Quick answers to common problems

Red Hat Enterprise Linux Server Cookbook

Over 60 recipes to help you build, configure, and orchestrate RHEL 7 Server to make your everyday administration experience seamless

William Leemans

[PACKT] PUBLISHING open source* community experience distilled

Red Hat Enterprise Linux Server Cookbook

Over 60 recipes to help you build, configure, and orchestrate RHEL 7 Server to make your everyday administration experience seamless

William Leemans

[PACKT] PUBLISHING open source community experience distilled

BIRMINGHAM - MUMBAI

Red Hat Enterprise Linux Server Cookbook

First published: December 2015

Production reference: 1151215

Published by Packt Publishing Ltd.
Livery Place
35 Livery Street
Birmingham B3 2PB, UK.

ISBN 978-1-78439-201-7

www.packtpub.com

Credits

Author
William Leemans

Reviewers
Kyung Huh
Marcus Young

Commissioning Editor
Kunal Parikh

Acquisition Editor
Reshma Raman

Content Development Editor
Arshiya Ayaz Umer

Technical Editor
Siddhesh Ghadi

Copy Editor
Shruti Iyer

Project Coordinator
Shipra Chawhan

Proofreader
Safis Editing

Indexer
Monica Ajmera Mehta

Graphics
Disha Haria

Production Coordinator
Conidon Miranda

Cover Work
Conidon Miranda

About the Author

William Leemans has over 20 years of experience in the IT industry in various positions and supporting several environments.

In 2005, he started his own consulting company, Critter BVBA, in the hope of offering open source solutions to his customers, who are mainly enterprises.

In 2010, William started supporting Red Hat products full time with the Federal Police, Belgium. Since then, he has moved on to support Red Hat products at Proximus and now Euroclear.

William is a strong open source supporter and contributes where he can. He has a couple of projects running at GitHub (`https://github.com/bushvin`). During the course of writing this book, William recertified himself as a Red Hat Certified Engineer, hoping to one day become a Red Hat Certified Architect.

When he's not tapping away at the keyboard of his laptop, William likes to play around with his two young children, listen to rock music (Foo Fighters, AC/DC, and Queens of the Stone Age are some of his favorites), and devising complicated and intricate plots for the stories that he runs at his biweekly roleplaying sessions with his friends.

Thank you, Caroline, my dear wife, for being my soul mate, supporting me during this lengthy process, and giving me the space, time, and motivation to see this to the end.

Thanks, Mom, Dad, and Fre, for your relentless and unwavering belief in me and support, regardless of my rebellious antics.

Thank you, Tim, for telling me that I should get into computers when you did. I wouldn't know where I would've ended up if you hadn't!

Thank you, Gilad, for being the friend I need in my darkest hours and my most joyous moments!

Thank you, Wednesday Guys, for the support and fun. Koeken troef!

Thank you, Dag, for taking me on to yet another rather interesting adventure in my life called Red Hat.

About the Reviewers

Kyung Huh is a senior consultant at Red Hat based in Korea. He is a Red Hat Certified Architect. Kyung has worked with Linux and open source software for more than 16 years as an instructor and consultant. He has a lot of experience in Linux environments and building virtualization and cloud infrastructures, such as Red Hat Enterprise Virtualization and Red Hat OpenStack Platform, on the field. Kyung also reviewed *Hybrid Cloud Management with Red Hat CloudForms* and *Getting Started with Red Hat Enterprise Virtualization* by Packt Publishing.

Marcus Young recently graduated with a degree in computer science and mathematics before getting involved in system administration and DevOps. He currently works in software automation using open source tools and technologies. Marcus' hobbies include playing ice hockey and making home-brewed beer. He also enjoys hardware projects based on microcontrollers and single-board computers.

Marcus authored *Implementing Cloud Design Patterns for AWS, Packt Publishing*, as well.

www.PacktPub.com

Support files, eBooks, discount offers, and more

For support files and downloads related to your book, please visit `www.PacktPub.com`.

Did you know that Packt offers eBook versions of every book published, with PDF and ePub files available? You can upgrade to the eBook version at `www.PacktPub.com` and as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at `service@packtpub.com` for more details.

At `www.PacktPub.com`, you can also read a collection of free technical articles, sign up for a range of free newsletters and receive exclusive discounts and offers on Packt books and eBooks.

`https://www2.packtpub.com/books/subscription/packtlib`

Do you need instant solutions to your IT questions? PacktLib is Packt's online digital book library. Here, you can search, access, and read Packt's entire library of books.

Why Subscribe?

- Fully searchable across every book published by Packt
- Copy and paste, print, and bookmark content
- On demand and accessible via a web browser

Free Access for Packt account holders

If you have an account with Packt at `www.PacktPub.com`, you can use this to access PacktLib today and view 9 entirely free books. Simply use your login credentials for immediate access.

Table of Contents

Preface

Gnu/Linux is the most important OS in the data center but how do you leverage it? How do you maintain and contain it? Many Gnu/Linux distributions try to answer these questions, but not all succeed. Red Hat Enterprise Linux is one that does answer these questions.

The next question is how do you, as a system administrator, manage a RHEL infrastructure? How do you deploy not just one system, but many? How do you make sure that it is secure and up to date? How can you monitor system components?

It may seem odd to you, but as a Red Hat Certified Engineer, I prefer the "lazy" approach—not as in "I can't be bothered," but as in "I like to do something once and do it good the first time and spend the rest of my time doing fun stuff."

In this book, I try to show you how to set up and configure systems, mainly by providing useful information to automate the setup, configuration, and management. This also explains the lack of the use of a GUI in this book. I'll be honest with you; I couldn't live without one on my laptop or desktop, but I do not believe servers should have a GUI. GUI-based applications tend not to have command-line counterparts, and I solemnly believe that if you cannot install, configure, manage, and maintain a piece of software through a script, it does not belong on a server.

This book does not pretend to be the de facto answer to all questions (that would be 42), but I do hope that you will learn something new and that, in turn, you will put this knowledge to good use. Remember, with great power, comes great responsibility!

What this book covers

Chapter 1, *Working with KVM Guests*, will not start by installing a basic RHEL system. It will start by introducing you to KVM if you don't already know it. You'll learn how to install and configure the KVM host and manage your KVM guests (the VMs). It will discuss the basics of adding resources on the fly, moving disks, and even moving the entire guest to another KVM host.

Chapter 2, *Deploying RHEL "En Masse"*, will explore the ways of installing a RHEL system, introducing you to kickstart deployments, which are used to streamline automated system installs. If you want to orchestrate your environment, this chapter will lay out the basics for you to build on.

Chapter 3, *Configuring Your Network*, will explore `NetworkManager` tools to manage your network configuration, including advanced topics such as VLANs, link aggregation, and bridges. It will show you how to leverage its command-line tools to automate your system's network configuration during its deployment or afterwards, when all is installed.

Chapter 4, *Configuring Your New System*, will explain how to configure the basics, such as log retention, time, and your boot environment. It will also introduce you to the new systemd, which is SysVinit's replacement, and to monitoring and managing your services.

Chapter 5, *Using SELinux*, will give you an overview, but a brief one, on how to manage and troubleshoot SELinux on your system. SELinux is becoming more and more important in today's world because of its security implementation, and it's better to know about it than to just turn it off because you can't handle it.

Chapter 6, *Orchestrating with Ansible*, will tell you all about Ansible, which was recently bought by Red Hat. It will show you how to create simple playbooks that easily deploy new systems and how to manage your system's configuration.

Chapter 7, *Puppet Configuration Management*, will show you how to set up and configure Puppet. It will also give you a peek at its configuration management capacities.

Chapter 8, *Yum and Repositories*, will take a look at yum repositories, how you can create your own mirrors of the existing (Red Hat) repositories, and how to leverage it to keep your RHEL environment up to date without breaking a sweat.

Chapter 9, *Securing RHEL 7*, will take security configuration and auditing problems a bit further. We'll explore how to configure setting up centralized secure authentication and privilege escalation. It will show you how you can operate a system that appears to be "hung" and trace the root cause of the event.

Chapter 10, *Monitoring and Performance Tuning*, will show you the basics of easy performance tuning and how to monitor your system's resources.

What you need for this book

The only thing you'll need for the recipes in this book is the Red Hat Enterprise Linux 7 Installation DVD, for which you can download an evaluation license from `https://access.redhat.com/downloads`. All software used in this book is either available through the RHEL media or the yum repositories specified in the recipes.

Who this book is for

This book is for the system administrators who want to learn about the new RHEL version and features that are included for management or certification purposes. Although this book provides a lot of information to get your Red Hat Certified System Administrator and/or Red Hat Certified Engineer certifications, it is by far a complete guide to get either!

To get the most of this book, you should have a working knowledge of the basic (RHEL) system administration and management tools.

Sections

In this book, you will find several headings that appear frequently (Getting ready, How to do it, How it works, There's more, and See also).

To give clear instructions on how to complete a recipe, we use these sections as follows:

Getting ready

This section tells you what to expect in the recipe, and describes how to set up any software or any preliminary settings required for the recipe.

How to do it...

This section contains the steps required to follow the recipe.

How it works...

This section usually consists of a detailed explanation of what happened in the previous section.

There's more...

This section consists of additional information about the recipe in order to make the reader more knowledgeable about the recipe.

See also

This section provides helpful links to other useful information for the recipe.

Conventions

In this book, you will find a number of text styles that distinguish between different kinds of information. Here are some examples of these styles and an explanation of their meaning.

Code words in text, database table names, folder names, filenames, file extensions, pathnames, dummy URLs, user input, and Twitter handles are shown as follows: "We can include other contexts through the use of the `include` directive."

A block of code is set as follows:

```
node /^www[0-9]+\.critter\.be$/ {
}
node /^repo[0-9]+\.critter\.be$/ {
}
```

Any command-line input or output is written as follows:

```
~]# yum install -y /tmp/puppetlabs-release-el-7.noarch.rpm
```

New terms and **important words** are shown in bold. Words that you see on the screen, for example, in menus or dialog boxes, appear in the text like this: "Clicking the **Next** button moves you to the next screen."

Warnings or important notes appear in a box like this.

Tips and tricks appear like this.

Reader feedback

Feedback from our readers is always welcome. Let us know what you think about this book—what you liked or disliked. Reader feedback is important for us as it helps us develop titles that you will really get the most out of.

To send us general feedback, simply e-mail `feedback@packtpub.com`, and mention the book's title in the subject of your message.

If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, see our author guide at `www.packtpub.com/authors`.

Customer support

Now that you are the proud owner of a Packt book, we have a number of things to help you to get the most from your purchase.

Downloading the color images of this book

We also provide you with a PDF file that has color images of the screenshots/diagrams used in this book. The color images will help you better understand the changes in the output. You can download this file from `https://www.packtpub.com/sites/default/files/downloads/RedHatEnterpriseLinuxServerCookbook_ColorImages.pdf`.

Errata

Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you find a mistake in one of our books—maybe a mistake in the text or the code—we would be grateful if you could report this to us. By doing so, you can save other readers from frustration and help us improve subsequent versions of this book. If you find any errata, please report them by visiting `http://www.packtpub.com/submit-errata`, selecting your book, clicking on the **Errata Submission Form** link, and entering the details of your errata. Once your errata are verified, your submission will be accepted and the errata will be uploaded to our website or added to any list of existing errata under the Errata section of that title.

To view the previously submitted errata, go to `https://www.packtpub.com/books/content/support` and enter the name of the book in the search field. The required information will appear under the **Errata** section.

Piracy

Piracy of copyrighted material on the Internet is an ongoing problem across all media. At Packt, we take the protection of our copyright and licenses very seriously. If you come across any illegal copies of our works in any form on the Internet, please provide us with the location address or website name immediately so that we can pursue a remedy.

Please contact us at `copyright@packtpub.com` with a link to the suspected pirated material.

We appreciate your help in protecting our authors and our ability to bring you valuable content.

Questions

If you have a problem with any aspect of this book, you can contact us at `questions@packtpub.com`, and we will do our best to address the problem.

1
Working with KVM Guests

In this chapter, we will cover the following recipes:

- Installing and configuring a KVM
- Configuring resources
- Building VMs
- Adding CPUs on the fly
- Adding RAM on the fly
- Adding disks on the fly
- Moving disks to another storage
- Moving VMs
- Backing up your VM metadata

Introduction

This book will attempt to show you how to deploy RHEL 7 systems without too much of a hassle. As this book is written with automation in mind, I will emphasize on command-line utilities rather than elaborating on its GUI counterparts, which are useless for automation.

This chapter explains how to build and manage KVM guests using the libvirt interface and various tools built around it. It will provide a brief overview on how to set up a KVM on RHEL and manage its resources. The setup provided in this overview is far from the ready enterprise as it doesn't provide any redundancy, which is generally required in enterprises. However, the recipes provided are relevant in enterprise setups as the interface stays the same. Most of the time, you will probably use a management layer (such as RHEV or oVirt), which will make your life easier in managing redundancy.

Libvirt is the API between the user and the various virtualization and container layers that are available, such as KVM, VMware, Hyper-V, and Linux Containers. Check `https://libvirt.org/drivers.html` for a complete list of supported hypervisors and container solutions.

As most tasks performed need to be automated in the end, I tend not to use any graphical interfaces as these do not allow an easy conversion into script. Hence, you will not find any recipes in this chapter involving a graphical interface. These recipes will primarily focus on `virsh`, the libvirt management user interface that is used to manage various aspects of your KVM host and guests. While a lot of people rely on the edit option of `virsh`, it doesn't allow you to edit a guest's configuration in real time. Editing your guest's XML configuration in this way will require you to shut down and boot your guest for the changes to take effect. A reboot of your guest doesn't do the trick as the XML configuration needs to be completely reread by the guest's instance in order for it to apply the changes. Only a fresh boot of the guest will do this.

The `virsh` interface is also a shell, so by launching `virsh` without any commands, you will enter the libvirt management shell. A very interesting command is `help`. This will output all the available commands grouped by keyword. Each command accepts the `--help` argument to show a detailed list of the possible arguments, and their explanation, which you can use.

Installing and configuring a KVM

This recipe covers the installing of virtualization tools and packages on RHEL 7.

By default, a RHEL 7 system doesn't come with a KVM or libvirt preinstalled. This can be installed in three ways:

- Through the graphical setup during the system's setup
- Via a kickstart installation
- Through a manual installation from the command line

For this recipe, you should know how to install packages using yum, and your system should be configured to have access to the default RHEL 7 repository (refer to *Chapter 8*, *Yum and Repositories*, for more information), which is required for the packages that we will use.

Alternatively, you could install packages from the installation media using `rpm`, but you'll need to figure out the dependencies yourself.

Check the dependencies of an `rpm` using the following command:

```
~]# rpm -qpR <rpm file>
```

This will output a list of binaries, libraries, and files that you need installed prior to installing this package.

Check which package contains these files through this command:

```
~]# rpm -qlp <rpm package>
```

As you can imagine, this is a tedious job and can take quite some time as you need to figure out every dependency for every package that you want to install in this way.

Getting ready

To install a KVM, you will require at least 6 GB of free disk space, 2 GB of RAM, and an additional core or thread per guest.

Check whether your CPU supports a virtualization flag (such as SVM or VMX). Some hardware vendors disable this in the BIOS, so you may want to check your BIOS as well. Run the following command:

```
~]# grep -E 'svm|vmx' /proc/cpuinfo
flags    : ... vmx ...
```

Alternatively, you can run the following command:

```
~]# grep -E 'svm|vmx' /proc/cpuinfo
flags    : ... svm ...
```

Check whether the hardware virtualization modules (such as `kvm_intel` and `kvm`) are loaded in the kernel using the following command:

```
~]# lsmod | grep kvm
kvm_intel              155648  0
kvm                    495616  1 kvm_intel
```

How to do it...

We'll look at the three ways of installing a KVM onto your system.

Manual installation

This way of installing a KVM is generally done once the base system is installed by some other means. You need to perform the following steps:

1. Install the software needed to provide an environment to host virtualized guests with the following command:

   ```
   ~]# yum -y install qemu-kvm qemu-img libvirt
   ```

   The installation of these packages will include quite a lot of dependencies.

2. Install additional utilities required to configure `libvirt` and install virtual machines by running this command:

   ```
   ~]# yum -y install virt-install libvirt-python python-virthost
   libvirt-client
   ```

3. By default, the `libvirt` daemon is marked to `autostart` on each boot. Check whether it is enabled by executing the following command:

   ```
   ~]# systemctl status libvirtd
   libvirtd.service - Virtualization daemon
      Loaded: loaded (/usr/lib/systemd/system/libvirtd.service;
   enabled)
      Active: inactive
        Docs: man:libvirtd(8)
              http://libvirt.org
   ```

4. If for some reason this is not the case, mark it for autostart by executing the following:

   ```
   ~]# systemctl enable libvirtd
   ```

5. To manually stop/start/restart the `libvirt` daemon, this is what you'll need to execute:

   ```
   ~]# systemctl stop libvirtd
   ~]# systemctl start libvirtd
   ~]# systemctl restart libvirtd
   ```

Kickstart installation

Installing a KVM during kickstart offers you an easy way to automate the installation of KVM instances. Perform the following steps:

1. Add the following package groups to your kickstarted file in the `%packages` section:

```
@virtualization-hypervisor
@virtualization-client
@virtualization-platform
@virtualization-tools
```

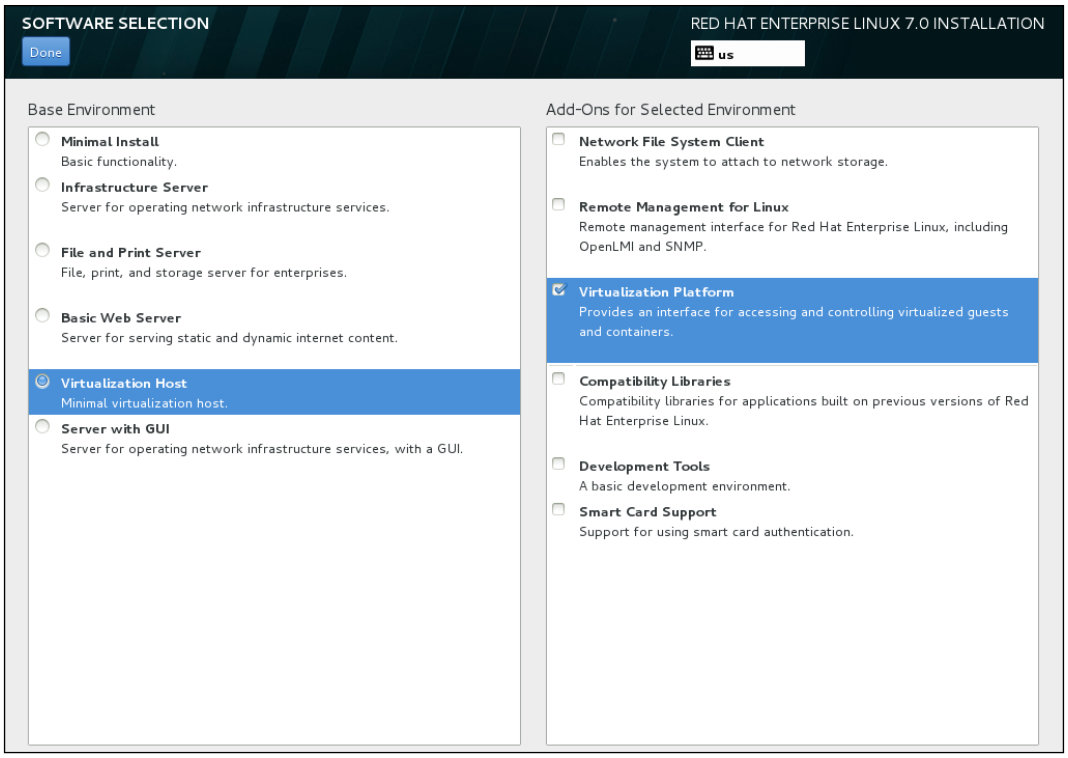
2. Start the installation of your host with this kickstart file.

Graphical setup during the system's setup

This is probably the least common way of installing a KVM. The only time I used this was during the course of writing this recipe. Here's how you can do this:

1. Boot from the RHEL 7 Installation media.
2. Complete all steps besides the **Software selection** step.

3. Go to **Software Selection** to complete the KVM software selection.
4. Select the **Virtualization host** radio button in **Base Environment**, and check the **Virtualization Platform** checkbox in **Add-Ons for Selected Environment**:



5. Finalize the installation.
6. On the **Installation Summary** screen, complete any other steps and click on **Begin Installation**.

See also

To set up your repositories, check out *Chapter 8*, *Yum and Repositories*.

To deploy a system using kickstart, refer to *Chapter 2*, *Deploying RHEL "En Masse"*.

For more in-depth information about using libvirt, go to `http://www.libvirt.org/`.

RHEL 7 has certain support limits, which are listed at these locations:

`https://access.redhat.com/articles/rhel-kvm-limits`

`https://access.redhat.com/articles/rhel-limits`

Configuring resources

Virtual machines require CPUs, memory, storage, and network access, similar to physical machines. This recipe will show you how to set up a basic KVM environment for easy resource management through libvirt.

A storage pool is a virtual container limited by two factors:

- The maximum size allowed by `qemu-kvm`
- The size of the disk on the physical machine

Storage pools may not exceed the size of the disk on the host. The maximum sizes are as follows:

- virtio-blk = 2^63 bytes or 8 exabytes (raw files or disk)
- EXT4 = ~ 16 TB (using 4 KB block size)
- XFS = ~8 exabytes

Getting ready

For this recipe, you will need a volume of at least 2 GB mounted on `/vm` and access to an NFS server and export.

We'll use `NetworkManager` to create a bridge, so ensure that you don't disable `NetworkManager` and have `bridge-utils` installed.

How to do it...

Let's have a look into managing storage pools and networks.

Creating storage pools

In order to create storage pools, we need to provide the necessary details to the KVM for it to be able to create it. You can do this as follows:

1. Create a `localfs` storage pool using `virsh` on `/vm`, as follows:

   ```
   ~]# virsh pool-define-as --name localfs-vm --type dir --target /vm
   ```

2. Create the target for the storage pool through the following command:

   ```
   ~# mkdir -p /nfs/vm
   ```

3. Create an NFS storage pool using `virsh` on NFS server:`/export/vm`, as follows:

```
~]# virsh pool-define-as --name nfs-vm --type network --source-
host nfsserver --source-path /export/vm –target /nfs/vm
```

4. Make the storage pools persistent across reboots through the following commands:

```
~]# virsh pool-autostart localfs-vm
~]# virsh pool-autostart nfs-vm
```

5. Start the storage pool, as follows:

```
~]# virsh pool-start localfs-vm
~]# virsh pool-start nfs-vm
```

6. Verify that the storage pools are created, started, and persistent across reboots. Run the following for this:

```
~]# virsh pool-list
 Name                 State      Autostart
-----------------------------------------
 localfs-vm           active     yes
 nfs-vm               active     yes
```

Querying storage pools

At some point in time, you will need to know how much space you have left in your storage pool.

Get the information of the storage pool by executing the following:

```
~]# virsh pool-info --pool <pool name>
Name:           nfs-vm
UUID:           some UUID
State:          running
Persistent:     yes
Autostart:      yes
Capacity:       499.99 GiB
Allocation:     307.33 GiB
Available:      192.66 GiB
```

As you can see, this command easily shows you its disk space allocation and availability.