

How to Install Kubernetes on Rocky Linux

April 27, 2023 KUBERNETES ROCKYOS

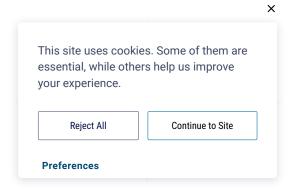
KB » DevOps and Development » How to Install Kubernetes on Rocky Linux

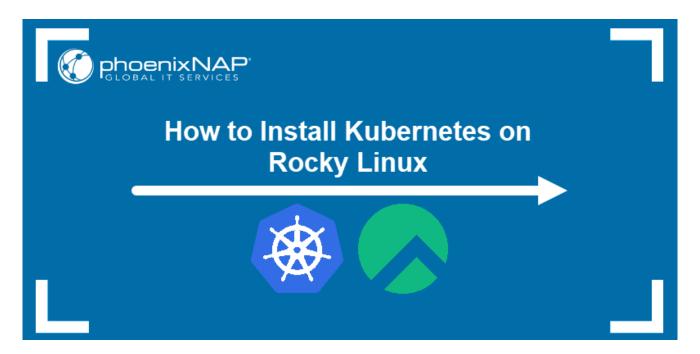
Introduction

Rocky Linux is one of the new **distributions** that emerged as an alternative to CentOS after **CentOS**'s **discontinuation** in 2021. As a free and open-source project, Rocky Linux aims to provide a viable replacement for enterprise operating systems in application development.

The server-centric and performance-oriented nature of Rocky Linux makes it a good choice for running containerized workloads. However, managing app containers at scale requires a **container orchestrator** like **Kubernetes**.

This article will guide you through installing Kubernetes on Rocky Linux.





Prerequisites

- Two or more machines running Rocky Linux (Bare Metal Cloud offers server instances that are deployed automatically with Rocky Linux).
- 2 GB of RAM and 2 CPU cores or more on each machine.
- Sudo or root access for each system.
- Ansible installed (for the 2nd method).

Install Kubernetes on Rocky Linux (Manual Method)

Manual installation of Kubernetes on Rocky Linux involves:

- Setting up a container runtime interface (CRI).
- · Making adjustments to security and networking configuration.
- Installing the essential Kubernetes tools.



Note: Execute the installation steps on **each node** (physical or virtual machine) you plan to add to the cluster.

Step 1: Install containerd

containerd is a **Docker**-made CRI tool that creates, executes, and supervises containers. Follow the procedure below to set it up on your Rocky Linux system.

1. Add the official Docker repository to your system. Docker does not maintain a separate repository for Rocky Linux, but the CentOS repo is fully compatible.

```
sudo dnf config-manager --add-repo https://download.docker.com/linux/centos/d
ocker-ce.repo
```

The output confirms the success of the operation.

```
[marko@localhost ~]$ sudo dnf config-manager --add-repo https://download.docker.com/lin
ux/centos/docker-ce.repo
[sudo] password for marko:
Adding repo from: https://download.docker.com/linux/centos/docker-ce.repo
[marko@localhost ~]$
```

2. Refresh the local repository information.

```
sudo dnf makecache
[marko@localhost ~]$ sudo dnf makecache
Docker CE Stable - x86_64
                                                        68 kB/s
                                                                   22 kB
                                                                             00:00
Extra Packages for Enterprise Linux 9 - x86 64
                                                        85 kB/s
                                                                   21 kB
                                                                             00:00
Rocky Linux 9 - BaseOS
                                                       8.5 kB/s
                                                                  4.1 kB
                                                                             00:00
                                                        12 kB/s | 4.5 kB
                                                                             00:00
Rocky Linux 9 - AppStream
Rocky Linux 9 - Extras
                                                       5.6 kB/s | 2.9 kB
                                                                             00:00
Metadata cache created.
[marko@localhost ~]$
```

3. Install the **containerd.io** package.

```
dnf install -y containerd.io
Running transaction
                                                                                 1/1
 Preparing
                 : containerd.io-1.6.20-3.1.el9.x86_64
                                                                                 1/2
 Installing
 Running scriptlet: containerd.io-1.6.20-3.1.el9.x86 64
                                                                                 1/2
 Obsoleting : runc-4:1.1.4-1.el9 1.x86 64
                                                                                 2/2
 Running scriptlet: runc-4:1.1.4-1.el9 1.x86 64
                                                                                 2/2
 Verifying
                 : containerd.io-1.6.20-3.1.el9.x86 64
                                                                                 1/2
 Verifying
                  : runc-4:1.1.4-1.el9 1.x86 64
                                                                                 2/2
Installed:
 containerd.io-1.6.20-3.1.el9.x86 64
Complete!
[marko@localhost ~]$
```

4. Back up the default configuration file for containerd:

```
sudo mv /etc/containerd/config.toml /etc/containerd/config.toml.bak
```

5. Create a new file with the default template:

```
containerd config default > config.toml
6. Open the file in a text editor. This tutorial uses nano.
   sudo nano config.toml
7. Find the SystemdCgroup field and change its value to true.
   SystemdCgroup = true
   GNU nano 5.6.1
                                             config.toml
                                                                                    Modified
              NoNewKeyring = false
              NoPivotRoot = false
              Root = ""
              ShimCgroup = ""
             SystemdCgroup = true
       [plugins."io.containerd.grpc.vl.cri".containerd.untrusted_workload_runtime]
         base runtime spec = ""
Save the file and exit.
```

8. Place the new file in the /etc/containerd directory:

```
sudo mv config.toml /etc/containerd/config.toml
```

9. Enable the containerd service:

```
systemctl enable --now containerd.service
```

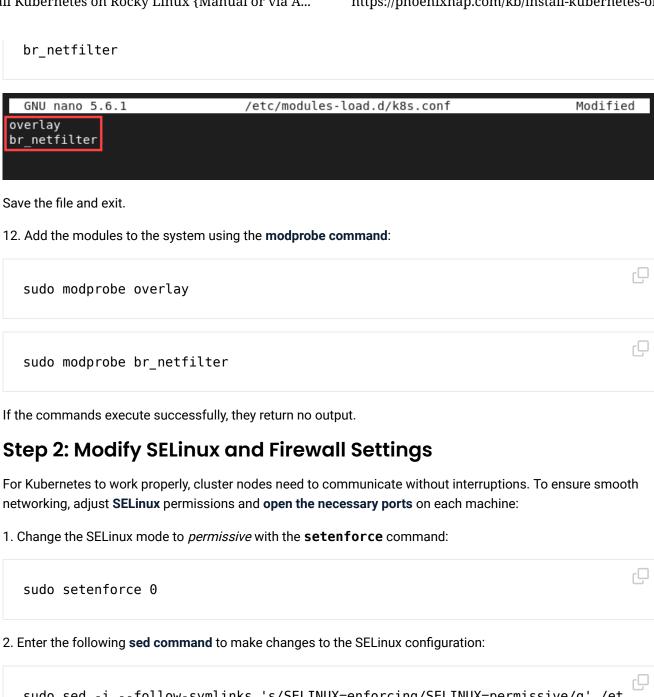
```
[marko@localhost ~]$ systemctl enable --now containerd.service
Created symlink /etc/systemd/system/multi-user.target.wants/containerd.service 	o /usr/l
ib/systemd/system/containerd.service.
[marko@localhost ~]$
```

10. Open the Kubernetes modules configuration file:

```
sudo nano /etc/modules-load.d/k8s.conf
```

11. Add the two modules required by the container runtime:

```
overlay
```



sudo sed -i --follow-symlinks 's/SELINUX=enforcing/SELINUX=permissive/g' /et
c/sysconfig/selinux

3. Confirm the changes by checking the SELinux status:

sestatus

The value of the **Current mode** field should be set to **permissive**.

```
[marko@localhost ~]$ sestatus
SELinux status:
                                enabled
SELinuxfs mount:
                                /sys/fs/selinux
SELinux root directory:
                                /etc/selinux
Loaded policy name:
                                targeted
Current mode:
                                permissive
Mode from config file:
                                permissive
Policy MLS status:
                                enabled
Policy deny unknown status:
                                allowed
Memory protection checking:
                                actual (secure)
Max kernel policy version:
                                33
[marko@localhost ~]$
```

4. Add **firewall** exceptions to allow Kubernetes to communicate via dedicated ports. On the **master node** machine, execute the following commands:

```
sudo firewall-cmd --permanent --add-port=6443/tcp
sudo firewall-cmd --permanent --add-port=2379-2380/tcp
sudo firewall-cmd --permanent --add-port=10250/tcp
sudo firewall-cmd --permanent --add-port=10251/tcp
sudo firewall-cmd --permanent --add-port=10259/tcp
sudo firewall-cmd --permanent --add-port=10257/tcp
sudo firewall-cmd --permanent --add-port=179/tcp
sudo firewall-cmd --permanent --add-port=4789/udp
```

The output confirms the success of the operation.

```
[marko@localhost ~]$ sudo firewall-cmd --permanent --add-port=6443/tcp
sudo firewall-cmd --permanent --add-port=2379-2380/tcp
sudo firewall-cmd --permanent --add-port=10250/tcp
sudo firewall-cmd --permanent --add-port=10251/tcp
sudo firewall-cmd --permanent --add-port=10259/tcp
sudo firewall-cmd --permanent --add-port=10257/tcp
sudo firewall-cmd --permanent --add-port=179/tcp
sudo firewall-cmd --permanent --add-port=4789/udp
success
success
success
success
success
success
success
success
[marko@localhost ~]$
```

5. On worker nodes, open the following ports:

```
sudo firewall-cmd --permanent --add-port=179/tcp
sudo firewall-cmd --permanent --add-port=10250/tcp
sudo firewall-cmd --permanent --add-port=30000-32767/tcp
sudo firewall-cmd --permanent --add-port=4789/udp
```

6. Reload the firewall configuration to enforce the changes.

Step 3: Configure Networking

Kubernetes requires filtering and **port forwarding** enabled for packets going through a network bridge. Perform the network configuration in the **k8s.conf** file:

1. Open the file in a text editor:

```
sudo nano /etc/sysctl.d/k8s.conf
```

2. Ensure the file contains the following lines:

```
net.ipv4.ip_forward = 1
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
```

```
GNU nano 5.6.1 /etc/sysctl.d/k8s.conf Modified

net.ipv4.ip_forward = 1
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
```

Save the file and exit.



Note: Read our tutorial to find out how can you save a file in Vim and exit.

3. Apply the changes with the sysctl command:

```
sudo sysctl --system
```

The system processes the k8s.conf file for changes.

```
[marko@localhost ~]$ sudo sysctl --system

* Applying /usr/lib/sysctl.d/10-default-yama-scope.conf ...

* Applying /usr/lib/sysctl.d/50-coredump.conf ...

* Applying /usr/lib/sysctl.d/50-libkcapi-optmem_max.conf ...

* Applying /usr/lib/sysctl.d/50-pid-max.conf ...

* Applying /usr/lib/sysctl.d/50-redhat.conf ...

* Applying /usr/lib/sysctl.d/50-redhat.conf ...

* Applying /etc/sysctl.d/99-sysctl.conf ...

* Applying /etc/sysctl.d/k8s.conf ...

* Applying /etc/sysctl.conf ...
```

Step 4: Disable Swap

For performance reasons and the maximum utilization of each node's resources, Kubernetes requires **virtual memory** to be disabled on each node.

1. Disable swap with the **swapoff** command.

```
sudo swapoff -a
```

2. Make the changes persist across reboots by typing:

```
sudo sed -e '/swap/s/^/#/g' -i /etc/fstab
```

Step 5: Install Kubernetes Tools

The following are the three main packages in a Kubernetes installation:

- kubeadm helps initialize a Kubernetes cluster.
- kubelet runs containers on each node.
- kubectl is the command-line utility for controlling the cluster and its components.

Install the packages by following the procedure explained below:

1. Create a repository file for Kubernetes:

```
sudo nano /etc/yum.repos.d/k8s.repo
```

2. Copy the repository specification below and paste it into the file.

```
[kubernetes]
name=Kubernetes
baseurl=https://packages.cloud.google.com/yum/repos/kubernetes-el7-x86_64
enabled=1
gpgcheck=1
repo_gpgcheck=1
gpgkey=https://packages.cloud.google.com/yum/doc/yum-key.gpg https://package
```

s.cloud.google.com/yum/doc/rpm-package-key.gpg

```
GNU nano 5.6.1 /etc/yum.repos.d/k8s.repo Modified
[kubernetes]
name=Kubernetes
baseurl=https://packages.cloud.google.com/yum/repos/kubernetes-el7-x86_64
enabled=1
gpgcheck=1
repo_gpgcheck=1
gpgkey=https://packages.cloud.google.com/yum/doc/yum-key.gpg https://packages.cloud.go>
```

Save the file and exit.

3. Refresh the local repository cache.

```
sudo dnf makecache
```

When prompted, type **Y** and press **Enter**.

```
[marko@localhost ~]$ sudo dnf makecache
Docker CE Stable - x86 64
                                                        23 kB/s | 3.5 kB
                                                                             00:00
Extra Packages for Enterprise Linux 9 - x86 64
                                                        35 kB/s |
                                                                  21 kB
                                                                             00:00
Kubernetes
                                                       323 B/s | 454 B
                                                                             00:01
                                                                             00:00
                                                       8.5 kB/s | 2.6 kB
Kubernetes
Importing GPG key 0x13EDEF05:
          : "Rapture Automatic Signing Key (cloud-rapture-signing-key-2022-03-07-08 0
Userid
Fingerprint: A362 B822 F6DE DC65 2817 EA46 B53D C80D 13ED EF05
           : https://packages.cloud.google.com/yum/doc/yum-key.gpg
Is this ok [y/N]:
```

4. Install the packages with the following command.

```
dnf install -y {kubelet,kubeadm,kubectl} --disableexcludes=kubernetes
```

[marko@localhost ~]\$ sudo dnf install -y {kubelet,kubeadm,kubectl}disableexcludes=kubernetes Last metadata expiration check: 0:01:09 ago on Tue 25 Apr 2023 03:02:54 PM CEST. Dependencies resolved.				
Package	Architectı	ıre Version	Repository	Size
Installing:				
kubeadm	x86 64	1.27.1-0	kubernetes	11 M
kubectl	x86 ⁻ 64	1.27.1-0	kubernetes	11 M
kubelet	x86_64	1.27.1-0	kubernetes	20 M
Installing dependencies:	_			
conntrack-tools	x86_64	1.4.5-17.el9_1	appstream	210 k
cri-tools	x86_64	1.26.0-0	kubernetes	8.6 M
kubernetes-cni	x86_64	1.2.0-0	kubernetes	17 M
libnetfilter_cthelper	x86_64	1.0.0-22.el9	appstream	23 k
libnetfilter_cttimeout	x86_64	1.0.0-19.el9	appstream	23 k
libnetfilter_queue	x86_64	1.0.5-1.el9	appstream	28 k
socat	x86_64	1.7.4.1-5.el9	appstream	300 k
Transaction Summary				
Install 10 Packages				

The system is now ready to deploy a Kubernetes cluster.

Install Kubernetes on Rocky Linux Using Ansible

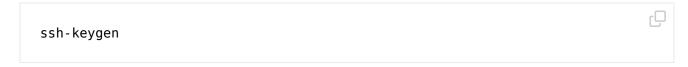
Ansible is an **IaC** tool that facilitates infrastructure deployment automation. It uses human-readable instruction files called **playbooks** to simplify and speed up repetitive deployments.

The following sections provide instructions for installing Kubernetes using Ansible.

Step 1: Connect Hosts

To enable communication between the Ansible host and the Kubernetes nodes, connect the machines via SSH.

1. Generate an SSH key:



When prompted, type the filename for the new key and press **Enter**. Next, press **Enter** two more times to create an empty passphrase.

2. Copy the credentials to each machine:

ssh-copy-id -i ~/.ssh/[ssh-key-name].pub root@[ip-address]

For example, to copy the **id_rsa** key to the machine with the **IP address 10.240.12.82**, type:

ssh-copy-id -i ~/.ssh/id_rsa.pub root@10.240.12.82

3. Create and go to the kube directory.

mkdir kube && cd kube

4. Create a file titled *hosts* using a text editor:

nano hosts

5. Paste the information about the nodes into the file. Split the info into two sections, *masters* and *workers*:

[masters]
master ansible_host=[ip-address] ansible_user=root

[workers]
worker1 ansible_host=[ip-address] ansible_user=root

Save the file and exit.

6. Test the connectivity between the nodes and the Ansible host by typing:

```
ansible -i hosts all -m ping
```

The output confirms that Ansible has pinged the machines successfully.

Step 2: Create Users

The first playbook that needs to be applied creates a user called *kube* on each machine. This user receives an authorized SSH key and permissions that allow it to run **sudo** commands without providing a password.

1. Create a playbook YML file in a text editor:

```
nano user-create.yml
```

2. Copy and paste the code below into the file.

```
    hosts: 'workers, masters' become: yes
    tasks:

            name: create a new user and name it kube user: name=kube append=yes state=present createhome=yes shell=/bin/bash
```

```
    name: allow the user to run sudo without requiring a password lineinfile:
        dest: /etc/sudoers
        line: 'kube ALL=(ALL) NOPASSWD: ALL'
        validate: 'visudo -cf %s'
    name: add authorized key for user
        authorized_key: user=kube key="{{item}}"
        with_file:
        - ~/.ssh/id_rsa.pub
```

Save the file and exit. The playbook now contains a set of tasks that Ansible will execute on the relevant connected machines.

3. Run the playbook by typing:

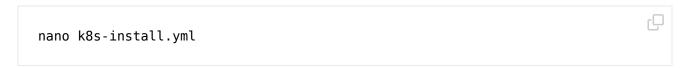
```
ansible-playbook -i hosts user-create.yml
```

The output shows the progress for each task.

Step 3: Install Kubernetes

After the necessary setup, create the playbook instructing Ansible to install Kubernetes tools on each node.

1. Create a YAML file in a text editor.



2. Copy and paste the following code into the file.

```
- hosts: "masters, workers"
  remote user: [current-user]
  become: yes
  become_method: sudo
  become user: root
  gather_facts: yes
  connection: ssh
  tasks:
     - name: create containerd configuration file
       file:
         path: "/etc/modules-load.d/containerd.conf"
         state: "touch"
     - name: set up containerd prerequisites
       blockinfile:
         path: "/etc/modules-load.d/containerd.conf"
         block: |
               overlay
               br_netfilter
     - name: load modules
       shell: |
               sudo modprobe overlay
               sudo modprobe br_netfilter
     - name: create network settings configuration file
       file:
         path: "/etc/sysctl.d/99-kubernetes-cri.conf"
         state: "touch"
     - name: set up containerd networking
       blockinfile:
         path: "/etc/sysctl.d/99-kubernetes-cri.conf"
         block: |
                net.bridge.bridge-nf-call-iptables = 1
                net.ipv4.ip\_forward = 1
                net.bridge.bridge-nf-call-ip6tables = 1
     - name: apply settings
       command: sudo sysctl --system
     - name: add docker repository
       shell: |
               sudo dnf config-manager --add-repo https://download.docker.co
m/linux/centos/docker-ce.repo
```

```
sudo dnf makecache
               sudo dnf install -y containerd.io
               sudo mkdir -p /etc/containerd
               sudo containerd config default | sudo tee /etc/containerd/conf
ig.toml
               sudo systemctl restart containerd
     - name: create k8s repo file
       file:
         path: "/etc/yum.repos.d/kubernetes.repo"
         state: "touch"
     - name: write repository information in the kube repo file
       blockinfile:
         path: "/etc/yum.repos.d/kubernetes.repo"
         block: |
                [kubernetes]
                name=Kubernetes
                baseurl=https://packages.cloud.google.com/yum/repos/kubernete
s-el7-x86_64
                enabled=1
                gpgcheck=1
                repo_gpgcheck=1
                gpgkey=https://packages.cloud.google.com/yum/doc/yum-key.gpg
https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg
     - name: install kubernetes
       shell: |
               sudo dnf install -y kubelet kubeadm kubectl
     - name: disable swap
       shell: |
               sudo swapoff -a
               sudo sed -i '/ swap / s/^{(.*)}$/#\1/g' /etc/fstab
```



Note: Do not forget to replace the **[current-user]** value in the **remote_user** field with the current username on your Ansible host.

Save the file and exit.

3. Execute the playbook by entering the following:

```
ansible-playbook -i hosts k8s-install.yml
```

When Ansible finishes all the operations, it displays a Play Recap.

Kubernetes has been successfully installed on all the nodes.

Conclusion

After completing this tutorial, you should know how to install Kubernetes on Rocky Linux and prepare for cluster deployment. The tutorial covered two methods for installation - manual and via Ansible-based.

If you are still looking for the best replacement for CentOS, read our comparison article **Rocky Linux vs. AlmaLinux**, to see how the two major competitors stack up against each other.

Was this article helpful? Yes No

Marko Aleksic

Marko Aleksić is a Technical Writer at phoenixNAP. His innate curiosity regarding all things IT, combined with over a decade long background in writing, teaching and working in IT-related fields, led him to technical writing, where he has an opportunity to employ his skills and make technology less daunting to everyone.

Next you should read

SysAdmin,
Virtualization
How to Install
Rocky Linux on
VMware
November 1, 2022

Virtual machine
software, such as
VMware, enables testdriving Rocky Linux. By
relying on hypervisors,
a host machine can
separate hardware
resources...

RE

A D

М

0

RΕ

DevOps and
Development,
SysAdmin,
Virtualization
How to Install
Docker on Rocky
Linux

November 2, 2022

This tutorial shows you how to install and perform the basic setup of Docker on Rocky Linux.

READ MORE

DevOps and
Development
When to Use
Kubernetes

March 23, 2023

Learning about the most common
Kubernetes use cases can help you assess whether it suits your needs. Read an overview of Kubernetes' advantages to help you decide...
READ MORE

Bare Metal Servers,
DevOps and
Development
Ansible Playbook
Dry Run

е m be 19 20 20 Α n si bl е p r 0 ٧i d е s а С h е С k m 0 d е in W hi С h у 0 u С а n te st а рl

N ov

а у b 0 0 k. Т hi s t ut 0 ri al s h 0 W s у 0 u h 0 W t 0 d 0 а d ry r u n 0 f а n Α n si bl е рl а

y b

O O k. .. R E A D M O R E

 Clive Chat
 ✓ Get a Quote
 ③ Support | 1-855-330-1509
 ¥ Sales | 1-877-588-5918

Privacy Center Do not sell or share my personal information

Contact Us

Legal

Privacy Policy

Terms of Use

DMCA

GDPR

Sitemap

©2024 Copyright phoenixNAP | Global IT Services. All Rights Reserved.