CHAPTER-14 INSTALL AND UPDATE SOFTWARE PACKAGES

Register Systems for Red Hat Support

Red Hat Subscription Management provides tools to entitle machines to product subscriptions, for administrators to get updates to software packages and to track information about support contracts and subscriptions that the systems use. Standard tools such as the dnf command obtain software packages and updates through a content distribution network that the Red Hat Content Delivery Network provides.

You can perform the following main tasks with the Red Hat Subscription Management tools:

- Register a system to associate it with the Red Hat account with an active subscription. With the Subscription Manager, the system can register uniquely in the subscription service inventory. You can unregister the system when not in use.
- Subscribe a system to entitle it to updates for the selected Red Hat products. Subscriptions have specific levels of support, expiration dates, and default repositories. The tools help to either auto-attach or select a specific entitlement.
- Enable repositories to provide software packages. By default, each subscription enables multiple repositories; other repositories such as updates or source code are enabled or disabled. A repository is a central location for storing and maintaining software packages.
- Review and track available or consumed entitlements. In the Red Hat Customer Portal, you might view the subscription information locally on a specific system or for a Red Hat account.

Simple Content Access

Simple Content Access (SCA) is a Red Hat subscription management capability. When you enable SCA for your organization, the entitlement process is simplified. SCA eliminates the requirement to attach subscriptions at a per-system level. You register your systems, enable the repositories that each system needs, and begin installing software packages.

Subscribe a System with the Command Line

DESCRIPTION	COMMANDS / OPTIONS
subscription-manager	**subscription-manager [options]** To register system without using gui --register -- username list -- available attach -- auto list - consumed unregister **Register a system by using the credentials of the Red Hat Customer Portal as the root user:** **[root@host ~]#** subscription-manager register --username <yourusername> **View available subscriptions for your Red Hat account:** **[root@host ~]#** subscription-manager list –available **Auto-attach a subscription:** [root@host ~]# subscription-manager attach –auto **View consumed subscriptions:** [root@host ~]# subscription-manager list –consumed **Unregister a system:** [root@host ~]# subscription-manager unregister

Activation Keys

An activation key is a preconfigured subscription management file that available for use with both Red Hat Satellite Server and subscription management through the Red Hat Customer Portal.

Entitlement Certificates

Digital certificates store current entitlement information on the local system. The registered system stores the entitlement certificates under the **/etc/pki directory**.

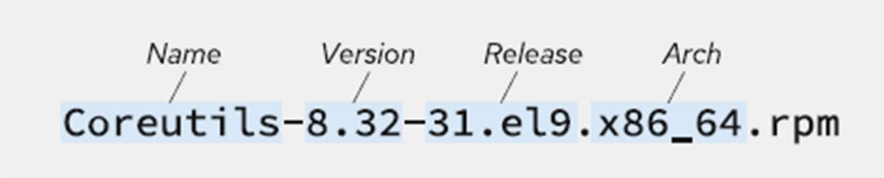
- **etc/pki/product** certificates indicates installed Red Hat products.
- **/etc/pki/consumer** certificates identifies the Red Hat account for registration.
- **/etc/pki/entitlement** certificates indicate which subscriptions are attached.

Explain and Investigate RPM Software Packages

Software Packages and RPM

The RPM Package Manager, which Red Hat originally developed, provides a standard way to package software for distribution. Managing software in the form of RPM packages is simpler than working with software that is extracted to a file system from an archive. With RPM packages, administrators can track which files the software package installs, which files the software package removes if you uninstall it, and it checks to ensure that supporting packages are present when you install it. The local RPM database on your system stores the information about installed packages. Red Hat provides all software for Red Hat Enterprise Linux as an RPM package

RPM package file **names** consist of **four elements** (plus the .rpm suffix):
name - versionrelease. architecture:



```
Coreutils-8.32-31.el9.x86_64.rpm
```

- **NAME** is one or more words describing the contents (coreutils).
- **VERSION** is the version number of the original software (8.32).
- **RELEASE** is the release number of the package based on that version, and is set by the packager, who might not be the original software developer (31.el9).
- **ARCH** is the processor architecture the package is compiled to run on. The x86_64 value indicates that this package is built for the 64-bit version of the x86 instruction set (as opposed to aarch64 for 64-bit ARM, and so on).

RPM packages are often downloaded from repositories. A repository is a central location for storing and maintaining RPM software packages. You require only the package name to install RPM packages from repositories.

- If multiple versions exist, then the RPM Package Manager installs the package with the later version number.
- If multiple releases of a single version exist, then the RPM Package Manager installs the package with the later release number.

Each RPM package is an archive with the following components:

- The files that the package installs in your system.
- Information about the package (metadata), such as the name, version, release, and architecture; a summary and description of the package; whether it requires other packages to be installed; licensing; a package change log; and other details.
- Scripts that might run when you install, update, or remove the package. These scripts might also run when you install, update, or remove other packages.

DESCRIPTION	COMMANDS / OPTIONS
Red Hat package Manager (rpm)	**Syntax:** rpm [options] package.rpm **-qa** list installed packages **-qf** Determine what package provides **-q** list the currently installe package version **-qi** get detailed package info **-ql** list the files that the package installs **-qc** list only the configuration files that package install **-qd** list only the doc files that the package installs **-q --** script list the shell script that run **-q--** changelog List the change log info **-qlp** list the files that the local package installs **-i** Install **-v** Verify **-ivh** To install Package from local system **Example:** [root@host ~]# **rpm -ivh** podman-4.0.0-6.el9.x86_64.rpm

Extracting RPM packages

You can extract files from an RPM package file without installing the package using the rpm2cpio utility.

The rpm2cpio utility converts and RPM package to a cpio archive. After the RPM package is converted to a cpio archive, the cpio command can be used to extract a list of files.
Use the cpio command with the -i option to extract files from standard input. Use the -d option to create subdirectories as needed, starting in the current working directory. Use the -v option for verbose output.

DESCRIPTION	COMMANDS / OPTIONS
Extracting RPM Packages	**Syntax:** rpm2cpio<package> | [options] **-i** options to extract files from standard input **-t** To list the files in an RPM package **-v** verbose output

Install and Update Software Packages with DNF

Manage Software Packages with DNF

DNF (Dandified YUM) replaced YUM as the package manager in Red Hat Enterprise Linux 9. DNF commands are functionally the same as YUM commands. For compatibility, YUM commands still exist as symbolic links to DNF:

DESCRIPTION	COMMANDS / OPTIONS
DNF package Management	**Syntax:** dnf [options] package list packageName search KEYWORD info PackageName provides PATHName install PACKAGEName update PACKAGEName list kernel remove PACKAGEName group list group info group install PACKAGEName tail -5 /var/log/dnf.rpm.log View Transaction History history history undo

Manage Package Module Streams with DNF

Traditionally, managing alternative versions of an application's software package and its related packages meant maintaining different repositories for each version. For developers who wanted the latest version of an application and administrators who wanted the most stable version of the application, the resulting situation was tedious to manage. Red Hat simplifies this process by using a technology **called Modularity**. With modularity, a **single repository** can **host multiple versions** of an application's package and its dependencies.

Module Streams

Each module has one or more module streams, which hold different versions of the content. Each of the streams receives updates independently. Think of the module stream as a virtual repository in the Application Stream physical repository.
For each module, you can enable only one of its streams, and this stream provides its packages.

Module Profiles

Each module can have one or more profiles. A profile is a list of packages that you can install together for a particular use-case such as for a server, client, development, minimal installation, or other.

Installing a module profile installs a particular set of packages from the module stream. You can subsequently install or uninstall packages normally. If you do not specify a profile, then the module installs its default profile.

Manage Modules with DNF

Red Hat Enterprise Linux 9 supports modular features of Application Stream. To handle the modular content, you can use the dnf module command. Otherwise, the dnf command works with modules similar to regular packages. You can find some important commands when managing modules in the following list:

DESCRIPTION	COMMANDS / OPTIONS
Manage Package Module Streams with DNF	**Syntax:** dnf [options] package **module list** List the available modules with the module **module list-name** **module info module-name** **module provides package**

Enable DNF Software Repositories

DESCRIPTION	COMMANDS / OPTIONS
	Syntax: dnf [options] package **Enable DNF Software Repositories** repolist all config-manager **eable /disable** **Example :** dnf config-manager --enable rhel-9-server- debug-rpms **config-manager** --add repoURL **repolist all** List all available repositories and their status **To enable and disable repositories** **Example :** dnf config-manager --enable rhel-9-server-debug-rpms **To add repositories to the machine** dnf config-manager \\ --add-repo="https://dl.fedoraproject.org/pub/epel/9/Everything/x86_64/"
Configuration Packages for Local Repositories	vim /etc/yum.repos.d/epel.repo **[epel]** **name=** Extra Packages for Enterprise Linux **baseurl=** https://download.example/pub/epel/$releasever/Everything **enabled=** 1 **gpgcheck=** 1 **gpgkey=** file:///etc/pki/rpm-gpg/RPM-GPG-KEY-EPEL [epel-source] **name=** Extra Packages for Enterprise Linux **baseurl=**https://download.example/pub/epel/$releasever/Everything/ **enabled=** 0 **gpgkey=** file:///etc/pki/rpm-gpg/RPM-GPG-KEY-EPEL- $releasever **gpgcheck=** 1