share on the ZFS storage appliance.
	
	When patching InfiniBand switches, you *must* specify an IP address in the URI. In all the other situations too, Oracle recommends that you specify the IP address, but you can specify the host name *if host-name resolution is configured*.
	
	If you do not specify this option, ExaPatch constructs the URI in the following format:
	
	`http://`*`IP_address`*`/shares/export/common/`*`path`*
	
	■ `IP` is the IP address of the active storage head of the ZFS storage appliance on the Exalogic machine.
	
	■ `path` is the directory in extracted PSU that contains the `exapatch_descriptor.py` file.
	
	Example:
	
	`http://`*`10.10.55.129`*`/shares/export/common/exalogic-lcd`
	`ata/16630395/Infrastructure/2.0.6.0.0`
	
	In this example, the path was specified as `exalogic-lcdata/16630395/Infrastructure/2.0.6.0.0`

`-v`
 	
or
 	
`--version`
	Displays the version number of ExaPatch.

1.7 Validating the Rack Configuration File

This section describes how to verify the rack components and validate the login passwords in the rack configuration file. It contains the following topics:

- Section 1.7.1, "Verifying the Rack Components in the Rack Configuration File"
- Section 1.7.2, "Validating the Password Directives in the Rack Configuration File"

1.7.1 Verifying the Rack Components in the Rack Configuration File

To verify that all the rack components are included in the rack configuration file, run ExaPatch with the `-a listComponents` option, as follows:

Note: For information about the locations from which you can run ExaPatch, see Section 1.5, "Location for Running ExaPatch."

```
# /exalogic-lctools/bin/exapatch -a listComponents [-r path_to_rack_
configuration.py]
```

For information about all the options of the ExaPatch command and the default value of options that are not specified, see Table 2, " ExaPatch Command Options".

Review the output of the command, and, if necessary, edit the rack configuration file to add or change components.

Sample Output (for an Exalogic Virtual Configuration)

```
Rack Components:
Compute-Node 10.10.54.171 root
Compute-Node 10.10.54.172 root
Compute-Node 10.10.54.173 root
Compute-Node 10.10.54.174 root
ILOM-ComputeNode 10.10.54.181 root
ILOM-ComputeNode 10.10.54.182 root
ILOM-ComputeNode 10.10.54.183 root
ILOM-ComputeNode 10.10.54.184 root
ILOM-ZFS 10.10.54.189 root
ILOM-ZFS 10.10.54.190 root
NM2-GW-IB-Switch 10.10.54.192 root
NM2-GW-IB-Switch 10.10.54.193 root
ZFS-Storage-Head 10.10.54.179 root
ZFS-Storage-Head 10.10.54.180 root
vServer-EC-EMOC-PC 192.168.20.8 root
vServer-EC-EMOC-PC 192.168.20.9 root
vServer-EC-OVMM 192.168.20.11 root
```

Note: The **`vserver-EC*`** entries are listed in the case of Exalogic *virtual* configurations only.

1.7.2 Validating the Password Directives in the Rack Configuration File

To validate whether ExaPatch can successfully log in to the components on the rack by using the password directives defined in the rack configuration file, run ExaPatch with the `-a checkAuthentication` option, as follows:

Note: For information about the locations from which you can run ExaPatch, see Section 1.5, "Location for Running ExaPatch."

```
# /exalogic-lctools/bin/exapatch -a checkAuthentication
```

For information about all the options of the ExaPatch command and the default value of options that are not specified, see Table 2, " ExaPatch Command Options".

If the authentication fails for any component, make sure that the password directive and the IP address for that component are specified correctly in the rack configuration file. Attempt to log in to the component by using the `ssh` *`user@host`* command, with the password corresponding to the directive in the rack configuration file. If the problem persists, contact Oracle Support and send the ExaPatch log files with your support request. *Do not* attempt to patch or upgrade the component until the `checkAuthentication` action succeeds.

Note: If you run ExaPatch on a compute node, ExaPatch cannot access the ILOM of that compute node.

Sample Output (for an Exalogic Virtual Configuration)

In this sample output, ExaPatch was run from the first compute node.

```
Login results:
Compute-Node         10.10.54.171   root       Succeeded
```

```
Compute-Node          10.10.54.172   root       Succeeded
Compute-Node          10.10.54.173   root       Succeeded
Compute-Node          10.10.54.174   root       Succeeded
ILOM-ComputeNode      10.10.54.181   root       Failed
ILOM-ComputeNode      10.10.54.182   root       Succeeded
ILOM-ComputeNode      10.10.54.183   root       Succeeded
ILOM-ComputeNode      10.10.54.184   root       Succeeded
ILOM-ZFS              10.10.54.189   root       Succeeded
ILOM-ZFS              10.10.54.190   root       Succeeded
NM2-GW-IB-Switch      10.10.54.192   root       Succeeded
NM2-GW-IB-Switch      10.10.54.193   root       Succeeded
ZFS-Storage-Head      10.10.54.179   root       Succeeded
ZFS-Storage-Head      10.10.54.180   root       Succeeded
vServer-EC-EMOC-PC    192.168.20.13  root       Succeeded
vServer-EC-EMOC-PC    192.168.20.14  root       Succeeded
vServer-EC-OVMM       192.168.20.11  root       Succeeded
```

Note: The **`vserver-EC*`** entries at the end of the output are listed in the case of Exalogic *virtual* configurations only.

1.8 Checking the Version Number of Each Component

To check the version number of each patchable component, run ExaPatch with the `-a getVersion` option, as follows:

```
# /exalogic-lctools/bin/exapatch -a getVersion
```

For information about all the options of the ExaPatch command and the default value of options that are not specified, see Table 2, " ExaPatch Command Options".

Save the output, for comparison with the version number after the patching is completed.

If the command fails for any component—that is, it does not display the version number, *do not* attempt to patch the component till the error is resolved. If the problem persists, contact Oracle Support and send the ExaPatch log files with your support request.

Note: If you run ExaPatch on a compute node, ExaPatch cannot access the ILOM of that compute node.

1.9 Generating a Rack History File

You can generate a rack history file using ExaPatch. The rack history file contains the history of all patches and versions for each component.

Generate a rack history file with the `-a refreshHistory` option, as follows:

```
# /exalogic-lctools/bin/exapatch -a refreshHistory [--rackHistory=path_to_rack_
history_file]
```

By default, ExaPatch creates a rack history file called `rack_history.xml` created in the `/exalogic-lcdata/inventory` directory when this command is run. After patching a component, ExaPatch automatically updates the rack history file. If you have generated the rack history file, you can manually update the version information in the rack history file using the `-a refreshHistory` command.

You can view the rack history for a specific component with the `-a getHistory` command, as follows

```
# /exalogic-lctools/bin/exapatch -a getHistory [component --rackHistory=path_to_
rack_history_file]
```

Example:

```
# /exalogic-lctools/bin/exapatch -a getHistory nm2-gw
Logging to file /var/log/exapatch_20131205020419.log

====== IBGatewaySwitch ======
HOSTNAME: ibswitch01.example.com
IP_ADDRESS: 10.10.54.192
------  init_refresh_event ------
VERSION: SUN DCS gw version: 2.0.8-1
Build time: Feb  6 2013 09:47:52
FPGA version: 0x34
SP board info:
Hardware Revision: 0x0006
Firmware Revision: 0x0000
BIOS version: SUN0R100
BIOS date: 06/22/2010
TIMESTAMP: 2013-12-04 02:23:10
------------------

====== IBGatewaySwitch ======
HOSTNAME: ibswitch02.example.com
IP_ADDRESS: 10.10.54.193
------  init_refresh_event ------
VERSION: SUN DCS gw version: 2.0.8-1
Build time: Feb  6 2013 09:47:52
FPGA version: 0x34
SP board info:
Hardware Revision: 0x0006
Firmware Revision: 0x0000
BIOS version: SUN0R100
BIOS date: 06/22/2010
TIMESTAMP: 2013-12-04 02:23:10
------------------
```

1.10 Running Pre-Patching Checks

Before patching a component, ExaPatch automatically runs pre-patching checks to verify if the component is ready for patching. You can manually run these checks with the `-a prePatchCheck` option, as follows:

```
# /exalogic-lctools/bin/exapatch -a prePatchCheck [component] -p path_to_
extracted_PSU_or_upgrade_kit [-r path_to_rack_configuration.py -d path_to_
exapatch_descriptor_file -h host]
```

If you do not specify a component, ExaPatch will run pre-patching checks on all the components. ExaPatch runs pre-patching checks depending on the component, as described in Table 3, " ExaPatch Pre-Patching Checks".

Table 3 ExaPatch Pre-Patching Checks

Component	Checks
Compute node	■ Checks whether the compute node is at a patchable version. ■ Checks whether the compute node is up. ■ Checks whether ExaPatch can successfully log in to the compute node. ■ Checks for free disk space.
Compute node and ZFS ILOMs	■ Checks whether the component is at a patchable version. ■ Checks whether the component is up. ■ Checks whether ExaPatch can successfully log in to the component.
Exalogic Configuration Utility files	■ Logs the version of the Exalogic Configuration Utility files. ■ Logs the validity of the Exalogic Configuration Utility files.
Exalogic Control Components	■ Exalogic Control service: ■ Checks whether the service is up. ■ Checks whether ExaPatch can successfully log in to the service. ■ Checks whether the Exalogic Control service is running. ■ Checks whether the EMOC OCDoctor utility exists in the `/var/opt/sun/xvm/OCDoctor` directory. ■ Proxy Controller service: ■ Checks whether the service is up. ■ Checks whether ExaPatch can successfully log in to the service. ■ Checks whether the Proxy Controller service is running. ■ Oracle VM Manager service: ■ Checks whether the service is up. ■ Checks whether ExaPatch can successfully log in to the service. ■ Checks whether the Oracle VM Manager Controller service is running and listening on port 54321. ■ Database service: ■ Checks whether the service is up. ■ Checks whether ExaPatch can successfully log in to the service. ■ Checks whether the database is healthy by running a basic SQL query.

Table 3 (Cont.) ExaPatch Pre-Patching Checks

Component	Checks
NM2-GW and NM2-36P switches	■ Checks whether the switch is at a patchable version. ■ Checks whether the switch is up. ■ Checks whether ExaPatch can successfully log in to the switch. ■ Checks for free disk space. ■ Checks whether a localhost entry exists in `/etc/hosts`. ■ Checks whether a GATEWAY entry exists in `/etc/sysconfig/network-scripts/ifcfg-eth0`. ■ Checks whether the contents of `/conf/configvalid` is 1. ■ Checks that the contents of `/config/etc/sysconfig/network-scripts/ifcfg-eth0` and `/etc/sysconfig/network-scripts/ifcfg-eth0` are not identical. ■ Runs the `fwverify` test.
Storage Appliance	■ Checks whether the storage head is at a patchable version. ■ Checks whether ExaPatch can access the storage head. ■ Checks for the head in Ready state: ■ Checks whether the state of the head is `AKCS_STRIPPED` or `AKCS_CLUSTERED`. ■ Only for non-clustered head: Checks whether the description of the head is set to `Ready`. ■ Checks whether the status of the head is `exported.` ■ Checks for the active head: ■ Checks whether the state of the head is `AKCS_OWNER`. ■ Checks whether the status of the head is `online`. ■ Checks whether there are no resilvering operations in progress ■ Checks whether there are no active maintenance issues. ■ Checks whether there are any firmware updates in progress.
Guest and Exalogic Control vServers	■ Checks whether the vServer is up. ■ Checks whether ExaPatch can successfully log in to the vServer. ■ Checks for free disk space.

2 Patching and Upgrading the Exalogic Infrastructure Using ExaPatch

This section provides the generic procedure for upgrading and patching various components of the Exalogic infrastructure by using ExaPatch. *The information in this section is for reference only*. For the specific steps that you should perform to patch or upgrade each component of the Exalogic infrastructure, see the readmes accompanying the PSU, upgrade kit, or one-off patch.

1. Ensure that any ongoing configuration and backup operations are completed. Otherwise, the patching or upgrade operation may fail.
2. Ensure that the prerequisites, described in the documentation accompanying the PSU, upgrade kit, or one-off patch, are fulfilled.
3. Run the following command in the `exapatch` directory, depending on the component/s that you want to patch:

   Note:

   - For information about the locations from which you can run ExaPatch, see Section 1.5, "Location for Running ExaPatch."
   - For information about all the options of the ExaPatch command and the default value of options that are not specified, see Table 2, " ExaPatch Command Options".

   - To patch the NM2-GW switches:

     ```
     # /exalogic-lctools/bin/exapatch -a patch nm2-gw
     ```

   - To patch the NM2-36P switch:

     ```
     # /exalogic-lctools/bin/exapatch -a patch nm2-36p
     ```

   - To patch the ZFS ILOM:

     ```
     # /exalogic-lctools/bin/exapatch -a patch zfs_ilom
     ```

   - To patch the ZFS software and workflows:

     ```
     # /exalogic-lctools/bin/exapatch -a patch zfs_software
     ```

     If any new ZFS workflows were installed, ExaPatch displays a warning message indicating that the new workflows may have to be executed manually. Follow the instructions in the readmes accompanying the PSU, upgrade kit, or one-off patch.

   - To patch the ILOM of a specific compute node:

     ```
     # /exalogic-lctools/bin/exapatch -a patch cn_ilom -h ILOM_IP_address
     ```

     Note: ExaPatch cannot patch the ILOM of the compute node on which it is running. To patch the ILOM of that compute node, run ExaPatch on a different compute node.

To patch the ILOMs of multiple compute nodes, specify multiple `-h` options. The compute-node ILOM IP addresses that you specify with the `-h` option must match the IP addresses in the rack configuration file.

Note: While the ILOM of a compute node is being updated, the node is shut down for the duration of the update process and will be started after the update. So do not attempt to update the ILOM of the compute node on which you are running ExaPatch.

- To patch a specific compute node:

  ```
  # /exalogic-lctools/bin/exapatch -a patch cn -h IP_address_of_compute_node
  ```

  Note: ExaPatch cannot patch the compute node on which it is running. To patch that compute node, run ExaPatch on a different compute node.

  To patch multiple compute nodes, specify multiple `-h` options. The compute-node IP addresses that you specify with the `-h` option must match the IP addresses in the rack configuration file.

  Note: While a compute node is being updated, the node is shut down for the duration of the update process and will be started after the update. So do not run ExaPatch on the compute node that you want to update.

- To patch the OVMM, PC1, and PC2 vServer templates:

  ```
  # /exalogic-lctools/bin/exapatch -a patch ectemplates
  ```

- To patch the Exalogic Control components:

  ```
  # /exalogic-lctools/bin/exapatch -a patch ecservices
  ```

  You may be prompted for the OVS schema password. If the OVS schema password was not changed after the initial installation of the machine, you can find the password in the Exalogic Configuration Utility configuration file `db.json`, as the value of the key `oracle_db_ovm_user_password`.

- To patch a specific guest vServer:

  ```
  # /exalogic-lctools/bin/exapatch -a patch vserver -h vServerIP
  ```

  To patch multiple guest vServers, specify multiple `-h` options. You will be prompted, once, for the root password of all the guest vServers. After the patching is completed, the guest vServers will reboot.

  Note that the guest vServers may be rebooted during the patching or upgrade process.

4. Perform any additional steps required, as described in the readmes accompanying the PSU, upgrade kit, or one-off patch.

3 Synchronizing the Exalogic Configuration Utility Files

For an Exalogic machine that was upgraded from EECS 2.0.4 to EECS 2.0.6, you must patch the Exalogic Configuration Utility (ECU) configuration files with the current configuration of the machine, by running the `patch` action on the `ecu_config` component. After patching the ECU files, you must download and install the ECU for EECS 2.0.6.

3.1 Prerequisites

- Ensure that the ECU files exist in the `/var/tmp/exalogic/ecu` directory on the first compute node in your Exalogic machine or the compute node on which Exalogic Configuration Utility was run.
- Ensure that you generated a patch history file, as described in Section 1.9, "Generating a Rack History File."

3.2 Patch the Exalogic Configuration Utility Files

To patch the ECU configuration files with the current configuration of the machine, do the following:

1. Verify that the ECU files are for EECS 2.0.4 as follows:
   a. Run the pre-patching checks on the ECU files:

      ```
      # /exalogic-lctools/bin/exapatch -a prePatchCheck ecu_config -h Master_
      Compute_Node_IP_Address
      ```

      `Master_Compute_Node_IP_Address` is the IP address or host name of either the first compute node in your Exalogic machine or the compute node on which Exalogic Configuration Utility was run.

   b. Open the `/var/log/exapatch_`*`timestamp`*`.log` file.
   c. Verify that the following line is present in the log file:

      ```
      Validating ECU Configuration EL204_IMPORTED
      ```

2. Patch the ECU files to the EECS 2.0.6 configuration by running the `patch` action:

   ```
   # /exalogic-lctools/bin/exapatch -a patch ecu_config -h Master_Compute_Node_IP_
   Address
   ```

   ExaPatch stores a backup of the EECS 2.0.4 ECU files in `/var/tmp/exalogic/ecu/ecuconv_backup.`*`timestamp`*`.tgz`.

3. Verify that the ECU files were updated for EECS 2.0.6 successfully as follows:
   a. Run the post-patching checks on the ECU files:

      ```
      # /exalogic-lctools/bin/exapatch -a postPatchCheck ecu_config -h Master_
      Compute_Node_IP_Address
      ```

      `Master_Compute_Node_IP_Address` is the IP address or host name of either the first compute node in your Exalogic machine or the compute node on which Exalogic Configuration Utility was run.

   b. Open the `/var/log/exapatch_`*`timestamp`*`.log` file.
   c. Verify that the following line is present in the log file:

```
Validating ECU Configuration EL206_IMPORTED
```

3.3 Download the ECU

To download the ECU, do the following:

1. Go to `https://edelivery.oracle.com`.

   Sign in by using your Oracle account.

2. Read and accept the **Oracle Software Delivery Cloud Trial License Agreement** and the **Export Restrictions**.

   Click **Continue**.

3. In the **Select a Product Pack** field, select **Oracle Fusion Middleware**.

   In the **Platform** field, select **Linux x86-64**.

   Click **Go**.

4. In the results displayed, select **Oracle Exalogic Elastic Cloud Software 11g Media Pack**, and click **Continue**.

5. Look for **Oracle Exalogic 2.0.6.0.0 Configuration Utility for Exalogic OracleVM x86-64 (64-bit)**, and click the **Download** button.

6. Save the ECU zip file to a location of your choice.

3.4 Install the ECU

To install the ECU, do the following:

1. Log in to the first compute node or the compute node where ECU was run previously.

2. Copy the ECU zip file you downloaded to the `/opt/exalogic` directory.

3. Extract the ECU zip file.

   The files will be extracted in the `/opt/exalogic/ecu` directory.

4. Define the `ECU_HOME` environment variable, by running the following command:

   ```
   # export ECU_HOME=/opt/exalogic/ecu
   ```

5. Navigate to the `ECU_HOME` directory.

   ```
   # cd $ECU_HOME
   ```

6. Run the install script to set up ECU:

   ```
   # ./install.sh
   ```

7. Verify whether the installation was completed correctly, by running the `ecu.sh` script, without specifying any arguments.

   ```
   # ./ecu.sh
   ```

   When the `ecu.sh` script is executed without any parameters, it displays help for the script.

3.5 Validate the ECU Files

To validate the ECU files, do the following:

1. Navigate to the `ECU_HOME` directory.

   ```
   # cd $ECU_HOME
   ```

2. Verify that the ECU can load the patched ECU files:

   ```
   # ./ecu.sh show_config_data
   ```

3. Validate the compute node data in the ECU files:

   ```
   # ./ecu.sh run_step_archaic 21 target=compute_node_number
   ```

   Replace `compute_node_number` with a number from 1 to N, where N is the number of compute nodes in your Exalogic rack. You must run this command for each compute node in your Exalogic rack. For an eighth rack, you must run the following commands:

   ```
   # ./ecu.sh run_step_archaic 21 target=1
   # ./ecu.sh run_step_archaic 21 target=2
   # ./ecu.sh run_step_archaic 21 target=3
   # ./ecu.sh run_step_archaic 21 target=4
   ```

4. Validate the InfiniBand switch data in the ECU files:

   ```
   # ./ecu.sh run_step_archaic 28 target=switch_number
   ```

   Replace `switch_number` with a number from 1 to N, where N is the number of InfiniBand switches in your Exalogic rack. You must run this command for each InfiniBand switch in your Exalogic rack. For an eighth rack, you must run the following commands:

   ```
   # ./ecu.sh run_step_archaic 28 target=1
   # ./ecu.sh run_step_archaic 28 target=2
   ```

5. Validate the storage appliance data in the ECU files:

   ```
   # ./ecu.sh run_step_archaic 32
   ```

6. Validate the ELControl vServer data in the ECU files:

   ```
   # ./ecu.sh run_step_archaic 40
   ```

7. Validate the Exalogic Control PC1 vServer data in the ECU files:

   ```
   # ./ecu.sh run_step_archaic 42
   ```

8. Validate the Exalogic Control PC2 vServer data in the ECU files:

   ```
   # ./ecu.sh run_step_archaic 44
   ```

3.6 Archive the ECU Files

To archive the ECU configuration files and logs, do the following:

1. Navigate to the `ECU_HOME` directory.

   ```
   # cd $ECU_HOME
   ```

2. Mount the `ExalogicControl` share in the `/mnt` directory:

```
# ./ecu.sh run_step_archaic 36
```

3. Archive the ECU Configuration files and logs to the `ExalogicControl` share:

   ```
   # ./ecu.sh run_step_archaic 48
   ```

4 Miscellaneous (Non-Patching) ExaPatch Actions

Besides using ExaPatch for applying patches to various Exalogic components, you can use it to perform the following additional tasks that are necessary at certain points during the patching or upgrade process.

> **Note:** The readmes accompanying the upgrade kit or PSU contains instructions about when to use the ExaPatch actions described in this section.

- Gracefully shut down the Exalogic Control vServers:

  ```
  # /exalogic-lctools/bin/exapatch -a ecvserversshutdown [-r rack_
  configuration.py]
  ```

- Start the Exalogic Control vServers:

  ```
  # /exalogic-lctools/bin/exapatch -a ecvserversstartup [-r rack_
  configuration.py]
  ```

- Run an ExaPatch plug-in (a Python script) included in the PSU, upgrade kit, or one-off patch:

  ```
  # /exalogic-lctools/bin/exapatch -a runExtension -p path_to_plugin [argument1]
   [argument2] [...]
  ```

  Note that information about specific ExaPatch plug-ins is provided in the documentation accompanying the PSU, upgrade kit, or one-off patch, as relevant.

- Stop the EM Ops Center services in the order Proxy Controller 2, Proxy Controller 1, Exalogic Control, Oracle VM Manager, Database:

  ```
  # /exalogic-lctools/bin/exapatch -a stopemoc [-r rack_configuration.py]
  ```

- Start the EM Ops Center services in the order Database, Oracle VM Manager, Exalogic Control, Proxy Controller 1, Proxy Controller 2:

  ```
  # /exalogic-lctools/bin/exapatch -a startemoc [-r rack_configuration.py]
  ```

- Restart the Oracle VM Server agent on all Oracle VM Servers:

  ```
  # /exalogic-lctools/bin/exapatch -a restartovsagent [-r rack_configuration.py]
  ```

5 Running Post-Patching Checks

After patching a component, ExaPatch automatically runs post-patching checks to verify if the component is ready for patching. You can manually run these checks with the `-a postPatchCheck` option, as follows:

```
# /exalogic-lctools/bin/exapatch -a postPatchCheck [component] -p path_to_
extracted_PSU_or_upgrade_kit [-r path_to_rack_configuration_file -d path_to_
exapatch_descriptor_file -h host]
```

If you do not specify a component, ExaPatch will run post-patching checks on all the components. The checks ExaPatch runs depends on the component, as described in Table 4, " ExaPatch Post-Patching Checks".

Table 4 ExaPatch Post-Patching Checks

Component	**Checks**
Compute node	▪ Checks whether the component has been patched. ▪ Checks whether the component is up. ▪ Checks whether ExaPatch can successfully log in to the component.
Compute node and ZFS ILOMs	▪ Checks whether the component has been patched. ▪ Checks whether the component is up. ▪ Checks whether ExaPatch can successfully log in to the component.
Exalogic Configuration Utility files	▪ Logs the version of the Exalogic Configuration Utility files. ▪ Logs the validity of the Exalogic Configuration Utility files.

Table 4 (Cont.) ExaPatch Post-Patching Checks

Component	Checks
Exalogic Control Components	■ Exalogic Control service: ■ Checks whether the service has been patched. ■ Checks whether the service is 6. ■ Checks whether ExaPatch can successfully log in to the service. ■ Checks whether the Exalogic Control service is running. ■ Checks whether the EMOC OCDoctor utility exists in the `/var/opt/sun/xvm/OCDoctor` directory. ■ Proxy Controller service: ■ Checks whether the service has been patched. ■ Checks whether the service is up. ■ Checks whether ExaPatch can successfully log in to the service. ■ Checks whether the Proxy Controller service is running. ■ Oracle VM Manager service: ■ Checks whether the service has been patched. ■ Checks whether the service is up. ■ Checks whether ExaPatch can successfully log in to the service. ■ Checks whether the Oracle VM Manager Controller service is running and listening on port 54322. ■ Database service: ■ Checks whether the service has been patched. ■ Checks whether the service is up. ■ Checks whether ExaPatch can successfully log in to the service. ■ Checks whether the database is healthy by running a basic SQL query.
NM2-GW and NM2-36P switches	■ Checks whether the component has been patched. ■ Runs the `fwverify` test. ■ Checks whether the component is up. ■ Checks whether ExaPatch can successfully log in to the component.
Storage Appliance	Checks whether both heads have been patched.

Table 4 (Cont.) ExaPatch Post-Patching Checks

Component	Checks
Guest and Exalogic Control vServers	■ Checks whether the vServer has been patched ■ Checks whether the vServer is up. ■ Checks whether ExaPatch can successfully log in to the vServer.

6 ExaPatch Log Files

By default, details of the patching process are logged in the `/var/log/exapatch_`*`timestamp`*`.log` file. The time stamp in the name of the log file is in the `YYYYMMDDHHMMSS` format.

You can change the log file location by using the `-l` *`logfile_path_and_name`* option while running ExaPatch.

7 Diagnosing and Troubleshooting Problems

This section provides solutions for errors that you may encounter while using ExaPatch.

- **Problem**: ExaPatch takes a long time to complete.

  Solution: ExaPatch may appear to take a long time because certain operations may be continuing in the background even after the component is patched. For example, after patching a compute node ILOM, ExaPatch waits for the compute node to restart.

  To track the current ExaPatch activity, run the following command on the host that is running ExaPatch, from a separate login shell:

  ```
  # tail -f exapatch_log_file
  ```

  When multiple components are patched in parallel, the ExaPatch logs for each component are stored temporarily (for the duration of the patching process) in separate log files with the suffix `Thread_`*`n`*: for example, `exapatch.log_Thread_1`, `exapatch.log_Thread_2`, and so on. After all the components are patched, the logs are moved to a single log file and the `Thread_`*`n`* files are deleted. So to track the ExaPatch activity while multiple components are being patched in parallel, run the following command:

  ```
  # tail -f exapatch_log_file_Thread_n
  ```

- **Problem**: Patching the ZFS software failed.

  Solution: Send the ExaPatch log files to Oracle Support.

- **Problem**: ExaPatch reported a failure when multiple compute nodes (base image or ILOM) were patched.

  Solution: Send the ExaPatch log files to Oracle Support.

- **Problem**: Upgrade of a single compute node failed.

  Solution: This issue may be because of insufficient disk space. The error log would point to a failed yum operation (size of the yum log file would be zero). Clear the disk space and run Exapatch again.

For all other problems, contact Oracle Support and send the ExaPatch log files with your support request.

8 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.

ExaPatch User's Guide, Release 1.2
E50725-06