Red Hat Enterprise Linux 10-beta

Managing software with the DNF tool

Managing content in the RPM repositories by using the DNF software management tool

Last Updated: 2025-03-04

Red Hat Enterprise Linux 10-beta Managing software with the DNF tool

Managing content in the RPM repositories by using the DNF software management tool

Legal Notice

Abstract

Find, install, and utilize content distributed through the RPM repositories by using the DNF tool.

Table of Contents

RHEL BETA RELEASE 4

CHAPTER 1. DISTRIBUTION OF CONTENT IN RHEL 10 5
1.1. REPOSITORIES 5
1.2. APPLICATION STREAMS 5

CHAPTER 2. CONFIGURING DNF 6
2.1. VIEWING THE CURRENT DNF CONFIGURATIONS 6
2.2. SETTING DNF MAIN OPTIONS 6
2.3. MANAGING DNF PLUG-INS 6
2.4. ENABLING AND DISABLING DNF PLUG-INS 6
2.5. EXCLUDING PACKAGES FROM DNF OPERATIONS 7

CHAPTER 3. SEARCHING FOR RHEL CONTENT 9
3.1. SEARCHING FOR SOFTWARE PACKAGES 9
3.2. LISTING SOFTWARE PACKAGES 9
3.3. DISPLAYING PACKAGE INFORMATION 10
3.4. LISTING PACKAGE GROUPS AND PACKAGES THEY PROVIDE 11
3.5. LISTING REPOSITORIES 11
3.6. SPECIFYING GLOBAL EXPRESSIONS IN DNF INPUT 12
3.7. ADDITIONAL RESOURCES 12

CHAPTER 4. INSTALLING RHEL CONTENT 13
4.1. INSTALLING PACKAGES 13
4.2. INSTALLING PACKAGE GROUPS 13
4.3. ADDITIONAL RESOURCES 14

CHAPTER 5. UPDATING RHEL CONTENT 15
5.1. CHECKING FOR UPDATES 15
5.2. UPDATING PACKAGES 15
5.3. UPDATING SECURITY-RELATED PACKAGES 15

CHAPTER 6. AUTOMATING SOFTWARE UPDATES IN RHEL 17
6.1. INSTALLING DNF AUTOMATIC 17
6.2. DNF AUTOMATIC CONFIGURATION FILE 17
6.3. ENABLING DNF AUTOMATIC 18
6.4. OVERVIEW OF THE SYSTEMD TIMER UNITS INCLUDED IN THE DNF-AUTOMATIC PACKAGE 19

CHAPTER 7. REMOVING RHEL CONTENT 21
7.1. REMOVING INSTALLED PACKAGES 21
7.2. REMOVING PACKAGE GROUPS 21
7.3. ADDITIONAL RESOURCES 21

CHAPTER 8. HANDLING PACKAGE MANAGEMENT HISTORY 22
8.1. LISTING DNF TRANSACTIONS 22
8.2. REVERTING DNF TRANSACTIONS 23
8.2.1. Reverting a single DNF transaction 23
8.2.2. Reverting multiple DNF transactions 24

CHAPTER 9. MANAGING CUSTOM SOFTWARE REPOSITORIES 25
9.1. DNF REPOSITORY OPTIONS 25
9.2. ADDING A DNF REPOSITORY 25
9.3. ENABLING A DNF REPOSITORY 26
9.4. DISABLING A DNF REPOSITORY 26

APPENDIX A. DNF COMMANDS LIST 27
A.1. COMMANDS FOR LISTING CONTENT IN RHEL 27
A.2. COMMANDS FOR INSTALLING CONTENT IN RHEL 28
A.3. COMMANDS FOR REMOVING CONTENT IN RHEL 28

RHEL BETA RELEASE

Red Hat provides Red Hat Enterprise Linux Beta access to all subscribed Red Hat accounts. The purpose of Beta access is to:

- Provide an opportunity to customers to test major features and capabilities prior to the general availability release and provide feedback or report issues.
- Provide Beta product documentation as a preview. Beta product documentation is under development and is subject to substantial change.

Note that Red Hat does not support the usage of RHEL Beta releases in production use cases. For more information, see the Red Hat Knowledgebase solution What does Beta mean in Red Hat Enterprise Linux and can I upgrade a RHEL Beta installation to a General Availability (GA) release?.

CHAPTER 1. DISTRIBUTION OF CONTENT IN RHEL 10

In the following sections, learn how the software is distributed in Red Hat Enterprise Linux (RHEL) 10.

1.1. REPOSITORIES

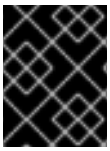
Red Hat Enterprise Linux distributes content through different repositories, for example:

BaseOS

Content in the BaseOS repository consists of the core set of the underlying operating system functionality that provides the foundation for all installations. This content is available in the RPM format and is subject to support terms similar to those in earlier releases of RHEL.

AppStream

Content in the AppStream repository includes additional user-space applications, runtime languages, and databases in support of the varied workloads and use cases.



IMPORTANT

Both the BaseOS and AppStream content sets are required by RHEL and are available in all RHEL subscriptions.

CodeReady Linux Builder

The CodeReady Linux Builder repository is available with all RHEL subscriptions. It provides additional packages for use by developers. Red Hat does not support packages included in the CodeReady Linux Builder repository.

1.2. APPLICATION STREAMS

Red Hat provides multiple versions of user-space components as Application Streams, and they are updated more frequently than the core operating system packages. This provides more flexibility to customize RHEL without impacting the underlying stability of the platform or specific deployments.

Application Streams are available in the following formats:

- RPM format
- Software Collections

RHEL 10 provides initial Application Stream versions as RPMs, which you can install by using the **dnf install** command.

CHAPTER 2. CONFIGURING DNF

The configuration of **DNF** and related utilities is stored in the **[main]** section of the **/etc/dnf/dnf.conf** file.

2.1. VIEWING THE CURRENT DNF CONFIGURATIONS

The **[main]** section in the **/etc/dnf/dnf.conf** file contains only the settings that have been explicitly set. However, you can display all settings of the **[main]** section, including the ones that have not been set and which, therefore, use their default values.

Procedure

- Display the global **DNF** configuration:

```
# dnf config-manager --dump
```

Additional resources

- **dnf.conf(5)** man page on your system

2.2. SETTING DNF MAIN OPTIONS

The **/etc/dnf/dnf.conf** file contains one **[main]** section. The key-value pairs in this section affect how **DNF** operates and treats repositories.

Procedure

1. Edit the **/etc/dnf/dnf.conf** file.
2. Update the **[main]** section according to your requirements.
3. Save the changes.

Additional resources

- The **[main] OPTIONS** and **OPTIONS FOR BOTH [main] AND REPO** sections in the **dnf.conf(5)** man page on your system.

2.3. MANAGING DNF PLUG-INS

Every installed plug-in can have its own configuration file in the **/etc/dnf/plugins/** directory. Name plug-in configuration files in this directory ***<plug-in_name>*.conf**. By default, plug-ins are typically enabled. To disable a plug-in in one of these configuration files, add the following to the file:

```
[main]
enabled=False
```

2.4. ENABLING AND DISABLING DNF PLUG-INS

In the **DNF** tool, plug-ins are loaded by default. However, you can influence which plug-ins **DNF** loads.

WARNING

Disable all plug-ins only for diagnosing a potential problem. **DNF** requires certain plug-ins, such as **product-id** and **subscription-manager**, and disabling them causes Red Hat Enterprise Linux to not be able to install or update software from the Content Delivery Network (CDN).

Procedure

- Use one of the following methods to influence how **DNF** uses plug-ins:
  - To enable or disable loading of **DNF** plug-ins globally, add the **plugins** parameter to the **[main]** section of the **/etc/dnf/dnf.conf** file.
    - Set **plugins=1** (default) to enable loading of all **DNF** plug-ins.
    - Set **plugins=0** to disable loading of all **DNF** plug-ins.
  - To disable a particular plug-in, add **enabled=False** to the **[main]** section in the **/etc/dnf/plugins/*<plug-in_name>*.conf** file.
  - To disable all **DNF** plug-ins for a particular command, append the **--noplugins** option to the command. For example, to disable **DNF** plug-ins for a single update command, enter:

    ```
    # dnf --noplugins update
    ```

  - To disable certain **DNF** plug-ins for a single command, append the **--disableplugin=*plugin-name*** option to the command. For example, to disable a certain **DNF** plug-in for a single update command, enter:

    ```
    # dnf update --disableplugin=<plugin_name>
    ```

  - To enable certain **DNF** plug-ins for a single command, append the **--enableplugin=*plugin-name*** option to the command. For example, to enable a certain **DNF** plug-in for a single update command, enter:

    ```
    # dnf update --enableplugin=<plugin_name>
    ```

2.5. EXCLUDING PACKAGES FROM DNF OPERATIONS

You can configure **DNF** to exclude packages from any **DNF** operation by using the **excludepkgs** option. You can define **excludepkgs** in the **[main]** or the repository section of the **/etc/dnf/dnf.conf** file.

NOTE

You can temporarily disable excluding the configured packages from an operation by using the **--disableexcludes** option.

Procedure

- Exclude packages from the **DNF** operation by adding the following line to the **/etc/dnf/dnf.conf** file:

```
excludepkgs=<package_name_1>,<package_name_2> ...
```

  Alternatively, use global expressions instead of package names to define packages you want to exclude. For more information, see Specifying global expressions in DNF input .

Additional resources

- **dnf.conf(5)** man page on your system
- Specifying global expressions in DNF input

CHAPTER 3. SEARCHING FOR RHEL CONTENT

In the following sections, learn how to locate and examine content in the AppStream and BaseOS repositories by using the **DNF** software management tool.

3.1. SEARCHING FOR SOFTWARE PACKAGES

To identify which package provides the software you require, you can use **DNF** to search the repositories.

Procedure

- Depending on your scenario, use one of the following options to search the repository:
  - To search for a term in the name or summary of packages, enter:

    ```
    $ dnf search <term>
    ```

  - To search for a term in the name, summary, or description of packages, enter:

    ```
    $ dnf search --all <term>
    ```

    Note that searching additionally in the description by using the **--all** option is slower than a normal search operation.

  - To search for a package name and list the package name and its version in the output, enter:

    ```
    $ dnf repoquery <package_name>
    ```

  - To search for which package provides a file, specify the file name or the path to the file:

    ```
    $ dnf provides <file_name>
    ```

3.2. LISTING SOFTWARE PACKAGES

You can use **DNF** to display a list of packages and their versions that are available in the repositories. If required, you can filter this list and, for example, only list packages for which updates are available.

Procedure

- List the latest versions of all available packages, including architectures, version numbers, and the repository they where installed from:

  ```
  $ dnf list --all
  ...
  postgresql.x86_64            16.4-1.el10      rhel-AppStream
  postgresql-contrib.x86_64    16.4-1.el10      rhel-AppStream
  postgresql-docs.x86_64       16.4-1.el10      rhel-AppStream
  postgresql-jdbc.noarch       42.7.1-6.el10    rhel-AppStream
  ...
  ```

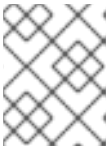
  The @ sign in front of a repository indicates that the package in this line is currently installed.

Alternatively, to display all available packages, including version numbers and architectures, enter:

```
$ dnf repoquery
...
postgresql-0:16.4-1.el10.x86_64
postgresql-contrib-0:16.4-1.el10.x86_64
postgresql-docs-0:16.4-1.el10.x86_64
postgresql-jdbc-0:42.7.1-6.el10.noarch
postgresql-odbc-0:16.00.0000-4.el10.x86_64
...
```

Optionally, you can filter the output by using other options instead of **--all**, for example:

- Use **--installed** to list only installed packages.
- Use **--available** to list all available packages.
- Use **--upgrades** to list packages for which newer versions are available.



NOTE

You can filter the results by appending global expressions as arguments. For more details, see Specifying global expressions in DNF input .

3.3. DISPLAYING PACKAGE INFORMATION

You can query **DNF** repositories to display further details about a package, such as the following:

- Version
- Release
- Architecture
- Package size
- Description

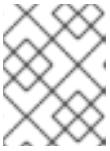
Procedure

- Display information about one or more available packages:

```
$ dnf info <package_name>
```

This command displays the information for the currently installed package and, if available, its newer versions that are in the repository. Alternatively, use the following command to display the information for all packages with the specified name in the repository:

```
$ dnf repoquery --info <package_name>
```



NOTE

You can filter the results by appending global expressions as arguments. For details, see Specifying global expressions in DNF input .

3.4. LISTING PACKAGE GROUPS AND PACKAGES THEY PROVIDE

Package groups bundle multiple packages, and you can use package groups to install all packages assigned to a group in a single step. However, before the installation, you must identify the name of the required package group.

Procedure

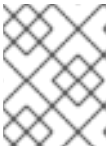
1. List both installed and available groups:

```
$ dnf group list
```

Note that you can filter the results by appending the **--installed** and **--available** option to the **dnf group list** command. By using the **--hidden** option, you can display hidden groups in the output.

2. List mandatory, optional, and default packages contained in a particular group:

```
$ dnf group info "<group_name>"
```



NOTE

You can filter the results by appending global expressions as arguments. For more details, see Specifying global expressions in DNF input .

3. Optional: View the number of installed and available groups:

```
$ dnf group summary
```

3.5. LISTING REPOSITORIES

To get an overview of repositories that are enabled and disabled on your system, you can list them.

Procedure

1. List all enabled repositories on your system:

```
$ dnf repolist
```

To display only certain repositories, append one of the following options to the command:

- Append **--disabled** to list only disabled repositories.
- Append **--all** to list both enabled and disabled repositories.

2. Optional: List additional information about the repositories:

```
$ dnf repoinfo <repository_name>
```

NOTE

You can filter the results by using global expressions. For details, see Specifying global expressions in DNF input.

3.6. SPECIFYING GLOBAL EXPRESSIONS IN DNF INPUT

You can filter the results of **dnf** commands by appending one or more global expressions as arguments.

Procedure

- Use one of the following methods if you use global expressions in **dnf** commands:
  - Enclose the entire global expression in single or double quotation marks:

    ```
    # dnf provides "*/<file_name>"
    ```

    Note that you must precede ***<file_name>*** either by / for an absolute path or */ to use a wildcard if the full path is unknown.
  - Escape the wildcard characters by preceding them with a backslash (\) character:

    ```
    # dnf provides \*/<file_name>
    ```

3.7. ADDITIONAL RESOURCES

- Commands for listing content in RHEL

CHAPTER 4. INSTALLING RHEL CONTENT

In the following sections, learn how to install Red Hat Enterprise Linux content by using the **DNF** software management tool.

4.1. INSTALLING PACKAGES

If a software is not part of the default installation, you can manually install it. **DNF** automatically resolves and installs dependencies.

Prerequisites

- Optional: You know the name of the package you want to install .

Procedure

- Use one of the following methods to install packages:
  - To install packages from the repositories, enter:

    ```
    # dnf install <package_name_1> <package_name_2> ...
    ```

    If you install packages on a system that supports multiple architectures, such as **i686** and **x86_64**, you can specify the architecture of the package by appending it to the package name:

    ```
    # dnf install <package_name>.<architecture>
    ```

  - To install a package if you only know the path to the file the package provides but not the package name, you can use this path to install the corresponding package:

    ```
    # dnf install <path_to_file>
    ```

  - To install a local RPM file, enter:

    ```
    # dnf install <path_to_RPM_file>
    ```

    If the package has dependencies, specify the paths to these RPM files as well. Otherwise, **DNF** downloads the dependencies from the repositories or fails if they are not available in the repositories.

4.2. INSTALLING PACKAGE GROUPS

Package groups bundle multiple packages, and you can use package groups to install all packages assigned to a group in a single step.

Prerequisites

- You know the name or ID of the group you want to install .

Procedure

- Install a package group:

```
# dnf group install <group_name_or_ID>
```

4.3. ADDITIONAL RESOURCES

- Commands for installing content in RHEL

CHAPTER 5. UPDATING RHEL CONTENT

With the **DNF** software management tool, you can check if your system has any pending updates. You can list packages that need updating and choose to update a single package, multiple packages, or all packages at once. If any of the packages you choose to update have dependencies, these dependencies are updated as well.

5.1. CHECKING FOR UPDATES

To identify which packages installed on your system have available updates, you can list them.

Procedure

- Check the available updates for installed packages:

```
# dnf check-update
```

  The output returns the list of packages and their dependencies that have an update available.

5.2. UPDATING PACKAGES

You can use **DNF** to update a single package, a package group, or all packages and their dependencies at once.

IMPORTANT

When applying updates to the kernel, **dnf** always installs a new kernel regardless of whether you are using the **dnf upgrade** or **dnf install** command. Note that this only applies to packages identified by using the **installonlypkgs** DNF configuration option. Such packages include, for example, the **kernel**, **kernel-core**, and **kernel-modules** packages.

Procedure

- Depending on your scenario, use one of the following options to apply updates:
  - To update all packages and their dependencies, enter:

```
# dnf upgrade
```

  - To update a single package, enter:

```
# dnf upgrade <package_name>
```

  - To update packages only from a specific package group, enter:

```
# dnf group upgrade <group_name>
```

5.3. UPDATING SECURITY-RELATED PACKAGES

You can use **DNF** to update security-related packages.

Procedure

- Depending on your scenario, use one of the following options to apply updates:
  - To upgrade to the latest available packages that have security errata, enter:

    ```
    # dnf upgrade --security
    ```

  - To upgrade to the last security errata packages, enter:

    ```
    # dnf upgrade-minimal --security
    ```

CHAPTER 6. AUTOMATING SOFTWARE UPDATES IN RHEL

DNF Automatic is an alternative command-line interface to **DNF** that is suited for automatic and regular execution by using systemd timers, cron jobs, and other such tools.

DNF Automatic synchronizes package metadata as needed, checks for updates available, and then performs one of the following actions depending on how you configure the tool:

- Exit
- Download updated packages
- Download and apply the updates

The outcome of the operation is then reported by a selected mechanism, such as the standard output or email.

6.1. INSTALLING DNF AUTOMATIC

To check and download package updates automatically and regularly, you can use the **DNF Automatic** tool that is provided by the **dnf-automatic** package.

Procedure

- Install the **dnf-automatic** package:

```
# dnf install dnf-automatic
```

Verification

- Verify the successful installation by confirming the presence of the **dnf-automatic** package:

```
# rpm -qi dnf-automatic
```

6.2. DNF AUTOMATIC CONFIGURATION FILE

By default, **DNF Automatic** uses **/etc/dnf/automatic.conf** as its configuration file to define its behavior.

The configuration file is separated into the following topical sections:

- **[commands]**
  Sets the mode of operation of **DNF Automatic**.

WARNING

Settings of the operation mode from the **[commands]** section are overridden by settings used by a systemd timer unit for all timer units except **dnf-automatic.timer**.

- **[emitters]**
  Defines how the results of **DNF Automatic** are reported.
- **[command]**
  Defines the command emitter configuration.
- **[command_email]**
  Provides the email emitter configuration for an external command used to send email.
- **[email]**
  Provides the email emitter configuration.
- **[base]**
  Overrides settings from the main configuration file of **DNF**.

With the default settings of the **/etc/dnf/automatic.conf** file, **DNF Automatic** checks for available updates, downloads them, and reports the results to standard output.

Additional resources

- **dnf-automatic(8)** man page on your system
- Overview of the systemd timer units included in the dnf-automatic package

6.3. ENABLING DNF AUTOMATIC

To run **DNF Automatic** once, you must start a systemd timer unit. However, if you want to run **DNF Automatic** periodically, you must enable the timer unit. You can use one of the timer units provided in the **dnf-automatic** package, or you can create a drop-in file for the timer unit to adjust the execution time.

Prerequisites

- You specified the behavior of **DNF Automatic** by modifying the **/etc/dnf/automatic.conf** configuration file.

Procedure

- To enable and execute a systemd timer unit immediately, enter:

  ```
  # systemctl enable --now <timer_name>
  ```

  If you want to only enable the timer without executing it immediately, omit the **--now** option.

  You can use the following timers:

  - **dnf-automatic-download.timer**: Downloads available updates.
  - **dnf-automatic-install.timer**: Downloads and installs available updates.
  - **dnf-automatic-notifyonly.timer**: Reports available updates.
  - **dnf-automatic.timer**: Downloads, downloads and installs, or reports available updates.

Verification

- Verify that the timer is enabled:

```
# systemctl status <systemd timer unit>
```

- Optional: Check when each of the timers on your system ran the last time:

```
# systemctl list-timers --all
```

Additional resources

- **dnf-automatic(8)** man page on your system
- Overview of the systemd timer units included in the dnf-automatic package

6.4. OVERVIEW OF THE SYSTEMD TIMER UNITS INCLUDED IN THE DNF-AUTOMATIC PACKAGE

The systemd timer units take precedence and override the settings in the **/etc/dnf/automatic.conf** configuration file when downloading and applying updates.

For example if you set **download_updates = yes** in the **/etc/dnf/automatic.conf** configuration file, but you have activated the **dnf-automatic-notifyonly.timer unit**, the packages will not be downloaded.

Table 6.1. systemd timers included in the dnf-automatic package

Timer unit	Function	Overrides the **apply_updates** and **download_updates** settings in the **[commands]** section of the **/etc/dnf/automatic.conf** file?
dnf-automatic-download.timer	Downloads packages to cache and makes them available for updating. This timer unit does not install the updated packages. To perform the installation, you must run the **dnf update** command.	Yes
dnf-automatic-install.timer	Downloads and installs updated packages.	Yes
dnf-automatic-notifyonly.timer	Downloads only repository data to keep the repository cache up-to-date and notifies you about available updates. This timer unit does not download or install the updated packages.	Yes

Timer unit	Function	Overrides the **apply_updates** and **download_updates** settings in the **[commands]** section of the **/etc/dnf/automatic.conf** file?
dnf-automatic.timer	The behavior of this timer when downloading and applying updates is specified by the settings in the **/etc/dnf/automatic.conf** configuration file. This timer downloads packages, but does not install them.	No

CHAPTER 7. REMOVING RHEL CONTENT

In the following sections, learn how to remove content in Red Hat Enterprise Linux 10 by using the **DNF** software management tool.

7.1. REMOVING INSTALLED PACKAGES

You can use **DNF** to remove a single package or multiple packages installed on your system. If any of the packages you choose to remove have unused dependencies, **DNF** uninstalls these dependencies as well.

Procedure

- Remove particular packages:

```
# dnf remove <package_name_1> <package_name_2> ...
```

7.2. REMOVING PACKAGE GROUPS

Package groups bundle multiple packages. You can use package groups to remove all packages assigned to a group in a single step.

Procedure

- Remove package groups by the group name or group ID:

```
# dnf group remove <group_name> <group_id>
```

7.3. ADDITIONAL RESOURCES

- Commands for removing content in RHEL

CHAPTER 8. HANDLING PACKAGE MANAGEMENT HISTORY

With the **dnf history** command, you can review the following information:

- Timeline of **DNF** transactions.
- Dates and times the transactions occurred.
- Number of packages affected by the transactions.
- Whether the transactions succeeded or were aborted.
- If the RPM database was changed between the transactions.

You can also use the **dnf history** command to undo the transactions.

8.1. LISTING DNF TRANSACTIONS

You can use the **DNF** software management tool to perform the following tasks:

- List the latest transactions.
- List the latest operations for a selected package.
- Display details of a particular transaction.

Procedure

- Depending on your scenario, use one of the following options to display transaction information:
  - To display a list of all the latest **DNF** transactions, enter:

    ```
    # dnf history
    ```

    The output contains the following information:

    - The **Action(s)** column displays which type of action was performed during a transaction, for example, Install (**I**), Upgrade (**U**), Remove (**E**), and other actions.
    - The **Altered** column displays the number of actions performed during the transaction. The number of actions can also be followed by the result of the transaction.
      For more information about the values of the **Action(s)** and **Altered** columns, see the **dnf(8)** man page.
  - To display a list of all the latest operations for a selected package, enter:

    ```
    # dnf history list <package_name>
    ```

  - To display details of a particular transaction, enter:

    ```
    # dnf history info <transaction_id>
    ```

NOTE

You can filter the results by appending global expressions as arguments. For more details, see Specifying global expressions in DNF input .

Additional resources

- **dnf(8)** man page on your system

8.2. REVERTING DNF TRANSACTIONS

Reverting a **DNF** transaction can be useful if you want to undo operations performed during the transaction. For example, if you installed several packages by using the **dnf install** command, you can uninstall these packages at once by reverting an installation transaction.

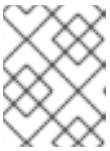
You can revert **DNF** transactions the following ways:

- Revert a single **DNF** transaction by using the **dnf history undo** command.
- Revert all **DNF** transactions performed between the specified transaction and the last transaction by using the **dnf history rollback** command.

8.2.1. Reverting a single DNF transaction

You can revert steps performed within a single transaction by using the **dnf history undo** command:

- If the transaction installed a new package, **dnf history undo** uninstalls the package.
- If the transaction uninstalled a package, **dnf history undo** reinstalls the package.
- The **dnf history undo** command also attempts to downgrade all updated packages to their previous versions if the older packages are still available.



NOTE

If an older package version is not available, the downgrade by using the **dnf history undo** command fails.

Procedure

1. Identify the ID of a transaction you want to revert:

```
# dnf history
ID | Command line     | Date and time     | Action(s)      | Altered
-------------------------------------------------------------------------
13 | install zip      | 2022-11-03 10:49  | Install        |    1
12 | install unzip    | 2022-11-03 10:49  | Install        |    1
```

2. Optional: Verify that this is the transaction you want to revert by displaying its details:

```
# dnf history info <transaction_id>
```

3. Revert the transaction:

```
# dnf history undo <transaction_id>
```

For example, if you want to uninstall the previously installed **unzip** package, enter:

```
# dnf history undo 12
```

8.2.2. Reverting multiple DNF transactions

You can revert all **DNF** transactions performed between a specified transaction and the last transaction by using the **dnf history rollback** command. Note that the transaction specified by the transaction ID remains unchanged.

Procedure

1. Identify the transaction ID of the state you want to revert to:

```
# dnf history
ID | Command line     | Date and time     | Action(s)   | Altered
-------------------------------------------------------------------
14 | install wget     | 2022-11-03 10:49  | Install     |    1
13 | install unzip    | 2022-11-03 10:49  | Install     |    1
12 | install vim-X11  | 2022-11-03 10:20  | Install     |  171 EE
```

2. Revert specified transactions:

```
# dnf history rollback <transaction_id>
```

For example, to revert to the state before the **wget** and **unzip** packages were installed, enter:

```
# dnf history rollback 12
```

Alternatively, to revert all transactions in the transaction history, use the transaction ID **1**:

```
# dnf history rollback 1
```

CHAPTER 9. MANAGING CUSTOM SOFTWARE REPOSITORIES

You can configure a repository in the **/etc/dnf/dnf.conf** file or in a **.repo** file in the **/etc/yum.repos.d/** directory.

IMPORTANT

Define your repositories in the **.repo** file instead of **/etc/dnf/dnf.conf**.

The **/etc/dnf/dnf.conf** file contains the **[main]** section and can contain one or more repository sections (**[*<repository-ID>*]**) that you can use to set repository-specific options. The values you define in individual repository sections of the **/etc/dnf/dnf.conf** file override values set in the **[main]** section.

9.1. DNF REPOSITORY OPTIONS

The **/etc/dnf/dnf.conf** configuration file contains repository sections with a unique repository ID in brackets (**[]**). You can use such sections to define individual **DNF** repositories.

IMPORTANT

Repository IDs in **[]** must be unique.

For a complete list of available repository ID options, see the **[*<repository-ID>*] OPTIONS** section of the **dnf.conf(5)** man page.

9.2. ADDING A DNF REPOSITORY

You can add a **DNF** repository to your system by using the **dnf config-manager --add-repo** command.

Procedure

1. Add a repository to your system:

   ```
   # dnf config-manager --add-repo <repository_URL>
   ```

   Note that repositories added by this command are enabled by default.

2. Review and, optionally, update the repository settings that the previous command has created in the **/etc/yum.repos.d/<repository_URL>.repo** file:

   ```
   # cat /etc/yum.repos.d/<repository_URL>.repo
   ```

WARNING

Obtaining and installing software packages from unverified or untrusted sources other than Red Hat certificate-based **Content Delivery Network** (CDN) is a potential security risk, and can lead to security, stability, compatibility, and maintainability issues.

9.3. ENABLING A DNF REPOSITORY

You can enable a **DNF** repository added to your system by using the **dnf config-manager** command.

Procedure

- Enable a repository:

```
# dnf config-manager --enable <repository_id>
```

9.4. DISABLING A DNF REPOSITORY

You can disable a **DNF** repository added to your system by using the **dnf config-manager** command.

Procedure

- Disable a repository:

```
# dnf config-manager --disable <repository_id>
```

APPENDIX A. DNF COMMANDS LIST

In the following sections, examine **DNF** commands for listing, installing, and removing content in Red Hat Enterprise Linux 10.

A.1. COMMANDS FOR LISTING CONTENT IN RHEL

The following are the commonly used **DNF** commands for finding content and its details in Red Hat Enterprise Linux 10:

Command	Description
dnf search *term*	Search for a package by using term related to the package.
dnf repoquery *package*	Search for enabled **DNF** repositories for a selected package and its version.
dnf list	List information about all installed and available packages.
dnf list --installed **dnf repoquery --installed**	List all packages installed on your system.
dnf list --available **dnf repoquery**	List all packages in all enabled repositories that are available to install.
dnf repolist	List all enabled repositories on your system.
dnf repolist --disabled	List all disabled repositories on your system.
dnf repolist --all	List both enabled and disabled repositories.
dnf repoinfo	List additional information about the repositories.
dnf info *package_name* **dnf repoquery --info *package_name***	Display details of an available package.
dnf repoquery --info --installed *package_name*	Display details of a package installed on your system.
dnf group summary	View the number of installed and available groups.
dnf group list	List all installed and available groups.

Command	Description
dnf group info *group_name*	List mandatory and optional packages included in a particular group.

A.2. COMMANDS FOR INSTALLING CONTENT IN RHEL

The following are the commonly used **DNF** commands for installing content in Red Hat Enterprise Linux 10:

Command	Description
dnf install *package_name*	Install a package.
dnf install *package_name_1 package_name_2*	Install multiple packages and their dependencies simultaneously.
dnf install *package_name.arch*	Specify the architecture of the package by appending it to the package name when installing packages on a *multilib* system (AMD64, Intel 64 machine).
dnf install */usr/sbin/binary_file*	Install a binary by using the path to the binary as an argument.
dnf install */path/*	Install a previously downloaded package from a local directory.
dnf install *package_url*	Install a remote package by using a package URL.
dnf group install *group_name*	Install a package group by a group name.
dnf group install *group_ID*	Install a package group by the groupID.

A.3. COMMANDS FOR REMOVING CONTENT IN RHEL

The following are the commonly used **DNF** commands for removing content in Red Hat Enterprise Linux 10:

Command	Description
dnf remove *package_name*	Remove a particular package and all dependent packages.
dnf remove *package_name_1 package_name_2*	Remove multiple packages and their unused dependencies simultaneously.

Command	Description
dnf group remove group_name	Remove a package group by the group name.
dnf group remove *group_ID*	Remove a package group by the groupID.