Offline Installation of an RPM Package and Its Dependencies

Last updated: March 31, 2023

Written by: baeldung (https://www.baeldung.com/linux/author/baeldung)

Installation (https://www.baeldung.com/linux/category/installation)

dnf (https://www.baeldung.com/linux/tag/dnf)

sudo (https://www.baeldung.com/linux/tag/sudo)

tar (https://www.baeldung.com/linux/tag/tar)

1. Overview

Red Hat-based Linux distributions use RPM files to provide programs and libraries. The installation of the application and its dependencies is facilitated by the package manager, usually *dnf*. The whole thing relies heavily on online repositories.

However, sometimes we're in a situation where our computer has no internet access. In this tutorial, we'll learn how to install a package and its dependencies offline.

2. The *repotrack* Command

With *repotrack* (https://man7.org/linux/man-pages/man1/repotrack.1.html) we can download all dependencies required by the program. It comes in the *dnf-utils* package. So, let's log in to the online computer and get dependencies of the *mc* (https://linux.die.net/man/1/mc) file browser:

```
$ repotrack --destdir /home/joe/prj/rpm mc
```

We've used the *—destdir* option to pass the folder for the RPM files.

For the next step, we should transfer the whole content of this directory to the offline computer. Afterwards, let's install the program and all its dependencies issuing *dnf (https://man7.org/linux/man-pages/man8/dnf.8.html) install* in the RPM's directory:

```
$ sudo dnf install *.x86_64.rpm --disablerepo=*
```

Notably, the *disablerepo=** switches off all active repositories. We must use it to curb *dnf* attempts to refresh repository data when offline. Also, we install only packages that match the x86_64 architecture of the target computer.

3. Installing the Application From Its Own Repository

As an alternative approach, let's wrap the dependencies for our program into a regular repository. Subsequently, we'll transfer the repository to the offline computer and install our program from it.

The important point is to download all dependencies. So, we'll prevent *dnf* from finding possible dependencies installed on the online computer.

In addition to *dnf*, we need the *createrepo* (https://linux.die.net/man/8/createrepo) command. We can install it from the package of the same name.

3.1. The Online Machine – Create the Repository

With *dnf*, we're going to use the *—installroot* option. **It points to an empty directory and causes *dnf* to suppose that no package is installed.** Thus, let's create this temporary directory:

```
$ mkdir /var/tmp/mc-installroot
```

The next directory we need is for packages:

```
$ mkdir /var/tmp/mc
```

Now, we're ready to issue the *dnf install* command:

```
$ sudo dnf install --downloadonly --installroot=/var/tmp/mc-installroot --releasever=37 --downloaddir=/var/tmp/mc mc
```

Let's look through the options. First, with the *downloadonly* option, *dnf* doesn't install or update any package. Then, *installroot* points to our directory. Next comes *release*ver with the target Fedora version. So, we can limit the number of downloaded packages. Finally, with *downloaddir,* we indicate the directory to store downloaded files.

In the next step, we're going to create a repository from the acquired files with *createrepo*:

```
$ createrepo /var/tmp/mc
```

As the *installroot* directory is no longer required, let's remove it:

```
$ sudo rm -rf /var/tmp/mc-installroot
```

Finally, let's compress our work and make it ready to be distributed:

```
$ sudo tar -czvf mc_repo.tar.gz /var/tmp/mc
```

3.2. The Offline Machine – Add the Repository

Next, we want to make *dnf* use the offline repo (/linux/rhel-setup-package-repo). To do so, let's add the repo configuration *mc-offline.repo* in the */etc/yum.repos.d* directory:

```
$ sudo nano /etc/yum.repos.d/mc-offline.repo
```

Here are the contents of this file:

```
[mc-offline]
name=fedora-37 - mc
baseurl=file:///var/tmp/mc
enabled=0
```

Now, let's take advantage of the automatic package installation done by *dnf.* Thus, we need to point to our repo with the *—enablerepo* option:

```
$ sudo dnf --disablerepo=* --enablerepo=mc-offline install mc
```

Let's check the command's output:

```
fedora-37 - mc                                   25 MB/s | 274
kB     00:00
Dependencies resolved.
================================================================
=================
 Package         Architecture  Version
Repository         Size
================================================================
=================
Installing:
 mc              x86_64        1:4.8.28-3.fc37          mc-
offline        1.9 M
Installing dependencies:
 gpm-libs        x86_64        1.20.7-41.fc37           mc-
offline         20 k
 slang           x86_64        2.3.3-1.fc37             mc-
offline        423 k

Transaction Summary
================================================================
=================
Install  3 Packages

# ...

Installed:
  gpm-libs-1.20.7-41.fc37.x86_64            mc-1:4.8.28-
3.fc37.x86_64
  slang-2.3.3-1.fc37.x86_64

Complete!
```

So, finally, we can enjoy *mc*!

4. Conclusion

In this article, we learned two ways of installing RPM packages and their dependencies offline.

First, we used the *repotrack* and *dnf* commands to deal directly with the RPM files. As a second approach, we followed the standard way of package installation. So, we created an offline repository for the desired

application on the online machine. Then, on the offline computer, we just installed the application from this repository.

Comments are closed on this article!