Oracle® Linux

Unbreakable Linux Network User's Guide for Oracle Linux 6 and Oracle Linux 7

E39381-25
August 2019

Oracle Legal Notices

Copyright © 2013, 2019, Oracle and/or its affiliates. All rights reserved.

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, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

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 and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

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.

Abstract

This manual provides information about using the Unbreakable Linux Network (ULN) for Oracle Linux 6 and Oracle Linux 7. For information specific to Oracle Linux 8, please see *Oracle® Linux 8: Managing Software on Oracle Linux*

Document generated on: 2019-08-21 (revision: 8187)

Table of Contents

Preface

The *Oracle® Linux: Unbreakable Linux Network User's Guide for Oracle Linux 6 and Oracle Linux 7* provides information about how to register your systems with the Unbreakable Linux Network (ULN), and includes procedures for creating a local yum server as well as a guide to using the `yum` command itself. This manual provides information that relates to usage on Oracle Linux 6 and Oracle Linux 7. For information specific to Oracle Linux 8, please see *Oracle® Linux 8: Managing Software on Oracle Linux*

Audience

This document is intended for administrators who want to use the ULN. It is assumed that readers are familiar with web technologies and have a general understanding of Linux system administration.

Document Organization

The document is organized as follows:

- Chapter 1, *About the Unbreakable Linux Network* provides an overview of ULN, including a description of the packages that are required for a system to connect to ULN. Information about how channels are named and how software errata are released to various channels is also covered.
- Chapter 2, *CSI Administration* describes how to manage and administer CSIs against your user accounts and systems from within ULN.
- Chapter 3, *ULN Registration* describes different ULN registration options.
- Section 3.3, “Migrating from RHN to ULN” describes how to migrate from RHN to ULN and includes tasks for obtaining a ULN account and registering a system with ULN.
- Section 4.1, “ULN Channel Subscription Management” describes the different interfaces and options for managing ULN channels and includes related channel management configuration tasks.
- Chapter 4, *ULN System Management* describes various tasks for managing systems that are registered with ULN.
- Chapter 5, *Creating and Using a Local ULN Mirror* describes how to create and use a yum server to act as a local mirror of the ULN channels.

Related Documents

The documentation for this product is available at:

https://docs.oracle.com/en/operating-systems/linux.html.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.

Convention	Meaning
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

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.

Chapter 1 About the Unbreakable Linux Network

Table of Contents

This chapter describes what the Unbreakable Linux Network (ULN) is and how it works. It includes a description of the packages required for a system to connect to ULN and also describes how channels are named and how software errata are released to the different channels.

If you have a subscription to Oracle Unbreakable Linux support, you can use the comprehensive resources of the Unbreakable Linux Network (ULN). ULN offers software patches, updates, and fixes for Oracle Linux and Oracle VM, as well as information on `yum`, Ksplice, and support policies. You can also download useful packages that are not included in the original distribution. The ULN Alert Notification Tool periodically checks with ULN and alerts you when updates are available. You can access ULN at https://linux.oracle.com/, where you will also find instructions for registering with ULN, for creating local `yum` repositories, and for switching from the Red Hat Network (RHN) to ULN.

If you want to use `yum` with ULN to manage your systems, you must register the systems with ULN and subscribe each system to one or more ULN channels. When you register a system with ULN, the channel that contains the latest version is chosen automatically, according to the architecture and operating system revision of the system. See Chapter 3, *ULN Registration* for more information.

When you run the `yum` command, it connects to the ULN server repository and downloads the latest software packages in RPM format onto your system. `yum` then presents you with a list of the available packages so that you can choose which packages you want to install.

1.1 ULN Access for Oracle Cloud Infrastructure

Compute nodes running Oracle Linux on Oracle Cloud Infrastructure and that are connected to a service gateway automatically have access to ULN content via the regional yum servers available on the Oracle Services Network. These yum servers differ from the publicly available Oracle Linux yum server in that they also mirror content available on restricted ULN channels.

Access to ULN content is provided by virtue of the support contract that you have for your Oracle Cloud Infrastructure account. You are able to access content on ULN without any requirement to register or use alternate tools to manage channel access, simplifying any software management that you need to perform on a compute node.

To enable access to restricted content via the regional yum servers, ensure that you have installed the appropriate `release-elx` packages and enabled the repositories that you require access to. For example, on Oracle Linux 7, you can run the following commands to access the `ol7_oci_included` repository, where tools like Oracle InstantClient, the Oracle Java Development Kit and Oracle Java Runtime Environment are located:

```
# yum install oci-included-release-el7
# yum-config-manager --enablerepo ol7_oci_included
```

Other ULN channels are also available directly via the Oracle Cloud Infrastructure regional yum servers. For instance, to access the Ksplice channels on an Oracle Linux 7 compute instance, you can do:

```
# yum install ksplice-release-el7
# yum-config-manager --enablerepo ol7_ksplice ol7_x86_64_userspace_ksplice
```

1.2 About the rhn-setup Package

The tools to register with ULN from an Oracle Linux or Oracle VM system are provided in the rhn-setup package. This package is available on the ol6_latest and ol7_latest yum repositories that are available on the Oracle Linux yum server. This package is usually also installed by default on a new installation of an Oracle Linux system.

You can also manually download the RPMs from ULN, directly, by browsing the appropriate channel and architecture for your system.

If you intend to migrate from the RedHat Network (RHN) to ULN, you should replace the matching package with the version provided by Oracle, to obtain access to the additional tools that make this possible. See Section 3.3, “Migrating from RHN to ULN” for more information.

1.3 About ULN Channels

ULN provides more than 100 unique channels, which support the i386, x86_64, IA64, and the 64-bit Arm architectures, for releases of Oracle Linux 4 update 6 and later and Oracle VM 2.1 and later.

You can choose that your system remain at a specific OS revision, or you can allow the system to be updated with packages from later revisions.

You should subscribe to the channel that corresponds to the architecture of your system and the update level at which you want to maintain it. Patches and errata are available for specific revisions of Oracle Linux, but you do not need to upgrade from a given revision level to install these fixes. ULN channels also exist for MySQL, Oracle VM, Oracle Ksplice, OCFS2, RDS, and productivity applications.

The following table describes the main channels that are available.

Channel	Description
_latest	Provides all the latest versions of the packages in a distribution, including any errata that are also provided in the patch channel. If no vulnerabilities have been found in a package, the package version might be the same as that included in the original distribution. For other packages, the version is the same as that provided in the patch channel for the highest update level. For example, the ol6_*arch*_latest channel for Oracle Linux 6 Update 3 contains a combination of the ol6_u3_*arch*_base and ol6_u3_*arch*_patch channels.
_archive	Provides older versions of packages that are added to a parent channel. The _archive suffix is usually added to the channel that it hosts archive packages for. For example, the _latest channels have equivalent _latest_archive channels to host older versions of packages that have been updated in the _latest channels. Packages are moved to an archive channel when newer versions of the same packages are added to the parent channel. This helps to keep the metadata for the parent channel manageable and also keeps the overall size of the channel down to a minimum. If you require an earlier version of a package, you can subscribe to the equivalent _archive channel to obtain it. When performing an installation or downgrade, you must specify the version of the package that you wish to install. Installing packages from an _archive channel may result in your system running software that has since been patched for security related issues. This could open your system up to vulnerabilities that could be exploited for malicious purposes.
_base	Provides the packages for each major version and minor update of Oracle Linux and Oracle VM. This channel corresponds to the released ISO media image. For example, there is a base channel for each of the update level of an Oracle Linux release . Oracle does not publish security errata and bug fixes on these channels.
_patch	Provides only those packages that have changed since the initial release of a major or minor version of Oracle Linux or Oracle VM. The patch channel always provides the most recent version of a package, including all fixes that have been provided since the initial version was released.
_addons	Provides packages that are not included in the base distribution, such as the package that you can use to create a local yum repository on Oracle Linux.
_oracle	Provides freely downloadable RPMs from Oracle that you can install on Oracle Linux, such as ASMLib and Oracle Instant Client.
_optional	Provides optional packages for Oracle Linux 7 that have been sourced from upstream. This channel includes most development packages (*-devel).
_developer	Provides packages that can be used to set up test and development environments for Oracle Linux. Packages released in this channel include tools that can be useful for developers and test engineers when setting up an environment.
_preview	Provides packages that are still under development at Oracle and are made available as technical previews for developer and test engineer usage. These packages are not for use in a production environment and are not supported by Oracle.
_developer_EPEL	Provides a mirror of the packages that are available on the EPEL (Extra Packages for Enterprise Linux) repository. The packages hosted in this channel are available for Oracle Linux users but are unsupported by Oracle.

Other channels may also be available, such as _beta channels for the beta versions of packages.

As each new, major version or minor update of Oracle Linux becomes available, Oracle creates new base and patch channels for each supported architecture to distribute new packages. The existing base and patch channels for the previous versions or updates remain available and do not include the new packages. The `_latest` channel distributes the latest possible version of any package, and tracks the top of the development tree independently of the update level.

Caution

You can choose to maintain your system at a specific update level of Oracle Linux and selectively apply errata to that level by subscribing the system to the `_base` and `_patch` channels and unsubscribing it from the `_latest` channel. However, patches are not added to the `_patch` channel for previous updates of Oracle Linux after a new update has been released. For example, after the release of Oracle Linux 7 Update 1, no further errata will be released on the `ol7_x86_64_u0_patch` channel.

Oracle recommends that you keep you system subscribed to the `_latest` channel. If you unsubscribe from the `_latest` channel, your system will become vulnerable to security-related issues when a new update is released.

For more information about the channels available for any system that you have registered with ULN, see Section 4.1, “ULN Channel Subscription Management”.

1.4 About Software Errata

Oracle releases important changes to the Oracle Linux and Oracle VM software as individual package updates, known as errata. These package updates are made available for download on ULN before they are gathered into a release or distributed through the `_patch` channel.

Errata packages can contain the following:

- Security advisories, which have names prefixed by `ELSA-*` (for Oracle Linux) and `OVMSA-*` (for Oracle VM).
- Bug fix advisories, which have names prefixed by `ELBA-*` and `OVMBA-*`.
- Feature enhancement advisories, which have names prefixed by `ELEA-*` and `OVMEA-*`.

To be notified when new errata packages are released, you can subscribe to the Oracle Linux and Oracle VM errata mailing lists at https://oss.oracle.com/mailman/listinfo/el-errata and https://oss.oracle.com/mailman/listinfo/oraclevm-errata.

If you are logged into ULN, you can also subscribe to these mailing lists by following the **Subscribe to Enterprise Linux Errata mailing list** and **Subscribe to Oracle VM Errata mailing list** links that are provided on the Errata tab.

Oracle publishes a complete list of errata made available on ULN at https://linux.oracle.com/errata. You can also see a published listing of Common Vulnerabilities and Exposures (CVEs) and explore their details and status at https://linux.oracle.com/cve.

1.5 For More Information About ULN

You can find out more information about ULN at https://linux.oracle.com/.

Chapter 2 CSI Administration

Table of Contents

Access to ULN requires at least one valid Customer Support Identifier (CSI). Your CSI is an identifier that is issued to you when you purchase Oracle Support for an Oracle product. You must provide a valid CSI that covers the support entitlement for each system that you register with ULN.

This chapter describes how you are able to manage and administer CSIs against your user accounts and systems from within ULN.

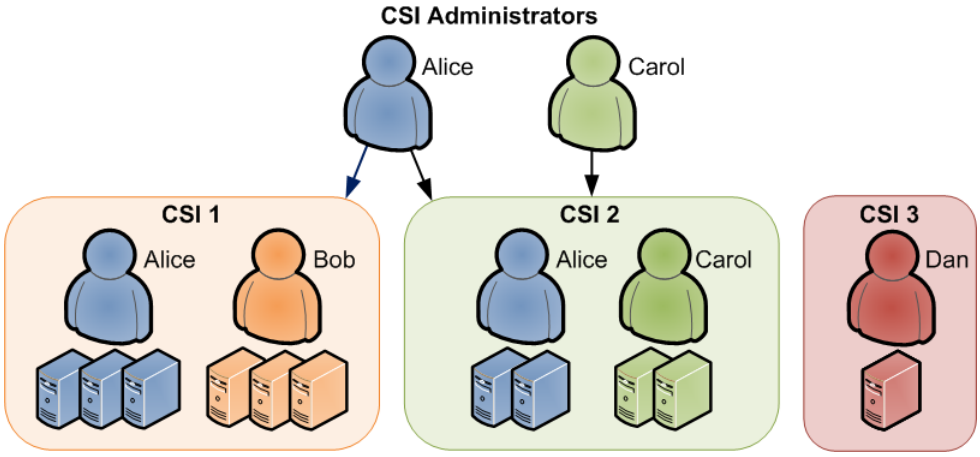
The CSI administration feature of ULN provides a unified view of all of your organization's CSIs and the systems that are registered with those CSIs. To be able to manage the registered systems, you must become an administrator for one or more of your organization's CSIs. To be able to view and change the details of any system that is not registered to your ULN user name, you must become an administrator for the CSI under which that system is registered.

If you are registered as a CSI administrator, you can access the CSI Administration tab while logged in to ULN and perform the following tasks:

- Assign yourself as administrator of a CSI, or assign someone else as administrator of a CSI. See Section 2.1, “Becoming a CSI Administrator”.
- List active CSIs, list the servers that are currently registered with an active CSI, and transfer those servers to another user or to another CSI. See Section 2.2, “Listing Active CSIs and Transferring Their Registered Servers”.
- List expired CSIs, list the servers that are currently registered with an expired CSI, and transfer those servers to another user or to another CSI. See Section 2.3, “Listing Expired CSIs and Transferring Their Registered Servers”.
- Remove yourself or someone else as administrator of a CSI. See Section 2.4, “Removing a CSI Administrator”.

Figure 2.1 shows a representative example of an organization with three CSIs, only two of which have CSI administrators.

Figure 2.1 Example of an Organization with three CSIs



CSI 1 has two registered users, Alice and Bob, who each have three systems registered to them.

CSI 2 also has two registered users, Alice and Carol, who each have two systems registered to them.

CSI 3 has one registered user, Dan, who has a single system registered to him.

Alice is registered as an administrator for both CSI 1 and CSI 2. She can view the details of both CSIs, including all systems and users that are registered with those CSIs. She can move systems between CSI 1 and CSI 2, and reassign systems between users in both CSI 1 and CSI 2. She can also assign additional administrators to CSI 1 and CSI 2, or remove administrators from CSI 1 and CSI 2. She cannot see any details for CSI 3.

Carol is registered as an administrator only for CSI 2. She can view the details of that CSI and of all systems and users that are registered with it, including Alice's systems. She can reassign systems between users in CSI 2, but she cannot move systems to the other CSIs. She can assign additional administrators to CSI 2, or remove administrators from CSI 2. She cannot see any details for CSI 1 or CSI 3.

Bob can view only the details of the systems that are registered to him in CSI 1. He cannot see any details for Alice's systems in CSI 1.

Dan is not registered as an administrator for CSI 3. He can view only the details of the system that is registered to him in CSI 3.

Neither Bob nor Dan can perform CSI administration tasks. For example, they cannot move systems between CSIs nor can they reassign systems to other users. However, as CSI 3 does not currently have an administrator, Dan can choose to become its administrator. As CSI 1 already has Alice as its administrator, Bob cannot become an administrator unless Alice grants him that privilege.

For Alice to become an administrator of CSI 3, Dan should register as the administrator of CSI 3 so that he can add Alice as an administrator.

2.1 Becoming a CSI Administrator

You can become an administrator of a CSI in one of the following ways:

- When you register with ULN, if no administrator is currently assigned to manage the CSI, you are prompted to click **Confirm** to become the CSI administrator. If you click **Cancel**, you cannot access the CSI administration feature.

- When logged into ULN, if you access the System tab and no administrator is currently assigned to manage one of the CSIs for which you are registered, you are prompted to choose whether to become the CSI administrator.

  To become a CSI administrator:

  1. Click the red link labeled **enter the CSI you would like to be the administrator for in this page**.
  2. On the Add CSI page, verify the CSI and click **Confirm**.

  Note

  On the Systems page, the CSIs of all systems that have no assigned administrator are also shown in red.

- If you are already an administrator of a CSI, you can add yourself as administrator of another CSI provided that you have registered either a server or your ULN user name with the other CSI.

  To assign yourself as administrator of an additional CSI:

  1. Log in to ULN and select the CSI Administration tab.
  2. On the Managed CSIs page, click **Add CSI**.
  3. On the Assign Administrator page, enter the CSI, and click **Add**.
  4. If there are existing administrators, the page lists these administrators and prompts you to click **Confirm** to confirm your request. Each administrator is sent an email to inform them that you have added yourself as an administrator of the CSI.

- An administrator for a CSI can add you as an administrator for the same CSI.

  To assign another administrator to a CSI:

  1. Log in to ULN as administrator of the CSI, and select the CSI Administration tab.
  2. On the Managed CSIs page, click **List Administrators**.
  3. On the CSI Administrators page, click **Assign Administrator**.
  4. On the Assign Administrator page in the Select New Administrator list, click the **+** icon that is next to the user name of the user that you want to add as an administrator. Their user name is added to the **Administrator** box.
  5. If you administer more than one CSI, select the CSI that the user will administer from the **CSI** drop down list.
  6. Click **Assign Administrator**.

  Note

  If you want to become the administrator of a CSI, but the person to whom it is registered is no longer with your organization, contact an Oracle support representative to request that you be made the administrator for the CSI.

2.2 Listing Active CSIs and Transferring Their Registered Servers

To list details of the active CSIs for which you are the administrator:

1. Log in to ULN as administrator of the CSI, and select the CSI Administration tab.
2. On the Managed CSIs page in the Select Managed CSI Services pane, select the **Active** link. The Managed Active CSI Services pane displays the service details for each active CSI that you administer.
3. Click the **View # Server(s)** link to display the details of the servers that are registered to an active CSI.
4. On the Registered Servers page, you can transfer one or more systems to another user or to another CSI that you administer.

Note

If you transfer a system to another user, at least one of the following conditions must be true:

- His or her user name must be registered to this CSI.
- One or more of the servers, for which they are the owner, must be registered to this CSI.
- He or she must be an administrator of at least one CSI for which you are also an administrator.

To transfer systems to another user:

a. Select the **Transfer System** check boxes for the systems that you want to transfer.

b. Click **Transfer Selected Systems to Another Owner**.

c. On the Transfer Registered System(s) - Owner page in the Transfer To column, click the red arrow icon that is next to the user name of the user to whom you want to transfer ownership.

d. On the Confirm Transfer Profile - Owner page, click **Apply Changes** to confirm the transfer to the new owner.

To transfer systems to another CSI:

a. Select the **Transfer System** check boxes for the systems that you want to transfer.

b. Click **Transfer Selected Systems to Another CSI**.

c. On the Transfer Registered System(s) - CSI page in the Transfer To column, click the red arrow icon that is next to the CSI to which you want to transfer the systems.

d. On the Confirm Transfer Profile - CSI page, click **Apply Changes** to confirm the transfer to the new CSI.

2.3 Listing Expired CSIs and Transferring Their Registered Servers

To list details of the expired CSIs for which you are the administrator:

1. Log in to ULN as administrator of the CSI, and select the CSI Administration tab.
2. On the Managed CSIs page in the Select Managed CSI Services pane, select the **Expired** link. The Managed Expired CSI Services pane displays the service details for each expired CSI that you administer.

3. Click the **View # Server(s)** link to display the details of the servers that are registered to an expired CSI.

4. On the Registered Servers page, you can transfer one or more systems to another user or to another CSI that you administer.

Note

If you transfer a system to another user, at least one of the following conditions must be true:

- His or her user name must be registered to this CSI.
- One or more of the servers, for which they are the owner, must be registered to this CSI.
- He or she must be an administrator of at least one CSI for which you are also an administrator.

To transfer systems to another user:

a. Select the **Transfer System** check boxes for the systems that you want to transfer.

b. Click **Transfer Selected Systems to Another Owner**.

c. On the Transfer Registered System(s) - Owner page in the Transfer To column, click the red arrow icon that is next to the user name of the user to whom you want to transfer ownership.

d. On the Confirm Transfer Profile - Owner page, click **Apply Changes** to confirm the transfer to the new owner.

To transfer systems to another CSI:

a. Select the **Transfer System** check boxes for the systems that you want to transfer.

b. Click **Transfer Selected Systems to Another CSI**.

c. On the Transfer Registered System(s) - CSI page in the Transfer To column, click the red arrow icon that is next to the CSI to which you want to transfer the systems.

d. On the Confirm Transfer Profile - CSI page, click **Apply Changes** to confirm the transfer to the new CSI.

2.4 Removing a CSI Administrator

To remove an administrator who is registered for a CSI:

1. Log in to ULN and select the CSI Administration tab.

2. On the Managed CSIs page, click **List Administrators**.

3. On the CSI Administrators page in the Delete? column, click the trash can icon that is next to the user name of the user that you want to remove as administrator for the CSI specified in the same row.

4. When prompted to confirm that you want to revoke administration privileges for the CSI from that user, click **OK**.

Chapter 3 ULN Registration

Table of Contents

This chapter describes ULN registration options. Typically, when you register a system with ULN, you provide your Oracle Single Signon (SSO) credentials as part of the registration process. This action automatically links the system to your own user name and creates a ULN profile for your user. It is also possible to register with ULN without registering a system. Using the approach to register with ULN enables your Oracle SSO credentials to be associated with a ULN profile prior to adding any systems to your account. Both approaches are described.

For users who are considering moving from RHN to ULN, Oracle provides a migration path that allows you to change which network you use to support your systems. Migration consists of downloading the Oracle versions of the required packages, installing the package and then running the system registration process with the tools provided in the new package. This procedure is also described in detail in this chapter.

3.1 Registering as a ULN User

When you register a system with ULN, your Oracle Single Signon (SSO) user name is also registered as your ULN user name. If you want to use ULN without first registering a system, you can register as a ULN user provided that you have a valid customer support identifier (CSI) for Oracle Linux support or Oracle VM support. To purchase Oracle Linux or Oracle VM support, go to the online Oracle Linux Store or contact your sales representative.

To register as a ULN user:

1. In a browser, go to https://linux.oracle.com/register.
2. If you do not have an SSO account, click **Create New Single Signon Account** and follow the onscreen instructions to create one.

   If you already have an SSO account, click **Sign On**.
3. Log in using your SSO user name and password.
4. On the Create New ULN User page, enter your CSI and click **Create New User**.

Note

If no administrator is currently assigned to manage the CSI, you are prompted to click **Confirm** to become the CSI administrator. If you click **Cancel**, you cannot access the CSI administration feature. See Chapter 2, *CSI Administration*.

If your user name already exists on the system, you are prompted to proceed to ULN by clicking the link **Unbreakable Linux Network**. If you enter a different CSI from your existing CSIs, your user name is associated with the new CSI in addition to your existing CSIs.

3.2 Registering an Oracle Linux System With ULN

To register an Oracle Linux 6 or Oracle Linux 7 system with ULN.

1. Run the `uln_register` command.

```
# uln_register
```

   Alternatively, if you use the GNOME graphical user desktop, select **System > Administration > ULN Registration** on Oracle Linux 6 or **Applications > System Tools > ULN Registration** on Oracle Linux 7. You can also register your system with ULN if you configure networking when installing Oracle Linux 6 or Oracle Linux 7.

2. When prompted, enter your ULN user name, password, and customer support identifier (CSI).

3. Enter a name for the system that will allow you to identify it on ULN, and choose whether to upload hardware and software profile data that allows ULN to select the appropriate packages for the system.

4. If you have an Oracle Linux Premier Support account, you can choose to configure an Oracle Linux 6 or Oracle Linux 7 system that is running a supported kernel to receive kernel updates from Oracle Ksplice.

The `yum-rhn-plugin` is enabled and your system is subscribed to the appropriate software channels.

If you use a proxy server for Internet access, see Section 4.3, “Configuring the Use of a Proxy Server”.

For information about registering to use Ksplice, see *Oracle® Linux: Ksplice User's Guide*.

3.3 Migrating from RHN to ULN

Note

You must have a ULN account before you can register a system with ULN. You can create a ULN account at https://linux.oracle.com/register.

To register your system with ULN instead of RHN:

1. Download the `uln_register.tgz` package from https://linux-update.oracle.com/rpms to a temporary directory.

   If the `rhn-setup-gnome` package is already installed on your system, also download the `uln_register-gnome.tgz` from the same URL.

2. Extract the packages using the following command.

```
# tar -xzf uln_register.tgz
```

   If the `rhn-setup-gnome` package is installed on your system, extract the packages from `uln_register-gnome.tgz`.

```
# tar -xzf uln_register-gnome.tgz
```

3. Change to the `uln_migrate` directory and install the registration packages.

```
# cd ./uln_migrate
# rpm -Uvh *.rpm
```

4. Run the `uln_register` command.

```
# uln_register
```

5. Follow the instructions on the screen to complete the registration. The `uln_register` utility collects information about your system and uploads it to Oracle.

For more information, see Section 3.2, “Registering an Oracle Linux System With ULN”.

Chapter 4 ULN System Management

Table of Contents

Systems that have been registered with ULN can be managed by using the ULN web interface at https://linux.oracle.com. You can use these tools to manage system information, such as the associated CSI for a system, the channels that a system is subscribed to, or to browse available updates and errata.

Some command-line tools are also provided to help perform certain tasks, such as channel subscription management, directly from the shell on the system itself.

System updates and package installation from ULN are handled by using the `yum` command directly on the system that is registered with ULN.

Some direct system configuration may be applied directly to the system, for example, to configure proxy settings or disable ULN updates for a particular package.

This chapter describes configuration steps and procedures to perform these tasks on a system that is registered with ULN.

4.1 ULN Channel Subscription Management

You can configure the channels that a system is subscribed to through the ULN Web interface. This chapter describes channel subscription in more detail.

4.1.1 Managing ULN Channel Subscription by Using the ULN Web Interface

If you have registered your system with ULN, you can subscribe the system to the channels that are available for the level of support that is associated with the CSI.

To subscribe your system to ULN channels:

1. Log in to https://linux.oracle.com with your ULN user name and password.
2. On the Systems tab, click the link named for the system in the list of registered machines.
3. On the System Details page, click **Manage Subscriptions**.
4. On the System Summary page, select channels from the list of available or subscribed channels and click the arrows to move the channels between the lists.
5. When you have finished selecting channels, click **Save Subscriptions**.

Note that you can view a complete listing of all available channels, for all operating systems and all architectures, by clicking on the Channels tab when you are logged into https://linux.oracle.com. You can

use the **Release** and **Architecture** drop-down selection boxes to limit the listing to a particular operating system release and architecture.

4.2 Modifying System Details

If you have registered your system with ULN, you can modify the details that ULN records for the system.

To update the details for your system:

1. Log in to https://linux.oracle.com with your ULN user name and password.
2. On the Systems tab, click the link named for the system in the list of registered machines.
3. On the System Details page, click **Edit**.
4. On the Edit System Properties page, you can change the name that is associated with your system, register it as a local yum server for your site, or change the CSI with which it is registered.

Note

You cannot change the CSI of a system unless it is registered to your user name.

5. When you have finished making changes, click **Apply Changes**.

4.3 Configuring the Use of a Proxy Server

If your organization uses a proxy server as an intermediary for Internet access, specify the `proxy` setting in the `/etc/yum.conf` file, as shown in the following example:

```
proxy=http://proxysvr.example.com:3128
```

If the proxy server requires authentication, additionally specify the `proxy_username`, and `proxy_password` settings.

```
proxy=http://proxysvr.example.com:3128
proxy_username=yumacc
proxy_password=clydenw
```

If you use the yum plugin (`yum-rhn-plugin`) to access the ULN, specify the `enableProxy` and `httpProxy` settings in `/etc/sysconfig/rhn/up2date` as shown in this example.

```
enableProxy=1
httpProxy=http://proxysvr.example.com:3128
```

If the proxy server requires authentication, additionally specify the `enableProxyAuth`, `proxyUser`, and `proxyPassword` settings.

```
enableProxy=1
httpProxy=http://proxysvr.example.com:3128
enableProxyAuth=1
proxyUser=yumacc
proxyPassword=clydenw
```

Caution

All `yum` users require read access to `/etc/yum.conf` or `/etc/sysconfig/rhn/up2date`. If these files must be world-readable, do not use a proxy password that is the same as any user's login password, and especially not `root`'s password.

4.4 Updating a System by Using Yum

Yum integration with ULN makes it possible to run most `yum` commands on the system after registering it with ULN and configuring the channels to which the system is subscribed. You can use the `yum install` and `yum update` commands to handle general package installation or updates.

To update a system to use the latest packages that are available on ULN, run:

```
# yum update
```

The ULN integration with `yum` enables you to run commands like `yum repolist` to obtain a listing of the ULN channels to which the system is subscribed. You can search for packages and obtain package information in the same way as you would if you were using `yum` to access the Oracle Linux yum server.

4.5 Disabling ULN Package Updates

You might need to disable package updates from ULN. For example, if you deleted your system from ULN, you would edit the `/etc/yum/pluginconf.d/rhnplugin.conf` file and change the value of `enabled` flag from `1` to `0` in the `[main]` section, as shown in the following example:

```
[main]
enabled = 0
gpgcheck = 1
```

To disable updates for particular packages, add an `exclude` statement to the `[main]` section of the `/etc/yum.conf` file. For example, to exclude updates for `VirtualBox` and `kernel`:

```
exclude=VirtualBox* kernel*
```

Note

Excluding certain packages from being updated can cause dependency errors for other packages. Your system could also become vulnerable to security-related issues if you do not install the latest updates.

4.6 Browsing Available Errata for a System

You can download a comma-separated values (CSV) report file of the errata that are available for a specific system registered on ULN and download any available errata RPMs, individually. You can also browse all of the available advisories that are available on ULN and download the errata RPMs for the supported combinations of the software release and the system architecture.

To download a CSV report or the errata RPMs for a specific system:

1. Log in to https://linux.oracle.com with your ULN user name and password.
2. On the Systems tab, click the link named for the system in the list of registered machines.

   The System Details page lists the available errata for the system in the Available Errata table, which might be split over several pages.
3. To download the CSV report file, click the link **Download All Available Errata for this System**.
4. To see more detail about an advisory and to download the RPMs:

   a. Click the link for the advisory.

b. On the System Errata Detail page for an advisory, you can download the RPMs for the affected releases and system architectures.

Note that updating the system by using the `yum update` command directly on the affected system, downloads these RPMs and updates the system with all available errata updates.

To browse all available advisories and download errata RPMs:

1. Log in to https://linux.oracle.com with your ULN user name and password.
2. Select the Errata tab.

   The Errata page displays a table of the available errata for all releases that are available on ULN.
3. On the Errata page, you can perform the following actions on the displayed errata:
   - To sort the table of available errata, click the title of the **Type**, **Severity**, **Advisory**, **Systems Affected**, or **Release Date** column. Click the title again to reverse the order of sorting.

     Note

     The **Systems Affected** column shows how many of your systems are potentially affected by an advisory.
   - To display or hide advisories of different types, select or deselect the **Bug**, **Enhancement**, and **Security** check boxes and click **Go**.
   - To display only advisories for a certain release of Oracle Linux or Oracle VM, select that release from the **Release** drop-down list and click **Go**.
   - To search within the table, enter a string in the **Search** field and click **Go**.
4. To see more detail about an advisory and to download the RPMs:
   a. Click the link for the advisory.
   b. On the Errata Detail page for an advisory, you can download the RPMs for the supported releases and system architectures. The **Superseded By Advisory** column displays a link to the most recent advisory (if any) that replaces the advisory you are browsing.

4.7 Removing a System From ULN

To remove a system that is registered with ULN:

1. Log in to https://linux.oracle.com with your ULN user name and password.
2. On the Systems tab, click the link named for the system in the list of registered machines.
3. On the System Details page, click **Delete**.

   Note

   You cannot delete a system unless it is registered to your user name.
4. When prompted to confirm the deletion, click **OK**.

Chapter 5 Creating and Using a Local ULN Mirror

Table of Contents

You can configure a local yum server to mirror the ULN channels within your network. This approach reduces the overhead that is associated with registering and managing systems within ULN, while still provisioning systems with all of the available software and updates that are available on ULN. Systems that are not able to connect to the Internet, either directly or by using a proxy, can also use this approach to keep up to date with the latest software.

This approach requires that the yum server is registered with ULN, has the available disk space to host the mirrored channels and is subscribed to the channels that it hosts. Systems that use the local ULN mirror are dependent on the synchronization of packages on the local yum server with the most recent updates provided by Oracle through ULN. If the local ULN mirror falls out of date, systems within your network may not be able to install critical security updates.

This chapter provides information on the requirements to host a local ULN mirror and the procedure to set up and configure the yum server for this purpose. Instructions are provided for configuring client systems to access and use the local yum server to obtain updates.

Note

If you are considering mirroring ULN channels on a local yum server, you should also investigate Spacewalk for Oracle Linux. Spacewalk provides tools to help with system maintenance, installation and package management, including tools to easily mirror ULN channels either from an intuitive web interface, or from a command line tool. For more information, see the Spacewalk for Oracle Linux documentation at https://docs.oracle.com/cd/E92593_01/index.html.

5.1 Prerequisites for the Local ULN Mirror

The system that you want to set up as a local ULN mirror must meet the following criteria:

- Must be registered with ULN. See Chapter 3, *ULN Registration*.
- Must be running Oracle Linux 6 (x86_64) or Oracle Linux 7 (x86_64).
- Must have at least 6 GB of memory to create the yum metadata.
- Must have enough disk space to store copies of the packages that it hosts. Note that disk space that is used for a mirror is only consumed and is never released. You should monitor disk space requirements regularly.

  The following table shows the approximate amount of space that is required for Oracle Linux channels:

Oracle Linux Channel	Space Required per Channel for Binaries Only	Space Required per Channel for Both Binaries and Source
`ol*_latest`	Up to 60 GB	Up to 120 GB
`ol*_addons`	8 GB	10 GB
`ol*_oracle`	1.5 GB	Not applicable
`ol*_base`	7 GB	14 GB
`ol*_patch`	6 GB	12 GB

The next table shows the approximate amount of space that is required for Oracle VM channels:

Oracle VM Channel	Space Required per Channel for Binaries Only	Space Required per Channel for Both Binaries and Source
`ovm*_latest`	2 GB	5 GB
`ovm*_base`	500 MB	1.5 GB
`ovm*_patch`	1 GB	2.5 GB

Note that since packages are updated frequently within these channels, the disk space requirements listed here are only approximations and may increase over time.

5.2 Setting up a Local ULN Mirror

To set up a local system as a local ULN mirror:

1. Enable the system as a **Yum Server** within ULN System Management. You can do this using the ULN web interface .

   This option disables system specific logic that is applied when a system attempts to subscribe to channels that do not apply to its architecture or platform version. For example, when a system is enabled as a **Yum Server** within ULN System Management, it is able to subscribe to channels for alternate architectures or operating system versions.

   Enable the Yum Server option using the ULN web interface

   a. Using a browser, log in at https://linux.oracle.com with the ULN user name and password that you used to register the system

   b. On the Systems tab, click the link named for your system in the list of registered machines.

   c. On the System Details page, click **Edit**.

   d. On the Edit System Properties page, select the **Yum Server** check box and click **Apply Changes**.

2. Subscribe the system to the channels that you intend to mirror. You can do this either using the ULN web interface, or by using the `uln-channel` command.

Note

You must subscribe the system to the `latest` and `addons` channels for the installed operating system release (Oracle Linux 6 or Oracle Linux 7) and the relevant system architecture to be able to install the `uln-yum-mirror` package. This package contains the `uln-yum-mirror` script that enables the system to act as a local ULN mirror.

If you subsequently update the list of channels to which the system is subscribed, the `uln-yum-mirror` script updates the channels that the system mirrors.

If you have an Oracle Linux Premier Support account and you want the yum server to host Ksplice packages for local Ksplice Offline clients, subscribe to the Ksplice for Oracle Linux channels for the architectures and Oracle Linux releases that you want to support.

For a complete and up-to-date list of the available release channels, log on to ULN at https://linux.oracle.com.

To subscribe your system to the channels that you want to mirror, use the ULN web interface .

Subscribe your system to channels using the ULN web interface

a. Log in to https://linux.oracle.com with your ULN user name and password.

b. On the Systems tab, click the link named for the system in the list of registered machines.

c. On the System Details page, click **Manage Subscriptions**.

d. On the System Summary page, select channels from the list of available or subscribed channels and click the arrows to move the channels between the lists.

e. When you have finished selecting channels, click **Save Subscriptions**.

3. Install the Apache HTTP server.

```
# yum install httpd
```

4. Create a base directory for the yum repositories, for example `/var/yum` or `/var/www/html/yum`.

```
# mkdir -p /var/www/html/yum
```

Note

The yum repository owner must have read and write permissions on this directory.

5. If you created a base directory for the yum repository that is not under `/var/www/html` and SELinux is enabled in enforcing mode on your system:

a. Use the `semanage` command to define the default file type of the repository root directory hierarchy as `httpd_sys_content_t`:

```
# /usr/sbin/semanage fcontext -a -t httpd_sys_content_t "/var/yum(/.*)?"
```

b. Use the `restorecon` command to apply the file type to the entire repository.

```
# /sbin/restorecon -R -v /var/yum
```

6. If you created a base directory for the yum repository that is not under `/var/www/html`, create a symbolic link in `/var/www/html` that points to the repository, for example:

```
# ln -s /var/yum /var/www/html/yum
```

7. Edit the HTTP server configuration file, `/etc/httpd/conf/httpd.conf`, as follows:

a. Specify the resolvable domain name of the server in the argument to `ServerName`.

```
ServerName server_addr:80
```

If the server does not have a resolvable domain name, enter its IP address instead.

b. Verify that the setting of the `Options` directive in the `<Directory "/var/www/html">` section specifies `Indexes` and `FollowSymLinks` to allow you to browse the directory hierarchy, for example:

```
Options Indexes FollowSymLinks
```

c. Save your changes to the file.

8. Start the HTTP server, and configure it to start after a reboot.

- On Oracle Linux 6, type the following commands:

```
# service httpd start
# chkconfig httpd on
```

- On Oracle Linux 7, enter the following commands:

```
# systemctl start httpd
# systemctl enable httpd
```

9. If you enabled a firewall on your system, configure it to allow incoming HTTP connection requests on TCP port 80.

- On Oracle Linux 6, type the following commands:

```
# iptables -I INPUT -p tcp -m state --state NEW -m tcp --dport 80 -j ACCEPT
# service iptables save
```

- On Oracle Linux 7, type the following commands:

```
# firewall-cmd --add-service=http
# firewall-cmd --permanent --add-service=http
```

10. Install the `uln-yum-mirror` package:

```
# yum install uln-yum-mirror
```

This package contains the `uln-yum-mirror` script that enables the system to act as a local ULN mirror.

Note

If you have not subscribed the system to the correct Oracle Linux `latest` and `addons` channels for your system, the command fails with the following error:

```
No package uln-yum-mirror available
```

11. To configure the operation of the `/usr/bin/uln-yum-mirror` script, edit the `/etc/sysconfig/uln-yum-mirror` file.

For example, if the base directory for the yum repositories is not `/var/www/html/yum`, set the value of the `REP_BASE` parameter to the correct base directory:

```
REP_BASE=/var/yum
```

Installing the `uln-yum-mirror` package also configures an `anacron` job (`/etc/cron.daily/uln-yum-mirror`) that updates the local yum repositories once every day. You can disable this job by setting the value of `CRON_ENABLED` to `0`:

```
CRON_ENABLED=0
```

For more information about the configuration options in `/etc/sysconfig/uln-yum-mirror` file, see Section 5.3, “ULN Mirror Configuration” .

The repositories are populated when the `anacron` job runs the `/usr/bin/uln-yum-mirror` script. Alternatively, you can run the script manually at any time to update the repositories. See Section 5.4, “Updating the Repositories on a Local ULN Mirror” .

5.3 ULN Mirror Configuration

The `/etc/sysconfig/uln-yum-mirror` file contains the following configuration parameters that affect the behavior of the `/usr/bin/uln-yum-mirror` script:

`ALL_PKGS`	Specifies whether `uln-yum-mirror` mirrors all versions of every available package or downloads only the latest version of each package. The default value of 1 causes `uln-yum-mirror` to mirror all versions of every available package. A value of 0 causes `uln-yum-mirror` to download only the latest version of each package.
`CRON_ENABLED`	Specifies whether `uln-yum-mirror` runs automatically once per day. The default value of 1 enables `uln-yum-mirror` to be run automatically as an `anacron` job. A value of 0 disables the job. You must run `uln-yum-mirror` manually to update the packages.
`HARDLINK_RPMS`	Specifies whether `uln-yum-mirror` runs `hardlinkpy` to create hard links between identical RPMs after the mirror process finishes. The default value of 1 enables hard linking, which saves storage space. It is not possible to create hard links across file systems. Set the value to 0 if the repository storage spans more than one file system.
`LOG_OUTPUT`	Specifies whether `uln-yum-mirror` logs its output. The default value of 1 enables logging. A value of 0 disables logging.
`REP_BASE`	Specifies the base directory for the repositories. The default setting is `/var/www/html/yum`. Do not change this setting unless you customize the configuration of the HTTP server.
`REP_EL` , `REP_ENG` , `REP_OL` , `REP_OVM` , `REP_UEK`	Specify the names of the repositories. If required, you can configure alternate names.
`REPO_FILE_DIR`	Not currently used.
`SRC`	Specifies whether `uln-yum-mirror` mirrors source RPMs in addition to binary RPMs. The default value of 0 prevents `uln-yum-mirror` from mirroring source RPMs. A value to 1 causes `uln-yum-mirror` to mirror source RPMs.
`YUM_GLOBAL_CACHE`	Specifies the `yum` global cache directory. The default setting is `/var/cache/yum`. Do not change this setting unless you customize the configuration of the HTTP server.

5.4 Updating the Repositories on a Local ULN Mirror

To update the repositories for the subscribed channels immediately without waiting for the `anacron` job to run or if you have disabled the job, enter the following command on the local ULN mirror server:

```
# /usr/bin/uln-yum-mirror
```

Note

If you have not yet set up the contents of the repositories, it can take many hours to download all the packages.

5.5 Configuring `yum` on a Local ULN Mirror

The following procedure configures the `yum` command on a server that is acting as a local ULN mirror to install package updates from itself rather than from ULN. This type of configuration can be important when the system is configured to mirror channels for alternate platforms or architectures. Because channel subscription logic is disabled for a system that is configured as a **Yum Server** within ULN System Management, the system could potentially install packages from channels with conflicting architectures or platforms. For this reason, you may want to configure the system to use the local ULN mirror that it is hosting, which enables you to control the system's own channel or repository subscriptions.

Note that the procedure does not affect the operation of the `uln-yum-mirror` script.

To configure a server that is acting as a local ULN Mirror to be able to install updated packages from itself:

1. Use the following command to list the channels that the server is mirroring from ULN:

```
# yum repolist
Loaded plugins: rhnplugin, security
This system is receiving updates from ULN.
0 packages excluded due to repository protections
repo id                  repo name                                    status
ol6_x86_64_addons        Oracle Linux 6 Add ons (x86_64)              367
ol6_x86_64_latest        Oracle Linux 6 Latest (x86_64)               35,995
ol6_x86_64_UEKR3_latest  Latest Unbreakable Enterprise Kernel
                         Release 3 for Oracle Linux 6 (x86_64)        41
```

   In this example, the server mirrors the `ol6_addons`, `ol6_x86_64_latest`, and `ol6_x86_64_UEKR3_latest` channels from ULN.

2. Edit `/etc/yum/pluginconf.d/rhnplugin.conf` and disable the mirrored channels by adding the following stanza for each channel:

```
[repo_id]
enabled=0
```

   For example, to disable the `ol6_addons`, `ol6_x86_64_latest`, and `ol6_x86_64_UEKR3_latest` channels, you would add the following stanzas:

```
[ol6_addons]
enabled=0

[ol6_x86_64_latest]
enabled=0

[ol6_x86_64_UEKR3_latest]
enabled=0
```

Note

If you subsequently subscribe the system to any additional channels on ULN, you must also disable those channels in `/etc/yum/pluginconf.d/rhnplugin.conf`.

3. Configure the server as a yum client as described in Section 5.6, “Configuring Access to a Local ULN Mirror” .

5.6 Configuring Access to a Local ULN Mirror

If you have set up a local ULN mirror, you can configure your local Oracle Linux systems to receive yum updates from that server.

To configure an Oracle Linux system as a yum client:

1. Import the GPG key:

```
# rpm --import /usr/share/rhn/RPM-GPG-KEY
```

2. Disable any existing yum repositories configured in the `/etc/yum.repos.d` directory. You can either edit any existing repository files and disable all entries by setting `enabled=0` or you can use `yum-config-manager`:

```
# yum-config-manager --disable \*
```

Alternately, you can rename any of the files in this directory so that they do not use the `.repo` suffix. This causes yum to ignore these entries. For example:

```
# cd /etc/yum.repos.d
# for i in *.repo; do mv $i $i.disabled; done
```

3. In the `/etc/yum.repos.d` directory, create the file `local-yum.repo`, which contains entries such as the following for an Oracle Linux 6 yum client:

```
[local_ol6_latest]
name=Oracle Linux $releasever - $basearch - latest
baseurl=http://local_uln_mirror/yum/OracleLinux/OL6/latest/$basearch/
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY
gpgcheck=1
enabled=1

[local_ol6_UEKR3_latest]
name=Unbreakable Enterprise Kernel Release 3 for Oracle Linux $releasever - $basearch - latest
baseurl=http://local_uln_mirror/yum/OracleLinux/OL6/UEKR3/latest/$basearch/
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY
gpgcheck=1
enabled=1

[local_ol6_addons]
name=Oracle Linux $releasever - $basearch - addons
baseurl=http://local_uln_mirror/yum/OracleLinux/OL6/addons/$basearch/
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY
gpgcheck=1
enabled=1
```

To distinguish the local repositories from the ULN repositories, prefix the names of their entries with a string such as `local_`.

Replace *`local_uln_mirror`* with the IP address or resolvable host name of the local ULN mirror.

The example configuration enables the `local_ol6_latest`, `local_ol6_UEKR3_latest`, and `local_ol6_addons` channels.

4. To test the configuration:

   a. Clear the yum metadata cache:

   ```
   # yum clean metadata
   ```

   b. Use `yum repolist` to verify the configuration, for example:

   ```
   # yum repolist
    Loaded plugins: rhnplugin, security
    This system is receiving updates from ULN.
   0 packages excluded due to repository protections
   repo id                          repo name                                      status
   local_ol6_addons                 Oracle Linux 6 - x86_64 - latest               367
   local_ol6_x86_64_latest          Oracle Linux 6 - x86_64 - latest               35,995
   local_ol6_x86_64_UEKR3_latest    Unbreakable Enterprise Kernel Release 3
                                    for Oracle Linux 6 - x86_64 - latest           41
   ```

   If `yum` cannot connect to the local ULN mirror, check that the firewall settings on the local ULN mirror server allow incoming TCP connections to the HTTP port (usually, port 80).

5. You can now run `yum update` to pick up new updates from the local ULN mirror.

Appendix A The Unbreakable Linux Network API

Table of Contents

This appendix describes the XML-RPC methods that the API provides for access to the Unbreakable Linux Network (ULN).

This API is based on XML-RPC, which enables applications to perform remote operations by encoding the procedure calls in XML and transmitting them over HTTP. For more information about XML-RPC, see http://www.xmlrpc.com/.

The API is accessed at the server entry point URL at https://linux-update.oracle.com/XMLRPC.

The following method namespaces are available:

`auth`	Contains methods for authenticating with ULN. See Section A.1, “Authentication Methods”.
`channel`	Contains methods for listing software channels on ULN. See Section A.2, “Channel Methods”.
`channel.software`	Contains methods for querying packages available within different channels on ULN. See Section A.3, “Channel Software Methods”.
`errata`	Contains methods for interacting with errata on ULN. See Section A.4, “Errata Methods”.
`packages`	Contains methods for querying package information for specified packages on ULN. See Section A.5, “Packages Methods”.
`system`	Contains methods for managing systems registered with ULN. See Section A.6, “System Methods”.

A.1 Authentication Methods

Authentication methods are provided in the `auth` namespace. The following methods are provided for authenticating with ULN:

- Section A.1.1, "`auth.login`"
- Section A.1.2, "`auth.logout`"

A.1.1 `auth.login`

The `login` method logs in to ULN using a specified user name and password.

The input parameters are:

Input Parameters

`string username`	The Oracle Single Sign On (SSO) user name to use for the session. For example: `myuser@example.com`
`string password`	The password to use for the session. For example: `secret`

Return Parameters

`string sessionKey`	The session key for the session. All other methods use the session key for the duration of the session. For example: `JyUVNoT74BFaRJ6fRjDIQ5idPmCaj5UJLb76E2f45Gc`

A.1.2 `auth.logout`

The `logout` method logs out of the ULN session specified by the session key.

Input Parameters

`string sessionKey`	The session key of the session to be terminated. For example: `JyUVNoT74BFaRJ6fRjDIQ5idPmCaj5UJLb76E2f45Gc`

Return Parameters

`int`	The method returns an `int` error code, which indicates whether the session terminated correctly. A value of `1` indicates a successful return.

A.2 Channel Methods

Channel methods are available in the `channel` namespace. The following method is provided for listing software channels that are available on ULN:

- Section A.2.1, "`channel.listSoftwareChannels`"

A.2.1 `channel.listSoftwareChannels`

The `listSoftwareChannels` method returns a list of software channels that are available to a session on ULN.

Input Parameters

`string sessionKey`	The session key for the session. For example: `JyUVNoT74BFaRJ6fRjDIQ5idPmCaj5UJLb76E2f45Gc`

Return Parameters

`array`	An array of channels with:		
	`struct (channel)`	A structure containing the following strings:	
		`string channel_arch`	The channel architectur For example: `x86_64`
		`string channel_end_of_life`	The channel end of life. Currently unused on ULN.
		`string channel_label`	The channel label. For example: `ol7_x86_`
		`string channel_name`	The channel name. For example: `Oracle Linux 7 Latest (x86_64)`
		`string channel_parent_label`	The channel parent label. Currently unused on ULN.

A.3 Channel Software Methods

Channel software methods are available in the `channel.software` namespace. The following methods can by used to query the packages that are available to a session from a channel on ULN.

A.3.1 channel.software.getDetails

The `getDetails` method returns the details of the given channel.

Input Parameters

`string sessionKey`	The session key for the session. For example: `JyUVNoT74BFaRJ6fRjDIQ5idPmCaj5UJLb76E2f45Gc`
`string channelLabel`	The channel label for the channel that you wish to query. For example: `ol7_x86_64_latest`

Return Parameters

`string channel_arch_name`	The channel architecture name. For example: `x86_64`		
`string channel_description`	The channel description. For example: `All packages released for Oracle Linux 7 (x86_64) including the latest errata packages. (x86_64)`		
`string channel_summary`	The channel summary, usually the same as the channel name. For example: `Oracle Linux 7 Latest (x86_64)`		
`struct metadata_urls`	A dictionary or associative array of metadata locations and checksum information, including the URLs to download channel metadata.		
	`struct filelists`	`string checksum_type`	The hashing algorithm used to generate the checksum. For example: `sha`
		`string checksum`	The checksum for the filelists metadata file. For example: `abc4ef3d6e6`

	string file_name	The file name for the filelists metadata at the channel location. For example: repodata filelist
	string url	The URL where the filelists metadata can be accessed. For example: https://uln.orac XMLRPC/ GET-REQ/ ol7_x86_ repodata filelist
struct group		This structure is returned optionally if this information is available.
	string checksum_type	The hashing algorithm used to generate the checksum. For example: sha
	string checksum	The checksum

			for the group metadata file. For example: 90acbe6860b
		string file_name	The file name for the group metadata at the channel location. For example: repodata/comps.xml
		string url	The URL where the group metadata can be accessed. For example: https://uln.oracle.XMLRPC/GET-REQ/ol7_x86_64_repodata/comps.xml
	struct other	string checksum_type	The hashing algorithm used to generate the checksum. For

			example: sha
		string checksum	The checksum for the other metadata file. For example: 20f6b193
		string file_name	The file name for the other metadata at the channel location. For example: repodata other.xm
		string url	The URL where the other metadata can be accessed. For example: https:// uln.orac XMLRPC/ GET- REQ/ ol7_x86_ repodata other.xm
	struct primary	string checksum_type	The hashing algorithm used

		to generate the checksum. For example: `sha`
	`string checksum`	The checksum for the primary metadata file. For example: `3992e1e77d4`
	`string file_name`	The file name for the primary metadata at the channel location. For example: `repodata/primary.xml`
	`string url`	The URL where the primary metadata can be accessed. For example: `https://uln.oracle.XMLRPC/GET-REQ/ol7_x86_64_repodata/primary.xml`

struct repomd	string file_name	The file name for the repomd metadata at the channel location. For example: repodata repomd.x
	string url	The URL where the repomd metadata can be accessed. For example: https:// uln.orac XMLRPC/ GET- REQ/ ol7_x86_ repodata repomd.x
struct updateinfo	This structure is returned optionally if this information is available.	
	string checksum_type	The hashing algorithm used to generate the checksum. For example: sha
	string checksum	The checksum

		for the updateinfo metadata file. For example: 6d11ecbceb5
	string file_name	The file name for the updateinfo metadata at the channel location. For example: repodata/updateinfo.
	string url	The URL where the updateinfo metadata can be accessed. For example: https://uln.oracle.XMLRPC/GET-REQ/ol7_x86_64_repodata/updateinfo.

A.3.2 channel.software.listAllPackages

The listAllPackages method returns a list of all packages that are available from a channel, including packages that are not the latest.

Input Parameters

string sessionKey	The session key for the session. For example: JyUVNoT74BFaRJ6fRjDIQ5idPmCaj5UJLb76E2f45Gc.

`string channelLabel`	The channel label for the channel that you wish to query.

Return Parameters

`array`	An array of all packages:		
	`struct (package)`	A structure containing the following strings:	
		`string package_arch_label`	The package architectur label. For example: `noarch`
		`string package_epoch`	The package epoch value, if specified. The epoch value can help RPM determine package version ordering if the versioning does not make sense or does not follow sequentiall For example: `1`
		`string package_id`	The package ID within the ULN

		infrastructure. For example: 11776733
	string package_last_modified	The date and timestamp for when a package was last modified. For example: 2018-09-27 19:31:13
	string package_name	The name of the package. For example: selinux-policy-mls
	string package_release	The package release information. For example: 192.0.6.el7
	string package_version	The package version number. For example: 3.13.1

A.3.3 channel.software.listErrata

The listErrata method returns a list of all errata that are associated with a channel.

Input Parameters

`string sessionKey` — The session key for the session. For example: `JyUVNoT74BFaRJ6fRjDIQ5idPmCaj5UJLb76E2f45Gc`.

`string channelLabel` — The channel label for the channel that you wish to query. For example: `ol7_x86_64_latest`

Return Parameters

`array` — An array of all errata associated with the channel label:

- `struct (errata)` — A structure containing the following strings:
  - `string errata_advisory_type` — The errata advisory type. For example: `Bug Fix Advisory`
  - `string errata_advisory` — The errata advisory label. For example: `ELBA-201`
  - `string errata_issue_date` — The date the errata was issued. For example: `2018-10-00:00:00`
  - `string errata_last_modified_date` — The date the errata was last modified . For example: `2018-10-00:00:00`

	string errata_synopsis	A brief synopsis of the errata. For example: glibc bug fix update
	string errata_update_date	The errata update date. For example: 2018-10-17 00:00:00

A.3.4 channel.software.listLatestPackages

The listLatestPackages method returns a list of the latest packages that are available from a channel.

Input Parameters

string sessionKey	The session key for the session. For example: JyUVNoT74BFaRJ6fRjDIQ5idPmCaj5UJLb76E2f45Gc.
string channelLabel	The channel label for the channel that you wish to query. For example: ol7_x86_64_latest

Return Parameters

array	An array of latest packages:	
	struct (package)	A structure containing the following strings:
		string package_arch_label — The package architecture label. For example: noarch
		string package_epoch — The package epoch value, if specified. The

		epoch value can help RPM determine package version ordering if the versioning does not make sense or does not follow sequentiall For example: 1
	string package_id	The package ID within the ULN infrastructu For example: 11776733
	string package_name	The name of the package. For example: selinux-policy-mls
	string package_release	The package release information For example: 192.0.6.

`string package_version`	The package version number. For example: `3.13.1`

A.4 Errata Methods

Errata methods are available in the `channel` namespace. The following methods are provided for interacting with errata that are available on ULN:

- Section A.4.1, “`errata.applicableToChannels`”
- Section A.4.2, “`errata.getDetails`”
- Section A.4.3, “`errata.listCves`”
- Section A.4.4, “`errata.listPackages`”

A.4.1 `errata.applicableToChannels`

The `applicableToChannels` method returns a list of all channels to which the specified erratum applies. .

Input Parameters

`string sessionKey`	The session key for the session. For example: `JyUVNoT74BFaRJ6fRjDIQ5idPmCaj5UJLb76E2f45Gc`.
`string advisoryName`	The name of the erratum (for example, `ELSA-2013-0269`).

Return Parameters

`array`	An array of channels:		
	`struct (channel)`	A structure containing the following strings:	
		`string channel_id`	The identifier for a channel in the ULN infrastructure. For example: `1844`
		`string channel_label`	The label for the

		channel. For example: ol7_x86_
	string channel_name	The full name for the channel. For example: Oracle Linux 7 Latest (x86_64)
	string parent_channel_label	The parent channel label. Not currently used on ULN.

A.4.2 errata.getDetails

The getDetails method returns detailed information for the specified erratum. Note that the method only fills in the errata_severity field for security errata.

Input Parameters

string sessionKey	The session key for the session. For example: JyUVNoT74BFaRJ6fRjDIQ5idPmCaj5UJLb76E2f45Gc
string advisoryName	The name of the erratum. For example: ELSA-2013-0269

Return Parameters

array		An array of detailed information associated with the erratum:
	struct (erratum)	A structure containing the following strings:
		string errata_description — The detailed description of the erratum. For example:

		[0:1.2.1-7. Add missing connection hostname check against X.509 certificate name \n- Resolves: CVE-2012-57
	string errata_issue_date	The date the erratum was issued. For example: 2/19/13
	string errata_last_modified_date	The date the erratum was last modified: For example: 2013-02-19 00:00:00
	string errata_notes	Notes associated with the erratum. Usually empty.
	string errata_references	References of the erratum. Usually empty.
	string errata_severity	The severity level

		set for the erratum . For example: `Moderate`
	`string errata_synopsis`	A brief synopsis of the erratum. For example: `axis security update`
	`string errata_topic`	The topic for the erratum. Usually empty.
	`string errata_type`	The type for the erratum. For example: `Security Advisory`
	`string errata_update_date`	The errata update date. For example: `2/19/13`

A.4.3 `errata.listCves`

The `listCves` method returns a list of Common Vulnerabilities and Exposures (CVE) IDs that are applicable to the specified erratum ID.

Input Parameters

`string sessionKey`	The session key for the session. For example: `JyUVNoT74BFaRJ6fRjDIQ5idPmCaj5UJLb76E2f45Gc`

`string advisoryName` The name of the erratum. For example: `ELSA-2018-2942`

Return Parameters

`array` An array of CVE IDs. If no matching CVE IDs are found, the array is empty.:

`string cve_name` The CVE ID associated with the erratum ID. For example: `CVE-2018-3136`

A.4.4 `errata.listPackages`

The `listPackage` method returns a list of all packages applicable to the specified erratum ID.

Input Parameters

`string sessionKey` The session key for the session. For example: `JyUVNoT74BFaRJ6fRjDIQ5idPmCaj5UJLb76E2f45Gc`

`string advisoryName` The name of the erratum. For example: `ELSA-2018-2942`

Return Parameters

`array` An array of packages:

`struct (package)` A structure containing the following strings:

`array download_urls` An array of URLs where the package can be downloaded from.

`string url`

`array providing_channels` An array

	listing channels providing this package. string l
string package_arch_label	The package architectur label. For example: i686
string package_build_date	The date and timestamp for when the package was built. For example: 2018-10-16:39:10
string package_build_host	For example: x86-ol7-builder-
string package_cookie	The package cookie value. Usually empty.

string package_description	The full description of the package. For example: The OpenJDK demos.
string package_epoch	The package epoch value, if specified. The epoch value can help RPM determine package version ordering if the versioning does not make sense or does not follow sequentially. For example: 1
string package_file	The package filename. For example: java-1.8.0-openjdk-demo-1.8.0.

string package_id	For example: 11807834
string package_last_modified_date	The date and timestamp for when the package was last modified. For example: 2018-10-16:39:10
string package_license	The license or licenses that a package is released under. For example: ASL 1.1 and ASL 2.0 and BSD and BSD with advertis and GPL + and GPLv2 and GPLv2 with exceptio and

		IJG and LGPLv2+ and MIT and MPLv2.0 and Public Domain and W3C and zlib
	string package_md5sum	The package md5sum value. For example: 1508de7bafe
	string package_name	The package name. For example: java-1.8.0-openjdk-demo
	string package_payload_size	The package payload size in bytes. For example: 4412184
	string package_release	The package release value. For example: 0.el7_5
	string package_size	The package size in

		bytes. For example: 4293131
	string package_summary	A summary of the contents of the package. For example: OpenJDK Demos
	string package_vendor	The package vendor name. For example: Oracle America
	string package_version	The package version. For example: 1.8.0.19
	struct package_checksums	A structure, listing package checksum values by type:
		string m

A.5 Packages Methods

Packages methods are available in the `packages` namespace. These methods are used for extracting information about the packages that are available to a session on the ULN. The following methods are provided for interacting with packages that are available on ULN:

- Section A.5.1, “`packages.getDetails`”
- Section A.5.2, “`packages.listProvidingErrata`”

A.5.1 `packages.getDetails`

The `getDetails` method returns detailed information about the specified package.

Input Parameters

`string sessionKey`	The session key for the session. For example: `JyUVNoT74BFaRJ6fRjDIQ5idPmCaj5UJLb76E2f45Gc`
`int pid`	The package identifier that should be queried, specified as an integer. For example: `11807834`

Return Parameters

`array`	An array of channels with:		
	`struct (package)`	A structure containing the following strings:	
		`array download_urls`	An array of URLs where the package can be downloaded from.
			`string url`
		`array providing_channels`	An array

		listing channels providing this package.
		string l
	string package_arch_label	The package architectur label. For example: i686
	string package_build_date	The date and timestamp for when the package was built. For example: 2018-10-16:39:10
	string package_build_host	The host where the package was built. For example: x86-ol7-builder-

`string package_cookie`	The package cookie value. Usually empty.
`string package_description`	The full description of the package. For example: `The OpenJDK demos.`
`string package_epoch`	The package epoch value, if specified. The epoch value can help RPM determine package version ordering if the versioning does not make sense or does not follow sequentially. For example: `1`
`string package_file`	The package

		filename. For example: `java-1.8 openjdk-demo-1.8`
	`string package_id`	For example: `11807834`
	`string package_last_modified_date`	The date and timestamp for when the package was last modified. For example: `2018-10-16:39:10`
	`string package_license`	The license or licenses that a package is released under. For example: `ASL 1.1 and ASL 2.0 and BSD and BSD with advertis and GPL + and`

		GPLv2 and GPLv2 with exceptions and IJG and LGPLv2+ and MIT and MPLv2.0 and Public Domain and W3C and zlib
	string package_md5sum	The package md5sum value. For example: 1508de7bafe
	string package_name	The package name. For example: java-1.8.0-openjdk-demo
	string package_payload_size	The package payload size in bytes. For example: 4412184
	string package_release	The package release value. For

	example: `0.el7_5`
`string package_size`	The package size in bytes. For example: `4293131`
`string package_summary`	A summary of the contents of the package. For example: `OpenJDK Demos`
`string package_vendor`	The package vendor name. For example: `Oracle America`
`string package_version`	The package version. For example: `1.8.0.19`
`struct package_checksums`	A structure, listing package checksum values by type:
	`string m`

A.5.2 packages.listProvidingErrata

The `listProvidingErrata` method returns a list of the errata that are associated with a package.

Input Parameters

`string sessionKey`	The session key for the session. For example: `JyUVNoT74BFaRJ6fRjDIQ5idPmCaj5UJLb76E2f45Gc`
`int pid`	The package identifier that should be queried, specified as an integer. For example: `11807834`

Return Parameters

`array`	An array of all errata associated with the package:		
	`struct (errata)`	A structure containing the following strings:	
		`string errata_advisory_type`	The errata advisory type. For example: `Security Advisory`
		`string errata_advisory`	The errata advisory label. For example: `ELSA-2018-2`
		`string errata_issue_date`	The date the errata was issued. For example: `2018-10-17 00:00:00`
		`string errata_last_modified_date`	The date

		the errata was last modified . For example: `2018-10-00:00:00`
	`string errata_synopsis`	A brief synopsis of the errata. For example: `java-1.8 openjdk security update`
	`string errata_update_date`	The errata update date. For example: `2018-10-00:00:00`

A.6 System Methods

System methods are available in the `system` namespace. These methods are used for managing systems that are registered on ULN. The following methods are available:

- Section A.6.1, “`system.deleteSystems`”

A.6.1 `system.deleteSystems`

The `deleteSystems` method removes a system from ULN, given its system ID.

Input Parameters

`string sessionKey`	The session key for the session. For example: `JyUVNoT74BFaRJ6fRjDIQ5idPmCaj5UJLb76E2f45Gc`.
`string serverId`	The system identifier that should be removed. This needs to be the id value within ULN. For example: `330213`

Return Parameters

`int`	The method returns an `int` error code, which indicates whether the system was deleted or not. A value of `0` indicates that the system was successfully removed from ULN.