Oracle Linux

Ksplice User's Guide

E39380-73
September 2025

Oracle Linux Ksplice User's Guide,

E39380-73

Contents

4 Using the Ksplice Uptrack Client

5 Using the Ksplice Uptrack API

Preface

Oracle Linux: Ksplice User's Guide provides information about how to install, configure, and use Oracle Ksplice to update kernel and user space packages on a running system and how to use the Ksplice Uptrack API.

1

About Oracle Ksplice

Oracle Ksplice updates select, critical components of a Linux installation with Critical, Important and select Moderate CVEs, based on the Oracle Linux vulnerability impact ratings, without the need to reboot.

Ksplice is freely available for Oracle customers who subscribe to Oracle Linux Premier Support and Oracle Cloud Infrastructure services. If you're an Oracle Linux Basic, Basic Limited, or Network Support subscriber, contact your sales representatives to discuss a potential upgrade of your subscription to a Premier Support plan.

Why Use Ksplice?

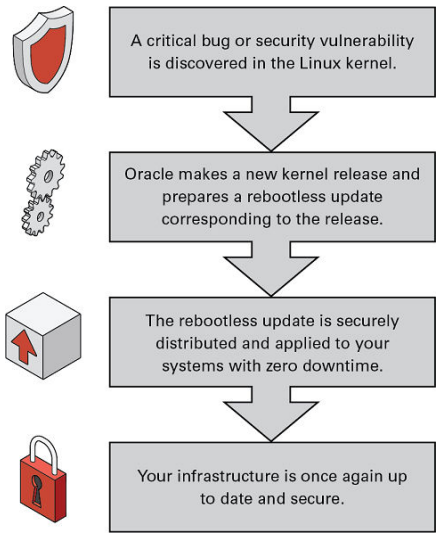
Ksplice can apply critical updates without rebooting. Traditionally, applying security updates to core operating system components requires you to manually install updated RPMs, schedule downtime, and reboot the server. Ksplice helps to keep systems secure and highly available by updating a running system with the latest kernel and key user space updates.

Ksplice rebootless updates:

- Save time and hassle by updating in seconds, while the system is running.
- Avoid downtime.
- Prevent disastrous security incidents by keeping systems updated.

Life Cycle of a Ksplice Update

When a critical bug or security vulnerability is discovered in the Linux kernel, Oracle produces a new kernel release and prepares a rebootless update corresponding to that release. The rebootless update is securely distributed by using the Ksplice Uptrack server and ULN. The Ksplice Enhanced or Ksplice Uptrack Client then applies this update to the system, with zero downtime. The infrastructure is updated and secure.



In-Memory Compared to On-Disk Updates

A Ksplice update occurs in memory and takes effect immediately upon application, which is different than an on-disk change that requires a reboot to take effect. However, you must continue to apply on-disk updates, even when using Ksplice, to ensure that updated package binaries can be used if the system or processes restart. On-disk updates are handled by subscribing to Unbreakable Linux Network (ULN) or by using a local ULN mirror.

Ksplice patches keep a system updated while it's running, but you must continue to install the regular kernel packages for released errata from ULN or the Oracle Linux Yum server so that the kernel is also updated on disk. The system is then ready for the next maintenance window or reboot. When you restart the system, you can boot it from the newer kernel version. Ksplice then uses the new kernel as a baseline for applying patches when they become available.

Available Architectures

Ksplice is available for the following platforms:

- Intel 64-bit (x86_64)
- AMD 64-bit (x86_64)
- 64-bit Arm (aarch64)

> **Note**
>
> Ksplice on the 64-bit Arm (aarch64) platform is only available with maintained Unbreakable Enterprise Kernel (UEK) releases. For more information, see the UEK release notes in the Unbreakable Enterprise Kernel documentation

Maintained Kernels

Only specific kernel versions are actively maintained by Ksplice.

- Kernels Actively Maintained With Ksplice

- Kernels No Longer Actively Maintained With Ksplice

> **Note**
>
> Ksplice on Oracle Cloud Infrastructure supports specific Linux distributions. For more information, see Oracle Ksplice on Oracle Cloud Infrastructure.

For questions about supported kernels, send an email to ksplice-support_ww@oracle.com.

Kernels Actively Maintained With Ksplice

With Oracle Linux Premier Support or Premier Limited subscriptions, you can use Ksplice to bring various Linux kernels up-to-date with the latest important security and bug fix patches. The following table shows the distributions and kernel versions that are automatically maintained with Ksplice.

> **Note**
>
> If the system is running RHEL and you recently migrated to Oracle Linux Premier Support, you must switch to RHCK to use Ksplice kernel patches. Oracle no longer maintains Ksplice patches for RHEL kernels.

Actively Maintained Kernel Type	More Information
UEK 8 (aarch64) starting with `6.12.0-0.20.20` (released Apr 14, 2025).	
UEK 8 (x86_64) starting with `6.12.0-0.20.20` (released Apr 14, 2025).	
UEK R7 (aarch64) starting with `5.15.0-0.30.19` (released Jun 30, 2022).	
UEK R7 (x86_64) starting with `5.15.0-0.30.19` (released Jun 30, 2022).	
UEK R6 (aarch64) starting with `5.4.17-2011.0.7` (released Mar 17, 2020).	
UEK R6 (x86_64) starting with `5.4.17-2011.1.2` (released Apr 27, 2020).	
Oracle Linux 10 Red Hat Compatible Kernels (RHCK) starting with the official release.	
Oracle Linux 9 Red Hat Compatible Kernels (RHCK) starting with the official release.	
Oracle Linux 8 Red Hat Compatible Kernels (RHCK) starting with the official release.	
Oracle Linux 7 Red Hat Compatible Kernels (RHCK) starting with the official release.	
Ubuntu 24.04 Noble kernels and Hardware Enablement (HWE) kernels, starting with the official release.	Kernels include `6.8` versions.
Ubuntu 22.04 Jammy kernels and HWE kernels, starting with the official release.	Kernels include `5.15` and `6.8` versions.

Actively Maintained Kernel Type	More Information
Ubuntu 20.04 Focal Fossa HWE kernels, starting with the official release	Kernels include `5.15` versions

> **Important**
>
> If you have booted the most recent available kernel and no Ksplice updates are available for that kernel, some Ksplice commands might fail or might return an error message notifying you that the kernel version isn't yet supported by Ksplice Uptrack. These commands only succeed when Ksplice updates are available for the kernel that's running on the system. As soon as an update becomes available, the command succeeds, and the update is applied.

Kernels No Longer Actively Maintained With Ksplice

The following kernels don't receive Ksplice updates, but any Ksplice updates previously issued are still available if you have a support contract.

To maintain any of the following kernels on a listed Linux distribution, you need to manually upgrade them by using the `yum update` or `dnf update` command, or in the case of Ubuntu, by using the `apt` command. Kernel updates that don't use Ksplice require system reboots to be effective.

Kernel Type	Kernel Version	Releases No Longer Actively Maintained
UEK R5	All versions	Oracle Linux 6 Oracle Linux 7
UEK R4	All versions	Oracle Linux 6
UEK R3	All versions	Oracle Linux 6 Oracle Linux 7
UEK R2	All versions	Oracle Linux 6
RHCK	All versions	Oracle Linux 6
Kernels shipped with RHEL 9.	All versions	RHEL 9
CentOS and RHEL 8 kernels.	All versions	RHEL or CentOS Linux 8
CentOS and RHEL 7 kernels.	All versions	RHEL or CentOS Linux 7
Kernels shipped in RHEL/CentOS Linux 6	All versions	RHEL or CentOS Linux 6
GA Kernels shipped in Ubuntu 20.04 LTS	5.4 GA	Ubuntu 20.04 LTS (Focal Fossa)
Kernels shipped in Ubuntu 18.04 LTS.	All versions	Ubuntu 18.04 LTS (Bionic Beaver)
Kernels shipped in Ubuntu 16.04 LTS	All versions	Ubuntu 16.04 LTS (Xenial Xerus)

Using the Ksplice Inspector Tool

Use Ksplice Inspector, which is a free, online tool that lists available Ksplice updates for Maintained Kernels.

Ksplice Inspector helps you find the updates that are available for the running kernel and what updates can be automatically applied in-memory by using either the Ksplice Enhanced client or the Ksplice Uptrack client. The tool helps you to proactively identify security vulnerabilities, which is a critical step in assessing potential cybersecurity issues. The tool is publicly available and doesn't require a support subscription.

The following procedure describes how to use the tool to find out about security issues for a particular system.

1. In a terminal, use the `uname` command to get information about the system.

   Open a terminal on the Linux system that you want to check, and then run the following command:

   ```
   echo "$(uname -s)//$(uname -m)//$(uname -r)//$(uname -v)"
   ```

2. Copy the output from the command into the Ksplice Inspector tool.

   Copy the output of the previous command into the Ksplice Inspector text input field, and then select **Find Updates**.

3. Review the available security patches.

   The tool indicates what security patches are already available to Ksplice customers.

Oracle Cloud Infrastructure Ksplice Support

You can monitor and manage automatic updates for Oracle Linux systems that are running within Oracle Cloud infrastructure (OCI) by using Ksplice.

Note the following key points about receiving automatic Ksplice updates on systems that are running within OCI:

- By default, Ksplice configuration is shipped with the OCI platform images by preconfiguring the Ksplice yum repositories and Ksplice online server URL.
- Bring Your Own Image (BYOI) configurations can use the same yum repository configuration file as the platform images (`/etc/yum.repos.d/ksplice-ol*.repo`), if copied there manually.

  > **Note**
  >
  > The `/etc/yum.repos.d/ksplice-ol`*N*`.repo` file comes from the `ksplice-release-el`*N* RPM, which is in the yum repository that's configured by `oci_included_ol`*N*`.repo` and is part of the `oci-included-release-el`*N* package (`/etc/yum.repos.d/oci-included-ol`*N*`.repo`).

- Systems running within OCI that have the Ksplice client configured in *online* mode don't need to be registered with ULN to access the Ksplice servers and receive automatic updates.
- Systems running within OCI that have the Ksplice client configured in *offline* mode don't need to be registered with ULN, nor do they require a local ULN mirror configuration to receive automatic updates.

For further information, see Oracle Ksplice on Oracle Cloud Infrastructure.

Oracle Enterprise Manager Ksplice Support

All Oracle Linux systems on which Enterprise Manager Agent is installed and the Ksplice software is configured can be monitored and managed through Oracle Enterprise Manager, within the Oracle Linux Home Ksplice region of the Enterprise Manager user interface (UI).

To learn more about using Oracle Enterprise Manager to monitor and use Ksplice patching on Oracle Linux hosts, see the Oracle Enterprise Manager Life Cycle Management Administrator's Guide.

2

Preparing to Use Oracle Ksplice

The prerequisites for using Oracle Ksplice depend on which client you select and whether you choose to use online or offline mode.

Before using Ksplice:

- Choose a Ksplice client
  - Select either the *Ksplice Enhanced Client* or the *Ksplice Uptrack Client*.
  - Decide whether to use the client in *online* or *offline* mode.
- Register your system with ULN (if using online mode)
- Configure a local Ksplice mirror (if using offline mode)

Choosing a Ksplice Client

You have the option to choose between the Ksplice Enhanced Client and the Ksplice Uptrack Client.

> **Note**
>
> The Ksplice Uptrack Client is deprecated on Oracle Linux 10 and might be removed in a future release. Although Ksplice Uptrack Client commands continue to work when you install the Ksplice Enhanced Client package, no `uptrack` package is available on Oracle Linux 10. We recommend using the Ksplice Enhanced Client on all systems.

Table 2-1 Features Supported by Each Ksplice Client

Ksplice Client	User Space Support	x86_64 Support	Arm (aarch64) Support	Known Exploit Detection Support
Ksplice Enhanced Client	Supported	Supported	Supported	Supported on x86_64 platform *only*
Ksplice Uptrack Client	Not supported	Supported	Supported	Not supported

About the Ksplice Enhanced Client

The Ksplice Enhanced Client provides more functionality than the Ksplice Uptrack Client.

In addition to the kernel updates that are applied by the Uptrack Client, the Enhanced Client can patch in-memory pages for the Ksplice-aware `glibc` and `openssl` shared libraries for user space processes. User space patching can install bug fixes and protect the system against security vulnerabilities without restarting processes and services.

Key features of the Enhanced Client include:

- Kernel and user space updates (the Uptrack Client only supports kernel updates)
- Known exploit detection
- Online and offline mode
- Use of the `ksplice` command

> **Note**
>
> The Enhanced Client shares the same configuration file as the Uptrack Client, which is the `/etc/uptrack/uptrack.conf` file. For more information about this file, see Ksplice Client Configuration Options.

About the Ksplice Uptrack Client

> **Note**
>
> The Ksplice Uptrack Client is deprecated on Oracle Linux 10 and might be removed in a future release. Although Ksplice Uptrack Client commands continue to work when you install the Ksplice Enhanced Client package, no `uptrack` package is available on Oracle Linux 10. We recommend using the Ksplice Enhanced Client on all systems.

Ksplice Uptrack can apply the latest kernel security errata for Common Vulnerabilities and Exposures (CVEs) without halting the system or restarting any applications. Ksplice Uptrack applies the updated patches in the background with negligible impact, and only requires a pause of a few milliseconds.

Key features of the Uptrack Client include:

- Kernel updates (to also apply user space updates, consider the Ksplice Enhanced Client instead)
- Online and offline mode
- Use of the `uptrack` command

About Ksplice Offline Mode

You can use either the Ksplice Enhanced Client or Ksplice Uptrack Client in offline mode. The offline version doesn't require a direct connection to the Oracle Uptrack server or to ULN. For example, you could use the `dnf` command to install an update package directly from a memory stick. However, a more typical method would be to configure a local ULN mirror that acts as a mirror for the Ksplice-aware ULN channels. Then, you can configure systems to receive `dnf` and Ksplice updates.

Oracle bundles all available Ksplice updates for each supported kernel version or user space package into an RPM that's specific to that version. Oracle updates this package every time a new Ksplice patch becomes available for the kernel. You can download the latest Ksplice update packages to the local ULN server periodically. Then, the Ksplice server can connect to the local server to receive updates without requiring direct access to the Oracle Uptrack server.

Offline mode doesn't support:

- Ksplice web interface
- Ksplice Uptrack API

> **Important**
>
> If you have booted the most recent available kernel and no Ksplice updates are available, an offline update RPM for that kernel might not yet exist. Offline update RPMs are made available shortly after the kernel releases. However, these RPMs might require more time to synchronize with the local repository that you have set up.

For more information, see:

- Configuring a Local Ksplice Mirror for the Offline Client
- Configuring the Ksplice Enhanced Client for Offline Mode
- Configuring Ksplice Uptrack Clients for Offline Mode
- Switching Between Online and Offline Ksplice Uptrack Installation Modes

Ksplice Client Configuration Options

The Ksplice client configuration is stored in the `/etc/uptrack/uptrack.conf` file.

Table 2-2 Ksplice Client Configuration Options

Option	Description

`accesskey`	The Ksplice Access Key is a unique identifier required to use the Ksplice service. The key associates the system with a ULN account and profile.
	The Ksplice Access Key for a system aligns with the key associated with a Ksplice account at https://status-ksplice.oracle.com/ in the **System Status** tab.
	An example entry might look as follows:
	`[Auth]`
	`accesskey = dfc21b3ce...354c84bc0ae5803e`

`https_proxy`	Proxy server URL and port for accessing the Uptrack server.
	If the system is registered with the Ksplice Uptrack repository, the client communicates with the Uptrack server by connecting to https://updates.ksplice.com:443.
	If the system is behind a firewall, you can either configure the firewall to allow the connection through port 443, or you can configure the client to use a proxy server. An example entry might look as follows:
	`[Network]`
	`https_proxy = https://proxy.example.org:8080`
	Ksplice clients also respect the `https_proxy` environment variable.

Table 2-2 (Cont.) Ksplice Client Configuration Options

Option	Description

`install_on_reboot`	Automatically reinstall updates at boot time if rebooting into the same kernel.
	If this option is enabled, Ksplice applies the same set of patch updates to the kernel after reboot to maintain continuity.
	To install all available updates at boot time if rebooting into the same kernel, enable this option. An example entry might look as follows:
	`[Settings]`
	`...`
	`install_on_reboot=yes`

`upgrade_on_reboot`	Automatically install all available updates at boot time, even if rebooting into a different kernel.
	If this option is enabled, Ksplice applies all available patch updates to the running kernel after reboot. Note that if the system is running in offline mode, it can only apply patches for RPM packages that are already available on the system.
	To install all available updates at boot time if rebooting into the same kernel, enable this option. An example entry might look as follows:
	`[Settings]`
	`...`
	`upgrade_on_reboot=yes`

`autoinstall`	Automatically install new updates as they become available. Note that if the system is running in offline mode, it can only apply patches for RPM packages that are already available on the system.
	You receive an email notification when Ksplice updates are available for the system.
	To instruct the client to install all updates automatically, enable this option. For example, the entry might appear as follows:
	`[Settings]`
	`...`
	`autoinstall = yes`
	Note
	Enabling the automatic installation of updates doesn't automatically update the Ksplice client itself. Oracle notifies you by email when you can upgrade the Ksplice software by using the `dnf` command.

Table 2-2 (Cont.) Ksplice Client Configuration Options

Option	Description

`skip_apply_after_pkg_install`	Prevent automatic application of Ksplice updates after installing uptrack-updates RPMs using the `dnf` command.
	This option is useful for testing, and for customized patch management where you might not want Ksplice to automatically apply patches, such as when running Ksplice in offline mode and you might need to decouple RPM package management from Ksplice patching.
	To disable the automatic application of Ksplice updates after uptrack update packages are installed, enable this option. For example, the entry might look as follows:
	`[Settings]`
	`...`
	`skip_apply_after_pkg_install = true`

Registering With ULN

To use Ksplice to apply automatic updates, the system must have access to the Internet and it must be registered with ULN.

Systems that are configured to use the Ksplice *offline* client must have access to a local ULN mirror to receive automatic updates. For instructions, see Configuring a Local Ksplice Mirror for the Offline Client.

The requirements for systems running within OCI are as follows:

- For instances that are configured to use the Ksplice *online* client, the system doesn't need to be registered with ULN, as these instances are preconfigured for automatic access to the Ksplice servers and all the Ksplice updates (applies to both the Ksplice online and offline clients).
- For instances that are configured to use the Ksplice *offline* client, you don't need to configure a local ULN mirror to receive automatic updates.

For more information about receiving automatic Ksplice updates on systems that are running within OCI, see Oracle Cloud Infrastructure Ksplice Support.

For more information about registering a system with ULN, see Oracle Linux: Managing Software on Oracle Linux.

After registering, you can install either the Ksplice Enhanced client software or the Ksplice Uptrack client software from Ksplice for Oracle Linux channel on ULN by using the `yum` or `dnf` command. After installation, Oracle allocates the Ksplice client an identification key that associates it with the Customer Support Identifier (CSI) for your account. You can configure the system to automatically receive updates from the Ksplice Uptrack server.

Automatic Registration

Your account is automatically registered to use the Ksplice Uptrack server if you have one of the following:

- Oracle Linux Premier support subscription

- Premier Limited support subscription
- Oracle Premier Support for Systems and Operating Systems subscription
- Systems running within OCI

Accessing the Ksplice Uptrack Web Interface

If your account has a valid CSI, you can sign in to the Ksplice Uptrack server web interface at https://status-ksplice.oracle.com/ by using your Oracle Account credentials. After signing in to the server, you can view the status of any registered systems, the patches that have been applied, and the patches that are available. You can also create access control groups for registered systems.

Available Ksplice Channels

Oracle Linux 10

Active Ksplice Channels for Oracle Linux 10

The following table describes the channels that are available for Ksplice in Oracle Linux 10.

Table 2-3 Active Ksplice Channels

Channel Name	Channel Label	Description
Ksplice for Oracle Linux 10 (x86_64)	`ol10_x86_64_ksplice`	Ksplice clients, updates, and dependencies for Oracle Linux 10 on x86_64 systems.
Ksplice for Oracle Linux 10 (aarch64)	`ol10_aarch64_ksplice`	Ksplice clients, updates, and dependencies for Oracle Linux 10 on aarch64 systems.
Ksplice-aware user space packages for Oracle Linux 10 (x86_64)	`ol10_x86_64_userspace_ksplice`	Latest packages for Ksplice-aware user space packages for Oracle Linux 10 (x86_64). This channel must only be used with the Ksplice Enhanced client.
Ksplice-aware user space packages for Oracle Linux 10 (aarch64)	`ol10_aarch64_userspace_ksplice`	Latest packages for Ksplice-aware user space packages for Oracle Linux 10 (aarch64). This channel must only be used with the Ksplice Enhanced client.

Oracle Linux 9

Active Ksplice Channels for Oracle Linux 9

The following table describes the channels that are available for Ksplice in Oracle Linux 9.

Table 2-4 Active Ksplice Channels

Channel Name	Channel Label	Description
Ksplice for Oracle Linux 9 (x86_64)	`ol9_x86_64_ksplice`	Ksplice clients, updates, and dependencies for Oracle Linux 9 on x86_64 systems.
Ksplice for Oracle Linux 9 (aarch64)	`ol9_aarch64_ksplice`	Ksplice clients, updates, and dependencies for Oracle Linux 9 on aarch64 systems.
Ksplice-aware user space packages for Oracle Linux 9 (x86_64)	`ol9_x86_64_userspace_ksplice`	Latest packages for Ksplice-aware user space packages for Oracle Linux 9 (x86_64). This channel must only be used with the Ksplice Enhanced client.
Ksplice-aware user space packages for Oracle Linux 9 (aarch64)	`ol9_aarch64_userspace_ksplice`	Latest packages for Ksplice-aware user space packages for Oracle Linux 9 (aarch64). This channel must only be used with the Ksplice Enhanced client.

Oracle Linux 8

Active Ksplice Channels for Oracle Linux 8

The following table describes the channels that are available for Ksplice in Oracle Linux 8.

Table 2-5 Active Ksplice Channels

Channel Name	Channel Label	Description
Ksplice for Oracle Linux 8 (x86_64)	`ol8_x86_64_ksplice`	Ksplice clients, updates, and dependencies for Oracle Linux 8 on x86_64 systems.
Ksplice for Oracle Linux 8 (aarch64)	`ol8_aarch64_ksplice`	Ksplice clients, updates, and dependencies for Oracle Linux 8 on aarch64 systems.
Ksplice-aware user space packages for Oracle Linux 8 (x86_64)	`ol8_x86_64_userspace_ksplice`	Latest packages for Ksplice-aware user space packages for Oracle Linux 8 (x86_64). This channel must only be used with the Ksplice Enhanced client.

Table 2-5 (Cont.) Active Ksplice Channels

Channel Name	Channel Label	Description
Ksplice-aware user space packages for Oracle Linux 8 (aarch64)	`ol8_aarch64_userspace_ksplice`	Latest packages for Ksplice-aware user space packages for Oracle Linux 8 (aarch64). This channel must only be used with the Ksplice Enhanced client.

Oracle Linux 7

Active Ksplice Channels for Oracle Linux 7

The following table describes the channels that are available for Ksplice in Oracle Linux 7.

Table 2-6 Active Ksplice Channels

Channel Name	Channel Label	Description
Ksplice for Oracle Linux 7 (x86_64)	`ol7_x86_64_ksplice_ELS`	Ksplice clients, updates, and dependencies for Oracle Linux 7 on x86_64 systems. Oracle Linux 7 is in extended support, see the Oracle Linux Lifetime Support Policy.
Ksplice-aware user space packages for Oracle Linux 7 (x86_64)	`ol7_x86_64_userspace_ksplice_ELS`	Latest packages for Ksplice-aware user space packages for Oracle Linux 7 (x86_64). This channel must only be used with the Ksplice Enhanced client. Oracle Linux 7 is in extended support, see the Oracle Linux Lifetime Support Policy.

Legacy Ksplice Channels

The following table describes legacy channels that are no longer updated but continue to be available for Ksplice in Oracle Linux under sustaining support. See Oracle Linux Lifetime Support Policy for more information.

Table 2-7 Legacy Ksplice Channels

Channel Name	Channel Label	Description
Ksplice for Oracle Linux 7 (aarch64)	`ol7_aarch64_ksplice`	Ksplice clients, updates, and dependencies for Oracle Linux 7 on aarch64 systems. Note that aarch64 platforms aren't covered for Oracle Linux 7 under extended support.
Ksplice-aware user space packages for Oracle Linux 7 (aarch64)	`ol7_aarch64_userspace_ksplice`	Packages for Ksplice-aware user space packages for Oracle Linux 7 (aarch64). This channel must only be used with the Ksplice Enhanced client. Note that aarch64 platforms aren't covered for Oracle Linux 7 under extended support.
Ksplice for Oracle Linux 6 (i386)	`ol6_i386_ksplice_ELS`	Ksplice clients, updates, and dependencies for Oracle Linux 6 on i386 systems.
Ksplice for Oracle Linux 6 (x86_64)	`ol6_x86_64_ksplice_ELS`	Ksplice clients, updates, and dependencies for Oracle Linux 6 on x86_64 systems.
Ksplice-aware user space packages for Oracle Linux 6 (x86_64)	`ol6_x86_64_userspace_ksplice_ELS`	Packages for Ksplice-aware user space packages for Oracle Linux 6 (x86_64).
Ksplice for Oracle Linux 5 (i386)	`ol5_i386_ksplice_ELS`	Ksplice clients, updates, and dependencies for Oracle Linux 5 on i386 systems.
Ksplice for Oracle Linux 5 (x86_64)	`ol5_x86_64_ksplice_ELS`	Ksplice clients, updates, and dependencies for Oracle Linux 5 on x86_64 systems.
Ksplice for Oracle VM 3 (x86_64)	`ovm3_x86_64_ksplice_ELS`	Ksplice updates for Oracle VM 3 on x86_64 systems.

Updating ULN Subscriptions

1. Sign in to ULN at https://linux.oracle.com. Provide the ULN username and password that you used to register the system.
2. On the Systems tab, select the link named for the system in the list of registered machines.
3. On the System Details page, select **Manage Subscriptions**.

- Oracle Linux 10
- Oracle Linux 9
- Oracle Linux 8

- Oracle Linux 7

Oracle Linux 10

The Ksplice Enhanced client and Ksplice-aware user space packages are available in the following channels on ULN:

- Ksplice for Oracle Linux 10 (x86_64) (`ol10_x86_64_ksplice`)
- Ksplice for Oracle Linux 10 (aarch64) (`ol10_aarch64_ksplice`)
- Ksplice-aware user space packages for Oracle Linux 10 (x86_64) (`ol10_x86_64_userspace_ksplice`)
- Ksplice-aware user space packages for Oracle Linux 10 (aarch64) (`ol10_aarch64_userspace_ksplice`)

Oracle Linux 9

The Ksplice Enhanced client and Ksplice-aware user space packages are available in the following channels on ULN:

- Ksplice for Oracle Linux 9 (x86_64) (`ol9_x86_64_ksplice`)
- Ksplice for Oracle Linux 9 (aarch64) (`ol9_aarch64_ksplice`)
- Ksplice-aware user space packages for Oracle Linux 9 (x86_64) (`ol9_x86_64_userspace_ksplice`)
- Ksplice-aware user space packages for Oracle Linux 9 (aarch64) (`ol9_aarch64_userspace_ksplice`)

Oracle Linux 8

The Ksplice Enhanced client and Ksplice-aware user space packages are available in the following channels on ULN:

- Ksplice for Oracle Linux 8 (x86_64) (`ol8_x86_64_ksplice`)
- Ksplice for Oracle Linux 8 (aarch64) (`ol8_aarch64_ksplice`)
- Ksplice-aware user space packages for Oracle Linux 8 (x86_64) (`ol8_x86_64_userspace_ksplice`)
- Ksplice-aware user space packages for Oracle Linux 8 (aarch64) (`ol8_aarch64_userspace_ksplice`)

Oracle Linux 7

The Ksplice Enhanced client and Ksplice-aware user space packages are available in the following channels on ULN:

- Ksplice for Oracle Linux 7 (x86_64) (`ol7_x86_64_ksplice`)
- Ksplice for Oracle Linux 7 (aarch64) (`ol7_aarch64_ksplice`)
- Ksplice-aware user space packages for Oracle Linux 7 (x86_64) (`ol7_x86_64_userspace_ksplice`)
- Ksplice-aware user space packages for Oracle Linux 7 (aarch64) (`ol7_aarch64_userspace_ksplice`)

4. On the System Summary page, select both the Ksplice user space and Ksplice channels from the list of available channels, then select the right arrow (**>**) to move them to the list of subscribed channels.
5. Accept the licensing terms for the Ksplice Enhanced client packages.
6. Save the subscription and log out of ULN.

Configuring a Local Ksplice Mirror for the Offline Client

To use the Ksplice Offline client, you must configure a local Ksplice mirror using a local ULN mirror. You can then download the latest Ksplice update packages to this server at regular intervals and configure any other systems to receive both yum and Ksplice updates.

For more information about offline clients, see:

- About Ksplice Offline Mode
- Configuring the Ksplice Enhanced Client for Offline Mode
- Configuring Ksplice Uptrack Clients for Offline Mode
- Switching Between Online and Offline Ksplice Uptrack Installation Modes

Setting Up a Local ULN Mirror

Configure an Oracle Linux host as a local ULN mirror to act as a Ksplice mirror.

A system that functions as a local ULN server mirrors channels in the Unbreakable Linux Network.

When you register an Oracle Linux system with ULN, that system is automatically subscribed to default channels in ULN, depending on the system's OS release and architecture. As such, the system can become a mirror to service clients that have the same OS and platform as the mirror.

However, you might also want the local ULN mirror to service clients that use different OS releases for other platforms. In this case, you would need to subscribe to any other channels that are required by those clients.

> **Note**
>
> Mirroring ULN channels is often slower than mirroring yum repositories. Only consider creating a ULN mirror for channels that aren't otherwise available on the Oracle Linux yum server. Where possible, set up mirrors of Oracle Linux yum server repositories instead.

Configuring the Local ULN Mirror

Setting up the system to be a local ULN mirror involves replicating channels from Unbreakable Linux Network.

The ULN mirror must meet be properly configured to access ULN. See Oracle Linux: Managing Software on Oracle Linux for more information.

For each step in this procedure, you can use either the ULN web interface or the `uln-channel` command. To display options that you can use with the `uln-channel` command, type `uln-channel -h`.

1. Enable the system as a yum server.

   As a yum server, the system can subscribe to channels for OS versions and platforms other than the system's own OS and platform.

   - **Using the ULN web interface**

   a. On a browser, sign in at https://linux.oracle.com using valid SSO credentials.

   b. On the Systems tab, select the link named for the system chosen to be a ULN mirror.

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

   d. On the Edit System Properties page, select the **Yum Server** checkbox.

   e. Select **Apply Changes**.

   - **Using the `uln-channel` command**

   a. On the system's terminal window, type:

   ```
   sudo uln-channel --enable-yum-server
   ```

   b. If prompted, specify the appropriate ULN username and authentication token.

2. Subscribe the system to the channels that you intend to mirror.

   - **Using the ULN web interface**

   a. On the System Details page of the chosen ULN mirror, select **Manage Subscriptions**.

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

   > **Note**
   >
   > If you have an Oracle Linux Support account and you want the mirror 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.

   c. When you have finished selecting channels, select **Save Subscriptions**.

   - **Using the `uln-channel` command**

   a. On the system's terminal window, type:

   ```
   sudo uln-channel -a -c channel [-c channel …]
   ```

   b. If prompted, specify the appropriate ULN username and authentication token.

   c. (Optional) To verify that the subscriptions completed successfully, type:

   ```
   sudo uln-channel -l
   ```

3. Create a protected and unprotected version of `/etc/dnf/plugins/spacewalk.conf`. For example:

   ```
   sudo cp /etc/dnf/plugins/spacewalk.conf /etc/dnf/plugins/
   spacewalk.conf.protected
   ```

```
sudo cp /etc/dnf/plugins/spacewalk.conf /etc/dnf/plugins/
spacewalk.conf.unprotected
```

Edit `/etc/dnf/plugins/spacewalk.conf.protected` to disable channels that you aren't using for the ULN mirror itself. See Localizing Subscriptions for the ULN Mirror Server. This procedure disables the channels that don't apply to the system.

4. Switch to the unprotected version of `/etc/dnf/plugins/spacewalk.conf` that enables all channels required for the mirror. For example:

```
cp /etc/dnf/plugins/spacewalk.conf.unprotected /etc/dnf/plugins/
spacewalk.conf
```

5. Mirror the ULN Channels to the location of the base directory for the mirror. For example:

```
sudo dnf reposync --delete --download-metadata -p /var/www/html/yum
```

You can use the `--repoid` and `--exclude` options with the `reposync` command to control exactly which repositories you're mirroring and to help reduce disk space requirements by excluding source packages. For example:

```
sudo dnf reposync --delete --download-metadata -p /var/www/html/yum \
 --repoid ol10_x86_64_ksplice --exclude *.src,*.nosrc
sudo dnf reposync --delete --download-metadata -p /var/www/html/yum \
 --repoid ol10_x86_64_userspace_ksplice --exclude *.src,*.nosrc
sudo dnf reposync --delete --download-metadata -p /var/www/html/yum \
 --repoid ol9_x86_64_ksplice --exclude *.src,*.nosrc
sudo dnf reposync --delete --download-metadata -p /var/www/html/yum \
 --repoid ol9_x86_64_userspace_ksplice --exclude *.src,*.nosrc
sudo dnf reposync --delete --download-metadata -p /var/www/html/yum \
 --repoid ol8_x86_64_ksplice --exclude *.src,*.nosrc
sudo dnf reposync --delete --download-metadata -p /var/www/html/yum \
 --repoid ol8_x86_64_userspace_ksplice --exclude *.src,*.nosrc
```

6. Switch to the protected version of `/etc/dnf/plugins/spacewalk.conf` that enables only those channels used by the system hosting the mirror. For example:

```
sudo cp /etc/dnf/plugins/spacewalk.conf.protected /etc/dnf/plugins/
spacewalk.conf
```

7. Consider creating a cron script or systemd service and timer to regularly update the mirror. For example, create a file at `/etc/cron.daily/uln-mirror-update` that automatically switches to the unprotected version of `/etc/dnf/plugins/spacewalk.conf`, updates the mirror based on the channels enabled, then switches back to the protected version of `/etc/dnf/plugins/spacewalk.conf`. If the system is configured to use itself as a mirror, the local yum repository configuration must be disabled while the mirror is updated:

```
#!/bin/bash
####### Regularly update yum repos ######

# Change DNF configuration to allow all repositories
cp /etc/dnf/plugins/spacewalk.conf.unprotected /etc/dnf/plugins/
spacewalk.conf
```

```
# Check whether the system is configured as a client of itself
if [ -f /etc/yum.repos.d/local-yum.repo ]; then
   mv /etc/yum.repos.d/local-yum.repo /etc/yum.repos.d/local-
yum.repo.disabled;
fi

# Run the reposync. You can change this command to specify the
# repoid and exclusions that you want for a more customized mirror

dnf reposync --delete --download-metadata -p /var/www/html/yum

# Change DNF configuration to use protected repositories
cp /etc/dnf/plugins/spacewalk.conf.protected /etc/dnf/plugins/
spacewalk.conf

# Enable the yum configuration again
if [ -f /etc/yum.repos.d/local-yum.repo.disabled ]; then
   mv /etc/yum.repos.d/local-yum.repo.disabled /etc/yum.repos.d/local-
yum.repo;
fi
```

Ensure that the file is executable. For example:

```
sudo chmod +x /etc/cron.daily/uln-mirror-update
```

Localizing Subscriptions for the ULN Mirror Server

Localizing the ULN mirror's channel subscriptions for the server hosting the mirror prevents the mirror's packages incompatible with the host system from being updated that would cause package collisions and damage package dependencies.

Ensure that you have subscribed to required channels to serve clients running different OS versions on different platforms, as described in Configuring the Local ULN Mirror.

This task is required for ULN mirrors that serve heterogeneous clients. In this case, the mirror subscribes to multiple channels, including channels the host server itself doesn't need. You would need to configure the host server to prevent its own channel subscriptions from being updated with packages targeted for other clients.

Suppose that the server hosting the mirror is an Oracle Linux 9 system but the mirror is also serving Oracle Linux 10 clients and Oracle Linux 8 clients on the x86_64 platform. The following steps would localize the Oracle Linux 9's channel subscriptions such that only those channels applicable to Oracle Linux 9 are enabled when not updating the ULN mirror:

1. Identify the channels to which the server is subscribed.

   ```
   sudo dnf repolist
   ```

   ```
   ol8_x86_64_ksplice              Ksplice for Oracle Linux 8 (x86_64)
   ol8_x86_64_userspace_ksplice    Ksplice-aware user space packages for
   Oracle Linux 8 (x86_64)
   ol9_x86_64_ksplice              Ksplice for Oracle Linux 9 (x86_64)
   ol9_x86_64_userspace_ksplice    Ksplice-aware user space packages for
   Oracle Linux 9 (x86_64)
   ```

```
ol9_x86_64_UEKR7                Oracle Linux 9 UEK Release 7 (x86_64)
ol9_x86_64_appstream            Oracle Linux 9 Application Stream
Packages (x86_64)
ol9_x86_64_baseos_latest        Oracle Linux 9 BaseOS  Latest(x86_64)
ol10_x86_64_ksplice              Ksplice for Oracle Linux 10 (x86_64)
ol10_x86_64_userspace_ksplice    Ksplice-aware user space packages for
Oracle Linux 10 (x86_64)
```

In addition to the system's own Oracle Linux 9 channels, the output would include Oracle Linux 8 and Oracle Linux 10 channels intended for clients.

2. Edit `/etc/dnf/plugins/spacewalk.conf.protected` to disable channel updates inapplicable to the server and enable those that are applicable.

   Use the following format:

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

   For the current example, you would disable all Oracle Linux 8 and Oracle Linux 10 channels:

   ```
   [ol8_x86_64_ksplice]
   enabled = 0

   [ol8_x86_64_userspace_ksplice]
   enabled = 0

   [ol10_x86_64_ksplice]
   enabled = 0

   [ol10_x86_64_userspace_ksplice]
   enabled = 0

   [ol9_x86_64_ksplice]
   enabled = 1

   [ol9_x86_64_userspace_ksplice]
   enabled = 1

   [ol9_x86_64_UEKR7]
   enabled = 1

   [ol9_x86_64_appstream]
   enabled = 1

   [ol9_x86_64_baseos_latest]
   enabled = 1
   ```

> **Note**
>
> If you subsequently subscribe the system to any other incompatible channels on ULN, you must also disable those channels in `/etc/dnf/plugins/spacewalk.conf`.

3. Edit `/etc/dnf/plugins/spacewalk.conf.unprotected` to enable all channels intended for the mirror.

   Use the following format:

   ```
   [repo_id]
   enabled=1
   ```

   For the current example, you would enable all channels:

   ```
   [ol8_x86_64_ksplice]
   enabled = 1

   [ol8_x86_64_userspace_ksplice]
   enabled = 1

   [ol10_x86_64_ksplice]
   enabled = 1

   [ol10_x86_64_userspace_ksplice]
   enabled = 1

   [ol9_x86_64_ksplice]
   enabled = 1

   [ol9_x86_64_userspace_ksplice]
   enabled = 1

   [ol9_x86_64_UEKR7]
   enabled = 1

   [ol9_x86_64_appstream]
   enabled = 1

   [ol9_x86_64_baseos_latest]
   enabled = 1
   ```

Installing the Ksplice Client From ULN

> **Note**
>
> If using Oracle Cloud Infrastructure (OCI), Ksplice is already installed by default (on all Oracle Linux instances created after August 25, 2017). For more information, see Oracle Ksplice on Oracle Cloud Infrastructure.

Before installing the Ksplice client:

- Verify that the system is running a supported Oracle Linux release with a supported version of either the Unbreakable Enterprise Kernel (UEK) or the Red Hat Compatible Kernel (RHCK) installed.
  Use the `uname -a` command to verify the kernel version. See Maintained Kernels. Ksplice applies updates to the running kernel *only*, so ensure that the running kernel is the one you want to update.
- For an online client, register the system with ULN and verify it has a connection to the Oracle Uptrack server.
- For an offline client, configure a local ULN mirror.

The following procedure applies *only* to Oracle Linux releases.

1. Subscribe to the necessary channels on ULN.

   See Updating ULN Subscriptions.

2. (Optional) Configure proxy settings.

   If you use an Internet proxy, you can configure the HTTP and HTTPS settings for the proxy in the shell as follows:

   - For the `sh`, `ksh`, or `bash` shells, use commands such as the following:

     ```
     sudo http_proxy=http://proxy_URL:http_port
     sudo https_proxy=http://proxy_URL:https_port
     sudo export http_proxy https_proxy
     ```

     For the `csh` shell, use commands such as the following:

     ```
     sudo setenv http_proxy=http://proxy_URL:http_port
     sudo setenv https_proxy=http://
     proxy_URL:https_port
     ```

3. Revert prelinked binaries and remove the `prelink` package.

   If `prelink` is installed, revert all the prelinked binaries and any dependent libraries to their original state, then remove the `prelink` package:

   ```
   sudo prelink -au
   sudo dnf remove prelink
   ```

   > **Note**
   >
   > `prelink` isn't installed and enabled by default Oracle Linux systems.

4. Install the `ksplice` package.

- Oracle Linux 10
- Oracle Linux 9
- Oracle Linux 8

- Oracle Linux 7

Oracle Linux 10

– For the Ksplice online client, use the following command:

```
sudo dnf install -y ksplice
```

– For the Ksplice offline client, use the following command:

```
sudo dnf install -y ksplice ksplice-offline
```

> **Note**
>
> No `uptrack` and `uptrack-offline` packages are available on Oracle Linux 10. The Uptrack clients are now included in the `ksplice` and `ksplice-offline` packages. The Uptrack clients are deprecated on Oracle Linux 10 and might be removed in a future release.

Oracle Linux 9

– For the Ksplice online client, use the following command:

```
sudo dnf install -y ksplice uptrack
```

– For the Ksplice offline client, use the following command:

```
sudo dnf install -y ksplice ksplice-offline uptrack-offline
```

Oracle Linux 8

– For the Ksplice online client, use the following command:

```
sudo dnf install -y ksplice uptrack
```

– For the Ksplice offline client, use the following command:

```
sudo dnf install -y ksplice ksplice-offline uptrack-offline
```

Oracle Linux 7

– For the Ksplice online client, use the following command:

```
sudo yum install -y ksplice uptrack
```

– For the Ksplice offline client, use the following command:

```
sudo yum install -y ksplice ksplice-offline uptrack-offline
```

The following packages are installed on the system:

`ksplice-core`
Contains the shared user space libraries, such as `glibc` and `openssl`, that support Ksplice patching.

`ksplice-helper`
Contains a helper library that enables user space executables to be patched by Ksplice.

`ksplice-helper-devel`
Contains the development environment for creating user space libraries that support Ksplice patching.

`ksplice-tools`
Contains the `ksplice` executable and `ksplice(8)` manual page.

5. Verify the Ksplice access key in the Uptrack configuration file.

   The access key for Ksplice Uptrack is retrieved from ULN and added to the `/etc/uptrack/uptrack.conf` file, as shown in the following example:

   ```
   [Auth]
   accesskey =
   0e1859ad8aea14b0b4306349142ce9160353297daee30240dab4d61f4ea4e59b
   ```

   You might need to manually add the key to the configuration file, if the system is unable to retrieve it from ULN. After install, check that the key is added to the configuration file and add it manually if it isn't present. To view the access key for the system, go to the **Systems** tab in the ULN web interface, or check https://status-ksplice.oracle.com/status.

6. Update the system.

   Update the system to install the Ksplice-aware versions of the user space libraries:

   ```
   sudo dnf update
   ```

- Oracle Linux 10
- Oracle Linux 9
- Oracle Linux 8
- Oracle Linux 7

Oracle Linux 10

To install only the libraries and not update any other packages, limit the update to the following channels, as appropriate:

- `ol10_x86_64_userspace_ksplice`
- `ol10_aarch64_userspace_ksplice`

For example, you would update the packages for the Oracle Linux Ksplice user-aware channels as follows:

```
sudo dnf --disablerepo=* --enablerepo=ol10_x86_64_userspace_ksplice update
```

Oracle Linux 9

To install only the libraries and not update any other packages, limit the update to the following channels, as appropriate:

- `ol9_x86_64_userspace_ksplice`
- `ol9_aarch64_userspace_ksplice`

For example, you would update the packages for the Oracle Linux Ksplice user-aware channels as follows:

```
sudo dnf --disablerepo=* --enablerepo=ol9_x86_64_userspace_ksplice update
```

Oracle Linux 8

To install only the libraries and not update any other packages, limit the update to the following channels, as appropriate:

- `ol8_x86_64_userspace_ksplice`
- `ol8_aarch64_userspace_ksplice`

For example, you would update the packages for the Oracle Linux Ksplice user-aware channels as follows:

```
sudo dnf --disablerepo=* --enablerepo=ol8_x86_64_userspace_ksplice update
```

Oracle Linux 7

To install only the libraries and not update any other packages, limit the update to the following channels, as appropriate:

- `ol7_x86_64_userspace_ksplice`
- `ol7_aarch64_userspace_ksplice`

For example, you would update the packages for the Oracle Linux Ksplice user-aware channels as follows:

```
sudo yum --disablerepo=* --enablerepo=ol7_x86_64_userspace_ksplice update
```

You can also use the `glibc*` and `openssl*` syntax with the `install` command for the package manager. To use this client to perform kernel updates, install it in the same way that you use the standard Uptrack client, for example:

```
sudo dnf install uptrack-updates-$(uname -r)
```

7. (Optional) Enable automatic installation of updates.

To enable the automatic installation of updates, change the entry in the `/etc/uptrack/uptrack.conf` file from `no` to `yes`:

```
autoinstall = yes
```

8. Reboot the system.

   Reboot the system for the changes to take effect.

   ```
   sudo systemctl reboot
   ```

Review the Ksplice configuration and update any other options that you might want to set. See Ksplice Client Configuration Options for more information.

3

Using the Ksplice Enhanced Client

Limitations of the Ksplice Enhanced Client

Be aware of the following important Oracle Ksplice limitations:

- Ksplice reports an error similar to the following if it can't apply updates to processes that don't have access to the `/var/cache/ksplice` directory:

  ```
  Ksplice was unable to load the update as the target process is in a
  different mount namespace or has changed root.  The service must be
  restarted to apply on-disk updates.
  Extra information: the process has changed root or mount namespace.
    └── rtkit-daemon (3680)
  ```

  This error might occur with processes that use `chroot` or those that run in a container. In such cases, you must restart the process to apply any available updates. For example, to restart the `rtkit-daemon` service, you would use the `systemctl restart rtkit-daemon` command.

  To avoid restarting a `chrooted` application that you maintain and compile, ensure that the `/var/cache/ksplice` directory is bind-mounted in the `chrooted` environment.

- Ksplice can't patch applications that use either `setcontext` or `swapcontext` from `glibc` to perform user space context switching between process threads.
- Because of certain kernel limitations, Ksplice doesn't patch the `init` process (PID 1).

  On Oracle Linux 7 and later, the `init` process, which is `systemd`, is automatically run again on system updates, so it doesn't require patching with Ksplice.

Using the ksplice Command For the Ksplice Enhanced Client

Summary

You manage the Ksplice Enhanced client by using the `ksplice` command. The `ksplice` command can perform user space patching, in addition to kernel patching.

Usage

The `ksplice` command performs actions on the following subsystems:

- `kernel`: action is performed on the kernel subsystem only
- `user`: action is performed on the user space subsystem only
- `all`: : action is performed on all subsystems

Actions, in the form of subcommands include:

- `list-target`: list the available targets that can be patched by the client
- `show`: show updates that have already been applied by the client
- `apply`: apply an update to the system specified by an update path
- `undo`: undo an update to the system specified by a unique Ksplice identifier
- `upgrade`: update the system with all available Ksplice updates
- `remove`: remove updates either by specified Ksplice identifiers or by using the `--all` option to remove all updates.

Command syntax is as follows:

```
ksplice [OPTIONS] SUBSYSTEM SUBCOMMAND
```

See the `ksplice(8)` manual page for more information.

Ksplice Subcommands

- **List targets.**

  To display all the running user space processes that the client can patch, use the `ksplice all list-targets` command, for example:

  ```
  sudo ksplice all list-targets
  ```

  Output might appear as follows:

  ```
  User-space targets:

  glibc-libm-2.34.100.0.1.ksplice1.el9_4.2:
   - crond (46435)
   - ksplice (51778)

  glibc-libc-2.34.100.0.1.ksplice1.el9_4.2:
   - crond (46435)
   - ksplice (51778)
   - less (51781)
  ```

```
openssl-libssl-3.0.7.27.0.3.ksplice1.el9:
 - ksplice (51778)

openssl-libcrypto-3.0.7.27.0.3.ksplice1.el9:
 - ksplice (51778)


Kernel version: Linux/x86_64/5.15.0-206.153.7.el9uek.x86_64/#2 SMP Thu May
9 15:59:05 PDT 2024
```

For each Ksplice-aware library, the command reports the running processes that would be affected by an update. The command also reports the effective version of the loaded kernel.

- **Show updates.**

  To display the updates that have been applied to the system, use the `ksplice all show` command:

  ```
  sudo ksplice all show
  ```

  Output might appear as follows:

  ```
  Ksplice user-space updates:
  chronyd (705)
  httpd (1503)
     ├── [h73qvumn]: CVE-2014-7817: Command execution in wordexp().
     └── [ml55ngz4]: CVE-2015-1781: Privilege escalation in gethostbyname_r().

  Ksplice kernel updates:
  Installed updates:
  [nf9nfyzj] Enablement update for live patching.
  [fe2qyrtu] Denial-of-service when checking if an address is a jump label.
  [bvjiimlr] Enable livepatching of jump labels.
  [id9g0y8c] Known exploit detection.
  [aq4p03vt] Known exploit detection for CVE-2019-9213.
  [pjd4ekqc] Known exploit detection for CVE-2017-1000253.
  [syt1v7t7] Known exploit detection for CVE-2022-0847.
  [rpa4ixvy] Known exploit detection for CVE-2022-27666.
  [hisf1nu9] Known exploit detection for CVE-2016-5195.
  ...
  [gsf5wlo8] CVE-2024-36934: Information leak in QLogic BR-series Ethernet
  driver.
  [e12zrdy5] CVE-2024-36919: Denial-of-service in QLogic Fiber-Channel-over-
  Ethernet offload driver.
  [ednh9erf] CVE-2024-36904: Remote code execution in TCP/IP networking
  stack.
  [8vkhpraf] CVE-2024-27398: Denial-of-service in Bluetooth Classic (BR/EDR)
  features.

  Effective kernel version is 5.15.0-208.159.3.el9uek
  ```

The command reports the updates that have been applied to running processes, and the updates to the kernel. In the example output, Ksplice applied updates for `CVE-2014-7817` and `CVE-2015-1781` to some user space processes.

To restrict the scope of the `ksplice` command to user space updates or kernel updates, specify `user` or `kernel` instead of `all` with the command.

To display the updates that have been applied to a process specified by its PID, use the `--pid=$PID` option with the `ksplice user show` command:

```
sudo ksplice user show --pid=705
```

Output similar to the following is displayed:

```
chronyd (705)
   ├── [h73qvumn]: CVE-2014-7817: Command execution in wordexp().
   └── [ml55ngz4]: CVE-2015-1781: Privilege escalation in gethostbyname_r().
```

- **Remove updates.**

  Use the `remove` subcommand to remove all the updates from a process, for example:

  ```
  sudo ksplice user remove --all --pid=705
  ```

  To remove a specific update that Ksplice has applied to a process, use the `undo` subcommand:

  ```
  sudo ksplice user undo --pid=705 h73qvumn
  ```

  > **Note**
  >
  > You can prevent Ksplice from patching specified executables and libraries. See Preventing the Ksplice Enhanced Client From Patching User Space Processes and Libraries.

  Ksplice patches are stored in the `/var/cache/uptrack` directory. Following a reboot, Ksplice automatically reapplies these patches early in the boot process before the network is configured so that the system is hardened before any remote connections can be established.

- **List and install available updates.**

  To list all the available Ksplice updates, use the `upgrade` subcommand:

  ```
  sudo ksplice -n kernel upgrade
  ```

  To install all the available Ksplice updates, use the `upgrade` subcommand as follows:

  ```
  sudo ksplice -y user upgrade
  ```

- **Show kernel version.**

  After Ksplice applies updates to a running kernel, the kernel has an effective version that's different than the original boot version displayed by the `uname -a` command.

Use the `ksplice kernel uname -r` command to display the effective version of the kernel:

```
sudo ksplice kernel uname -r
```

The `ksplice kernel uname` command supports the commonly used `uname` flags, including `-a` and `-r`, and also provides a way for applications to detect that the kernel has been patched. The effective version is based on the version number of the latest patch that Ksplice Uptrack has applied to the kernel.

Example 3-1 Example Usage

The following examples show ways in which you can view information about Ksplice updates and administer Ksplice updates on a system.

View the updates that Ksplice Uptrack has made to the running kernel:

```
sudo ksplice kernel show
```

View the updates that are available to be installed:

```
sudo ksplice kernel show --available
```

Remove all updates from the kernel:

```
sudo ksplice kernel remove --all
```

Prevent Ksplice from reapplying the updates at the next system reboot, create the empty file `/etc/uptrack/disable`:

```
touch /etc/uptrack/disable
```

Or, you can specify `nouptrack` as a parameter on the boot command line when you next restart the system.

Preventing the Ksplice Enhanced Client From Patching User Space Processes and Libraries

If you don't want Ksplice to patch the user space processes for certain executables or libraries, you can specify the information in a `/etc/ksplice/blacklist.d` configuration file. The following is an example of a `localblacklist.conf` file. The example shows how you would prevent Ksplice from patching any process that corresponds to any executable in the `/opt/app/bin` or `/usr/local/bin` directory, or from patching any shared library with a name matching `liblocal-*`.

The following example shows the format of the rules, which are Python regular expressions:

```
[executables]
^/opt/apt/bin/.*$
^/usr/local/bin/.*$
```

```
[targets]
^liblocal-.*$
```

Configuring the Ksplice Enhanced Client for Offline Mode

Before configuring an offline client, you must set up a local ULN mirror that can act as a Ksplice mirror. See Configuring a Local Ksplice Mirror for the Offline Client for more information.

The offline version of the Ksplice Enhanced Client removes the requirement that a server on an intranet has a direct connection to the Oracle Uptrack server or ULN. For more information about running Ksplice offline, see About Ksplice Offline Mode.

The following procedure describes how to configure the Ksplice Enhanced Client for offline mode.

1. Import the GPG key.

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

2. Disable existing yum repositories in `/etc/yum.repos.d`.

   You can either edit any existing repository files and disable all the entries by setting `enabled=0`; or, you can use `dnf config-manager`, for example:

   ```
   sudo dnf config-manager --disable \*
   ```

   Or, you can rename any of the files in this directory so that they don't use the `.repo` suffix. This change causes the `dnf` command to ignore these entries, as shown in the following example:

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

3. Create a `local-yum.repo` configuration for the system to use the local ULN mirror.

   In the `/etc/yum.repos.d` directory, create the `local-yum.repo` file, which contains entries such as the following for an Oracle Linux 9 yum client:

   ```
   [local_ol9_x86_64_ksplice]
   name=Ksplice for Oracle Linux $releasever - $basearch
   baseurl=http://local_uln_mirror/yum/OracleLinux/OL9/ksplice/$basearch/
   gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-oracle
   gpgcheck=1
   enabled=1

   [local_ol9_x86_64_ksplice_userspace]
   name=Ksplice aware userspace packages for Oracle Linux $releasever
   - $basearch
   baseurl=http://local_uln_mirror/yum/OracleLinux/OL9/userspace/
   ksplice/$basearch/
   gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-oracle
   gpgcheck=1
   enabled=1
   ```

```
[local_ol9_baseos_latest]
name=Oracle Linux 9 BaseOS Latest ($basearch)
baseurl=http://local_uln_mirror/yum/OracleLinux/OL9/baseos/
latest/$basearch/
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-oracle
gpgcheck=1
enabled=1

[local_ol9_appstream]
name=Oracle Linux 9 Application Stream Packages ($basearch)
baseurl=http://local_uln_mirror/yum/OracleLinux/OL9/appstream/$basearch/
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-oracle
gpgcheck=1
enabled=1

[local_ol9_addons]
name=Oracle Linux $releasever - $basearch - addons
baseurl=http://local_uln_mirror/yum/OracleLinux/OL9/addons/$basearch/
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-oracle
gpgcheck=1
enabled=1
```

- Replace *local_uln_mirror* with the IP address or resolvable host name of the local ULN mirror.
- To distinguish the local repositories from the ULN repositories, optionally prefix the labels for each entry with a string such as `local_`. Note that you must also edit the uptrack configuration, described later in this procedure.
- The example configuration enables the `local_ol9_x86_64_ksplice`, `local_ol9_x86_64_ksplice_userspace`, `local_ol9_baseos_latest`, `local_ol9_appstream`, and `local_ol9_addons` channels.

4. Test the configuration.

   a. Clear the yum metadata cache.

   ```
   sudo dnf clean metadata
   ```

   b. Verify the configuration.

   ```
   sudo dnf repolist
   ```

   If the `dnf` commands can't 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 (port 80).

5. Remove prelink.

   If `prelink` is installed, revert all the prelinked binaries and dependent libraries to the original states and then remove the `prelink` package as follows:

   ```
   prelink -au
   sudo dnf remove prelink
   ```

   The `prelink` package isn't installed and enabled by default on Oracle Linux 7, Oracle Linux 8 or Oracle Linux 9.

6. Install the offline version of the enhanced client package.

   ```
   sudo dnf install ksplice-offline
   ```

7. Update the Uptrack configuration in `/etc/uptrack/uptrack.conf` for channel prefixes.

   Add a configuration directive to the `/etc/uptrack/uptrack.conf` file to provide the enhanced client with the label of the local, user space channel in the local ULN mirror yum repository.

   > **Note**
   >
   > You can skip this step if you didn't use the `local_` prefix for the channel label, and this label is an exact match of the label that's used on ULN. If you used the `local_` prefix or labeled this channel differently, add the following lines, but instead of *local_ol9_x86_64_ksplice_userspace*, specify the same label that you used for the Ksplice user space channel, for example:
   >
   > ```
   > [User]
   > yum_userspace_ksplice_repo_name =
   > local_ol9_x86_64_ksplice_userspace
   > ```

8. Install packages for offline updates.

   To install offline update packages, install the relevant packages, for example:

   ```
   sudo dnf install ksplice-updates-glibc ksplice-updates-openssl
   ```

   After you have installed these packages, the offline version of the enhanced client behaves the same way as the online version.

9. Update the system.

   Update the system to install the Ksplice-aware versions of the user space libraries:

   ```
   sudo dnf update
   ```

   To install only the libraries and not any other packages, limit the update to the Ksplice user space channel, for example, `ol9_x86_64_userspace_ksplice` channel:

   ```
   sudo dnf --disablerepo=* --enablerepo=ol9_x86_64_userspace_ksplice update
   ```

   Or, you can use the following command:

   ```
   sudo dnf update *glibc *openssl*
   ```

   You might also use this client to perform kernel updates in the same way that you can use the standard uptrack client:

   ```
   sudo dnf install uptrack-updates-$(uname -r)
   ```

10. (Optional) Enable automatic installation of updates.

To enable the automatic installation of updates, change the entry in `/etc/uptrack/uptrack.conf` from `no` to `yes`, as shown in the following example:

```
autoinstall = yes
```

11. Reboot the system.

    Reboot the system for the changes to take effect.

    ```
    sudo systemctl reboot
    ```

Using the Known Exploit Detection Feature on the Ksplice Enhanced Client

> **Note**
>
> Known exploit detection support is available for the Ksplice Enhanced client only. On the 64-bit ARM (aarch64) platform, known exploit detection requires Oracle Linux 8 or later, with the UEK7 (5.15.0) or later kernel.

Oracle provides the known exploit detection feature for supported systems that have the Ksplice Enhanced client installed. This feature reports attempted exploitation by known attack vectors. When new Common Vulnerabilities and Exposures (CVEs) are discovered and patched with Ksplice, Oracle might add tripwires to the code that fire when an erroneous condition is triggered, thus enabling you to monitor systems for suspicious activity.

> **Note**
>
> Because not all security issues have tripwires added, and also because it's possible to trigger tripwires under normal operations, further analysis of erroneous conditions might be required.

Running Known Exploit Detection on the Ksplice Enhanced Client

You can run the Ksplice known exploit detection on supported Oracle Linux systems that have the Ksplice Enhanced client installed. This feature works for both the online and offline Ksplice Enhanced client.

To run known exploit detection with the default configuration:

1. Install the `ksplice-known-exploit-detection` package.

   ```
   sudo dnf install ksplice-known-exploit-detection
   ```

2. Update the uptrack configuration to enable known exploit detection.

Add the following lines to the `/etc/uptrack/uptrack.conf` file:

```
[Known-Exploit-Detection]
enabled = yes
```

3. Enable known exploit detection in the running kernel.

   Enable the feature by running the `kernel upgrade` command:

   ```
   sudo ksplice kernel upgrade
   ```

4. Verify that known exploit detection is running in the kernel.

   Verify that the feature has been enabled for the current kernel:

   ```
   cat /proc/sys/kernel/known_exploit_detection
   ```

   If the value is `0` or the file is missing, then the kernel hasn't enabled kernel exploit detection. If the value is `1`, known exploit detection is enabled on the system.

The helper file, `/usr/sbin/log-known-exploit`, is invoked directly by the kernel. To invoke the help manually to check the configuration or perform dry-run tests, use the following command:

```
/usr/sbin/log-known-exploit --help
```

You can specify the following extra options and arguments with this command:

-h, --help
Display the help message and exit.

-c, --config *`/etc/example.conf`*
Specify a compatible configuration file. Defaults to `/etc/log-known-exploit.conf`.

-f, --force
Run the command without checking for root permissions.

-n, --dry-run
Simulate the output and expected actions that would be performed by the helper file.

-d, --dummy
Use sample data to verify that report logging is configured correctly.

Configuring Known Exploit Logging and Email Notification Options

Configuration options for the known exploit detection feature are set in `/etc/log-known-exploit.conf`. You can edit the configuration file to control the following behaviors.

- To set up email alerts, edit the `[email]` section to enable the functionality and to provide a recipient email address for delivery:

  ```
  [email]
  enabled: 1
  recipients: admin@example.com
  ```

  The default configuration for the Ksplice known exploit detection feature logs exploit attempts to `syslog` by using the normal `syslog` facilities.

> **Note**
>
> Email alerts require that the system is already configured to handle outbound or local mail delivery using a mail transfer agent.

- To define the logging behavior for tripwires that aren't specified, add a value for `default` to the list.

  For example, to avoid logging any tripwire reports unless they're specified, do the following:

  ```
  [actions]
  default: ignore
  ```

- To specify which tripwire reports must be logged or ignored, add rules to the `[actions]` configuration section.

  For example, list the CVEs to be reported and the CVEs that can be ignored:

  ```
  [actions]
  CVE-2024-12345: report
  CVE-2024-12346: ignore
  ```

  Note that these configuration entries override the default configuration.
  You can also temporarily disable or enable tripwires for different CVEs, to override the stored configuration. See Temporarily Disabling and Enabling Tripwires.

Temporarily Disabling and Enabling Tripwires

You can disable or enable a specific tripwire manually when you need to troubleshoot.

- Temporarily *disable* a tripwire.

  To disable a specific tripwire until the next reboot, remove the CVE reference from the `/proc/sys/kernel/known_exploit_detection_tripwires` file as follows:

  ```
  echo -n '-CVE-2024-12345' | sudo tee /proc/sys/kernel/
  known_exploit_detection_tripwires
  ```

- Temporarily *enable* a tripwire.

  To enable a specific tripwire, append the CVE reference to the same configuration file again:

  ```
  echo -n '+CVE-2024-12345' | sudo tee /proc/sys/kernel/
  known_exploit_detection_tripwires
  ```

Removing the Ksplice Enhanced Client Software

The following procedure describes how to remove the Ksplice Enhanced Client from a system.

1. Remove the Ksplice Enhanced client software.

   ```
   sudo dnf -y remove ksplice
   ```

2. Remove the offline version of the Ksplice Enhanced client software, if installed.

   ```
   sudo dnf -y remove ksplice-offline
   ```

3. Remove the Ksplice-aware versions of the `glibc+openssl` packages from the system.

   a. Unsubscribe all the subscribed Ksplice-aware user space channels from the yum repository.

   b. Manually downgrade the Ksplice-aware packages using the dnf shell.

   ```
   dnf shell
   > erase ksplice-helper
   > downgrade glibc* openssl*
   > run
   ```

   Note

   The following single command performs the same downgrade action without needing manual entry and can be used for automation purposes:

   ```
   printf 'erase ksplice-helper\n downgrade glibc* openssl*\n run'
   | dnf -y shell
   ```

4

Using the Ksplice Uptrack Client

> **Note**
>
> The Ksplice Uptrack Client is deprecated on Oracle Linux 10 and might be removed in a future release. Although Ksplice Uptrack Client commands continue to work when you install the Ksplice Enhanced Client package, no `uptrack` package is available on Oracle Linux 10. We recommend using the Ksplice Enhanced Client on all systems.

Using Uptrack Commands to Manage Ksplice Updates

Summary

Several `uptrack` commands are available to manage the Ksplice Uptrack Client. This reference provides an overview of the commands available and how to use them.

For the Enhanced Client, see Using the ksplice Command For the Ksplice Enhanced Client.

Uptrack Commands

The following uptrack commands are available:

- `uptrack-install`: downloads and installs updates specified by identifiers
- `uptrack-remove`: removes updates specified by identifiers
- `uptrack-show`: shows the list of installed updates; or shows the details for an update specified by an identifier
- `uptrack-uname`: show information about the effective kernel
- `uptrack-upgrade`: lists, downloads, and installs the latest available updates

For more information about each of these commands see the `uptrack(8)` manual page.

Example 4-1 Example Usage

- List all available updates.

```
sudo uptrack-upgrade -n
```

- Install all available Ksplice updates.

  ```
  sudo uptrack-upgrade -y
  ```

- Display the effective version of the kernel.

  ```
  sudo uptrack-uname -r
  ```

  You can compare this to the original boot version displayed by the `uname -a` command.

  The `uptrack-uname` command supports commonly used `uname` flags, including `-a` and `-r`, and also provides a way for applications to detect that the kernel has been patched. The effective version is based on the version number of the latest patch that Ksplice has applied to the kernel.

- View updates made to running kernel

  ```
  uptrack-show
  ```

- View the updates that are available for installation.

  ```
  uptrack-show --available
  ```

- Remove all the updates from the kernel.

  ```
  uptrack-remove --all
  ```

- Prevent Ksplice from reapplying the updates at the next system reboot and create the empty file `/etc/uptrack/disable`.

  ```
  touch /etc/uptrack/disable
  ```

  Or, you can specify the `nouptrack` argument as a parameter on the boot command line when you next reboot the system.

Updating the Ksplice Uptrack Client to a Specific Effective Kernel Version

You might want to limit the set of updates that `uptrack-upgrade` installs. For example, the security policy at a site might require a senior administrator to approve Ksplice updates before you can install these updates on production systems. In such cases, you can direct `uptrack-upgrade` to upgrade to a specific effective kernel version instead of the latest available version.

> **Note**
>
> You can only select a specific effective version when using the offline Ksplice client and offline update RPM packages. This ability keeps production systems at a tested update level temporarily, while the latest updates are tested in an integration or UAT environment.

1. Install the `uptrack-updates` package for the current kernel.

   ```
   sudo dnf -y install uptrack-updates-$(uname -r)
   ```

   > **Important**
   >
   > If you have booted the most recent available kernel and no Ksplice updates are available, this command might fail or might return an error message notifying you that the kernel version isn't yet supported by Ksplice Uptrack. This command only succeeds when Ksplice updates are available for the kernel that the system is running.

2. Use the `uptrack-uname -r` command to display the current effective kernel version.

   ```
   sudo uptrack-uname -r
   ```

3. List all effective kernel versions.

   To list all the effective kernel versions that are available, specify the `--list-effective` option to the `uptrack-upgrade` command, for example:

   ```
   sudo uptrack-upgrade --list-effective
   ```

   Output similar to the following is displayed:

   ```
   Available effective kernel versions:

   5.15.0-205.149.5.4.el9uek.x86_64/#2 SMP Wed May 8 15:31:38 PDT 2024
   5.15.0-206.153.7.1.el9uek.x86_64/#2 SMP Wed May 22 20:24:12 PDT 2024
   5.15.0-207.156.6.el9uek.x86_64/#2 SMP Thu Jun 6 02:32:40 PDT 2024
   ```

4. Remove installed updates.

   Remove the installed updates to revert the effective kernel version to the earliest that's available, which is 205.149.5.4 in the following example:

   ```
   sudo uptrack-remove --all
   sudo uptrack-uname -r
   ```

   The current effective kernel version is displayed:

   ```
   5.15.0-205.149.5.4.el9uek.x86_64
   ```

5. Set the effective kernel version.

   You can set the effective kernel version that you want the system to use by using either of the following methods:

   - Specify the `--effective` option to the `uptrack-upgrade` command.

For example, to update from 205.149.5 to 206.153.7.1 instead of updating to the latest 207.156.6, use the `--effective` option to specify 206.153.7.1:

```
sudo uptrack-upgrade --effective="5.15.0-206.153.7.1.el9uek.x86_64/#2
SMP Wed May 22 20:24:12 PDT 2024"
```

The effective kernel version is displayed after the upgrade:

```
...
Effective kernel version is 5.15.0-206.153.7.1.el9uek
```

You can check that the effective kernel version matches:

```
sudo uptrack-uname -r
```

Output similar to the following is displayed:

```
5.15.0-206.153.7.1.el9uek.x86_64
```

This method is suitable for setting the effective kernel version on individual systems.

- Use the `effective_version` option in the `/etc/uptrack/uptrack.conf` file to set an effective package version for the `uptrack-upgrade` command. This method works the same as specifying `--effective` on the command line.

  Because `uptrack-upgrade` runs automatically whenever you update the `uptrack-updates` package on a system, the following entry would limit the effective kernel version to 206.153.7.1:

  ```
  effective_version = 5.15.0-206.153.7.1.el9uek.x86_64/#2 SMP Wed May 22
  20:24:12 PDT 2024
  ```

  This method is convenient for setting the effective version for a package on several production systems, where the content of the `/etc/uptrack/uptrack.conf` file can be obtained from a centrally maintained primary copy.

Switching Between Online and Offline Ksplice Uptrack Installation Modes

To switch from one Ksplice client software version (or mode) to another Ksplice software version, for example, switch from a Ksplice online installation to a Ksplice offline installation, you must first remove the existing Ksplice client software from the system. You can then install the new version of the Ksplice client software.

> ⚠ **Caution**
>
> Failure to remove an existing Ksplice client software version before installing a new Ksplice client software version results in transaction check errors during the package installation process.

- Switch to the *offline* client.

  If you have the Ksplice Uptrack client software installed on the system and you want to install the offline version of the Ksplice Enhanced client software, you would need to first remove the Ksplice Uptrack client software, and then install the Ksplice offline Enhanced client software, for example:

  ```
  sudo dnf remove uptrack ksplice-tools
  sudo dnf install ksplice-offline
  ```

- Switch to the *online* client.

  To switch from an offline installation to an online installation, for example, to switch from the offline Ksplice Uptrack client software to the Ksplice Uptrack (online) client software, use the following commands:

  ```
  sudo dnf remove ksplice-offline ksplice-tools
  sudo dnf install uptrack
  ```

Configuring Ksplice Uptrack Clients for Offline Mode

Before configuring an offline client, you must set up a local ULN mirror that can act as a Ksplice mirror. See Configuring a Local Ksplice Mirror for the Offline Client for more information.

The offline Ksplice client eliminates the need for having a server on an intranet that has a direct connection to the Oracle Uptrack server. Also, a Ksplice offline client doesn't require a network connection to apply the update package to the kernel. For example, you could use the `dnf` command to install the update package directly from a memory stick. For more information about running Ksplice offline, see About Ksplice Offline Mode.

The following procedure describes how to configure the Ksplice Uptrack client for offline mode.

1. Import the GPG key.

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

2. Disable existing yum repositories in `/etc/yum.repos.d`.

   You can either edit any existing repository files and disable all the entries by setting `enabled=0`; or, you can use `dnf config-manager`, for example:

   ```
   sudo dnf config-manager --disable \*
   ```

   Or, you can rename any of the files in this directory so that they don't use the `.repo` suffix. This change causes the `dnf` command to ignore these entries, as shown in the following example:

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

3. Create a `local-yum.repo` configuration for the system to use the local ULN mirror.

In the `/etc/yum.repos.d` directory, create the `local-yum.repo` file, which contains entries such as the following for an Oracle Linux 9 yum client:

```
[local_ol9_x86_64_ksplice]
name=Ksplice for Oracle Linux $releasever - $basearch
baseurl=http://local_uln_mirror/yum/OracleLinux/OL9/ksplice/$basearch/
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-oracle
gpgcheck=1
enabled=1

[local_ol9_baseos_latest]
name=Oracle Linux 9 BaseOS Latest ($basearch)
baseurl=http://local_uln_mirror/yum/OracleLinux/OL9/baseos/
latest/$basearch/
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-oracle
gpgcheck=1
enabled=1

[local_ol9_appstream]
name=Oracle Linux 9 Application Stream Packages ($basearch)
baseurl=http://local_uln_mirror/yum/OracleLinux/OL9/appstream/$basearch/
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-oracle
gpgcheck=1
enabled=1

[local_ol9_addons]
name=Oracle Linux $releasever - $basearch - addons
baseurl=http://local_uln_mirror/yum/OracleLinux/OL9/addons/$basearch/
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-oracle
gpgcheck=1
enabled=1
```

- Replace *local_uln_mirror* with the IP address or resolvable host name of the local ULN mirror.
- To distinguish the local repositories from the ULN repositories, optionally prefix the labels for each entry with a string such as `local_`.
- The example configuration enables the `local_ol9_x86_64_ksplice`, `local_ol9_baseos_latest`, `local_ol9_appstream`, and `local_ol9_addons` channels.
- **Note**

  The Ksplice offline client package is unable to install user space updates, so don't enable any `*_userspace_ksplice` channels unless you intend to use the offline version of the Ksplice Enhanced client.

4. Install the Ksplice offline client package.

   ```
   sudo dnf -y install uptrack-offline
   ```

5. Test the configuration.

a. Clear the yum metadata cache.

```
sudo dnf clean metadata
```

b. Verify the configuration.

```
sudo dnf repolist
```

If the `dnf` commands can't 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 (port 80).

6. Install the Ksplice updates that are available for the kernel.

```
sudo dnf -y install uptrack-updates-$(uname -r)
```

As new Ksplice updates are made available, use the same command to pick up and apply these updates.

Consider creating a systemd service unit to install the correct uptrack updates for the running kernel at boot time and a timer unit to run the service daily. The following procedure describes how to set up such a service and enable it.

a. Create a systemd service unit configuration file.

Create a systemd unit file at `/etc/systemd/system/uptrack-update-packages.service`, to include the following configuration:

```
[Unit]
Description=Update uptrack updates for running kernel
After=network-online.target
Wants=network-online.target

[Service]
Type=oneshot
ExecStart=/bin/bash -c 'dnf -y install uptrack-updates-$(uname -r)'
```

The unit file starts after the network is online, so that the `dnf` command can run and complete.

b. Create a systemd timer unit configuration file.

Create a systemd unit file at `/etc/systemd/system/uptrack-update-packages.timer`, to include the following configuration:

```
[Unit]
Description=Timer unit to update uptrack updates for running kernel

[Timer]
OnUnitActiveSec=1d
RandomizedDelaySec=10min
Persistent=true

[Install]
WantedBy=timers.target
```

This unit runs the service at boot and then runs daily, thereafter. The service must run daily, because uptrack-update packages are only released for the kernel when uptrack updates are available.

c. Reload the systemd daemon to recognize the new service.

```
sudo systemctl daemon-reload
```

d. Enable and run the new systemd timer.

```
sudo systemctl enable --now uptrack-update-packages.timer
```

Note that the service might return an error similar to the following, until an uptrack-update package becomes available:

```
systemd[1]: Failed to start Update uptrack updates for running kernel.
```

To display information about Ksplice updates, use the `rpm -qa | grep uptrack-updates` and `uptrack-show` commands.

Using the SNMP Plugin for Ksplice Uptrack

The SNMP plugin for Ksplice works with Oracle Enterprise Manager to monitor the status of Ksplice on systems. It also works with any monitoring solution that's compatible with SNMP.

Installing and Configuring the SNMP Plugin

Verify the system meets all prerequisites:

- The `net-snmp` package must be installed.
- The `net-snmp-utils` package must be installed to test the configuration using the `snmpwalk` command.
- The `snmpd` service must be configured to start automatically.
- SELinux must either be disabled or set to permissive mode on the system.

This procedure describes how to install the SNMP plugin on the system that you want to monitor.

> **Note**
>
> This procedure uses the `dnf` command to describe many package management actions. On releases earlier than Oracle Linux 8, substitute the commands with the appropriate `yum` commands.

1. Subscribe the system to the appropriate Ksplice channels on ULN.

   Subscribe the system to the appropriate Ksplice channel for the installed Oracle Linux distribution and system architecture, for example, `ol9_x86_64_ksplice` for Oracle Linux 9 on x86_64.

2. Install the `ksplice-snmp-plugin` package on the system.

   ```
   sudo dnf -y install ksplice-snmp-plugin
   ```

3. (Optional) If you plan to test the configuration by using the `snmpwalk` command, install the `net-snmp-utils` package.

   ```
   sudo dnf -y install net-snmp-utils
   ```

4. Configure the system to use the SNMP plugin by editing the `/etc/snmp/snmpd.conf` file.

   The following example shows how the entries in this file might look on an Oracle Linux 9 system:

   ```
   # Setting up permissions
   # ======================
   com2sec local localhost public
   com2sec mynet source public

   group local v1 local
   group local v2c local
   group local usm local
   group mynet v1  mynet
   group mynet v2c mynet
   group mynet usm mynet

   view all included .1 80

   access mynet "" any noauth exact all none none
   access local "" any noauth exact all all none

   syslocation Oracle Linux 9
   syscontact sysadmin <root@localhost>

   # Load the plugin
   # ===============
   dlmod kspliceUptrack /usr/lib/ksplice-snmp/kspliceUptrack.so
   ```

   a. In the `com2sec mynet` community entry, replace *source* with the IP address or resolvable host name of the server that hosts the SNMP monitoring software, or with a subnet address represented as *IP_address* / *netmask*, for example, `com2sec mynet 192.168.10.0/24 private`.

      For IPv6 configuration, specify an IPv6 address and netmask to a `com2sec6 mynet` community entry, for example, `com2sec6 mynet fec0::/64 private`.

   b. In the `syslocation` entry, replace the argument for the identifier of the system being monitored.

   c. In the `dlmod` entry that loads the `kspliceUptrack.so` plugin, replace the *lib* path element with `lib` on a 32-bit system and `lib64` on a 64-bit system.

   This sample configuration file is suitable for the purposes of testing.

5. Restart the SNMP service.

   ```
   sudo systemctl restart snmpd
   ```

For information about configuring SNMP, see the documentation at https://www.net-snmp.org/docs/readmefiles.html. See also the `snmpd(8)` and `snmpd.conf(5)` manual pages.

Testing the SNMP Plugin

You can use the `snmpwalk` command to check information and test the SNMP plugin.

1. Check the installed version of Ksplice.

   ```
   snmpwalk -v 1 -c public -O e localhost KSPLICE-UPTRACK-MIB::kspliceVersion
   ```

   Sample output:

   ```
   KSPLICE-UPTRACK-MIB::kspliceVersion.0 = STRING: 1.2.80
   ```

2. Check if available updates for a kernel have been installed.

   ```
   snmpwalk -v 1 -c public -O e localhost KSPLICE-UPTRACK-MIB::kspliceStatus
   ```

   Sample output (which shows the kernel is out of date):

   ```
   KSPLICE-UPTRACK-MIB::kspliceStatus.0 = STRING: outofdate
   ```

3. Compare the installed kernel with the Ksplice effective version.

   ```
   snmpwalk -v 1 -c public -O e localhost KSPLICE-UPTRACK-
   MIB::kspliceBaseKernel
   snmpwalk -v 1 -c public -O e localhost KSPLICE-UPTRACK-
   MIB::kspliceEffectiveKernel
   ```

   Sample output (which shows the base kernel and effective kernel are the same, implying no updates have been applied):

   ```
   KSPLICE-UPTRACK-MIB::kspliceBaseKernel.0 = STRING:
   5.15.0-304.171.4.1.el9uek
   KSPLICE-UPTRACK-MIB::kspliceEffectiveKernel.0 = STRING:
   5.15.0-304.171.4.1.el9uek
   ```

4. Show a list of all the updates that have been applied to the kernel.

   ```
   snmpwalk -v 1 -c public -O e localhost KSPLICE-UPTRACK-
   MIB::ksplicePatchTable
   ```

   In this example, we receive no output, meaning no updates have been applied. This confirms why the base and effective kernel versions are the identical and why the kernel is out of date.

5. Show a list of updates that can be installed.

   ```
   snmpwalk -v 1 -c public -O e localhost KSPLICE-UPTRACK-
   MIB::kspliceAvailTable
   ```

Sample output:

```
KSPLICE-UPTRACK-MIB::kspliceavailIndex.0 = INTEGER: 0
KSPLICE-UPTRACK-MIB::kspliceavailIndex.1 = INTEGER: 1
KSPLICE-UPTRACK-MIB::kspliceavailIndex.2 = INTEGER: 2
...
KSPLICE-UPTRACK-MIB::kspliceavailDesc.23 = STRING: CVE-2011-4325: Denial
of service in NFS direct-io.
KSPLICE-UPTRACK-MIB::kspliceavailDesc.24 = STRING: CVE-2011-4348: Socking
locking race in SCTP.
KSPLICE-UPTRACK-MIB::kspliceavailDesc.25 = STRING: CVE-2011-1020,
CVE-2011-3637: Information leak, DoS in /proc.
```

6. After fully upgrading the kernel by using Ksplice Uptrack, you can run the following `snmpwalk` commands to verify that the kernel is updated.

```
snmpwalk -v 1 -c public -O e localhost KSPLICE-UPTRACK-MIB::kspliceStatus
```

Sample output:

```
KSPLICE-UPTRACK-MIB::kspliceStatus.0 = STRING: uptodate
```

7. Check that no updates are available for installation, and also that the patches that have been applied.

```
snmpwalk -v 1 -c public -O e localhost KSPLICE-UPTRACK-
MIB::kspliceAvailTable
snmpwalk -v 1 -c public -O e localhost KSPLICE-UPTRACK-
MIB::ksplicePatchTable
```

Output similar to the following is displayed:

```
KSPLICE-UPTRACK-MIB::ksplicepatchIndex.0 = INTEGER: 0
KSPLICE-UPTRACK-MIB::ksplicepatchIndex.1 = INTEGER: 1
KSPLICE-UPTRACK-MIB::ksplicepatchIndex.2 = INTEGER: 2
...
```

Removing the Ksplice Uptrack Client Software

This procedure describes how to remove the Ksplice Uptrack Client software.

- Select the appropriate command to remove the software from the system depending on whether you're running the Ksplice Uptrack Client in online or offline mode.
  - To remove the online Ksplice Uptrack software from a system:

    ```
    sudo dnf -y remove uptrack
    ```

  - To remove the offline Ksplice Uptrack software from a system:

    ```
    sudo dnf -y remove uptrack-offline
    ```

5

Using the Ksplice Uptrack API

The Ksplice Uptrack API is a RESTful web API that you can query for the status of systems that are running Oracle Ksplice Uptrack. The API provides information about the updates that any systems have, and status of any out-of-date, inactive, or unsupported systems.

You can use the command line tools that are included with the Python bindings; or, you can write custom scripts by using the bindings. You can also write a custom interface by using HTTP requests. The Python bindings include the `check_uptrack` and `check_uptrack_local` plugins for Nagios. These plugins can be used to monitor the status of systems.

> **Note**
>
> The Ksplice Uptrack API doesn't support user space updates. However, the online version of the Ksplice Enhanced client can patch shared libraries for user space processes that are running on an Oracle Linux.

> **Note**
>
> You can't use the Ksplice Uptrack API to monitor systems that are running Ksplice Offline client because these systems aren't registered with https://status-ksplice.oracle.com.

For more information about the Ksplice Uptrack API, visit http://www.ksplice.com/.

Installing the API Command Line Tools

The API command line tools are included with the Python bindings for the API in the `python-ksplice-uptrack` package. This package is available in the Ksplice for Oracle repositories on ULN at `linux.oracle.com` or the Ksplice Uptrack for Oracle Linux repositories at `www.ksplice.com`.

1. Check that you have a valid support subscription.

   Ensure that you have a valid Oracle Linux Premier subscription, a Premier Limited subscription, or an Oracle Premier Support for Systems and Operating Systems subscription.

   These subscriptions automatically register the system to use Ksplice. See Registering With ULN for more details.

2. Install the `python-ksplice-uptrack` package.

   ```
   sudo dnf install -y python-ksplice-uptrack
   ```

   The package includes Python bindings, API tools, and Nagios plugins.

Working With API Credentials

1. Sign in to https://status-ksplice.oracle.com.
2. Select the **Settings** tab.
3. View the information under the **API information** heading.
   - **API Username**: displays the API username that you can use to access the API
   - **API Key**: displays the API key that you can use to access the API
   - **API URL**: displays the API base URL that you can use with the different API endpoints

   You can select the **Generate a new API key?** checkbox and click **Save Changes** to change the API key.

> **Note**
>
> Generating a new API key invalidates the existing API key.

Using the Ksplice Uptrack API Commands

The Python bindings include the following commands, which cover common uses of the Ksplice Uptrack API.

- uptrack-api-authorize
- uptrack-api-describe
- uptrack-api-list

The commands use the API user name and key for authentication.

- Setting the API Username and Key Variables
- Specifying a Proxy

Setting the API Username and Key Variables

If you set the API username and key as variables that can be loaded each time you run any of the API commands, you don't need to supply these variables as command line arguments to the scripts.

See also Working With API Credentials.

The following options to set the variables are available:

- Set the `username` and `api_key` variables in the `/etc/uptrack-api.conf` file.

Place the variables under an `[uptrack]` section heading, for example:

```
[uptrack]
username = jo.admin@example.com
api_key  = 3af3c2c1ec407feb0fdc9fc1d8c4460c
```

- Set the `UPTRACK_API_USERNAME` and `UPTRACK_API_KEY` environment variables.

  For example:

  ```
  export UPTRACK_API_USERNAME=jo.admin@example.com
  export UPTRACK_API_KEY=3af3c2c1ec407feb0fdc9fc1d8c4460c
  ```

Specifying a Proxy

If you access the internet by using a proxy, specify the connection information in the `[uptrack]` section of the `/etc/uptrack-api.conf` file, as shown in the following example:

```
https_proxy = [protocol://][username:password@]proxy[:port]
```

In the previous example, *protocol* is either specified as `http` or `https`, *username* and *password* authenticate you with the proxy (if required), and *proxy* and *port* are the host name/IP address and port number that you use to connect to the proxy server.

The following example shows how you might specify this connection information:

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

Note that the proxy *must* support HTTPS connections.

uptrack-api-authorize

The `uptrack-api-authorize` command uses the `authorize` API call to change the authorization for a single system, as shown in the following examples.

- To deny a system:

  ```
  uptrack-api-authorize -u api_username -k api_key uuid deny
  ```

  Output might appear as follows:

  ```
  Successfully denied access for uuid.
  ```

- To allow a system:

  ```
  uptrack-api-authorize -u api_username -k api_key uuid allow
  ```

  Output might appear as follows:

  ```
  Successfully allowed access for uuid .
  ```

> **Note**
>
> To view the API username and API key, sign in to https://status-ksplice.oracle.com and then select the **Settings** tab. You can also optionally set these so that you don't need to specify them each time you run a command. See Setting the API Username and Key Variables.
>
> The UUID of a registered system is stored in `/var/lib/uptrack/uuid` on the system. An example of a UUID is `e82ba0ae-ad0a-4b92-a776-62b502bfd29d`.

uptrack-api-describe

The `uptrack-api-describe` command uses the `describe` API call to get detailed information about a single system, which is specified by its UUID, for example:

```
uptrack-api-describe -u api_username -k api_key uuid
```

Output might appear as follows:

```
prod1.example.com (192.168.1.100)
Effective kernel: 6.12.0-0.20.20.el9uek
This machine is no longer active
Last seen on 2025-05-12T11:19:21Z
OS status: Up to date
```

Or, you can specify the `--this-machine` option if you're running the script on the system you want to check:

```
uptrack-api-describe -u api_username -k api_key --this-machine
```

Output might appear as follows:

```
qa.example.com (192.168.1.200)
Effective kernel: 5.15.0-304.171.4.el8uek
This machine is active
Last seen on 2025-05-12T11:22:07Z
OS status: Out of date:
   * Install dv4m9d3r Improved update for Denial-of-service when checking if
an address is a jump label.
   * Install d6eukcmk Enablement update for live patching.
   * Install bk3v6mgo Denial-of-service when checking if an address is a jump
label.
   * Install m96ph72r Enable livepatching of jump labels.
   * Install eu43qhnj Known exploit detection.
   * Install 74a4h3ep Known exploit detection for CVE-2019-9213.
...
```

> **Note**
>
> To view the API username and API key, sign in to https://status-ksplice.oracle.com and then select the **Settings** tab. You can also optionally set these so that you don't need to specify them each time you run a command. See Setting the API Username and Key Variables.

uptrack-api-list

The `uptrack-api-list` command uses the `machines` API call to return a list of all systems and their status, for example:

```
uptrack-api-list -u api_username -k api_key
```

Output might appear as follows:

```
- dev1.example.com (192.168.1.102): outofdate
- qa1.example.com (192.168.1.103): outofdate (inactive)
- prod1.example.com (192.168.1.100): uptodate
- prod2.example.com (192.168.1.101): uptodate
```

> **Note**
>
> To view the API username and API key, sign in to https://status-ksplice.oracle.com and then select the **Settings** tab. You can also optionally set these so that you don't need to specify them each time you run a command. See Setting the API Username and Key Variables.

About the API Implementation

Learn about the API version, authentication, request format, supported requests, and a sample interaction.

Authentication

Authentication to the Uptrack API server uses a username and an API key that are specified in custom HTTP headers.

All requests must include the following HTTP headers:

- `X-Uptrack-User`: the API username of the user who is making the request.
- `X-Uptrack-Key`: the API key of the user who is making the request.

API Request Format

API requests or responses include JSON-encoded data in the request body.

Set the following HTTP headers in all requests:

- `Content-Type`: `application/json`

- `Accept:: application/json`

These headers aren't required, as the API only uses JSON-encoded data, but future versions of the API might use other data-encoding formats.

API Endpoints

This document describes version 1 of the API. All requests go to paths that begin with `https://uptrack.api.ksplice.com/api/1/`. The following endpoints are available:

- GET /api/1/machines
- GET /api/1/machine/$UUID/describe
- POST /api/1/machine/$UUID/authorize
- POST /api/1/machine/$UUID/group

Interaction Sample

The following example, is provided as a reference *only* and shows how to list the systems managed for a Ksplice account by using the API.

```
curl -X GET "https://uptrack.api.ksplice.com/api/1/machines" \
  -H "Accept: application/json" \
  -H "X-Uptrack-User: jo.admin@example.com" \
  -H "X-Uptrack-Key: 3af3c2c1ec407feb0fdc9fc1d8c4460c"
```

The server authenticates the request and responds with a list of the systems, for example:

```
[
  {
    "status": "uptodate",
    "autoinstall": true,
    "uuid": "00086980-47cb-4486-81c0-4fe3299bad90",
    "mmap_min_addr": 65536,
    "ip": "192.168.248.238",
    "hostname": "host1.example.com",
    "uptrack_client_version": "1.2.80",
    "active": true,
    "authorization": "allowed",
    "last_seen": "2025-05-12T11:19:21Z"
  }
]
```

GET /api/1/machines

The `GET /api/1/machines` API request returns a list of all the registered systems. This list includes inactive systems that have uninstalled Uptrack or any systems that haven't reported to the Uptrack server recently. The list doesn't include systems that you have hidden by using the web interface. The response shows a list of systems, which are represented as dictionaries, as shown in the following example:

```
[
  {
    "status": "uptodate",
    "autoinstall": true,
```

```
    "uuid": "00086980-47cb-4486-81c0-4fe3299bad90",
    "mmap_min_addr": 65536,
    "ip": "192.168.248.238",
    "hostname": "host1.example.com",
    "uptrack_client_version": "1.2.80",
    "active": true,
    "authorization": "allowed",
    "last_seen": "2025-05-12T11:19:21Z"
  }
]
```

The following fields are provided in the response:

`status`
Contains one of the following values:

- `outofdate` - Updates are available for installation on the system.
- `unsupported` - The system's kernel isn't supported by Ksplice Uptrack.
- `uptodate` - All available updates have been installed on the system.

`authorization`
Contains one of the following values:

- `allowed` - The system is allowed to communicate with the Uptrack servers and to receive updates.
- `denied` - The system has been denied access to the Uptrack servers by using the web interface, `uptrack-api-authorize`, or the `authorize` API call.
- `pending` - This account has the default deny policy set for new systems, and the system hasn't yet been authorized.

`autoinstall`
Indicates whether `autoinstall` is set on the system.

`mmap_min_addr`
Is the value of `/proc/sys/vm/mmap_min_addr` or `None` for clients before version 1.0.3.

`uptrack_client_version`
Is the version of the Uptrack client that the system is running.

GET /api/1/machine/$UUID/describe

The `GET /api/1/machine/$UUID/describe` API request returns information about the system with the specified UUID. The UUID of a system is stored in `/var/lib/uptrack/uuid` and can be retrieved by using the `machines` query. The response is a dictionary of the same form that `GET /api/1/machines` returns, except that it includes the following fields:

`effective_kernel`
Ksplice has applied all the important security and reliability updates that are needed to bring the system into line with this kernel version.

`group`
The group to which the system is assigned. You can also use the web interface to manage system groups.

installed_updates
A list of 2-element dictionaries of the form {'ID': *update_id*, 'Name': *update_name*} that represent the updates installed on the system. *update_id* is the ID code of an update (for example, diptbg4f) and *update_name* is a short descriptive name for the update (for example, CVE-2010-0415: Information Leak in sys_move_pages).

original_kernel
The kernel version of the system before any Ksplice updates were applied.

steps
A list of two-element lists of the form [*action*, {'ID': *update_id*, 'Name': *update_name*}], representing the updates that need to be installed or removed to update the system. For the *action* argument, you can specify Install or Remove. Note that an existing update is removed if it superseded by a more recent version.

POST /api/1/machine/$UUID/authorize

The POST /api/1/machine/$UUID/authorize API request authorizes the system with the specified UUID to access the Uptrack service if you have configured the account to deny access to new systems.

The content is a dictionary of the following form:

```
{authorized: boolean}
```

Specify the *boolean* argument as true to authorize the system or false to revoke authorization.

POST /api/1/machine/$UUID/group

The POST /api/1/machine/$UUID/group API request changes the group of the system with the specified UUID.

The content is a dictionary that uses the following form:

```
{group_name: string}
```

In the previous example, *string* is the name of the new group. The group is created if it doesn't already exist. Note that if the account doesn't have a system with the specified UUID, the request results in an HTTP 404 error.

To remove a system from a group, you can set the group to a different name, or you can specify an empty string for no group.

Configuring the check_uptrack Nagios Plugin

> **Note**
>
> The Nagios software doesn't include the python-ksplice-uptrack package. For information about obtaining and using Nagios, visit the official Nagios website at http://www.nagios.org.

Configure the `check_uptrack` Nagios plugin as follows:

1. Set the `username` and `api_key` variables in the configuration file `/etc/uptrack-api.conf` under an `[uptrack]` section heading, for example:

   ```
   [uptrack]
   username = jo.admin@example.com
   api_key  = 3af3c2c1ec407feb0fdc9fc1d8c4460c
   ```

2. If you access the Internet by using a proxy, specify the connection information in the `[uptrack]` section of `/etc/uptrack-api.conf`:

   ```
   https_proxy = [protocol://][username:password@]proxy[:port]
   ```

   In the previous example, *protocol* is `http` or `https`, *username* and *password* authenticate you with the proxy (if required), and *proxy* and *port* are host name/IP address and port that you use to connect o the proxy server. The connection information you specify might be similar to the following:

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

   The proxy *must* support HTTPS connections.

3. Configure the `check_uptrack` plugin in the Nagios configuration file, at `/usr/local/nagios/etc/nagios.cfg`.

   The following minimal configuration can be used to run the plugin:

   ```
   # Dummy host with which to associate the Uptrack service
   define host {
          host_name                       uptrack-service
          notifications_enabled           0
          max_check_attempts              1
          notification_interval           0
          check_period                    never
          contacts                        server-admins
   }

   define service {
          host_name                       uptrack-service
          service_description             Ksplice Uptrack Update Status
          check_command                   check_uptrack
          notifications_enabled           1
          normal_check_interval           60
          retry_check_interval            15
          max_check_attempts              4
          notification_options            w,c,r
          contacts                        server-admins
   }

   define command {
          command_name     check_uptrack
          command_line     /usr/lib/nagios/plugins/check_uptrack
   }
   ```

```
define command {
       command_name     check_uptrack_opts
       command_line     /usr/lib/nagios/plugins/check_uptrack -w $ARG1$ -
c $ARG2$
}
```

Using the Nagios Plugin

To monitor all systems by using the Nagios plugin, run the following command:

```
sudo /usr/lib/nagios/plugins/check_uptrack
```

The previous command produces a summary of systems in the standard Nagios plugin format, as shown in the following example:

```
2 machines are OUTOFDATE!|uptodate=1280;outofdate=1;unsupported=0;inactive=3
  prod1.example.com (192.168.1.1) is OUTOFDATE
  prod2.example.com (192.168.1.2) is OUTOFDATE
```

If you specify the `-c` or `-w` options with a comma-separated list of the arguments that also specify the `i`, `o`, or `u` options for inactive, out-of-date, or unsupported systems, the `check_uptrack` command displays critical or warning notices for systems that match the criteria.

For example, the following command returns warning notices for any systems that are inactive or unsupported, and critical notices for any systems that are out of date:

```
sudo /usr/lib/nagios/plugins/check_uptrack -w u,i -c o
```

To monitor the local system, use the `check_uptrack_local` plugin:

```
sudo /usr/lib/nagios/plugins/check_uptrack_local
```

The output from the `check_uptrack_local` command is similar to the output from the `check_uptrack` command. However, for out-of-date systems, the command also lists the updates that are required to update the system.

> **Note**
>
> The `check_uptrack_local` command reads the local uptrack update cache; however, it doesn't use the settings from the `/etc/uptrack-api.conf` file.