

Oracle Linux 8

Upgrading Systems With Leapp



F36401-32
May 2024



Oracle Linux 8 Upgrading Systems With Leapp,

F36401-32

Copyright © 2022, 2024, Oracle and/or its affiliates.

Contents

Preface

Documentation License	v
Conventions	v
Documentation Accessibility	v
Access to Oracle Support for Accessibility	v
Diversity and Inclusion	v

1 About Leapp

Supported Leapp Features	1-1
Upgrading Oracle Linux 7 Systems	1-1
Upgrading Oracle Linux 7 Oracle Cloud Infrastructure Instances	1-2
Upgrading Oracle Linux 7 Systems That Use RDMA	1-3
Upgrading Oracle Linux 7 KVM Hosts	1-3
Upgrading Oracle Linux 7 Systems with Oracle Databases	1-4
Requirements for Upgrading	1-4
Kernels Upgradeable With Leapp	1-5

2 Preparing for the Upgrade

3 Upgrading the System

Assessing the Capability of the System for Upgrading	3-1
Running the Preupgrade	3-1
Analyzing the Leapp Report	3-2
Python Version Issue	3-2
GPG Key Issue	3-2
Providing Information to the Leapp Answerfile	3-3
Performing the Upgrade	3-4
Verifying the Upgrade	3-5

4 Completing Postupgrade Tasks

5 Troubleshooting Oracle Linux Upgrades

Tools for Troubleshooting	5-1
Known Issues	5-1
Upgrade Issues	5-1
Security and Authentication Issues	5-2
System Management Issues	5-3
File Systems and Storage Issues	5-3
Networking Issues	5-4
Virtualization and Containers Issues	5-5
Development Tools Issues	5-6
Hardware Related Issues	5-6
Leapp Overlay Size Issues	5-6

A Supported Repositories in Leapp Upgrades

Repository Mappings	A-1
Using Command Arguments to Enable Repositories	A-1

B Updated dhclient Script

Preface

[Oracle Linux 8: Upgrading Systems With Leapp](#) provides information about how to use the Leapp utility to perform system upgrades from Oracle Linux 7 to the current Oracle Linux 8 release.

Documentation License

The content in this document is licensed under the [Creative Commons Attribution–Share Alike 4.0 \(CC-BY-SA\)](#) license. In accordance with CC-BY-SA, if you distribute this content or an adaptation of it, you must provide attribution to Oracle and retain the original copyright notices.

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.
<i>italic</i>	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
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 <https://www.oracle.com/corporate/accessibility/>.

Access to Oracle Support for Accessibility

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit <https://www.oracle.com/corporate/accessibility/learning-support.html#support-tab>.

Diversity and Inclusion

Oracle is fully committed to diversity and inclusion. Oracle respects and values having a diverse workforce that increases thought leadership and innovation. As part of our initiative to build a more inclusive culture that positively impacts our employees, customers, and partners, we are working to remove insensitive terms from our products and documentation. We are also

mindful of the necessity to maintain compatibility with our customers' existing technologies and the need to ensure continuity of service as Oracle's offerings and industry standards evolve. Because of these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

1

About Leapp

The Leapp utility is a framework for updating and upgrading operating systems and applications. The utility's component packages enable the creation of different workflows into profiles for updating software.

Leapp operations consist of two phases:

- The preupgrade phase, where system checks are performed to verify if the software can be upgraded.
- The actual upgrade, which process is based on configuration files that map packages between previous and current versions of the software packages.

Caution:


Currently, the Leapp utility is used to upgrade the OSs only, namely, from the *current* Oracle Linux 7 release to the *current* Oracle Linux 8 version. The procedures in this document *don't apply to and are unsupported on* any other OSs or versions.

Supported Leapp Features

The Leapp utility can be used to upgrade local or remote Oracle Linux 7 systems and instances on Oracle Cloud Infrastructure that are based on the Oracle Linux 7 image.

Upgrading Oracle Linux 7 Systems

For Oracle Linux systems, the following table lists supported and unsupported features by the Leapp utility.

Upgradeable With Leapp	Not Upgradeable With Leapp
<p>Platforms (latest shipping updates)</p> <ul style="list-style-type: none"> x86_64 Arm (aarch64) <p>Operating Systems</p> <ul style="list-style-type: none"> Current Oracle Linux 7 version only <p>Profiles</p> <ul style="list-style-type: none"> Minimal Install Virtualization Host Server with GUI Basic Web Server Infrastructure Server File and Print Server 	<p>Systems installed with the following:</p> <ul style="list-style-type: none"> ISV applications and middleware <div style="border: 1px solid #0070C0; padding: 5px; margin: 10px 0;"> <p> Note:</p> <p>ISVs must provide and validate Leapp actors to coordinate their application upgrades.</p> </div> <ul style="list-style-type: none"> Oracle Linux Manager or Spacewalk for Oracle Linux Installations that did not use ISO images such as Ceph and GlusterFS Migration of disks that are encrypted with LUKS Systems that are registered with ULN Systems in FIPS mode or running SecureBoot Systems configured with Docker or Podman containers Systems configured with high-availability clustering software such as Pacemaker / Corosync or other 3rd party clustering solutions Instances that are being managed with the Oracle OS Management Hub service

Upgrading Oracle Linux 7 Oracle Cloud Infrastructure Instances

The Leapp utility can also upgrade both the x86_64 and Arm (aarch64) platforms that are running Oracle Linux 7 instances on Oracle Cloud Infrastructure.

The following table lists available and unavailable features:

Supported with Leapp	Unsupported with Leapp
<p>Images</p> <ul style="list-style-type: none"> Oracle Linux 7 Unbreakable Enterprise Kernel Release 6 Platform Image <p>See https://docs.oracle.com/iaas/Content/Compute/References/images.htm</p>	<p>Images</p> <ul style="list-style-type: none"> Oracle Autonomous Linux 7 Bring Your Own (BYOI) Images <p>See https://docs.oracle.com/iaas/Content/Compute/References/bringyourownimage.htm.</p> <ul style="list-style-type: none"> Oracle Cloud Infrastructure Marketplace images (Oracle Linux KVM, Oracle Cloud Developer, and so on.)

Supported with Leapp	Unsupported with Leapp
<p>Shapes</p> <ul style="list-style-type: none"> All Flexible Shapes See https://docs.oracle.com/iaas/Content/Compute/References/computeshapes.htm#flexible. All Virtual Machine Shapes See https://docs.oracle.com/iaas/Content/Compute/References/computeshapes.htm#vmshapes. <p>Features</p> <ul style="list-style-type: none"> Instances that are being managed with the Oracle OS Management Service See https://docs.oracle.com/iaas/os-management/osms/osms-getstarted.htm. 	<p>Shapes</p> <ul style="list-style-type: none"> Bare Metal Shapes (Standard, Dense I/O, GPU, HPC) See, https://docs.oracle.com/iaas/Content/Compute/References/computeshapes.htm#baremetalsshapes.. <p>Features</p> <ul style="list-style-type: none"> Any unupgradeable features listed in the table in Upgrading Oracle Linux 7 Systems. Instances that are being managed with the Oracle OS Management Hub service

Upgrading Oracle Linux 7 Systems That Use RDMA

The following table lists the scope of Leapp support for upgrading Oracle Linux 7 systems and instances that use Remote Direct Memory Access (RDMA) :

Supported with With Leapp	Unsupported with Leapp
<ul style="list-style-type: none"> Oracle Linux 7 with UEKR5 Oracle RDMA (requires a kernel upgrade to UEKR6) Oracle Linux 7 with UEKR6 Oracle RDMA 	<ul style="list-style-type: none"> Oracle Linux 7 with UEKR3 OFED 2.0 Oracle Linux 7 with UEKR4 OFED

For details and instructions on obtaining newer versions of RDMA packages, see the latest UEK release notes at [Unbreakable Enterprise Kernel documentation](#).

Upgrading Oracle Linux 7 KVM Hosts

The Leapp utility can be used to upgrade Oracle Linux 7 systems that host KVM virtual machines. Systems must fulfill the other Leapp criteria listed in the previous sections. The following table lists the scope of KVM host support.



Note:

The Oracle Linux KVM Image isn't an Oracle Cloud Infrastructureplatform image and not supported by Leapp.

Supported with Leapp	Unsupported with Leapp
<ul style="list-style-type: none"> Upgrading the Oracle Linux 7 Latest KVM packages to the Oracle Linux 8 KVM AppStream Upgrading the Oracle Linux 7 KVM Utilities to the Oracle Linux 8 KVM AppStream 	<ul style="list-style-type: none"> Upgrading from the Oracle Linux 7 Latest packages to the Oracle Linux 8 KVM AppStream packages Upgrading from the Oracle Linux 7 KVM Utilities to the Oracle Linux 8 AppStream packages Upgrading KVM hosts while KVM virtual machines (guests) are running.

For repository mappings between preupgrade stage and postupgrade stage that involve KVM clients, see [Supported Repositories in Leapp Upgrades](#).

Upgrading Oracle Linux 7 Systems with Oracle Databases

The Leapp utility can be used to upgrade Oracle Linux 7 systems that host Oracle Database both single host or in a Real Application Clusters (RAC) configuration across multiple hosts. Systems must fulfill the other Leapp criteria listed in the previous sections. The following table lists the scope of Oracle Database support.

Supported with Leapp	Unsupported with Leapp
<ul style="list-style-type: none"> Upgrading Oracle Linux 7 running Oracle Database 19c to Oracle Linux 8 on x86_64 Upgrading Oracle Database 12c to 19c on Oracle Linux 7 before upgrading to Oracle Linux 8 on x86_64 Upgrading Oracle Database 18c to 19c on Oracle Linux 7 before upgrading to Oracle Linux 8 on x86_64 	<ul style="list-style-type: none"> Upgrading from Oracle Database 21c on Oracle Linux 7 to Oracle Database 21c Oracle Linux 8 Upgrading directly from Oracle Database 12c on Oracle Linux 7 to Oracle Database 19c Oracle Linux 8 Upgrading directly from Oracle Database 19c on Oracle Linux 7 to Oracle Database 21c Oracle Linux 8 Upgrading Oracle Linux 7 with an Oracle Databases on a Bare Metal Oracle Cloud Infrastructure to Oracle Linux 8 Upgrading Oracle Linux 7 running any instant client to Oracle Linux 8

Requirements for Upgrading

To upgrade an Oracle Linux 7 system or instance, ensure that either one meets the following requirements:

- The minimum installation requirements as listed in System Requirements in [Oracle Linux 8: Installing Oracle Linux](#) are met.

In particular, ensure that the system has disk space to complete the Leapp upgrade. Disk space in the `/boot` partition is especially paramount. The partition must have at least 250 MB of disk space to accommodate the installation of the Red Hat Compatible Kernel (RHCK) and Unbreakable Enterprise Kernel (UEK), `initramfs`, `kdump` images, and so on. Examine the preupgrade report which might notify you if insufficient disk space is detected. For more information about the preupgrade phase, see [Assessing the Capability of the System for Upgrading](#).

- Only packages provided by Oracle are installed. Upgrade stability isn't guaranteed if third-party packages are present in the system.
- Oracle Linux yum server at <https://yum.oracle.com> or a corresponding yum mirror is accessible.

If accessing repositories from a mirror or a local repository, ensure that both Oracle Linux7 and Oracle Linux 8 channels are mirrored.

- x86_64 deployments are running Unbreakable Enterprise Kernel Release 5 or later versions or the Red Hat Compatible Kernel (RHCK).
- aarch64 deployments are running the Unbreakable Enterprise Kernel Release 6.

Check the following references for information that might have an impact on the upgrade process:

- [Oracle Linux 8: Release Notes for Oracle Linux 8.6](#)
- [Known Issues](#)

Kernels Upgradeable With Leapp

The following table provides guidance about which kernel upgrades can be performed with the Leapp utility. The table assumes that the Oracle Linux 7 host satisfies the requirements listed in [Requirements for Upgrading](#).

	Starting Kernel (Oracle Linux 7)	Ending Kernel (Oracle Linux 8)	Supported
x86_64 not using Btrfs file system	RHCK	RHCK	Yes ¹
	RHCK	UEK	No
	UEK	UEK	Yes
	UEK	RHCK	No
x86_64 using Btrfs file system	RHCK	RHCK	No ²
	RHCK	UEK	Yes
	UEK	UEK	Yes
	UEK	RHCK	No ²
aarch64	UEK	UEK	Yes ³

¹Unbreakable Enterprise Kernel Release 6 remains on the system or instance after the upgrade. If preferred, the administrator can remove this kernel.

²RHCK in Oracle Linux 8 doesn't support the Btrfs file system.

³RHCK isn't distributed nor available for the aarch64 platform.

2

Preparing for the Upgrade

Complete the steps as applicable to prepare for an upgrade from Oracle Linux 7 to Oracle Linux 8. Unless specified otherwise, all of the procedures for upgrading an Oracle Linux 7 system also apply upgrading an Oracle Linux 7 instance on Oracle Cloud Infrastructure.

1. Set up a means to connect remotely through a console.

This document assumes that you're performing a Leapp upgrade remotely. In this case, a console is necessary so you can monitor the progress of the upgrade process, especially as the upgrade performs automatic reboots.

The following list shows console connection options you can use:

- Oracle Cloud Infrastructure instance: Create a console connection by following the instructions at https://docs.oracle.com/iaas/Content/Compute/References/serialconsole.htm#Instance_Console_Connections.
- Oracle Linux server: Use Oracle Integrated Lights Out Manager (ILOM). See <https://docs.oracle.com/en/servers/management/ilom/index.html>.
- Oracle Private Cloud Appliance: Use the Instance Console Connection. See <https://docs.oracle.com/en/engineered-systems/private-cloud-appliance/index.html>.
- Oracle Linux Virtualization Manager or Oracle Linux Kernel based Virtual Machines (KVM): User `virt-viewer`, `virt-manager`, or Cockpit Web Console. See [Oracle Linux Virtualization Manager documentation](#).

Note:

If you connect to the system by using SSH or by using VNC to a VNC service running on the system, you're disconnected during the upgrade process and are unable to log in until the upgrade is completed.

2. If you are upgrading an Oracle Linux instance on Oracle Cloud Infrastructure, verify if Oracle OS Management Service is running on the instance. Do the following:
 - a. From Oracle Cloud Infrastructure, open the navigation menu and click **Compute**. Under Compute, click **Instances**.
 - b. Select the instance you want to upgrade from Oracle Linux 7 to Oracle Linux 8.
 - c. From the Resources section, click **OS Management**.
 - If the Oracle OS Management Service description specifies "No OS management information is available for this resource," then the Oracle OS Management Service agent is not running on the instance.
 - If the description provides other information about the instance, then the Oracle OS Management Service agent is running on the instance.
3. Do one of the following:
 - If the instance is managed by Oracle OS Management Service:
 - a. From the Resources section, click **OS Management**.

- b. From the Overview tab, click the horizontal ellipsis (...) and select **View OS Management Details**.
- c. From the Resources area, select **Software Sources**.
- d. Click **Add** and select the check box next to **Leapp Upgrade Utilities for Oracle Linux 7 Server**.
- e. Click **Add**.
- If the instance is not managed by Oracle OS Management Service, ensure that the Oracle OS Management Service agent is disabled:
 - a. Select the **Oracle Cloud Agent** tab.
 - b. Disable the **OS Management Service Agent** process if it is enabled. This process takes awhile to complete.
 - c. After waiting sufficiently, check for osms-agent processes and ensure that none are running. Connect to the process with a terminal, and run the following command:

```
ps -ef | grep osms-agent
```

For more information, refer to <https://docs.oracle.com/en-us/iaas/Content/Compute/Tasks/manage-plugins.htm#disable-one-plugin>.

Note:

An Oracle Linux system may run the Oracle OS Management Service agent, but may not be managed by the Oracle OS Management Service. For more information about Oracle OS Management Service, see <https://docs.oracle.com/iaas/os-management/osms/osms-getstarted.htm>.

4. Perform a backup.

Always back up a system so that the system can be restored to its former state if the upgrade fails. Oracle highly recommends making a full backup of your database, if your instance of Oracle Linux includes a supported database version. For more information, see [Upgrading Oracle Linux 7 Systems with Oracle Databases](#).

Note:

For an Oracle Linux 7 instance in Oracle Cloud Infrastructure, perform a boot volume backup. For instructions, see <https://docs.oracle.com/iaas/Content/Block/Concepts/bootvolumebackups.htm>.

- 5. Shut down all production workloads that have been set up to run on the system, as the upgrade is intrusive and requires several reboots.
- 6. Disable Secure Boot if it's running.

To check the status of Secure Boot, choose from one of the following commands:

- Using `bootctl status`, for example:

```
sudo bootctl status
```

```
...
Secure Boot: disabled
...
```

- Using `mokutil --sb-state`, for example:

```
sudo mokutil --sb-state

SecureBoot disabled
```

If Secure Boot is enabled, you would need to access the system's firmware at boot time to disable the option.

Note:

It is not possible to disable Secure Boot if it is already enabled on an Oracle Cloud Infrastructure instance.

7. If the system has network mounted file systems, unmount them and then insert related entries in the `/etc/fstab` file inside comment marks.

See [File Systems and Storage Issues](#).

8. If the system is behind a proxy, configure the proxy settings in `/etc/yum.conf`, for example:

```
proxy=proxy-url:port
```

See Yum Configuration in [Oracle Linux 7: Managing Software](#).

9. If you installed the `yum-plugin-versionlock` package, clear any packages with locked versions.

```
sudo yum versionlock clear
```

10. Obtain the latest Oracle Linux 7 packages.

```
sudo yum update -y
```

Note:

After the update is completed, the following message might be displayed:

```
warning: /etc/yum.repos.d/oracle-linux-ol7.repo
created as /etc/yum.repos.d/oracle-linux-ol7.repo.rpmnew
```

This warning appears if an `oracle-linux-ol7.repo` file already exists before updating the Oracle Linux 7 packages. The update process creates the `.rpmnew` file to avoid overwriting any customizations that might be in the current file.

In this case, use the `.rpmnew` file to guide you in making the necessary modifications to your existing `.repo` file. Incorporate any new information into your `.repo` file. The `ol7_leappol8_leapp` repository description **must** be listed in your final `oracle-linux-ol7.repo` file for the upgrade to proceed.

11. If you're upgrading Oracle Linux 7 KVM hosts, stop all the virtual machines that might be running.

The command lists the virtual machines. From the list, stop specific virtual machines that are running.

- a. List the available virtual machines.

```
sudo virsh list --all
```

- b. From the list, stop individual virtual machines that are running.

```
sudo virsh shutdown vm-name
```

12. If the system is registered with ULN or a ULN mirror, unregister the system.

See the following documentation for this step.

- Removing a System From ULN in [Oracle Linux: Unbreakable Linux Network User's Guide for Oracle Linux 6 and Oracle Linux 7](#)
- Checking Yum Configuration in <https://yum.oracle.com/getting-started.html#checking-yum-configuration>.

13. Reboot the system.

```
sudo reboot
```

14. Install the Leapp utility while enabling certain repositories, using the following command:

```
sudo yum install -y leapp-upgrade --enablerepo=ol7_leapp,ol7_latest
```

3

Upgrading the System

This chapter discusses the different stages of a system upgrade, which are the assessment phase and the upgrade phase. The main commands to use for these stages are `leapp preupgrade` and `leapp upgrade`, and followed by command arguments. For a list of these arguments, use the `-help` or `--help` argument, for example:

```
sudo leapp preupgrade --help
```

Unless specified otherwise, all the procedures for upgrading an Oracle Linux 7 system also apply to upgrading an Oracle Linux 7 instance on Oracle Cloud Infrastructure.

Assessing the Capability of the System for Upgrading

The `preupgrade` phase checks whether the system is fully ready for the upgrade.

! Important:

Refer also to [Known Issues](#) to better prepare the system for a Leapp upgrade.

Running the Preupgrade

Through the `preupgrade` phase, you can check whether the system is fully ready for the upgrade.

Running the `preupgrade` phase is recommended to ensure that the system is cleared of issues that might impede the upgrade. In this phase, you generate an assessment report that identifies risks to upgrading. The report also provides recommendations for resolving those risks.

1. Grant root SSH login permissions in the `/etc/ssh/sshd_config` file.

```
PermitRootLogin yes
```

2. If you're using a proxy server, edit the `/etc/yum.repos.d/leapp-upgrade-repos-ol8.repo` by adding the proxy setting for each repository entry.

To add the setting in a single operation, you can run the following command:

```
sudo sed -i '/^enabled=0.*/a proxy=http://proxy-host:proxy-port' /etc/yum.repos.d/leapp-upgrade-repos-ol8.repo
```

3. Run the `preupgrade` command.

Use the appropriate command argument for a system or an Oracle Cloud Infrastructure instance.

- On a system:

```
sudo leapp preupgrade --oraclelinux [--enablerepo repository]
```


- On an instance in Oracle Cloud Infrastructure that is not running the Oracle OS Management Service agent:

```
sudo leapp preupgrade --oci [--enablerepo repository]
```

For detailed information about the arguments, see [Using Command Arguments to Enable Repositories](#).

This process generates a process log, a report, and a file called `answerfile`.

Analyzing the Leapp Report

The `/var/log/leapp/leapp-report.txt` identifies potential risks to the upgrade. The risks are classified as high, medium, or low. A high risk that would prevent an upgrade is further classified as an inhibitor. The report summarizes the issues behind the identified risk and also suggests remediations if any are needed.

Ensure that you complete the recommended remedies to clear risks that are labeled high and can inhibit the upgrade process.

After addressing the reported risks, run the `preupgrade` command again. In the regenerated report, verify that all serious risks are cleared.

To better illustrate the contents of the report, consider the following examples:

Python Version Issue

The Leapp report might post the following:

```
Risk Factor: high
Title: Difference in Python versions and support in OL 8
Summary: In OL 8, there is no 'python' command. Python 3 (backward incompatible)
        is the primary Python version and Python 2 is available with limited support an
        d limited set of packages. Read more here: https://docs.oracle.com/en/operating-
        systems/oracle-linux/8/python/
Remediation: [hint] Please run "alternatives --set python /usr/bin/python3" after upgrade
```

As the example shows, for some risks, the report would suggest actions you should perform *after* the upgrade. Therefore, the risk, although high, is not labeled as an inhibitor. The remedy can be performed later.

GPG Key Issue

The report might warn about the `gpg-pubkey`.

```
Risk Factor: high
Title: Packages not signed by Oracle found on the system
Summary: The following packages have not been signed by Oracle and may be
        removed during the upgrade process in case Oracle-signed packages to be removed
        during the upgrade depend on them:
- gpg-pubkey
```

To resolve this issue, run the following command:

```
sudo rpm -qa | grep gpg-pubkey
```

If the command output lists only the Oracle Linux 7 public key `gpg-pubkey-ec551f03-53619141`, the issue can be safely ignored. Otherwise, any other unsigned packages or `gpg-pubkey` entries in the report must be manually analyzed, as they might be removed during the upgrade.

Providing Information to the Leapp Answerfile

In addition to completing the recommendations of `/var/log/leapp/leapp-report.txt`, you must also provide answers to all the items in `/var/log/leapp/answerfile`.

An inhibitor might be reported both in `/var/log/leapp/answerfile` and `/var/log/leapp/leapp-report.txt`, with the latter file providing an alternative remedy. Despite overlapping contents, always examine both files to ensure a successful upgrade.

The `/var/log/leapp/answerfile` file consists of specific verification checks that Leapp performs on the system. A verification check contains information about the system and also prompts you for confirmation on the action to be performed. The file provides context and information to help guide you on the response required.

Note:

All verification checks listed in the `answerfile` must be answered. Unanswered items cause the upgrade process to halt.

The following is a sample entry from `/var/log/leapp/answerfile`:

```
[remove_pam_pkcs11_module_check]
# Title:                None
# Reason:               Confirmation
# ===== remove_pam_pkcs11_module_check.confirm =====
# Label:                Disable pam_pkcs11 module in PAM configuration? If no, the
upgrade
                        process will be interrupted.
# Description:          PAM module pam_pkcs11 is no longer available in RHEL-8 since it
was
                        replaced by SSSD.
# Type:                 bool
# Default:              None
# Available choices:    True/False
# Unanswered question. Uncomment the following line with your answer
# confirm =
```

Based on the example, each verification check is identified with a section heading in square brackets, such as `remove_pam_pkcs11_module_check`. The heading is followed by descriptions of the issue and the valid responses to address the issue.

To provide responses to `answerfile`, choose from one of the following methods:

- Use the `leapp answer` command.
Run this command on the specific section that needs correcting. For example, to confirm the PAM module verification, you would type:

```
sudo leapp answer --section remove_pam_pkcs11_module_check.confirm=True
```

- Edit the contents of `/var/log/leapp/answerfile`.

Go to the specific section that you want to confirm, such as `[remove_pam_pkcs11_module_check]`, uncomment its `confirm =` line and specify the answer, for example:

```
confirm = True
```

 **Note:**

Systems that use the Btrfs file system must also confirm in `/var/log/leapp/answerfile` the switch to using UEK. As noted in [Kernels Upgradeable With Leapp](#), using the Btrfs file system in Oracle Linux 8 requires the UEK kernel. To confirm the kernel upgrade, issue this command to update `answerfile`:

```
sudo leapp answer --section
confirm_UEK_install_and_default_boot_kernel.confirm=True
```

Alternatively, you can edit the specific section in `answerfile`.

Performing the Upgrade

After you have properly completed the `/var/log/leapp/answerfile` and verified that `/var/log/leapp/leapp-report.txt` no longer reports risks, upgrade the system as follows:

1. Using a console, connect to the system or the Oracle Cloud Infrastructure instance that you're upgrading.
 - If you're upgrading a remote system configured with a VNC server, connect to the system by using a VNC client.
 - If you're working on an Oracle Cloud Infrastructure instance, connect to the instance through the console connection you previously created in [Preparing for the Upgrade](#). For instructions, see [Connecting to the Serial Console in https://docs.oracle.com/iaas/Content/Compute/References/serialconsole.htm#Instance_Console_Connections](https://docs.oracle.com/iaas/Content/Compute/References/serialconsole.htm#Instance_Console_Connections).

For example, on a local terminal window, the command that's provided to connect to the instance might resemble the following syntax:

```
ssh -o ProxyCommand='ssh additional-commands
```

If the command doesn't work at first use, you might need to specify the `-i path-to-key` option, for example:

```
ssh -i path-to-key -o ProxyCommand='ssh -i path-to-key additional-commands
```

2. On a separate terminal window of the system or instance to be upgraded, run the `upgrade` command with the appropriate command argument, depending on whether you're upgrading a system or an Oracle Cloud Infrastructure instance.

- On a system:

```
sudo leapp upgrade --oraclelinux [--enablerepo repository]
```

- On an instance in Oracle Cloud Infrastructure that is not running Oracle OS Management Service agent:

```
sudo leapp upgrade --oci [--enablerepo repository]
```

For detailed information about the command arguments, see [Using Command Arguments to Enable Repositories](#).

3. At the end of the upgrade, reboot the system.

```
sudo reboot
```

4. While the system reboots, monitor the progress on the console.

At the completion of the boot process, the utility automatically proceeds with upgrading packages. This operation takes awhile to complete and also includes multiple automatic reboots.

⚠ Caution:

Do *not* interrupt the ongoing processes at this stage. Wait until the login screen appears, which indicates that the entire upgrade process has completed. Only then can you begin to use the system.

5. When the login screen appears on the console, log in with the proper credentials.

After the completion of an instance upgrade, the instance retains its Oracle Linux 7 base image on the Instance Details page of the Oracle Cloud Infrastructure console, for example, `Oracle-Linux-7.9-2020-11.10-1`. You can apply a custom tag so you can track the upgrades that have been performed on the instance after its creation.

! Important:

See [Oracle Linux 8 documentation](#) for information about new features, changes, and deprecated items in Oracle Linux 8. Thus, you can identify post upgrade tasks that you might need to complete. For example, after the upgrade, network configurations and settings from Oracle Linux 7 continue to operate based on legacy network scripts. However, network scripts are deprecated in Oracle Linux 8. Therefore, consider reconfiguring the upgraded system's network settings to be managed by `NetworkManager`. For more information, see [Oracle Linux 8: Setting Up Networking](#).

Verifying the Upgrade

Upon completion, the upgrade process generates the same files as the preupgrade phase: a process log, a report, and the `/var/log/leapp/answerfile`. On the console, perform the following steps:

1. Examine the `/var/log/leapp/leapp-report.txt` and fulfill any important recommendations to be completed after the upgrade process.
2. Perform the following verifications:

To verify the system's new OS version, type:

```
cat /etc/oracle-release
```

To check the system's kernel version, type this command to verify that the kernel contains the `e18` substring:

```
uname -r
```

You can also identify the system's default kernel with the following command:

```
sudo grubby --default-kernel
```

4

Completing Postupgrade Tasks

! Important:

The following tasks aren't comprehensive. Depending on the setup, you might need to perform other procedures to return the newly upgraded system back into operation. Review the `/var/log/leapp/leapp-report.txt` that's generated after the upgrade. This report might contain more recommendations to ensure that the upgraded system remains in a supported state.

1. Configured the python version.

```
sudo alternatives --set python /usr/bin/python3
```

2. Enable the firewall.

```
sudo systemctl start firewalld
sudo systemctl enable firewalld
```

3. Check that the network connections are operational, for example, by logging in to the system by using SSH.
4. If you have an instance managed by Oracle OS Management Service, do the following:
 - a. From Oracle Cloud Infrastructure, open the navigation menu and click **Compute**. Under Compute, click **Instances**.
 - b. Select the instance you upgraded from Oracle Linux 7 to Oracle Linux 8.
 - c. From the Resources section, click **OS Management**.
 - d. From the Overview tab, click the horizontal ellipsis (...) and select **View OS Management Details**.
 - e. From the Resources area, select **Software Sources**.
 - f. Remove all Oracle Linux 7 sources. Select **Select All** then click **Remove**.
 - g. Add the Oracle Linux 8 Base OS Latest software source. This becomes the software source for the upgraded managed instance.
 - h. Add any other required software sources.

 **Note:**

The Oracle OS Management Service can take some time to become aware of the changes to the managed instance Oracle Linux version. For example, in some cases this may take between 10 to 20 minutes. Additionally, the following error messages may appear in the `/var/log/oracle-cloud-agent/plugins/osms/agent.log` after upgrading an instance being managed by the Oracle OS Management Service.

```
2024-03-26 11:42:54,555 - INFO - gRPC Start: OSMS Agent service
started
2024-03-26 11:42:54,569 - ERROR - Error checking actions (next
check in 240 seconds): [Errno 2] No such file or directory:
'osms': 'osms'
```

These error messages are resolved after running the `sudo alternatives --set python /usr/bin/python3` command in step 1.

5. If you had `yum` customizations before the upgrade, restore them in the upgraded system's `/etc/dnf/dnf.conf` file, for example:

```
proxy=proxy-url:port
```

6. On upgraded Oracle Linux 7 instances on Oracle Cloud Infrastructure, update the script to avoid losing SSH connectivity.
 - a. Use a text editor to open a new `/etc/dhcp/exit-hooks.d/dhclient-exit-hook-set-hostname.sh-ol8` file.
 - b. Add to this script the contents as provided in [Updated dhclient Script](#).
 - c. Replace the current `dhclient-exit-hook` script with the updated file that you just created. Type:

```
sudo cp /etc/dhcp/exit-hooks.d/dhclient-exit-hook-set-hostname.sh-ol8 /etc/dhcp/
exit-hooks.d/dhclient-exit-hook-set-hostname.sh
```

7. Restore network mounted file systems that you unmounted prior to the upgrade. See [File Systems and Storage Issues](#).
8. Confirm the Python installation.

Significant differences exist in how Python is used in Oracle Linux 8. Notably, you need to explicitly select the Python version that you're using. Further, if you have installed third-party libraries by using `pip`, you might need to manually clean and reinstall these libraries as required. For more information, see [Oracle Linux 8: Installing and Managing Python](#).

9. If upgrading KVM hosts, restart the KVM virtual machines.

```
sudo virsh start vm-name
```

10. Set SELinux to run in Enforcing mode.

During the upgrade, the Leapp utility sets SELinux to run in `Permissive` mode. To restore the setting: To revert to `Enforcing` mode and verify the setting, type:

```
sudo setenforce enforcing
```

You can verify the mode of SELinux as follows:

```
getenforce
```

Enforcing

To make this setting persist across system reboots, add the following line to `/etc/selinux/config`:

```
SELINUX=enforcing
```

11. Reevaluate then reapply the security policies such as setting cryptographic policies.
If you disabled Secure Boot during the preparation steps, reenable it in the system's firmware that you access at boot time.
12. Inspect the system for unneeded configurations and files.

Note:

Some of these unneeded files might be reported in the generated `/var/log/leapp/leapp-report.txt` after the upgrade. Ensure that you review this report and complete its post upgrade recommendations.

This step aims to ensure that the configurations are consistent with the new OS version. The completion of this step would vary, depending on what you deem is important to retain from the previous system's state. Consider the following guidelines:

- Remove kernels and kernel modules that are no longer applicable. For example, if the system uses the Btrfs file system, then you can only use the UEK kernel. Therefore, consider removing the RHCK kernel and any earlier versions of the UEK kernel. Also, you can also rebuild the rescue kernel.
- If you remove kernels, you might also need to update the GRUB menu so that the menu options only reflect the actual kernels on the system.
- Review `/etc/yum.repos.d` for entries that might need to be addressed, such as customized repositories.

For example, during system updates, `*.rpmnew` files might be created to prevent overwriting corresponding existing `*.rpm` files. You would need to use the contents of the `*.rpmnew` files to guide you when modifying the corresponding `*.rpm` files.

- Remove residual packages from the previous Oracle Linux version.
 - a. Edit `/etc/dnf/dnf.conf` by removing or commenting out `exclude=` lines that refer to leapp packages, for example:


```
#exclude=python2-leapp,snactor,leapp-upgrade-el7toel8,leapp
```
 - b. Use commands such as `rpm -qa` to list packages that can be removed.


```
rpm -qa | grep el7
rpm -qa | grep leapp
```
 - c. Use the `sudo dnf remove` command to remove the packages listed by the queries.

Caution:

Residual `el7` packages that remain on the system do not receive updates. Vulnerability scanners or other security audits might report warnings or failures about these packages.

13. Remove the `/root/tmp_leapp_py3` directory, which is no longer needed.
14. If you removed the system from ULN to perform the upgrade, register the system again and configure the appropriate channels.

For more information, see Registering an Oracle Linux System With ULN and ULN Channel Subscription Management in [Oracle Linux: Managing Software on Oracle Linux](#).

5

Troubleshooting Oracle Linux Upgrades

This chapter provides troubleshooting information and describes known issues that might affect the upgrade process.

Tools for Troubleshooting

Use the following options to generate more output when you are generating the preupgrade report or performing the actual upgrade:

- `--verbose` displays warnings, error messages, and other critical information.
- `--debug` adds debug information in addition to the same output as the `--verbose` option.

You can use the following resources and tools for obtaining troubleshooting information:

- `/var/log/leapp/leapp-report.txt`
- `/var/log/leapp/leapp-upgrade.log`
- `/var/log/leapp/dnf-debugdata/`: a directory for debug information. Note that this directory is created only if you use the `--debug` option when issuing either the `preupgrade` or the `upgrade` command.
- `journalctl` command

Known Issues

The following are known issues that you might encounter when upgrading an Oracle Linux 7 system to Oracle Linux 8.

Upgrade Issues

- **Optional Resilient Storage group isn't supported with Leapp**

Leapp doesn't support the optional Resilient Storage group. The presence in the system of certain packages from that group might prevent a leapp upgrade to complete.

For example, the Resilient Storage group includes the `lvm2-cluster` package. In an Oracle Linux 7 installation, this package can be optionally added as part of either the Infrastructure Server and File profile or the Print Server profile. However, the package is an inhibitor and would cause the Leapp upgrade to fail.

BugID 33573562

- **Leapp might report missing packages that are marked for installation**

The `/var/log/leapp-preupgrade.log` or `/var/log/leapp-upgrade.log` files might report a warning similar to the following:

```
Warning: Packages marked by Leapp for install not found in repositories
metadata: rpcgen python3-pyxattr libnsl2-devel rpcsvc-proto-devel
```

These packages are in the Oracle Linux8 Codeready Builder repository, which is a developer repository and is disabled by default.

If the system requires these packages, then during the preupgrade or the upgrade phase, add the `--enablerepo ol8_codeready_builder` option to the appropriate Leapp command, for example:

```
sudo leapp upgrade --oraclelinux --enablerepo ol8_codeready_builder
```

Repositories that have been enabled during the Leapp upgrade remain enabled on the Oracle Linux 8 system after the upgrade completes.

Alternatively, after completing the upgrade, you can manually install the packages required for your installation by using the `dnf` command.

Bug ID 32827043

- **MySQL-related *.e17 packages might remain after an upgrade**

In an Oracle Cloud Infrastructure instance, existing MySQL packages might remain after the upgrade is completed. You can verify their existence with the following command:

```
rpm -qa | grep e17 | grep -v leapp | grep -v kernel  
  
mysql-community-client-plugins-8.0.25-1.e17.x86_64
```

To ensure that these packages are correctly updated, enable the `ol8_MySQL80` repository also when you run the upgrade.

```
sudo leapp upgrade --oci --enablerepo ol8_mysql80
```

- **Some e17 packages might not be upgraded**

The same `rpm -qa` command syntax in the previous item that detects MySQL-related *.e17 packages might also list more *.e17 packages on the system that weren't upgraded. Packages might not be upgraded if they were installed from repositories that aren't supported by Leapp, such as developer repositories. For such packages, do the following:

1. Go to <https://yum.oracle.com> and check the Oracle Linux 8 repositories that would serve the packages you need.
2. After the upgrade is completed, manually install the packages from those Oracle Linux 8 repositories.
3. After all the necessary packages have been installed, remove the residual e17 packages from the system.

Bug ID 32878386

- **(aarch64) Upgrade log might report errors related to the vmd module**

After completing an upgrade on aarch64 systems, the Leapp upgrade log might report the following message:

```
dracut-install: Failed to find module 'vmd'
```

The VMD module doesn't apply to the Arm architecture and therefore, the error message can be safely ignored.

Bug ID 34172552

Security and Authentication Issues

- **TCP Wrappers not used in Oracle Linux 8**

TCP Wrappers aren't available in Oracle Linux 8. If you're using TCP Wrappers to control network traffic in the Oracle Linux 7 system through `/etc/hosts.deny` and `/etc/hosts.allow` files, then after the upgrade you must create applicable firewall rules to implement the same control in Oracle Linux 8.

For more information about implementing firewall rules, see [Oracle Linux 8: Configuring the Firewall](#).

- **Oracle Linux 7 authentication configuration not converted after upgrade**

Leapp does not convert Oracle Linux 7 authentication configuration that uses the deprecated `authconfig` utility to the corresponding Oracle Linux 8 authentication that uses the `authselect` command. After the upgrade, you need to reconfigure authentication appropriately in Oracle Linux 8.

For more information about using the `authselect` utility, see [Oracle Linux 8: Setting Up System Users and Authentication](#).

- **Upgrade blocked if system has configured `pam_krb5` and `pam_pkcs11` PAM modules**

The deprecated `pam_krb5` and `pam_pkcs11` PAM modules in Oracle Linux 7 are removed during the upgrade process. Before performing the upgrade, you must first reconfigure the system's PAM configuration to disable these modules. Otherwise, the system becomes locked.

For instructions to disable these modules, see [Oracle Linux 7: Setting Up System Accounts and Authentication](#). See also the discussion about these modules in [Providing Information to the Leapp Answerfile](#).

System Management Issues

- **Ksplice Uptrack software displays error messages**

During the upgrade, the Oracle Ksplice Uptrack software might report errors similar to the following:

```
[ 256.033527] upgrade[390]: Upgrading : uptrack-1.2.74-0.el8.noarch 577/1453
[ 256.037151] upgrade[390]: Running scriptlet: uptrack-1.2.74-0.el8.noarch 577/1453
...
[ 256.045914] upgrade[390]: Traceback (most recent call last):
[ 256.049230] upgrade[390]: File "/usr/lib/uptrack/access-key-from-uls", line 9, in
<module>
[ 256.051376] upgrade[390]: from up2date_client import up2dateAuth, rpcServer
[ 256.056490] upgrade[390]: File "/usr/share/rhn/up2date_client/up2dateAuth.py",
line 74
[ 256.059251] upgrade[390]: os.chmod(path, 0600)
[ 256.060997] upgrade[390]:
[ 256.062842] upgrade[390]: SyntaxError: invalid token
```

The report is a known but harmless issue, which you can ignore. After the upgrade is completed, Ksplice continues to operate normally.

File Systems and Storage Issues

- **Systems with Btrfs in a RAID configuration can't be upgraded**

A system that uses the Btrfs file system in a RAID configuration can't be upgraded. In the `/var/log/leapp/leapp-report.txt` that's generated by the `preupgrade` command, this configuration is flagged as an inhibitor and no remedy is provided. If you upgrade the system and that configuration is detected, the upgrade process halts.

- **Upgrade blocked if `winbind` and `wins` Samba modules are used**

If `winbind` and `wins` Samba modules are used in the `/etc/nsswitch.conf`, the upgrade is blocked. As a workaround, remove these modules from the file first, then perform the upgrade. After the upgrade is complete, restore these module entries to the file.

For more information about configuring these modules, see [Oracle Linux 8: Managing Shared File Systems](#).

- **Hosts with network mounted file systems can't be upgraded**

Leapp doesn't support upgrading systems with mounted file systems on network storage, NFS, or iSCSI. As a workaround, unmount the file systems and comment out their entries from `/etc/fstab`. After the upgrade is completed, you can restore the entries and remount the file systems.

Networking Issues

- **Possible upgrade error if system has several NICs with the same prefix as NIC that's used by kernel**

The in-place upgrade process might cause an error if the system to be upgraded has more than one NIC that shares the same prefix as the NIC that's used by the kernel, for example `eth`. After the upgrade, the system's network connectivity is lost.

For more information, see About Network Interface Names in [Oracle Linux 8: Setting Up Networking](#).

- **NetworkManager might not start after the upgrade completes**

After the upgrade, the system's `NetworkManager` might not start because of the failure of its name resolution service. The failure can be verified by checking the status of the service.

```
systemctl status systemd-resolved.service
```

```
• systemd-resolved.service - Network Name Resolution
  Loaded: loaded (/usr/lib/systemd/system/systemd-resolved.service;
disabled; >
  Active: inactive (dead)
  Docs: man:systemd-resolved.service(8)
        https://www.freedesktop.org/wiki/Software/systemd/resolved
```

The `/var/log/messages` file also reports the following error:

```
dbus-daemon[742]: [system] Activation via systemd failed for unit
'dbus-org.freedesktop.resolve1.service': Unit
dbus-org.freedesktop.resolve1.service not found.
```

To resolve this issue, choose one of the following workarounds:

- **Configure `NetworkManager` to not use `systemd-resolved.service`.**

Add the following entries to the `/etc/NetworkManager/conf.d/no-systemd-resolved.conf` file:

```
[main]
systemd-resolved=false
```

- **Enable the `systemd-resolved.service` as follows:**

```
systemctl enable systemd-resolved.service
```

```
Created symlink /etc/systemd/system/dbus-org.freedesktop.resolve1.service →
/usr/lib/systemd/system/systemd-resolved.service.
```

```
Created symlink
/etc/systemd/system/multi-user.target.wants/systemd-resolved.service →
/usr/lib/systemd/system/systemd-resolved.service.

systemctl start systemd-resolved.service
```

You can also adopt other methods that are more consistent with the network name resolution model that you're using for the specific setup. For useful information, see [About Network Interface Names in Oracle Linux 8: Setting Up Networking](#).

Virtualization and Containers Issues

- **Docker removed during upgrade process**

Docker is removed as part of the upgrade process. If you intend to run containers on the upgraded system, be prepared to migrate the container infrastructure to Podman.

For more information, see [Oracle Linux: Podman User's Guide](#).

- **KVM virtual machine snapshots might not be listed after an upgrade**

After an upgrade, the `libvirtd` service might report snapshot-related error messages similar to the following:

```
libvirtd[53328]: internal error: Failed to parse snapshot XML from file
'/var/lib/libvirt/qemu/snapshot/path-to-previous-snapshot-file'
```

Furthermore, listing available snapshots from prior to the upgrade generates an empty list.

```
sudo virsh snapshot-list previous-snapshot-file
```

```

Name      Creation Time    State
-----

```

As a workaround, reboot the system. At the end of the boot process, the snapshots are listed and available again.

- **libvirtd service might fail to restart in nested virtualization configurations**

In nested virtualization setups, the `libvirtd` service might not restart in the nested KVM host after the upgrade.

As a workaround, reboot the nested KVM host.

- **Some KVM virtual machines might not start after the upgrade**

If the KVM host is running the RHCK kernel, virtual machines or guests that existed before the upgrade might not start after the Leapp upgrade completes. When start these guests, an error message similar to the following might be displayed:

```
sudo virsh start domain-name

error: Failed to start domain domain-name
error: the CPU is incompatible with host CPU: Host CPU does not provide
required features: hle, rtm
```

This error is reported because the RHCK kernel disables Intel Transactional Synchronization Extensions (TSX) by default and can't provide the Hardware Lock Elision (HLE) and Restricted Transactional Memory (RTM) features that the guest is expecting.

As a workaround, perform the following steps:

1. Edit `/etc/default/grub` by adding `tsx=on` to the `GRUB_CMDLINE_LINUX` directive, as shown in the following example in bold:

```
...  
GRUB_CMDLINE_LINUX="LANG=en_US.UTF-8 ... tsx=on"
```

2. Regenerate the `grub.cfg` file.

```
sudo grub2-mkconfig -o /boot/grub2/grub.cfg
```

3. Reboot the KVM host.

Development Tools Issues

- **Python 2 replaced by Python 3**

In Oracle Linux 8, the `python` command isn't available by default. Python 3 is the primary Python version and isn't backward compatible with Python 2. Also, Python 2 is available in Oracle Linux 8 but only with limited support. If you have Python packages or scripts on the system to be upgraded, migrate these to Python 3 as soon as possible.

For more information, see [Oracle Linux 8: Installing and Managing Python](#).

Hardware Related Issues

- **Systems with unrecognized hardware can't be upgraded**

Support for certain hardware, such as the `e1000` driver, has been removed from RHCK 8. The upgrade can't proceed on platforms that have such hardware installed. Even though UEK might continue to support the hardware, the upgrade procedure is still inhibited if the hardware is detected on the system.

Leapp Overlay Size Issues

- **Upgrading might require increased overlay size**

Upgrading Oracle Linux 7 systems with a huge number of packages to Oracle Linux 8 might fail because of insufficient space in the Leapp overlay file systems that are used during the upgrade. You might see the following error message:

```
Error: Transaction test error:  
installing package package-name needs 4MB on the / filesystem
```

As a workaround, increase the `LEAPP_OVL_SIZE` variable. The default size is 4096. The actual size you would need might be larger depending on the specific setup. Use the following command:

```
sudo export LEAPP_OVL_SIZE=new-size
```

A

Supported Repositories in Leapp Upgrades

This appendix shows repositories that are used in a system or instance upgrade that uses the Leapp utility.

Repository Mappings

The following table shows repository correspondences between Oracle Linux 7 and Oracle Linux 8. The table helps you to identify the corresponding repositories that the Leapp utility makes available after the host has completed upgrading to Oracle Linux 8.

Oracle Linux 7 Yum Repositories	Oracle Linux 8 DNF Repositories	Notes
ol7_latest	ol8_baseos_latest ol8_appstream	All Oracle Linux 8 upgrades require the BaseOS and AppStream repositories.
ol7_kvm_utils (x86_64) ol7_latest (aarch64)	ol8_kvm_appstream	In Oracle Linux 7 for the aarch64 platform, the KVM packages are in the ol7_latest repository.
ol7_UEKR5	ol8_UEKR6	Oracle Linux 8 requires UEK R6 as a minimum UEK version.
ol7_UEKR6	ol8_UEKR6	
ol7_addons	ol8_addons	
ol7_ksplince	ol8_ksplince	Available for Oracle Cloud Infrastructure instances only.
ol7_\$basearch_userspace_ksplince	ol8_\$basearch_userspace_ksplince	Available for Oracle Cloud Infrastructure instances only.
ol7_oci_included	ol8_oci_included	Available for Oracle Cloud Infrastructure instances only.
ol7_optional_latest	ol8_codeready_builder	Suggested for developer systems only.
ol7_UEKR5_RDMA	ol8_UEKR6_RDMA	
ol7_UEKR6_RDMA	ol8_UEKR6_RDMA	

This table shows the repository mappings for the Oracle Linux KVM Stack.

Using Command Arguments to Enable Repositories

As more products are upgradeable with future versions of the Leapp utility, the number of repositories that need to be enabled after the upgrade might also increase. The Leapp upgrade commands would become complicatedly long as you manually list the repositories to be enabled in the command syntax.

Oracle has provided the following convenience switches or arguments that can be used with the `leapp preupgrade` or `upgrade` commands. When used, these arguments automatically apply the `--enablerepo` subcommand to repositories that are appropriate to the host that you're upgrading.

--oraclelinux

This argument is used on system upgrades that you perform either locally or remotely. The argument detects the system's architecture and automatically uses the repositories that are applicable to the architecture.

When you use this argument, the following repositories are automatically enabled:

- `ol8_baseos_latest`
- `ol8_appstream`
- `ol8_UEKR6`¹

--oci

This argument is used on Oracle Cloud Infrastructure instance upgrades. The repositories covered by this argument are a superset of the `--oraclelinux` argument.

When you use this argument, the following repositories are automatically enabled:

- `ol8_baseos_latest`
- `ol8_appstream`
- `ol8_UEKR6`¹
- `ol8_addons`
- `ol8_ksplite`
- `ol8_oci_included`

¹ For `aarch64` platforms, the `ol8_UEKR6` repository does not exist. Instead, UEK R6 is part of the `ol8_baseos` repository.

Caution:

The `--oraclelinux` and `--oci` arguments for enabling default repositories are mutually exclusive.

For example, the `leapp preupgrade --oraclelinux` command would be equivalent to the following syntax:

```
sudo leapp preupgrade --enablerepo 'ol8_baseos_latest' --enablerepo  
'ol8_appstream' --enablerepo 'ol8_UEKR6'
```

Use the `--enablerepo` option to enable more required repositories that aren't in the default list of repositories that are enabled by the argument that you're using. You must use the option for every additional repository you want to enable, for example:

```
sudo leapp preupgrade --oraclelinux --enablerepo 'ol8_addons' --enablerepo  
'ol8_codeready_builder' ...
```


B

Updated dhclient Script

The `dhclient-exit-hook-set-hostname.sh-ol8` is intended for use on Oracle Linux instances on Oracle Cloud Infrastructure that have been upgraded from Oracle Linux 7 to Oracle Linux 8. This script includes code for the proper handling of connectivity to those upgraded instances.

```
#!/bin/bash
#
# Copyright (C) 2022 Oracle. All rights reserved.
#
# This program is free software; you can redistribute it and/or modify it under
# the terms of the GNU General Public License as published by the Free Software
# Foundation, version 2. This program is distributed in the hope that it will
# be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General
# Public License for more details. You should have received a copy of the GNU
# General Public License along with this program; if not, write to the Free
# Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA
# 021110-1307, USA.

log () {
    logger -t "${0##*/}" "$*"
}

log "set hostname, /etc/hosts, /etc/resolv.conf begin $(date):"

OPC_CONF="/etc/oci-hostname.conf"

# import the oci-hostname configuration info
if [ -f $OPC_CONF ]; then
    . $OPC_CONF
fi

# ER-28862654 - Add custom header text to /etc/resolv.conf
if ! grep -q "docs.cloud.oracle.com" /etc/resolv.conf; then
    sed -i '1i; Any changes made to this file will be overwritten whenever the\
; DHCP lease is renewed. To persist changes you must update the\
; /etc/oci-hostname.conf file. For more information see\
;[https://docs.cloud.oracle.com/iaas/Content/Network/Tasks/managingDHCP.htm#notes]\
;' /etc/resolv.conf
fi

function retry_command()
{
    retry_attempts=30
    retry_interval_sec=2
    while [ "$retry_attempts" -gt 0 ]; do

        command_success=true
        "$@" || { command_success=false; }
        if [ "$command_success" == false ]; then
            (( retry_attempts-- ))
            log "Error occurred running command $@. Will retry in $retry_interval_sec seconds"
            sleep $retry_interval_sec
        else

```

```

        log "Successfully executed the command ${0}"
        break
    fi
done

# Check if issue running command still existed after all retry_attempts
if [ "$command_success" == false ]; then
    log "ERROR: failed to execute command '${0}' (Retried $retry_attempts times)"
    return 1
fi
}

#Usage: add_entries <file name> <keyword> <an array of the corresponding values for the
keyword>
#We pass array by name so if the array name is 'arr', pass it as 'arr' instead of $arr
#This function can be used to add entries to files with a mapping format.
#For example, /etc/hosts has <ip> mapped to <fqdn/host alias>
#The function checks to see if a line containing the given 'keyword' is in the file
#If so, we check the given array of values against the existing values for the keyword
in that line.
#Append the values specified in the array to the line if it doesn't already exist.
#If the file does not contain a line with the given keyword,
#the function will add a new line with the given keyword mapped to all values in the
given array.
function add_entries()
{
    local file=${1}
    local keyword=${2}
    local values=${3[@]}
    values=("${!values}")
    if ! grep -qw "^$keyword" $file; then
        log "Line with '$keyword' not found in $file"
        new_entry="$keyword"
        for value in "${values[@]}"
        do
            new_entry="$new_entry $value"
        done
        log "Adding '$new_entry' to $file"
        echo "$new_entry" >> $file

    else
        log "Found line with '$keyword'"
        target_line=$(grep -w "^$keyword" $file)
        for value in "${values[@]}"
        do
            #First case needs spaces around $value to make sure it's not the prefix or
            suffix of another value
            #Second case checks if $value is at the end of the line
            if [[ $target_line == *" $value *" ]] || [[ $target_line == "*" $value " " ]];
            then
                log "'$value' already exists in line"
            else
                log "Adding '$value' to line"
                sed -i "s/^\<keyword\>.*$/& $value/g" $file
            fi
        done
    fi
}

# This function updates the hostname
# Arguments:

```

```

# Arg1 -- OS version information to set hostname accordingly
# Arg2 -- Hostname that needs to be set
function update_hostname()
{
    local os_version=${1}
    local new_host_name=${2}

    log "Updating hostname"

    # 1. run hostname command
    if [ $os_version -eq 6 ]; then
        # use short hostname for /etc/sysconfig/network
        # https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/6/html/
Installation_Guide/sn-Netconfig-x86.html
        new_host_name_config="HOSTNAME=$new_host_name"
        log "Update /etc/sysconfig/network with new host name $new_host_name_config"

        if grep --quiet '^HOSTNAME=' /etc/sysconfig/network; then
            log "HOSTNAME exists in /etc/sysconfig/network. Updating its value"
            sed -i "s/^HOSTNAME=.*$/$new_host_name_config/g" /etc/sysconfig/network
        else
            log "Adding HOSTNAME to /etc/sysconfig/network"
            echo "$new_host_name_config" >> /etc/sysconfig/network
        fi

        log "Running hostname command: hostname $new_host_name"
        hostname $new_host_name

    elif [ $os_version -eq 7 ]; then
        log "Running hostnamectl command: hostnamectl set-hostname $new_host_name"
        hostnamectl set-hostname $new_host_name >> $log_file 2>&1
    elif [ $os_version -eq 8 ]; then
        systemctl is-active --quiet NetworkManager
        nm_status=$?
        if [ $nm_status -eq 0 ]; then
            log "Running nmcli command: nmcli general hostname $new_host_name"
            nmcli general hostname $new_host_name
        else
            log "Running hostnamectl command: hostnamectl set-
hostname $new_host_name"
            hostnamectl set-hostname $new_host_name
        fi
    fi
}

# This function updates /etc/hosts and /etc/resolv.conf
# Arguments:
# Arg1 -- new IP address
# Arg2 -- new hostname of the system
function update_hosts_resolv()
{
    local new_ip_address=${1}
    local new_host_name=${2}

    # Retry params
    fqdn_success=1
    retries=12

    # Remove old entry from /etc/hosts so that we avoid getting
    # stale information from ipcalc.

```

```

# First though, save the entries for
# restoration if ipcalc encounters failures.
old_vals=$(grep "^$new_ip_address" /etc/hosts)
old_fqdn=$(echo $old_vals | awk -F " " '{print $2}')
old_host_name=$(echo $old_vals | awk -F " " '{print $3}')
log "Pre-existing fqdn is $old_fqdn and hostname is $old_host_name"
# Now remove old entry
sed -i "/^\<$new_ip_address\>.*$/d" /etc/hosts

while [[ $fqdn_success -ne 0 ]] && [[ $retries -ne 0 ]]; do
    # Get fqdn
    fqdn=$(retry_command ipcalc -h $new_ip_address)
    if [[ "$fqdn" != *.*.*.* ]]; then
        # DNS can take upto 120 seconds to return info. Retry as long.
        sleep 10
        (( retries-- ))
    else
        fqdn_success=0
    fi
done

if [ $fqdn_success -ne 0 ]; then
    log "ERROR: ipcalc unsuccessful despite multiple retries for 120 seconds"
    # Restore previously existing hostname entry but first:
    # Check for existing exact matches of hostname and delete them, if any.
    sed -i -e "/[[:space:]]\<$old_host_name\>[[:space:]]\./d; /[[:space:]]\<$old_host_name\>$/d" /etc/hosts
    fqdn=${fqdn#HOSTNAME=}
    echo "fqdn=$fqdn" >> $log_file

    old_host_values=("$old_fqdn" "$old_host_name")

# Pre-existing FQDN is null for the first boot.
if [ -z "$old_fqdn" ]; then
    echo "Pre-existing fqdn is . Skip updating /etc/hosts" >> $log_file
else
    add_entries "/etc/hosts" "$new_ip_address" old_host_values
fi

else

# ipcalc returns HOSTNAME=xxxx, need to remove "HOSTNAME="
fqdn=${fqdn#HOSTNAME=}

# get subnet_domain_name
subnet_domain_name=${fqdn#$new_host_name.}

# verify that the subnet domain is valid, we expect it is of the
# form <subnet-name>.<vcn-name>.<oraclevcn>.<com>
if [[ "$subnet_domain_name" != *.*.*.* ]]; then
    log "ERROR: invalid subnet domain name '$subnet_domain_name'."
else
    # get vcn domain name - everything after the first dot in the subnet domain
name
    vcn_domain_name=${subnet_domain_name#*.}
    log "fqdn=$fqdn"
    log "subnet_domain_name=$subnet_domain_name"
    log "vcn_domain_name=$vcn_domain_name"

# 2. Update /etc/hosts if needed
# Check for existing exact matches of hostname and delete them, if any.

```

```

        sed -i -e "/[[:space:]]\<$new_host_name\>[[:space:]]\./d; /[[:space:]]
\<$new_host_name\>$/d" /etc/hosts

        new_host_values=("$fqdn" "$new_host_name")
        # Pass array by name
        add_entries "/etc/hosts" "$new_ip_address" new_host_values

        # 3. Update /etc/resolv.conf
        # This is a temp fix till we have a resolution for a proper dhcp response
        new_search_domains=("$subnet_domain_name" "$vcn_domain_name")
        add_entries "/etc/resolv.conf" "search" new_search_domains
        echo -e "[main]\ndns = none\n" > /run/NetworkManager/conf.d/10-oci-
dhclient.conf
        fi
    fi
}

# This function updates /etc/resolv.conf
# Arguments:
#   Arg1 -- new IP address
#   Arg2 -- new hostname of the system
function update_resolv()
{
    local new_ip_address=${1}
    local new_host_name=${2}

    # Retry params.
    retries=12
    fqdn_success=1

    while [[ $fqdn_success -ne 0 ]] && [[ $retries -ne 0 ]]; do
        # Since the hostname might have been changed in /etc/hosts and we're not
        # updating it, only using ipcalc might give us stale hostname information.
        # To get the DNS provided hostname, use host and compare it with ipcalc's
        # generated version. If they don't match use the one from host.
        host_name=$(retry_command host $new_ip_address | awk -F " " '{print $5}')
        ipcalc_name=$(retry_command ipcalc -h $new_ip_address)
        if [[ "$ipcalc_name" != *.*.*.* ]] || [[ "$host_name" != *.*.*.* ]]; then
            # DNS can take up to 120 seconds to return info. Retry as long.
            sleep 10
            (( retries-- ))
        else
            fqdn_success=0
        fi
    done

    if [[ $fqdn_success -ne 0 ]]; then
        log "ERROR: ipcalc and host commands failed " \
        log "despite multiple retries for 120 seconds."
    else

        # ipcalc returns HOSTNAME=xxxx, need to remove "HOSTNAME="
        ipcalc_name=${ipcalc_name#HOSTNAME=}
        log "ipcalc returned hostname: $ipcalc_name"

        # host substring will have a "." at the end. Drop it
        host_name=${host_name%?.}
        log "host returned hostname: $host_name"

        use_ipcalc_hostname=0
        if [ "$host_name" = "$ipcalc_name" ]; then
            # either one will do

```

```

        fqdn=$ipcalc_name
    else
        # Likely that ipcalc has a user changed hostname.
        # This will not match the new_host_name one.
        # Use the one from host. But before that confirm
        # that host has returned a valid name.
        if [[ "$host_name" != *.*.*.* ]]; then
            log "Invalid hostname $host_name from host command"
            fqdn=$ipcalc_name

            # Need an additional check here for the host_name
            # being identical to the new_host_name as the user
            # could have changed the /etc/host host_name
            # entry which would not match the dhclient one.
            ipcalc_host_name=$(echo $ipcalc_name | awk -F "." '{print $1}')
            if [[ "$ipcalc_host_name" != "$new_host_name" ]]; then
                log "ipcalc returned host $ipcalc_host_name"
                log "dhclient returned host $new_host_name"
                log "Using ipcalc returned hostname"
                use_ipcalc_hostname=1
            fi
        else
            fqdn=$host_name
        fi
    fi

    # get subnet_domain_name
    if [[ "$use_ipcalc_hostname" -eq 1 ]]; then
        subnet_domain_name=${fqdn#$ipcalc_host_name.}
    else
        subnet_domain_name=${fqdn#$new_host_name.}
    fi

    # verify that the subnet domain is valid, we expect it is of the
    # form <subnet-name>.<vcn-name>.<oraclevcn>.<com>
    if [[ $subnet_domain_name != *.*.*.* ]]; then
        log "WARNING: invalid subnet domain name '$subnet_domain_name' seen."
    else
        # get vcn domain name - everything after the first dot in the subnet domain
name
        vcn_domain_name=${subnet_domain_name#*.}
        log "fqdn=$fqdn"
        log "subnet_domain_name=$subnet_domain_name"
        log "vcn_domain_name=$vcn_domain_name"

        # Update /etc/resolv.conf
        new_search_domains=("$subnet_domain_name" "$vcn_domain_name" "new_search_domains")
        add_entries "/etc/resolv.conf" "search" new_search_domains
        echo -e "[main]\ndns = none\n" > /run/NetworkManager/conf.d/10-oci-
dhclient.conf
        fi
    fi
}

# This function adds NM_CONTROLLED=no entry to the primary interface config file
# So that network manger does not take cotrol when installed.
# Arguments:
# Arg1 -- primary_ip

function disable_NMcontrol()
{
    local primary_ip=${1}

```

```

# find the primary interface
primary_if=$(ifconfig | grep -B1 $primary_ip | head -n1 | awk -F '[: ]' '{print $1}')

# generate the primary interface's ifconfig filepath.
cfg_file="/etc/sysconfig/network-scripts/ifcfg-${primary_if}"

# check if the file is present.
if [ ! -f $cfg_file ]; then
    log "$cfg_file not found, skip NM_CONTROLLED setting."
    return
fi

# check if the keyword is present or not
if ! grep -qw "^NM_CONTROLLED" $cfg_file; then
    # append the line..
    echo "NM_CONTROLLED=no" >> $cfg_file
else
    # modify the line
    sed -i "s/^\<NM_CONTROLLED\>.*$/NM_CONTROLLED=no/g" $cfg_file
fi
}

os_version=0
if [ -f /etc/os-release ]; then
    os_string=$(grep -w VERSION /etc/os-release | awk -F "\"" '{print $2}')
    log "INFO: Obtained $os_string from /etc/os-release"
    if [[ "$os_string" == "8.*" || "$os_string" == "8"* ]]; then
        os_version=8
    elif [[ "$os_string" == "7.*" || "$os_string" == "7"* ]]; then
        os_version=7
    elif [[ "$os_string" == "6.*" ]]; then
        os_version=6
    fi
fi

if [ $os_version == 0 ]; then
    log "INFO: Getting OS version via uname -mrs"
    kernel_version=$(uname -mrs)
    if [[ "$kernel_version" == *"el8"* ]]; then
        os_version=8
    elif [[ "$kernel_version" == *"el7"* ]]; then
        os_version=7
    elif [[ "$kernel_version" == *"el6"* ]]; then
        os_version=6
    fi
fi

if [ $os_version == 0 ]; then
    log "ERROR: Could not obtain valid OS version. Exiting.."
    exit 1
fi

# OL8 use NetworkManager. DHCP does not provide a reason.
if [[ "$os_version" -ne 8 ]]; then
    log "$(date): Script Reason: $reason"
else
    log "$(date): Script Reason: NM-controlled system"
fi

# For non-OL8 OSs get the primary vnic ip only if interface has been
# initialized.

```

```

# Ref: https://www.isc.org/wp-content/uploads/2018/02/dhcp44cscript.html#PREINIT
if [ "$os_version" == 8 -o "$reason" != "PREINIT" ]; then
    primary_ip=$(retry_command curl -H "Authorization: Bearer Oracle" http://
169.254.169.254/opc/v2/vnics/ -sf | jq -r '[0] | .privateIp')
    log "$(date): Primary IP obtained: $primary_ip"
fi

# This script is invoked whenever dhclient is run.
# For non-ol8 instances, We want to skip hostname update if
# $new_ip_address != $primary_ip
# so we don't run this for all interfaces
# For ol8, with the use of NM, we don't get information on the
# new_ip_address, therefore we just use the primary_ip each time.
if [ -z "$primary_ip" ]; then
    log "Skip updating hostname because primary ip is empty."
elif [[ "$os_version" -ne 8 ]] && [[ "$new_ip_address" != "$primary_ip" ]]; then
    log "Skip updating hostname because this was not invoked for the primary vnic"
else
    if [ $os_version -ne 8 ]; then
        # For non-ol8, add NM_Controlled="no" to primary network interface
        # configuration file
        disable_NMcontrol $primary_ip

        # reason why this hook was invoked. It is set by dhclient script when
        # the OS is non-ol8
        log "reason=$reason"

        # https://linux.die.net/man/8/dhclient-script
        if [ "$reason" = "BOUND" ] || [ "$reason" = "RENEW" ] || [ "$reason" =
"REBIND" ] || [ "$reason" = "REBOOT" ]; then
            log "os version = $os_version"
            #These variables are set by dhclient script
            log "new_ip_address=$new_ip_address"
            log "new_host_name=$new_host_name"
            log "new_domain_name=$new_domain_name"
        else
            log "Not updating because reason=$reason"
        fi
    fi

    if [[ $PRESERVE_HOSTINFO -eq 2 ]]; then
        log "Skip updating hostname, /etc/hosts and /etc/resolv.conf"
        log "as per PRESERVE_HOSTINFO=${PRESERVE_HOSTINFO} setting"
        return 0
    fi

    #Retrieve hostname from metadata if its empty
    if [ -z $new_host_name ]; then
        new_host_name=$(retry_command curl -sf -H "Authorization: Bearer Oracle" http://
169.254.169.254/opc/v2/instance/ | jq '.hostname' -r)
    fi

    # Will get retrieved for ol8 since we do not have DHCP provided vars.
    if [ -z $new_ip_address ]; then
        new_ip_address=$(retry_command curl -H "Authorization: Bearer Oracle" http://
169.254.169.254/opc/v2/vnics/ -sf | jq -r '[0] | .privateIp')
    fi

    if [ -z $new_host_name ]; then
        echo "ERROR: new_host_name is empty after retrieving it from metadata json.
Exiting."
        exit_status=1
    fi

```



```
else
log "new_ip_address=$primary_ip"
if [[ $PRESERVE_HOSTINFO -eq 0 ]]; then
# update the hostname with new hostname
update_hostname $os_version $new_host_name
# update hosts and resolv conf files
update_hosts_resolv $new_ip_address $new_host_name
elif [[ $PRESERVE_HOSTINFO -eq 1 ]]; then
log "Skip updating hostname as per"
log "PRESERVE_HOSTINFO=${PRESERVE_HOSTINFO} setting"
# update hosts and resolv conf files
update_hosts_resolv $new_ip_address $new_host_name
elif [[ $PRESERVE_HOSTINFO -eq 3 ]]; then
log "Skip updating hostname and /etc/hosts as per"
log "PRESERVE_HOSTINFO=${PRESERVE_HOSTINFO} setting"
log "Updating subnet in /etc/resolv"
# update resolv conf file alone
update_resolv $new_ip_address $new_host_name
fi
fi
log "sethostname, /etc/hosts, /etc/resolv.conf END"
```