NI Platform on Linux Desktop User Manual

2025-12-16

Contents

NI Platform on Linux Desktop User Manual

The NI Platform on Linux Desktop User Manual provides detailed descriptions of the product functionality and the step by step processes for use.

Looking for Something Else?

For information not found in the User Manual for your product, such as specifications and API reference, browse ***Related Information***.

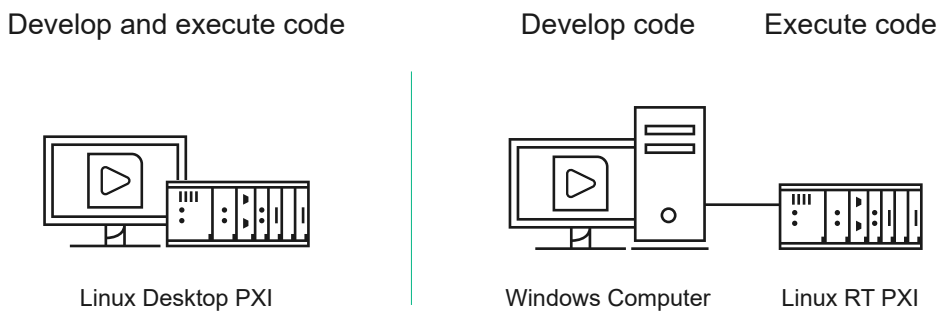
Related information:

- Introduction to NI Linux Real-Time
- NI Supported Operating Systems Roadmap
- NI Linux Desktop User Community
- NI Linux Device Drivers Download

About the NI Platform on Linux Desktop

The NI Platform on Linux Desktop is a set of hardware, software, and drivers designed for users creating test and measurement applications with Linux.

You can integrate NI instruments into systems running on Linux and program NI instruments using LabVIEW, Python, C, and other programming languages. NI Platform on Linux Desktop allows you to develop and execute code on the same machine running a desktop OS. NI also offers NI Linux Real-Time, a Linux distribution with industrial-grade, real-time capabilities. You can use NI Linux RT on embedded systems that are programmed and maintained remotely from another system running a desktop OS.



Note This manual applies to Linux desktop only.

Related information:

- Introduction to NI Linux Real-Time

NI Platform on Linux Desktop Key Features

- Support for most NI PXI and desktop hardware
- Support for third-party instruments through NI-VISA and the Hardware Configuration Utility, and the Instrument Driver Network (IDNet)
- Use of the stable, secure Linux operating system
- Option to program using LabVIEW, Python, C, and other programming languages

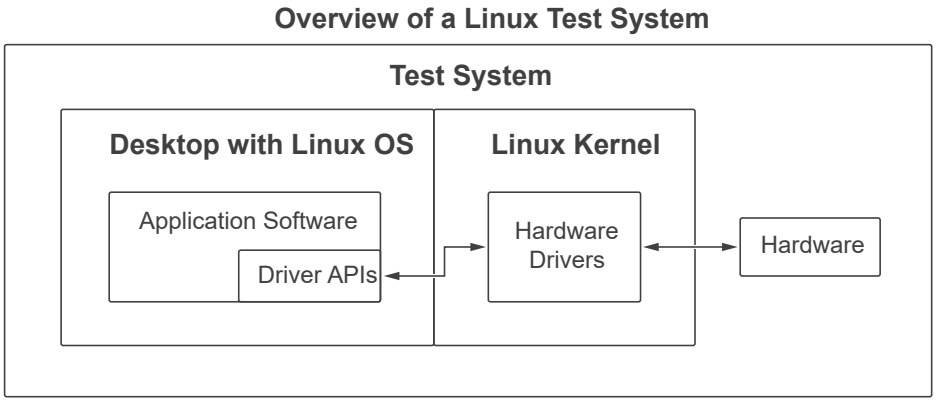
- Support for various Linux distributions:
  - Ubuntu
  - Red Hat Enterprise Linux
  - OpenSUSE
- DKMS technology in NI drivers allows users to build kernel driver support into non-supported Linux distributions

Who needs the NI Platform on Linux Desktop?

Support for Linux Desktop allows users who prefer the Linux operating system to use it for their test and measurement applications involving NI hardware.

Components of a System with Linux Desktop Support

You can use the NI Platform on Linux Desktop to create a test system that requires hardware, drivers, and software. Use the following list of typical components as a starting point for building your test system.



- **Application Software**—Allows users to develop code using the driver APIs
- **Desktop with Linux OS**—Provides the operating system functionality on which other software is built
- **Driver APIs**—Allows users to develop test and measurement code that performs I/O on NI hardware
- **Linux Kernel**—The basis of any Linux distribution. The kernel provides support for some PC hardware and it can load modules that provide support for additional hardware.
- **NI Driver**—Provides support for hardware. Hardware support includes the following:
  - Configuration utilities and API libraries that build on the Linux desktop environment and libraries
  - Kernel modules that plug into the Linux kernel
- **NI Hardware**—Equipment, such as PXI modules, that the Linux desktop interacts with through drivers

NI Platform on Linux Desktop System Requirements

Your system must meet the minimum requirements to run and use the NI Platform on Linux Desktop.

Refer to ***NI Hardware and Software Operating System Compatibility*** to verify Linux support for your instruments.

Related information:

- NI Hardware and Software Operating System Compatibility

NI Platform on Linux Desktop New Features and Changes

Learn about updates, including new features and behavior changes, introduced in each version of NI Platform on Linux Desktop.

Discover what is new in the latest releases of NI Platform on Linux Desktop.

> **Note** If you cannot find new features and changes for your version, it might not include user-facing updates. However, your version might include non-visible changes such as bug fixes and compatibility updates. For information about non-visible changes, refer to your product ***Release Notes***.

NI Platform on Linux Desktop 2024 Q1 New Features and Changes

- LabVIEW now supports openSUSE 15.5

NI Platform on Linux Desktop 2023 Q4 New Features and Changes

- Repositories for openSUSE 15.5
- Repositories will no longer be updated for openSUSE 15.3

NI Platform on Linux Desktop 2023 Q3 New Features and Changes

- Hardware Configuration Utility support for RF composite devices

NI Platform on Linux Desktop 2023 Q2 New Features and Changes

- Hardware Configuration Utility now allows configuring switch blocks, creating simulated devices on all Linux distributions
- Ubuntu 22.04, Ubuntu 20.04, RHEL 9, RHEL 8, openSUSE 15.4, and openSUSE 15.3 now supports PXIe-7903, PCIe-8383, NI I/O Trace, and NI System Configuration

Updates and Changes for NI Platform on Linux Desktop Extended Support Versions

Browse updates and changes made in NI Platform on Linux Desktop versions on extended support.

Note If you cannot find changes for your version, it might be a more recent version, documented as a new feature. Or, your version might not have included user-facing updates. You can find more information about non-visible changes, such as bug fixes, compatibility updates, and stability adjustments or maintenance adjustments, in the product ***Release Notes***, available on ni.com.

NI Platform on Linux Desktop 2023 Q1 New Features and Changes

- Red Hat Enterprise Linux 9 support for NI-DMM, NI-FGEN, FlexRIO, NI-SCOPE, NI-SWITCH, and NI-Sync
- openSUSE 15.3 and openSUSE 15.4 support for NI-DMM, NI-FGEN, FlexRIO, NI-SCOPE, and NI-SWITCH
- Ubuntu 22.04 support for NI-DCPower, NI-DMM, NI-FGEN, FlexRIO, NI-SCOPE, and NI-SWITCH
- LabVIEW Host Interface API for FlexRIO, R Series, cRIO, and RIO

NI Platform on Linux Desktop 2022 Q4 New Features and Changes

- Repositories for Red Hat Enterprise Linux 9, Ubuntu LTS 22.04, openSUSE 15.3, and openSUSE 15.4
- Repositories will no longer be updated for Red Hat Enterprise Linux 7, Ubuntu LTS 18.04, openSUSE 15.1, and openSUSE 15.2
- Hardware Configuration Utility support for NI-RFSA and NI-RFSG non-composite devices

- NI-SWITCH Ubuntu LabVIEW API package added to Ubuntu 20.04 repository

NI Platform on Linux Desktop 2022 Q3 New Features and Changes

- NI-RFSA and NI-RFSG adds support for Ubuntu 18.04 and Ubuntu 20.04
- NI-DMM and NI-SCOPE adds LabVIEW support for Ubuntu
- NI-FGEN adds support for Hardware Configuration Utility
- Ubuntu repositories add support for the same products as the RedHat Enterprise Linux and OpenSUSE repositories
- LabVIEW adds support for the following distributions.
  - openSUSE Leap 15.3
  - openSUSE Leap 15.4
  - Red Hat Enterprise Linux 9
  - Ubuntu 22.04 LTS

NI Platform on Linux Desktop 2022 Q2 New Features and Changes

- NI-FGEN support
- NI Automotive Ethernet support
- PXI-6683H support for NI-Sync
  - GPS, IRIG-B, PPS, IEEE-1588, IEEE 802.1as synchronization
- PXIe-419x support for NI-DCPower
- TS-15xxxx support for NI-DAQmx
- LabVIEW Ubuntu API support for NI-Sync and NI-DAQmx

Related information:

- NI Hardware and Software Operating System Compatibility

Migrating an NI Test System to Linux Desktop

The functionality of an NI test system determines the details of migration to a Linux desktop operating system.

Use the following procedure to ensure that you are aware of the general considerations for migrating a system.

1. Select a Linux operating system distribution and kernel that NI supports.
2. Determine which NI hardware, software, and drivers in your current test system are supported on your chosen Linux operating system. Replace any unsupported system elements with equivalent components or alternative implementations.
3. Install your chosen Linux OS on the desktop machine you are migrating your system to.
4. Install NI application-level software and drivers on your Linux desktop machine.
5. Install test system software, such as LabVIEW VIs and projects, on your Linux desktop machine. You do not need to convert these files to open and run them on a Linux desktop machine.

   Note Toolkits with operating system-dependent functionality, such as toolkits that utilize ActiveX, do not function when migrated to a Linux operating system.

6. Connect your hardware to your Linux desktop machine.
7. Run the Hardware Configuration Utility to ensure that your target machine detects all of your devices.
8. Run your test application on your Linux desktop machine to see what functionality works correctly and where errors occur.

   Note LabVIEW cannot build installers on a Linux operating system. For test systems that rely on LabVIEW to build installers, you must replace this functionality with Linux-based tools.

If your test system does not function as expected after migration, contact NI support or reach out to the NI Linux User Community for assistance.

Related concepts:

- NI Platform on Linux Desktop System Requirements

Related tasks:

- Supported Linux Distributions
- Installing NI Drivers and Software on Linux Desktop

Related information:

- NI Linux User Community
- NI Hardware and Software Operating System Compatibility

Supported Linux Distributions

Choose which supported distribution to use with your NI software.

NI formally tests and supports three families of Linux distributions. Choose the family and distribution that best suits your goals.

Distribution	Forked from	Beginner Friendly	Free to Use
Ubuntu	Debian	Yes	Yes
Red Hat Enterprise Linux	Red Hat Linux, Fedora	Yes	No
OpenSUSE	SLS/Slackware	No	Yes

To ask other Linux users how they chose their OS, visit the NI Linux User Community.
Related information:

- NI Supported Operating Systems Roadmap
- NI Linux User Community
- Ubuntu
- Red Hat Enterprise Linux
- OpenSUSE

Supported Driver Packages for Linux Distributions

To install the specified packages for the following NI drivers, use the package manager for your Linux distribution.

Table 1. Supported NI Driver Packages for Linux Distributions

Driver	Required Package	Optional Packages
NI-488.2	ni-488.2	libni4882-devel ni-488.2-iotrace-support
Automotive Diagnostic Command Set Toolkit	ni-adcs	ni-adcs-labview-20xy-supportlib nidiagcs-devel
NI-DAQmx	ni-daqmx	ni-daqmx-labview-20xy-support ni-daqmx-labview-supportlib nidaqmx-devel
NI-DCPower	ni-dcpower	ni-dcpower-daqmx ni-dcpower-labview-20xy-support ni-dcpower-labview-support ni-dcpower-devel
NI-DMM	ni-dmm	ni-dmm-daqmx ni-dmm-labview-20xy-support ni-dmm-labview-support

Driver	Required Package	Optional Packages
		ni-dmm-devel
ECU Measurement and Calibration Toolkit	ni-ecumc	ni-ecumc-labview-20xy-support libniemcc-devel
NI-FGEN	ni-fgen	ni-fgen-daqmx ni-fgen-devel ni-fgen-labview-support ni-fgen-labview-20xy-support
FlexRIO	ni-flexrio	ni-flexrio-795x ni-flexrio-796x ni-flexrio-797x libniflexrioapi-devel ni-flexrio-modulario-libs-devel
Hardware Configuration Utility	ni-hwcfg-utility	—
PXI Platform Services	ni-pxiplatformservices	ni-pxiplatformservices-labview-20xy-support ni-pxiplatformservices-labview-support ni-pxiplatformservices-devel
NI-RIO	ni-fpga-interface	ni-rio-mxie
NI R Series	ni-rseries	—
NI-RFSA	ni-rfsa	ni-rfsa-daqmx ni-rfsa-labview-20xy-support ni-rfsa-devel
NI-RFSG	ni-rfsg	ni-rfsg-daqmx

Driver	Required Package	Optional Packages
		ni-rfsg-labview-20xy-support ni-rfsg-devel
NI-SCOPE	ni-scope	ni-scope-daqmx ni-scope-labview-20xy-support ni-scope-labview-support ni-scope-devel
NI-Serial	ni-serial	—
NI-SWITCH	ni-switch	ni-switch-daqmx ni-switch-daqmx-config ni-switch-labview-support ni-switch-labview-20xy-examples ni-switch-devel
NI-Sync	ni-sync	ni-sync-devel ni-sync-labview-20xy-support ni-sync-labview-support
System Configuration	ni-syscfg-runtime	ni-syscfg-labview-20xy-support ni-syscfg-labview-support ni-iotrace ni-syscfg-devel
NI-VISA	ni-visa	ni-visa-labview-20xy-support ni-visa-labview-support ni-visa-devel

Driver	Required Package	Optional Packages
NI-XNET	ni-xnet	ni-xnet-labview-20xy-support libnixnet-devel
NI RIO enables connections through the FPGA Interface API to remote RIO devices such as CompactRIO, Single-Board RIO, and Ethernet RIO.		

The following are categories of optional packages:

- **LabVIEW support packages**—These packages install the driver APIs to LabVIEW. LabVIEW support packages have driver and "labview" in the package name. There are two types of LabVIEW support packages:
  - Version-specific LabVIEW support packages—An older style of LabVIEW support packages that lists a year. Only use this support package when the LabVIEW support package matches the version of LabVIEW that you are using.
  - Version-agnostic LabVIEW support packages—A newer style of LabVIEW support packages. Use this support package when you cannot find a LabVIEW support package that matches the version of LabVIEW that you are using.
- **Development packages**—These packages install the header files that are required to compile code with the driver, but that are not required to execute code compiled somewhere else. Typically, development packages are required on the development machine but not always on a deployed system. Development package names end with "-devel".
- **Driver-DAQmx support packages**—The following packages have a few hardware devices that have separate drivers built on top of NI-DAQmx. These packages function as the top-level driver package for those particular hardware devices.
  - NI-DCPower
  - NI-DMM
  - NI-FGEN
  - NI-RFSA
  - NI-RFSG
  - NI-SCOPE
  - NI-SWITCH

When Will NI Support My Linux OS?

Visit the NI Supported Operating Systems Roadmap to learn about past support and tentative plans for when NI will begin and end software support on various operating systems.

Use the NI Linux User Community to connect to people who use unsupported Linux distributions and versions.

Related information:

- NI Supported Operating Systems Roadmap
- NI Linux User Community

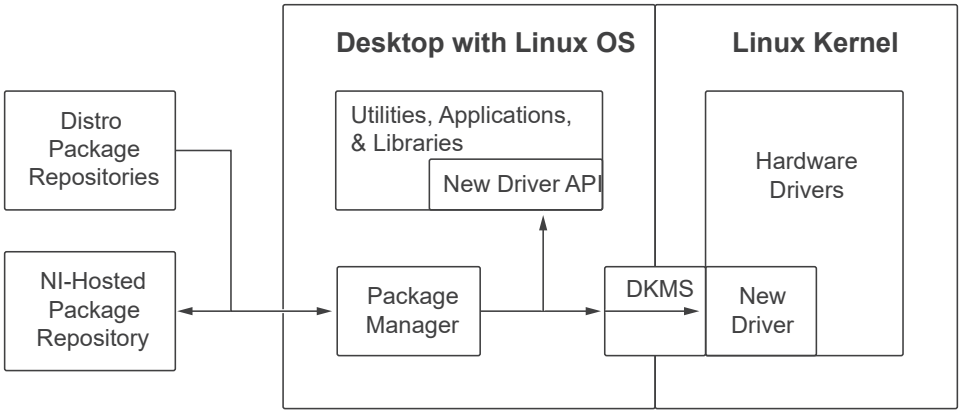
How Do NI Installers Handle Kernel Updates?

Learn about how the components of the installation process interact to maintain stability across different distributions and kernels.

The Linux kernel manages memory, hardware, processes, and system calls on a machine. Linux distributions add a select group of utilities that operate on top of the kernel, such as the following:

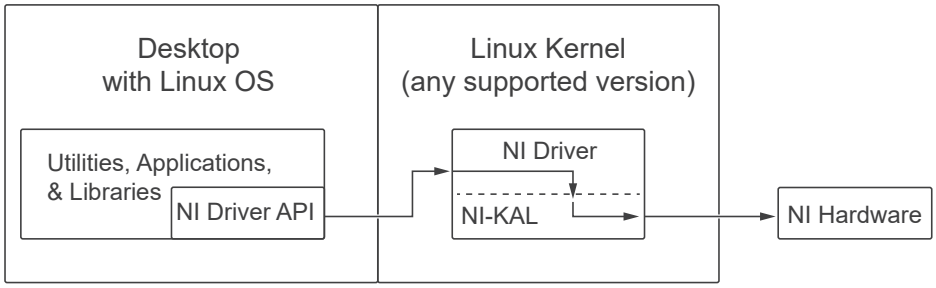
- Package managers
- File explorers
- Applications
- Libraries
- Command line interpreter
- Desktop environment

To install NI software, you use the package manager of the distribution to get NI software packages onto the system. Driver software installations also include configuration utilities, API libraries, and kernel modules. The native package manager installs these three types of packages, but adding kernel modules to the kernel requires an additional step. NI driver software uses Dynamic Kernel Module Support (DKMS) to add NI driver kernel modules to the kernel.



The mechanism that a kernel module uses to plug into the Linux kernel breaks easily between different versions of the kernel. To ensure that NI software can support the rapidly progressing kernel versions, NI created the Kernel Abstraction Layer (NI-KAL). NI-KAL does the following:

- Smooths out the differences between versions of the Linux kernel
- Allows the same NI software stack to work on any of the kernels that NI supports



Driver Errors with Ubuntu 22.04 and 24.04 New Installation or Kernel 6.8 (or Later) Upgrade

When setting up or replicating an existing Ubuntu 22.04 or 24.04 system, runtime failure messages may occur.

Symptoms

Because this scenario affects overall driver function, drivers may return various error messages, including messages about internal errors or hardware failure. For example:

```
The device could not be calibrated.
Status Code: -50152
A hardware failure has occurred. The operation could note be completed as
specified.
```

Affected Distributions and Kernels

This issue has been reproduced on the following systems:

- Ubuntu 22.04.04 HWE (kernel 6.8.0-40-generic)
- Ubuntu 24.04.01 (kernel 6.8.0-40-generic)

You can check the kernel version using the command: `uname -r`

Confirming the Problem

Use one of the following two methods to identify whether the PC has this issue.

- Look for `PTE Read access is not set` kernel messages in the system journal:
  1. Open a new Terminal window.
  2. Enter the following command:

     ```
     journalctl -k | grep -i NO_PASID
     ```

  3. Look for journal entries similar to this example:

     ```
     Sep 09 14:00:10 my_test_machine kernel: DMAR: [DMA Read NO_PASID] Request
     device
     [03:00.0] fault addr 0x1166b8000 [fault reason 0x06] PTE Read access is not
     set
     ```

  4. Correlate the timestamp to a previous failure.

  To check if similar errors occurred before the last restart, enter the following command instead:

  ```
  journalctl | grep -i NO_PASID
  ```

- Look for a kernel message specifying that the IOMMU default domain type is set to `Translated`:
  1. Open a new Terminal window.
  2. Enter the command:

     ```
     journalctl -k | grep -i IOMMU
     ```

  3. Look for journal entries similar to this example:

     ```
     [ 0.262686] iommu: Default domain type: Translated
     ```

Solution

The recommended solution is to configure the IOMMU default domain type to `Passthrough` mode using the following steps.

Notice The following will edit the GRUB configuration file. NI strongly recommends that you create a copy of the file before editing. If the file contains incorrect data, the PC may not boot.

The following procedure does not include steps for creating a copy of the GRUB configuration file. Create a backup copy of the existing configuration file before you modify the configuration. Read the complete process before you begin.

1. Edit the GRUB configuration file:
   a. Open a Terminal window.
   b. At the Command Prompt, enter the following command to open the file in the VI editor:

      ```
      sudo vi /etc/default/grub
      ```

   c. Enter the user password if prompted.
   d. Move the cursor to the line specifying the variable.

      ```
      GRUB_CMDLINE_LINUX_DEFAULT
      ```

   e. Press the <i> key to put the VI editor into insert mode.
   f. Add the command “ `iommu=pt`”.

      For example: `GRUB_CMDLINE_LINUX_DEFAULT="quiet splash iommu=pt"`

      Note Do not modify or delete the existing variable definition text. Add the command at the position shown in the preceding example.

   g. Press the <Esc> key to return the VI editor to command mode.
   h. Save the GRUB configuration file and exit the VI editor using the command `:wq`
2. In the Terminal window, enter the following command to regenerate the GRUB configuration file:

   ```
   sudo update-grub
   ```

3. Restart the system.

4. Verify that the GRUB configuration has been properly updated and that the IOMMU default domain type is configured to pass-through mode.
   a. Open a new Terminal window.
   b. Enter the command:

      ```
      journalctl -k | grep -i IOMMU
      ```

   c. Look for journal entries similar to this example:

      ```
      [ 0.000000] Command line: BOOT_IMAGE=/boot/vmlinuz-6.8.0-41-generic
      root=UUID=d27287cc-623b-4c29-93ec-23a5eb00177a ro quiet splash iommu=pt
      vt.handoff=7
      ...
      [ 0.261029] iommu: Default domain type: Passthrough (set via kernel command
      line)
      ```

Note If you are using Ubuntu 22.04 you can install Kernel 6.5 as an alternative to the recommended solution.

Related information:

- NI Supported Operating Systems Roadmap
- NI Hardware and Software Operating System Compatibility
- NI Linux Device Drivers 2025 Q2 Known Issues
- NI Linux Device Drivers 2025 Q1 Known Issues
- NI Linux Device Drivers 2024 Q3 Known Issues
- NI Linux Device Drivers 2024 Q1 Known Issues
- NI Linux Device Drivers 2023 Q4 Known Issues
- NI Linux Device Drivers 2023 Q3 Known Issues

Installing NI Drivers and Software on Linux Desktop

Download the repository registration package to allow the native package manager for your Linux distribution to install NI driver and software packages.

1. Download the Linux device driver repository registration packages from the Linux Device Driver Download page.
2. Use the following table to determine the repository registration package type that best meets your needs.

Option	Description
Static	Static repositories are updated only for critical patches. Use a static repository when you want to reproduce a specific software stack. You can identify static repository registration packages by the date-based suffix. For example, `ni-rhel9-drivers-2022Q4`.
Stream	Software in this repository is updated regularly with the latest versions of each driver package as it is released. Use a stream repository to keep all NI drivers and software up to date. Run an update command or an upgrade command on systems that have registered a stream repository to get the latest released version of NI packages. You can identify stream repository registration packages by the `-stream` suffix. For example, `ni-rhel9-drivers-stream`.

3. Complete the related task that matches your distribution. NI recommends using the online installation.

Related concepts:

- Supported Driver Packages for Linux Distributions

Related information:

- Linux Device Driver Download

Installing NI Products (Ubuntu)

Install NI drivers and software on Ubuntu.

> **Notice** Select NI drivers are incompatible with default IOMMU settings on Linux Kernel 6.8 and later. This incompatibility results in generic hardware or internal errors while using NI hardware. Refer to ***Driver Errors with Ubuntu 22.04 and 24.04 New Installation or Kernel 6.8 (or Later) Upgrade*** for more information.

Complete the steps in ***Installing NI Drivers and Software on Linux Desktop***.

1. Open the Command Prompt and run the following commands to apply the latest system updates to avoid installation issues.
   ```
   sudo apt update

   sudo apt dist-upgrade
   ```

   > **Note** This command may update your kernel to the latest version.

2. Restart your system.
3. Open the Command Prompt and run the following command to install the repository registration package.
   ```
   sudo apt install ./filename.deb
   ```

   Where

   - *filename.deb* is the repository registration package you chose in ***Installing NI Drivers and Software on Linux Desktop***.
4. Run the following command to refresh the package list.
   ```
   sudo apt update
   ```
5. Install the following basic Ubuntu packages on the target system.

- dkms
- expat
- libopenal1

> **Note** You must install these packages to run the NI packages.

6. Run the following command for each driver or application you want to install.
   `sudo apt install package-name`

   Where

   - *package-name* is the name of the package as it appears in ***Supported Driver Packages for Linux Distributions***.

   > **Note** Some drivers require multiple packages. Refer to ***Supported Driver Packages for Linux Distributions*** for driver-specific information.

7. NI recommends running the following command to install the Hardware Configuration Utility.
   `sudo apt install ni-hwcfg-utility`
8. Run the following command to build NI kernel drivers.
   `sudo dkms autoinstall`
9. Restart your system.

Related concepts:

- Supported Driver Packages for Linux Distributions

Related tasks:

- Installing NI Drivers and Software on Linux Desktop

Related reference:

- Driver Errors with Ubuntu 22.04 and 24.04 New Installation or Kernel 6.8 (or Later) Upgrade

Installing NI Products (Red Hat Enterprise Linux)

Install NI drivers and software on Red Hat Enterprise.

Complete the steps in ***Installing NI Drivers and Software on Linux Desktop***.

1. Open the Command Prompt and run the following command to apply the latest system updates to avoid installation issues.
   `sudo yum update`

   > **Note** This command may update your kernel to the latest version.

2. Restart your system.
3. Open the Command Prompt and run the following command to install chkconfig.
   `sudo yum install chkconfig`
4. Run the following command to install the repository registration package.
   `sudo yum install filename.rpm`

   Where

   - *filename.rpm* is the repository registration package you chose in ***Installing NI Drivers and Software on Linux Desktop***.
5. Run the following command for each driver or application you want to install.
   `sudo yum install package-name`

   Where

   - *package-name* is the name of the package as it appears in ***Supported Driver Packages for Linux Distributions***.

   > **Note** Some drivers require multiple packages. Refer to ***Supported Driver Packages for Linux Distributions*** for driver-specific information.

6. NI recommends running the following command to install the Hardware Configuration Utility.
   `sudo yum install ni-hwcfg-utility`

7. Run the following command to build NI kernel drivers.
   `sudo dkms autoinstall`
8. Restart your system.

Related concepts:

- Supported Driver Packages for Linux Distributions

Related tasks:

- Installing NI Drivers and Software on Linux Desktop

Installing NI Products (openSUSE)

Install NI drivers and software on openSUSE.

Complete the steps in ***Installing NI Drivers and Software on Linux Desktop***.

1. Open the Command Prompt and run the following command to apply the latest system updates to avoid installation issues.
   `sudo zypper update`

   Note This command may update your kernel to the latest version.
2. Restart your system.
3. Open the Command Prompt and run the following command to install insserv.
   `sudo zypper install insserv`
4. Run the following command to install the repository registration package.
   `sudo zypper --no-gpg-checks install ./filename.rpm`

   Where

   - *filename.rpm* is the repository registration package you chose in ***Installing NI Drivers and Software on Linux Desktop***.
5. Run the following command to refresh the package list.
   `sudo zypper refresh`
6. Run the following command for each driver or application you want to install.

```
sudo zypper install package-name
```

Where

- *package-name* is the name of the package as it appears in ***Supported Driver Packages for Linux Distributions***.

> **Note** Some drivers require multiple packages. Refer to ***Supported Driver Packages for Linux Distributions*** for driver-specific information.

7. NI recommends running the following command to install the Hardware Configuration Utility.
   ```
   sudo zypper install ni-hwcfg-utility
   ```
8. Run the following command to build NI kernel drivers.
   ```
   sudo dkms autoinstall
   ```
9. Restart your system.

Related concepts:

- Supported Driver Packages for Linux Distributions

Related tasks:

- Installing NI Drivers and Software on Linux Desktop

Installing NI Products Offline (Ubuntu)

Install NI drivers and software on Ubuntu offline.

> **Notice** Select NI drivers are incompatible with default IOMMU settings on Linux Kernel 6.8 and later. This incompatibility results in generic hardware or internal errors while using NI hardware. Refer to ***Driver Errors with Ubuntu 22.04 and 24.04 New Installation or Kernel 6.8 (or Later) Upgrade*** for more information.

Complete the steps in ***Installing NI Drivers and Software on Linux Desktop*** and ensure the following:

- You have root permissions for the system.
- The system has internet access.
- The system is running the same distribution as the offline system on which you want to install.

1. On your online system, open the Command Prompt and run the following command to install prerequisite software.
   `sudo apt-get install python3 apt-mirror dpkg-dev`
2. Run the following commands to apply the latest system updates to avoid installation issues.
   `sudo apt update`

   `sudo apt dist-upgrade`

   Note This command may update your kernel to the latest version.
3. Restart your system.
4. Open the Command Prompt and run the following command to install the repository registration package.
   `sudo apt install ./filename.deb`

   Where

   - *filename.deb* is the repository registration package you chose in ***Installing NI Drivers and Software on Linux Desktop***.
5. Download the `offline_deb_repo.py` script.
6. Run the following command to make the script `offline_deb_repo.py` executable by changing the permissions.
   `chmod +x offline_deb_repo.py`
7. Complete the steps based on your goals.

Option	Description
Clone the entire repo	a. Check the files in the `/etc/apt/sources.list.d/` directory for a file that matches the source repository to clone.

Option	Description
	For example, if the file name is `ni-software-2022-focal.list`, the script expects the name `ni-software-2022-focal`. b. If the repository does not have a name, use the following format to specify the apt source entry: `deb https://site.example.com/debian distribution component1 component2 component3`. For example, `deb https://download.ni.com/ni-linux-desktop/2022/Q1/deb/ni/focal focal ni`. c. To clone the repository, run the following command from the directory that contains the script. `./offline_deb_repo.py repo --collapse repo-name` Where ▪ *repo-name* is the name or apt source entry for the repository.

Create a repo for specific products	a. Open the Command Prompt and run the following command to create a GPG key on the root keyring.
	
	`sudo gpg --generate-key`
	
	b. Enter additional information as prompted.
	
	The result looks similar to the following example.
	
	`pub rsa3072 2022-03-02 [SC] [expires: 2024-03-01]BB4931F582F3BF628815C834E5E662294857752D`
	`uid First Last <email@example.com>`
	`sub rsa3072 2022-03-08 [E] [expires: 2024-03-01]`
	
	c. Copy the long hexadecimal number on the second line. This is the key ID.
	d. Run the following command.
	
	`./offline_deb_repo.py packages --key-id key id package-names`

Option	Description
	Where ▪ *key id* is the key ID you copied in the previous step. ▪ *package-names* is a space-separated list of the package names you want to include.

By default, the script clones the new repository in the `offline_repo` subdirectory within the directory where the script runs. Pass the `--output` argument to specify a different location.

8. Copy the repository to the target system.

> **Note** Choose a location that will not be modified accidentally.

9. Register the repository on the system.

> **Note** You must disable GPG checking of repository packages so that NI software installs properly because NI does not currently sign their packages. The following instructions add the repository and disable GPG checking.

a. Create a file at the following location as root.
`/etc/apt/sources.list.d/`*`repo-name.list`*

Where

- *repo-name* is the name you give your repository.

b. As root, enter the following content in the file.

```
deb [signed-by=/srv/offline_repo/dists/series/key.asc]
file:///srv/offline_repo series ni
```

Where

- *series* is the Ubuntu version `jammy`, `noble`, or `focal`.

10. Install the following basic Ubuntu packages on the target system.
    - dkms
    - expat

- libopenal1

Note You must install these packages to run the NI packages.

Complete the installation by following the online install instructions in ***Installing NI Products (Ubuntu)***.

Related tasks:

- Installing NI Drivers and Software on Linux Desktop
- Installing NI Products (Ubuntu)

Related reference:

- Driver Errors with Ubuntu 22.04 and 24.04 New Installation or Kernel 6.8 (or Later) Upgrade

Related information:

- offline_deb_repo.py

Installing NI Products Offline (Red Hat Enterprise Linux)

Install NI drivers and software on Red Hat Enterprise offline.

Complete the steps in ***Installing NI Drivers and Software on Linux Desktop*** and ensure the following:

- You have root permissions for the system.
- The system has internet access.
- The system is running the same distribution as the offline system on which you want to install.

1. On your online system, open the Command Prompt and run one of the following commands to install prerequisite software based on your distribution.

Option	Description
RHEL/CentOS 8	`sudo dnf install python36`

Option	Description
	`createrepo`
RHEL/CentOS 7	`sudo yum install python36 createrepo yum-utils`

2. Run the following command to apply the latest system updates to avoid installation issues.
   `sudo yum update`

   Note This command may update your kernel to the latest version.
3. Restart your system.
4. Open the Command Prompt and run the following command to install the repository registration package.
   `sudo yum install filename.rpm`

   Where

   - *filename.rpm* is the repository registration package you chose in ***Installing NI Drivers and Software on Linux Desktop***.
5. Download the `offline_rpm_repo.py` script.
6. Complete the steps based on your goals.

Option	Description
Clone the entire repo	a. Run the following command to list all known and enabled repositories on the system. **Distribution** / **Command** RHEL/CentOS 8 / `sudo dnf repolist`

<table>
<tr><th>Option</th><th>Description</th></tr>
<tr><td></td><td>
<table>
<tr><th>Distribution</th><th>Command</th></tr>
<tr><td>RHEL/CentOS 7</td><td><code>sudo yum repolist</code></td></tr>
</table>
b. To clone the repository, run the following command from the directory that contains the script.

<code>./offline_rpm_repo.py repo --norepopath repo-name</code>

Where

▪ repo-name is the name of the repository.
</td></tr>
<tr><td>Create a repo for specific products</td><td>Open the Command Prompt and run the following command.

<code>./offline_rpm_repo.py packages package-names</code>

Where

◦ package-names is a space-separated list of the package names you want to include.</td></tr>
</table>

By default, the script clones the new repository in the `offline_repo` subdirectory within the directory where the script runs. Pass the `--output` argument to specify a different location.

7. Copy the repository to the target system.

Note Choose a location that will not be modified accidentally.

8. To disable GPG checking, create a file at the following location as root. `/etc/yum.repos.d/`*`repo-name`*`.repo`

Where

- *repo-name* is the name you give your repository.

9. As root, enter the following content in the file.

```
[repo-name]

name=readable-name

enabled=1

gpgcheck=0

baseurl=file:///srv/offline_repo
```

Where

- *repo-name* is the name you give your repository.
- *readable-name* is the human-readable name you give your repository.

Complete the installation by following the online install instructions in ***Installing NI Products (Red Hat Enterprise Linux)***.

Related tasks:

- Installing NI Drivers and Software on Linux Desktop

Related information:

- offline_rpm_repo.py

Installing NI Products Offline (openSUSE)

Install NI drivers and software on openSUSE offline.

Complete the steps in ***Installing NI Drivers and Software on Linux Desktop*** and ensure the following:

- You have root permissions for the system.

- The system has internet access.
- The system is running the same distribution as the offline system on which you want to install.

1. On your online system, open the Command Prompt and run the following command to install prerequisite software.

   ```
   sudo zypper install python36 createrepo
   ```

2. Run the following command to apply the latest system updates to avoid installation issues.

   ```
   sudo zypper update
   ```

   Note This command may update your kernel to the latest version.

3. Restart your system.
4. Open the Command Prompt and run the following command to install insserv.
   `sudo zypper install insserv`
5. Run the following command to install the repository registration package.
   `sudo zypper --no-gpg-checks install ./filename.rpm`

   Where

   - *filename.rpm* is the repository definition you chose in ***Installing NI Drivers and Software on Linux Desktop***.
6. Download the `offline_rpm_repo.py` script.
7. Complete the steps based on your goals.

Option	Description
Clone the entire repo	a. Run the following command to list all known and enabled repositories on the system. `sudo zypper repos --show-enabled-only`

Option	Description
	Note On OpenSUSE Leap 15.2, the script requires the alias of the repository and not the name. b. To clone the repository, run the following command from the directory that contains the script. `./offline_rpm_repo.py repo --norepopath repo-name` Where ▪ *repo-name* is the name of the repository. By default, the script clones the new repository in the `offline_repo` subdirectory within the directory where the script runs. Pass the `--output` argument to specify a different location.
Create a repo for specific products	Open the Command Prompt and run the following command. `./offline_rpm_repo.py packages package-names` Where ◦ *package-names* is a space-separated list of the package names you want to include.

By default, the script clones the new repository in the `offline_repo` subdirectory within the directory where the script runs. Pass the `--output` argument to specify a different location.

8. Copy the repository to the target system.

Note Choose a location that will not be modified accidentally.

9. Run the following command to disable GPG checking.
   `sudo zypper addrepo --no-gpgcheck --name repo-name file:///srv/offline_repo`

   Where

   - *repo-name* is the name you give your repository.

Complete the installation by following the online install instructions in ***Installing NI Products (openSUSE)***.

Related tasks:

- Installing NI Drivers and Software on Linux Desktop
- Installing NI Products (openSUSE)

Related information:

- offline_rpm_repo.py