DNF is the New Default Package Manager Of Fedora 22

 SK

May 28, 2015

Fedora 22 was released two days ago with many added and improved features. As we all know, Fedora is upstream for Red Hat Enterprise Linux which is commercially supported by Red Hat, Inc. It is an American multinational software company providing open-source software products to the enterprise community. Since Fedora 22 comes up with plenty of new features, one of the notable feature is **DNF**.

Good Bye YUM, Hello DNF!

As you may know, **DNF** is the replacement of popular package manager **YUM** of Fedora and other RPM based distros. DNF was forked from YUM in January 2012, and available for experimenting since Fedora 18. DNF stands for **'Dandified Yum'**. According to the Fedora development team, DNF is now fully matured and capable of replacing YUM, and it will be the default package manager in Fedora 22 and newer versions. Yes, Fedora 22 ditched YUM, and bring DNF as it's new default package manager.

DNF is the next upcoming major version of YUM. It does package management using **RPM**, **libsolv** and **hawkey** libraries. For metadata handling and package downloads it utilizes **librepo**. To process and effectively handle the comps data it uses **libcomps**.

In this tutorial, let us see how to use DNF package manager in Fedora 22.

Install DNF

DNF and all its dependencies are available in Fedora 18 and later, including the rawhide Fedora.

If you want to test DNF on your Fedora systems, run the following command from your Terminal.

```
sudo yum install dnf
```

Note: You don't have to install it on Fedora 22, because it comes preinstalled with Fedora 22.

Usage

Synopsis:

```
dnf [options] <command> [<args>...]
```

DNF usage is very similar to YUM.

For example, to install a new package using YUM, we use the following command:

```
sudo yum install <package-name>
```

Similarly, we can install package using DNF as shown below.

To install a package using DNF:

```
sudo dnf install <package-name>
```

To remove a package:

```
sudo dnf remove <package-name>
```

To update the system:

```
sudo dnf update
```

To upgrade the system:

```
sudo dnf upgrade
```

Complete list of DNF Commands

Available commands are:

```
check-update
clean
dist-sync
distribution-sync
downgrade
erase
group
help
history
info
install
list
makecache
provides
reinstall
repolist
search
update
update-to
upgrade
upgrade-to
```

Options

Just like YUM, we can use options to perform a particular action while using DNF commands.

The list of available options are given below:

```
--assumeno

     answer no for all questions

--best

     Try the best available package versions in transactions.

-C, --cacheonly

     Run entirely from system cache, don’t update cache

-c <config file>, --config=<config file>

     config file location

-d <debug level>, --debuglevel=<debug level>

     Debugging output level.

--disableexcludes=[all|main|<repoid>]

     Disable the config file excludes. Takes one of three options:

          - all, disables all config file excludes

          - main, disables excludes defined in the [main] section

          - repoid, disables excludes defined for the given repo

-e <error level>, --errorlevel=<error level>

     Error output level.

-x <package-spec>, --exclude=<package-spec>

     Exclude packages specified by a name or a glob from the operation.

-h, --help

     Shows the help.

--installroot=<path>

     set install root

--nogpgcheck

     skip checking GPG signatures on packages

-q, --quiet
```

```
     quiet operation

-R <minutes>, --randomwait=<minutes>

     maximum command wait time

--releasever=<release>

     configure DNF as if the distribution release was <release>.

--rpmverbosity=<debug level name>

     debugging output level for rpm

--showduplicates

     show duplicates, in repos, in list/search commands

-v, --verbose

     verbose operation, show debug messages.

--version

     show Yum version and exit

-y, --assumeyes

     answer yes for all questions
```

To see the complete DNF Command reference, please visit here.

DNF Configuration

By default, DNF uses the global configuration file at **`/etc/dnf/dnf.conf`** and all ***.repo** files found under **`/etc/yum.repos.d`** directory.

The contents of dnf.conf file:

```
sudo nano /etc/dnf/dnf.conf

[main]

gpgcheck=1

installonly_limit=3

clean_requirements_on_remove=true
```

There are two types of sections in the configuration files: **main** and **repository**.

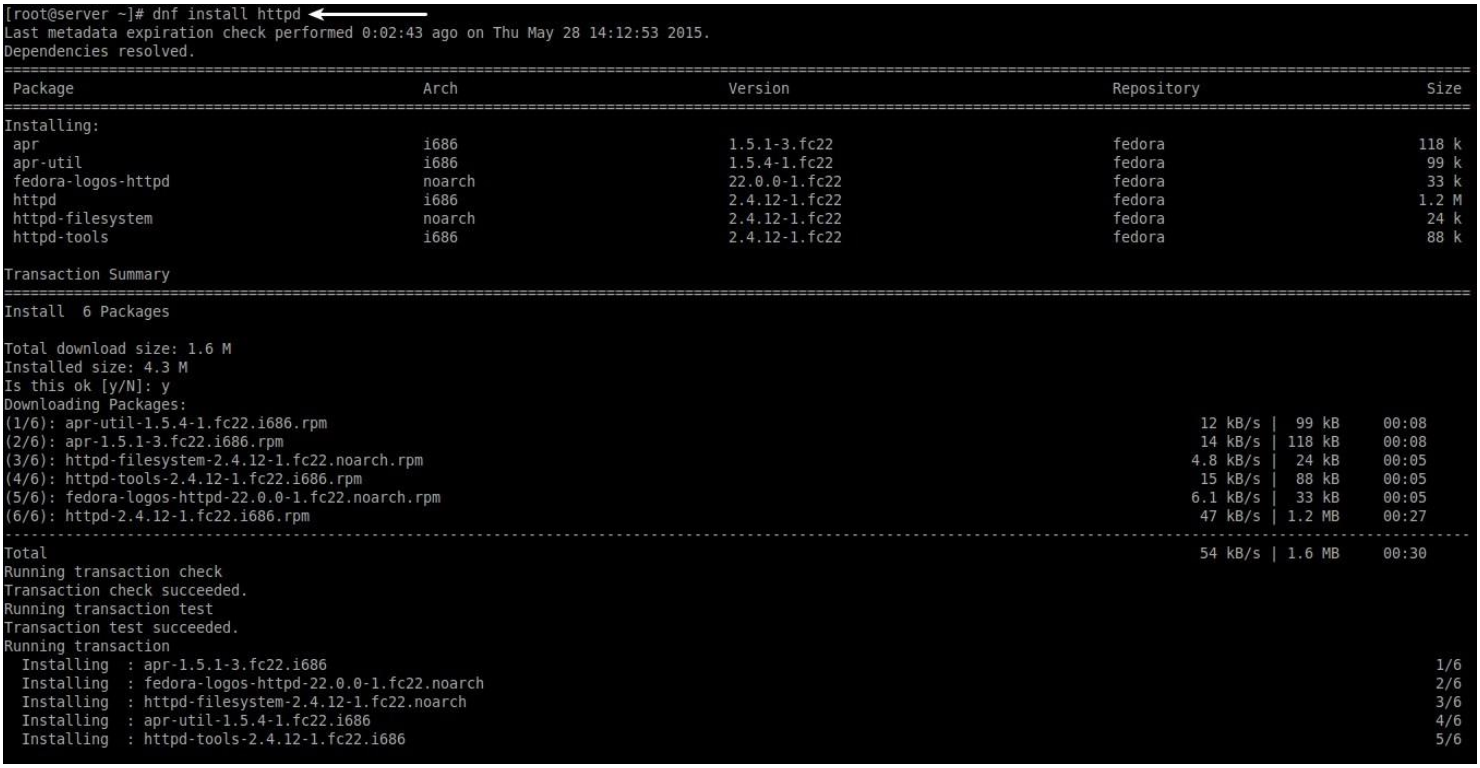
The **main** section defines all global configuration options. There should be only one main section.

The **repository** sections define the configuration for each (remote or local) repository.

For more details about DNF Configuration, please refer here.

DNF in action:

Let me install a package (ex.httpd) using DNF to view how it looks in action.



As you see in the above screenshot, nothing is changed, we just replaced the word “yum” with “dnf”. The remaining part of the command is exactly same as the way we use in YUM. However, there is some small changes in DNF.

Changes in DNF CLI compared to Yum

- No `--skip-broken` ;
- Update and Upgrade Commands are the Same ;
- `clean_requirements_on_remove` on by default ;
- No `resolvedep` command ;
- No `deplist` command ;
- Excludes and repo excludes apply to all operations ;

- Yum's conf directive `includepkgs` is just `include ;`
- `protected_packages` is supported via plugin ;
- `dnf remove kernel` deletes all packages called `kernel ;`
- `dnf provides /bin/<file>` does not find any packages on Fedora ;
- `skip_if_unavailable` enabled by default ;
- `overwrite_groups` dropped, comps functions acting as if always disabled ;
- `mirrorlist_expire` dropped ;
- metalink not recognized in the `mirrorlist` repo option ;
- `group_package_types` dropped ;
- `upgrade_requirements_on_install` dropped ;
- `dnf history rollback` check dropped ;
- Packages replacement without `yum shell` or `yum swap ;`
- `dnf history info last ;`
- Dependency processing details are not shown in the CLI ;
- `dnf provides` complies with the Yum documentation of the command ;
- `--enableplugin` not recognized ;
- Bandwidth limiting ;
- The usage of Delta RPM files ;
- Handling .srpm files and non-existent packages ;
- Promoting package to install to a package that obsoletes it.

Among popular package managers such as **APT-GET**, **YUM** and **ZYPPER**, DNF is new to the show. Let us wait and see how DNF will perform over YUM and other package managers in near future. You, however, can also use both YUM and DNF package managers in your Fedora system. But, be mindful that DNF and Yum keep additional data about each installed package and every performed transaction. This data is currently not shared between the two managers so if the admin installs half of the packages with DNF and the other half with Yum then each program can not benefit from the information held by the other one. The practical bottom line is that commands like `autoremove` can not take a completely informed decision and thus have to "play it safe" and remove only a subset of dependencies they would be able to otherwise. Similar situation exists with groups.