

Containers.

Debian.

Document how you installed Docker and Kubernetes. Make sure you've installed them on both servers and your documentation includes instructions for installs on both distros. Target Market is someone technically inclined but unfamiliar with virtualization. Include screenshots to walk them through the install process, as well as explanations of what you're doing and any troubleshooting that was needed.

Docker installation

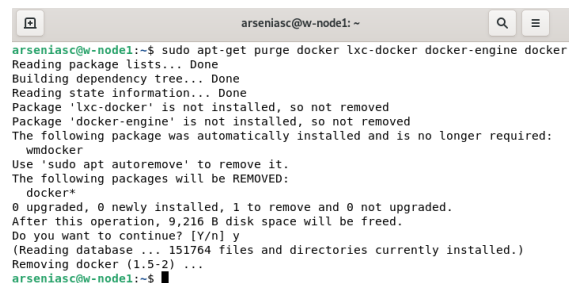
Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly. Docker helps to create containers. Today we are going to learn how to install docker this useful tool.

Instructions

After you successfully have logged in on your server, follow the next steps to install docker

1. Uninstall Default Docker Packages

-Type the following to uninstall the old docker package ,**sudo apt-get purge docker lxc-docker docker-engine docker.io**. If you skip this step, it could lead you to have some issues when running docker in the future.



```
arseniasc@w-node1: ~  
arseniasc@w-node1:~$ sudo apt-get purge docker lxc-docker docker-engine docker.io  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
Package 'lxc-docker' is not installed, so not removed  
Package 'docker-engine' is not installed, so not removed  
The following package was automatically installed and is no longer required:  
  vmdocker  
Use 'sudo apt autoremove' to remove it.  
The following packages will be REMOVED:  
  docker*  
0 upgraded, 0 newly installed, 1 to remove and 0 not upgraded.  
After this operation, 9,216 B disk space will be freed.  
Do you want to continue? [Y/n] y  
(Reading database ... 151764 files and directories currently installed.)  
Removing docker (1.5-2) ...  
arseniasc@w-node1:~$
```

-Next, update the apt package index. Type **sudo apt-get update**.

-Later, install the require packages needed to run docker. Please type **sudo apt-get install apt-transport-https ca-certificates curl gnupg2 software-properties-common**.

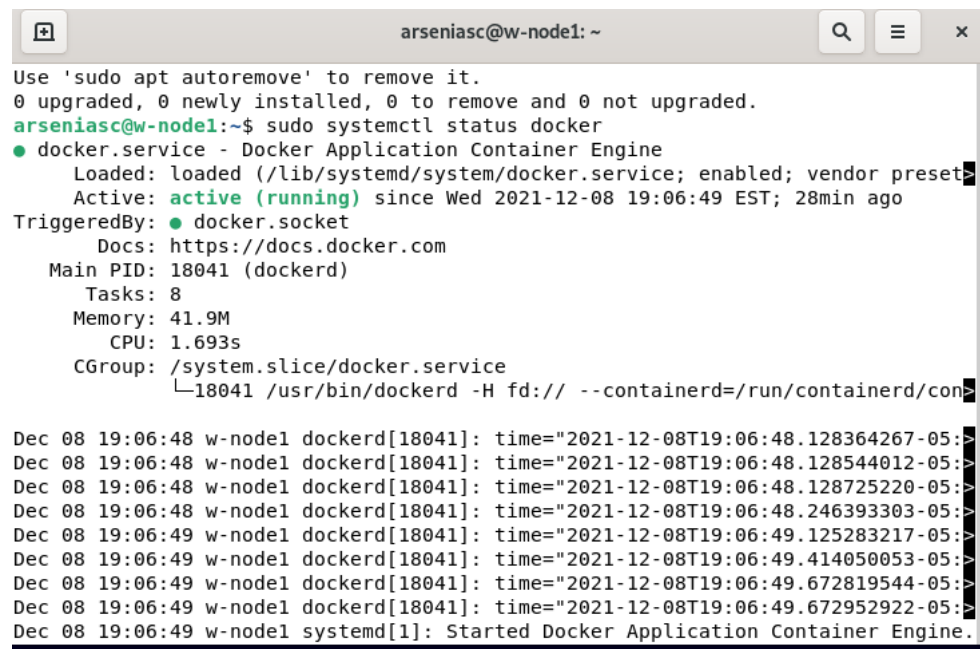
-Later, we must download the key to verify the integrity of packages before installing, type the following to install the key `curl -fsSL https://download.docker.com/linux/debian/gpg | sudo apt-key add -`

-Now, we are going to install the repository to our system, by typing the following command we will be able to do that `sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/debian buster stable"`

-Update the apt packages again, typing this command on the command line `sudo apt-get update`

-Later, we are going to install the docker engine type this command `sudo apt-get install docker-ce docker-ce-cli containerd.io`

-If you want to check the status of your docker type the following : `sudo systemctl status docker`

A terminal window titled 'arseniasc@w-node1: ~' showing the output of the command 'sudo systemctl status docker'. The output indicates that the 'docker.service' is 'active (running)'. It provides details such as the loaded path, active state, triggered by 'docker.socket', documentation link, main PID (18041), tasks (8), memory usage (41.9M), CPU time (1.693s), and CGroup. At the bottom, there is a log of system messages showing the Docker Application Container Engine starting successfully at 19:06:49.

```
arseniasc@w-node1: ~
Use 'sudo apt autoremove' to remove it.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
arseniasc@w-node1:~$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
   Active: active (running) since Wed 2021-12-08 19:06:49 EST; 28min ago
     TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
    Main PID: 18041 (dockerd)
       Tasks: 8
      Memory: 41.9M
         CPU: 1.693s
       CGroup: /system.slice/docker.service
               └─18041 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/con>

Dec 08 19:06:48 w-node1 dockerd[18041]: time="2021-12-08T19:06:48.128364267-05:>
Dec 08 19:06:48 w-node1 dockerd[18041]: time="2021-12-08T19:06:48.128544012-05:>
Dec 08 19:06:48 w-node1 dockerd[18041]: time="2021-12-08T19:06:48.128725220-05:>
Dec 08 19:06:48 w-node1 dockerd[18041]: time="2021-12-08T19:06:48.246393303-05:>
Dec 08 19:06:49 w-node1 dockerd[18041]: time="2021-12-08T19:06:49.125283217-05:>
Dec 08 19:06:49 w-node1 dockerd[18041]: time="2021-12-08T19:06:49.414050053-05:>
Dec 08 19:06:49 w-node1 dockerd[18041]: time="2021-12-08T19:06:49.672819544-05:>
Dec 08 19:06:49 w-node1 dockerd[18041]: time="2021-12-08T19:06:49.672952922-05:>
Dec 08 19:06:49 w-node1 systemd[1]: Started Docker Application Container Engine.
```

-After the system finish processing the download, you will be all set. In the image below you can see that docker is in your installed app list. To see that your installed apps you can type `apt list --installed`

```
arseniasc@debian: ~  
ioc-debian/stable,now 0.5 all [installed]  
locbook-xml/stable,now 4.5-9 all [installed,automatic]  
locker/stable,now 1.5-2 all [installed]  
lofstools/stable,now 4.2-1 amd64 [installed,automatic]  
lpgk/stable,now 1.20.9 amd64 [installed]  
:2fsprogs/stable,now 1.46.2-2 amd64 [installed]  
:ject/stable,now 2.36.1-8 amd64 [installed]  
:macs-bin-common/stable,now 1:27.1+1-3.1 amd64 [installed,automatic]  
:macs-common/stable,now 1:27.1+1-3.1 all [installed,automatic]
```

Kubernetes installation

Kubernetes, also referred to as K8s, is an open-source platform used to manage Linux Containers across private, public and hybrid cloud environments. Now we are going to learn how to install Kubernetes.

Instructions :

-To install Kubernetes, we need some packages to be installed before install the Kubernetes repository.

Type the following to install the package:

sudo apt-get install -y apt-transport-https ca-certificates curl

```
arseniasc@debian: ~  
arseniasc@debian:~$ sudo apt-get install -y apt-transport-https ca-certificates curl  
[sudo] password for arseniasc:  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
ca-certificates is already the newest version (20210119).  
The following NEW packages will be installed:  
  apt-transport-https curl  
0 upgraded, 2 newly installed, 0 to remove and 0 not upgraded.  
Need to get 427 kB of archives.  
After this operation, 603 kB of additional disk space will be used.  
Get:1 http://deb.debian.org/debian bullseye/main amd64 apt-transport-https all 2  
.2.4 [160 kB]  
Get:2 http://deb.debian.org/debian bullseye/main amd64 curl amd64 7.74.0-1.3+b1  
[267 kB]  
Fetched 427 kB in 0s (1,564 kB/s)  
Selecting previously unselected package apt-transport-https.  
(Reading database ... 151390 files and directories currently installed.)  
Preparing to unpack .../apt-transport-https_2.2.4_all.deb ...  
Unpacking apt-transport-https (2.2.4) ...  
Selecting previously unselected package curl.
```

-Also, we need to **download the Google Cloud public signing key** to download this type this command:

sudo curl -fsSL /usr/share/keyrings/kubernetes-archive-keyring.gpg
<https://packages.cloud.google.com/apt/doc/apt-key.gpg>

-Now, we are going to add the Kubernetes apt repository

```
arseniasc@debian: ~  
arseniasc@debian:~$ echo "deb [signed-by=/usr/share/keyrings/kubernetes-archive-keyring.gpg] https://apt.kubernetes.io/ kubernetes-xenial main" | sudo tee /etc/apt/sources.list.d/kubernetes.list  
deb [signed-by=/usr/share/keyrings/kubernetes-archive-keyring.gpg] https://apt.kubernetes.io/ kubernetes-xenial main  
arseniasc@debian:~$
```

-Then after adding the Kubernetes Repository we are going to update the apt package index.

Type the following to update the apt package

-Lastly, we are going to install the Kubernetes services **kubelet, kubeadm and kubectl**. Type the following command to install them:

sudo apt-get install -y kubelet kubeadm kubectl

A small paragraph describing how the Minikube tutorial went, what gave you trouble (if anything) and how you fixed it. Include screenshots to show how it went and what steps you did. You don't need to include everything, just a small selection is fine. Likely less than 2 pages will be generated here.

Minibuke

Is a tool that lets you run Kubernetes locally

Instructions

First, update the apt packages index. Type the following command to get it done **sudo apt update** and the **sudo apt upgrade** if any update is needed.

We need some packages to enable us to execute others commands later during the installation. So, install this package with the instructions below.

sudo apt install curl wget apt-transport-https -y as you see I already has this package on my system I installed it before.

```
arseniasc@debian: ~  
arseniasc@debian:~$ sudo apt install curl wget apt-transport-https -y  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
apt-transport-https is already the newest version (2.2.4).  
curl is already the newest version (7.74.0-1.3+b1).  
wget is already the newest version (1.21-1+b1).  
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.  
arseniasc@debian:~$
```

Now, we are going to use the **wget** command to download the latest Minikube library type the following command to do that :

```
arseniasc@debian: ~  
arseniasc@debian:~$ wget https://storage.googleapis.com/minikube/releases/latest  
/minikube-linux-amd64  
--2021-12-08 15:37:23-- https://storage.googleapis.com/minikube/releases/latest  
/minikube-linux-amd64  
Resolving storage.googleapis.com (storage.googleapis.com)... 142.250.65.208, 142  
.251.32.112, 142.251.35.176, ...  
Connecting to storage.googleapis.com (storage.googleapis.com)|142.250.65.208|:44  
3... connected.  
HTTP request sent, awaiting response... 200 OK  
Length: 69568775 (66M) [application/octet-stream]  
Saving to: 'minikube-linux-amd64'  
  
minikube-linux-amd64 100%[=====] 66.35M 3.15MB/s in 26s  
  
2021-12-08 15:37:49 (2.57 MB/s) - 'minikube-linux-amd64' saved [69568775/6956877  
5]  
arseniasc@debian:~$
```

Next, copy the binary file to the `/usr/local/bin` path type the following,

`sudo cp minikube-linux-amd64 /usr/local/bin/minikube`

Lastly, add **execute permission to the file**, type the following to do that **`sudo chmod +x /usr/local/bin/minikube`**.

Now, we are going to check the Minikube version type **`minikube version`**

```
arseniasc@debian: ~  
arseniasc@debian:~$ minikube version  
minikube version: v1.24.0  
commit: 76b94fb3c4e8ac5062daf70d60cf03ddcc0a741b  
arseniasc@debian:~$
```

Running minibuke

When I try to run minibuke it gave me this **error** shown in the image below

```
arseniasc@w-node1: ~  
arseniasc@w-node1:~$ minikube start  
😊 minikube v1.24.0 on Debian 11.1  
💡 Unable to pick a default driver. Here is what was considered, in preference  
order:  
  ■ docker: Not healthy: "docker version --format {{.Server.Os}}-{{.Server.Ve  
rsion}}" exit status 1: Got permission denied while trying to connect to the Do  
cker daemon socket at unix:///var/run/docker.sock: Get "http://%2Fvar%2Frun%2Fd  
ocker.sock/v1.24/version": dial unix /var/run/docker.sock: connect: permission  
denied  
  ■ docker: Suggestion: Add your user to the 'docker' group: 'sudo usermod -a  
G docker $USER && newgrp docker' <https://docs.docker.com/engine/install/linux-  
postinstall/>  
  ■ kvm2: Not installed: exec: "virsh": executable file not found in $PATH  
  ■ podman: Not installed: exec: "podman": executable file not found in $PATH  
  ■ vmware: Not installed: exec: "docker-machine-driver-vmware": executable f  
ile not found in $PATH  
  ■ virtualbox: Not installed: unable to find VBoxManage in $PATH  
  
❌ Exiting due to DRV_NOT_HEALTHY: Found driver(s) but none were healthy. See  
above for suggestions how to fix installed drivers.  
arseniasc@w-node1:~$
```

Let's fix it.

-Install some updates needed for the new version of docker.

After 2018 was implemented rootless Docker/Moby. Rootless Docker has been merged to the Docker/Moby upstream since Docker 19.03. to install those packages follow the next steps.

-First, type the following `curl -o rootless-install.sh -fsSL https://get.docker.com/rootless`

-Later, type this to install `sh rootless-install.sh`, when I typed this command and hit enter, in thew image below you can see my result

```
arseniasc@w-node1: ~  
arseniasc@w-node1:~$ curl -o rootless-install.sh -fsSL https://get.docker.com/ro  
otless  
arseniasc@w-node1:~$ sh rootless-install.sh  
# Installing stable version 20.10.11  
# Missing system requirements. Please run following commands to  
# install the requirements and run this installer again.  
# Alternatively iptables checks can be disabled with SKIP_IPTABLES=1  
  
cat <<EOF | sudo sh -x  
apt-get install -y uidmap  
EOF
```

-The system needs additional requirements, now you are going to follow these step to install the requirements:

Type `sudo -get install -y uidmap`

```
arseniasc@w-node1: ~  
bash: EOF: command not found  
arseniasc@w-node1:~$ sudo apt-get install -y uidmap  
[sudo] password for arseniasc:  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
The following package was automatically installed and is no longer required:  
  wmdocker  
Use 'sudo apt autoremove' to remove it.  
The following NEW packages will be installed:  
  uidmap  
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.  
Need to get 223 kB of archives.  
After this operation, 305 kB of additional disk space will be used.  
Get:1 http://deb.debian.org/debian bullseye/main amd64 uidmap amd64 1:4.8.1-1 [223 kB]  
Fetched 223 kB in 0s (876 kB/s)  
Selecting previously unselected package uidmap.  
(Reading database ... 151761 files and directories currently installed.)  
Preparing to unpack .../uidmap_1%3a4.8.1-1_amd64.deb ...  
Unpacking uidmap (1:4.8.1-1) ...  
Setting up uidmap (1:4.8.1-1) ...  
Processing triggers for man-db (2.9.4-2) ...
```

-Next, we will export to PATH type this command **PATH=\$HOME/bin:\$PATH** and hit enter.

-Now we need to connect to the rootless daemon, we can do that by typing this command export **DOCKER_HOST=unix://\$XDG_RUNTIME_DIR/docker.sock**

-Set the driver to **Driver= docker** as default each time when you run minibuke

sudo -E minikube start --driver=docker, this one does not need root privileges. After that please add \$user to docker group.

sudo usermod -aG docker \$USER && newgrp docker

run minibuke again. **minikube start --driver=docker**, now should be good.

```
Activities Terminal Dec 8 22:35  
arseniasc@Debian: ~  
arseniasc@Debian:~$ sudo usermod -aG docker $USER && newgrp docker  
[sudo] password for arseniasc:  
arseniasc@Debian:~$ minikube start --driver=docker  
🐳 minikube v1.24.0 on Debian 11.1  
🔧 Using the docker driver based on user configuration  
👉 Starting control plane node minikube in cluster minikube  
📦 Pulling base image ...  
📥 Downloading Kubernetes v1.22.3 preload ...  
> preloaded-images-k8s-v13-v1...: 501.73 MiB / 501.73 MiB 100.00% 1.75 MiB  
> gcr.io/k8s-minikube/kicbase: 355.78 MiB / 355.78 MiB 100.00% 978.32 KiB  
🔥 Creating docker container (CPUs=2, Memory=2200MB) ...  
📦 Preparing Kubernetes v1.22.3 on Docker 20.10.8 ...  
  ▪ Generating certificates and keys ...  
  ▪ Booting up control plane ...  
  ▪ Configuring RBAC rules ...  
🔍 Verifying Kubernetes components...  
  ▪ Using image gcr.io/k8s-minikube/storage-provisioner:v5  
🌟 Enabled addons: storage-provisioner, default-storageclass  
🎉 Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default  
arseniasc@Debian:~$
```

To open minikube type the following **minikube ssh**

```
docker@minikube: ~  
arseniasc@Debian:~$ minikube ssh  
docker@minikube:~$ exit  
logout  
arseniasc@Debian:~$ minikube ssh  
Last login: Thu Dec 9 03:42:37 2021 from 192.168.49.1  
docker@minikube:~$
```

You are all set.

OPTIONAL: A text document including your thoughts and struggles as you went through the ansible tutorial. What worked? What didn't?

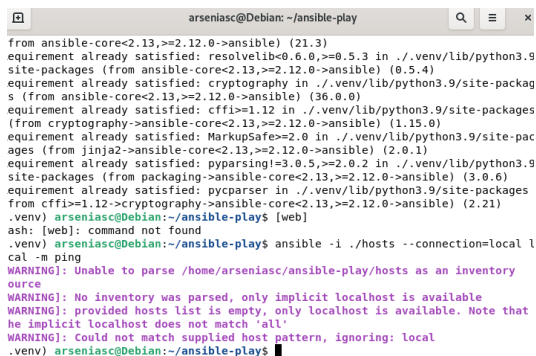
Ansible installation on Debian

What work?

It was nice the start using Ansible, it looks like ansible is a great tool to manage complex task in an easier way. I downloaded python 3, I create the directory to play with ansible, I create the python virtual environment and enabled it. I installed ansible.

What didn't?

As I said above, I could install ansible on Virtualenv's everything was working until I started the configuration. Everything started to give errors I try to fix it in different ways, but it did not work. In the image attached below you can see an error it gave me.



```
arseniasc@Debian: ~/ansible-play
from ansible-core<2.13,>=2.12.0->ansible) (21.3)
requirement already satisfied: resolvelib<0.6.0,>=0.5.3 in ./venv/lib/python3.9
site-packages (from ansible-core<2.13,>=2.12.0->ansible) (0.5.4)
requirement already satisfied: cryptography in ./venv/lib/python3.9/site-packag
s (from ansible-core<2.13,>=2.12.0->ansible) (36.0.0)
requirement already satisfied: cffi>=1.12 in ./venv/lib/python3.9/site-packages
(from cryptography->ansible-core<2.13,>=2.12.0->ansible) (1.15.0)
requirement already satisfied: MarkupSafe>=2.0 in ./venv/lib/python3.9/site-pac
ages (from jinja2->ansible-core<2.13,>=2.12.0->ansible) (2.0.1)
requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in ./venv/lib/python3.9
site-packages (from packaging->ansible-core<2.13,>=2.12.0->ansible) (3.0.6)
requirement already satisfied: pycparser in ./venv/lib/python3.9/site-packages
from cffi>=1.12->cryptography->ansible-core<2.13,>=2.12.0->ansible) (2.21)
.venv) arseniasc@Debian:~/ansible-play$ [web]
ash: [web]: command not found
.venv) arseniasc@Debian:~/ansible-play$ ansible -i ./hosts --connection=local l
cal -m ping
WARNING]: Unable to parse /home/arseniasc/ansible-play/hosts as an inventory
source
WARNING]: No inventory was parsed, only implicit localhost is available
WARNING]: provided hosts list is empty, only localhost is available. Note that
the implicit localhost does not match 'all'
WARNING]: Could not match supplied host pattern, ignoring: local
.venv) arseniasc@Debian:~/ansible-play$
```


CentOS

Docker installation

Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly. Docker helps to create containers. Today we are going to learn how to install docker this useful tool.

Instructions

After you successfully have logged in on your server, follow the next steps to install docker

1. Uninstall Default Docker Packages

-Type the following to uninstall the old docker package , **sudo yum remove docker**

If you skip this step, it could lead you to have some issues when running docker in the future.

Update the yum package type **sudo yum update**

-Set up the repository

Install the **yum-utils** package this will offer the configuration manager.

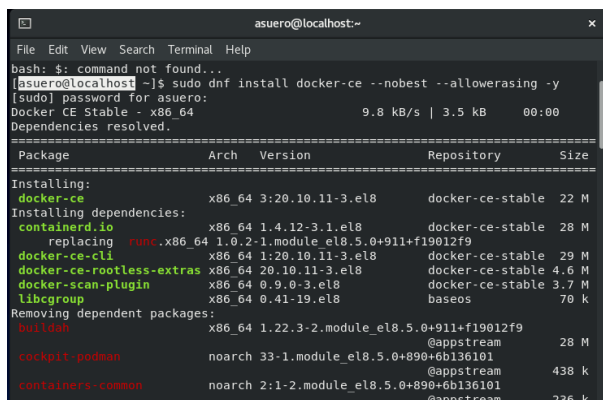
To install the utils type **sudo yum install -y yum-utils**

-Add the repository

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

-Later, Install Docker Engine

To install the newest version of docker type the following **command sudo dnf install docker-ce --nobest --alloweraseing -y**



```
asueiro@localhost:~  
File Edit View Search Terminal Help  
bash: $: command not found...  
[asueiro@localhost ~]$ sudo dnf install docker-ce --nobest --alloweraseing -y  
[sudo] password for asueiro:  
Docker CE Stable - x86_64 9.8 kB/s | 3.5 kB 00:00  
Dependencies resolved.  
=====
```

Package	Arch	Version	Repository	Size
Installing:				
docker-ce	x86_64	3:20.10.11-3.el8	docker-ce-stable	22 M
Installing dependencies:				
containerd.io	x86_64	1.4.12-3.1.el8	docker-ce-stable	28 M
replacing runc	x86_64	1.0.2-1.module_el8.5.0+911+f19012f9	docker-ce-stable	29 M
docker-ce-cli	x86_64	1:20.10.11-3.el8	docker-ce-stable	4.6 M
docker-ce-rootless-extras	x86_64	20.10.11-3.el8	docker-ce-stable	3.7 M
docker-scan-plugin	x86_64	0.9.0-3.el8	baseos	70 k
libcgrouper	x86_64	0.41-19.el8	baseos	70 k
Removing dependent packages:				
buildah	x86_64	1.22.3-2.module_el8.5.0+911+f19012f9	@appstream	28 M
cockpit-podman	noarch	33-1.module_el8.5.0+890+6b136101	@appstream	438 k
containers-common	noarch	2:1-2.module_el8.5.0+890+6b136101	@appstream	236 k

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>**

exclude=kubelet kubeadm kubectl

EOF

-update the dnf type **dnf upgrade -y**

Install the kubernetes services

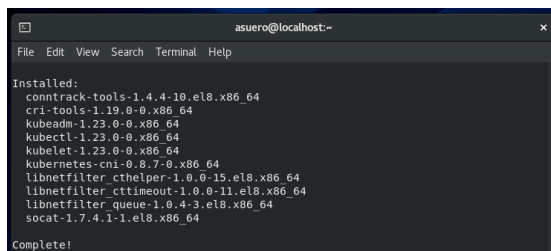
-Now, we are going to add the Kubernetes apt repository

-Then after adding the Kubernetes Repository we are going to update the apt package index.

Type the following to update the apt package

-Now, we are going to install the Kubernetes services, **kubelet, kubeadm and kubectl**. Type the following command to install them:

dnf install -y kubelet kubeadm kubectl --disableexcludes=Kubernetes



```
asuro@localhost:~$ dnf install -y kubelet kubeadm kubectl --disableexcludes=Kubernetes
Installed:
  conntrack-tools-1.4.4-10.el8.x86_64
  cri-tools-1.19.0-0.x86_64
  kubeadm-1.23.0-0.x86_64
  kubectl-1.23.0-0.x86_64
  kubelet-1.23.0-0.x86_64
  kubernetescni-0.8.7-0.x86_64
  libnetfilter_cthelper-1.0.0-15.el8.x86_64
  libnetfilter_cttimeout-1.0.0-11.el8.x86_64
  libnetfilter_queue-1.0.4-3.el8.x86_64
  socat-1.7.4.1-1.el8.x86_64
Complete!
```

-Enable Kubernetes service to do that type, **systemctl enable kubelet**

Start Kubernetes to do that type **systemctl start kubelet**

A small paragraph describing how the Minikube tutorial went, what gave you trouble (if anything) and how you fixed it. Include screenshots to show how it went and what steps you did. You don't need to include everything, just a small selection is fine. Likely less than 2 pages will be generated here.

Minibuke

Minikube is a tool that lets you run Kubernetes locally

Instructions

First, update the yum packages index. Type the following command to get it done **sudo yum update**

We need some packages to be installed before to install minikube. To install those packages, follow the instructions below

-Install epel by typing this command **sudo yum -y install epel-release**

-Install KVM hypervisor for the virtualization **sudo yum -y install libvirt qemu-kvm virt-install virt-top libguestfs-tools bridge-utils**

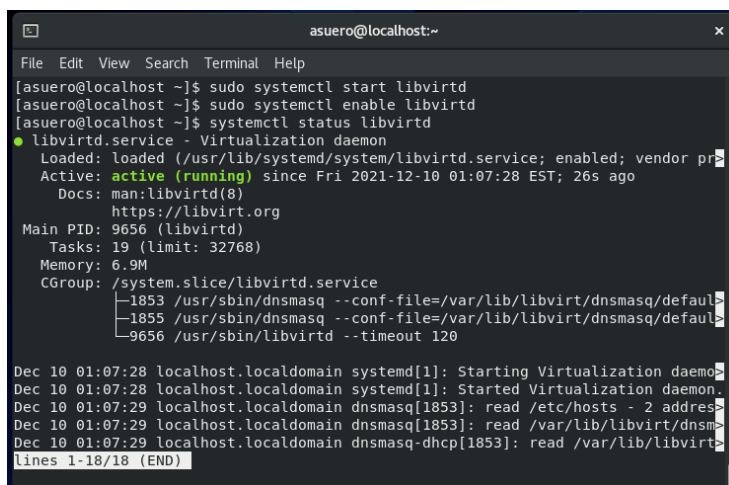
-Now, start and enable libvirtd service type

sudo systemctl start libvirtd

sudo systemctl enable libvirtd

-Check the status of libvirtd service

systemctl status libvirtd

A terminal window titled 'asuerdo@localhost:~' showing the execution of several commands to install and manage the libvirtd service. The commands are: 'sudo systemctl start libvirtd', 'sudo systemctl enable libvirtd', and 'systemctl status libvirtd'. The output of the status command shows that the service is loaded, enabled, and active (running) since Fri 2021-12-10 01:07:28 EST. It also displays details about the service's main PID, tasks, memory usage, and CGroup. At the bottom, there are logs from systemd and dnsmasq indicating the successful start of the virtualization daemon and the reading of host addresses.

```
asuerdo@localhost:~  
File Edit View Search Terminal Help  
[asuerdo@localhost ~]$ sudo systemctl start libvirtd  
[asuerdo@localhost ~]$ sudo systemctl enable libvirtd  
[asuerdo@localhost ~]$ systemctl status libvirtd  
● libvirtd.service - Virtualization daemon  
   Loaded: loaded (/usr/lib/systemd/system/libvirtd.service; enabled; vendor pre  
   Active: active (running) since Fri 2021-12-10 01:07:28 EST; 26s ago  
     Docs: man:libvirtd(8)  
           https://libvirt.org  
  Main PID: 9656 (libvirtd)  
    Tasks: 19 (limit: 32768)  
   Memory: 6.9M  
    CGroup: /system.slice/libvirtd.service  
            └─1853 /usr/sbin/dnsmasq --conf-file=/var/lib/libvirt/dnsmasq/default  
            └─1855 /usr/sbin/dnsmasq --conf-file=/var/lib/libvirt/dnsmasq/default  
            └─9656 /usr/sbin/libvirtd --timeout 120  
  
Dec 10 01:07:28 localhost.localdomain systemd[1]: Starting Virtualization daemo  
Dec 10 01:07:28 localhost.localdomain systemd[1]: Started Virtualization daemon.  
Dec 10 01:07:29 localhost.localdomain dnsmasq[1853]: read /etc/hosts - 2 addres  
Dec 10 01:07:29 localhost.localdomain dnsmasq[1853]: read /var/lib/libvirt/dnsm  
Dec 10 01:07:29 localhost.localdomain dnsmasq-dhcp[1853]: read /var/lib/libvirt  
lines 1-18/18 (END)
```

-Add your user to libvirt group, type **sudo usermod -a -G libvirt \$(whoami)**

-Restart libvirtd **sudo systemctl restart libvirtd.service**

Now we can start installing Minikube

-Install the Minikube package using wget

wget https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64

```
asuario@localhost:~  
File Edit View Search Terminal Help  
[asuario@localhost ~]$ sudo vi /etc/libvirt/libvirtd.conf  
[sudo] password for asuario:  
[asuario@localhost ~]$ wget https://storage.googleapis.com/minikube/releases/latest  
st/minikube-linux-amd64  
--2021-12-10 01:28:36-- https://storage.googleapis.com/minikube/releases/latest  
/minikube-linux-amd64  
Resolving storage.googleapis.com (storage.googleapis.com)... 142.250.81.240, 142  
.251.32.112, 142.251.35.176, ...  
Connecting to storage.googleapis.com (storage.googleapis.com)|142.250.81.240|:44  
3... connected.  
HTTP request sent, awaiting response... 200 OK  
Length: 69568775 (66M) [application/octet-stream]  
Saving to: 'minikube-linux-amd64'  
  
minikube-linux-amd64 100%[=====] 66.35M 1.99MB/s in 19s  
  
2021-12-10 01:28:55 (3.48 MB/s) - 'minikube-linux-amd64' saved [69568775/6956877  
5]  
[asuario@localhost ~]$
```

-Now, use the **chmod** command to give the file execute permission:

chmod +x minikube-linux-amd64

- Lastly, move the file to the **/usr/local/bin** directory:

sudo mv minikube-linux-amd64 /usr/local/bin/minikube

***Done!** you have minikube on your server.

-To check the status of the installation type: **minikube version**

```
asuario@localhost:~  
File Edit View Search Terminal Help  
[asuario@localhost ~]$ minikube version  
minikube version: v1.24.0  
commit: 76b94fb3c4e8ac5062daf70d60cf03ddcc0a741b  
[asuario@localhost ~]$
```

Running minibuke

When I try to run minibuke it gave me this **error** shown in the image below

```
asuario@localhost:~  
File Edit View Search Terminal Help  
[asuario@localhost ~]$  
[asuario@localhost ~]$ minikube start  
❌ minikube v1.24.0 on Centos 8.5.2111  
💡 Unable to pick a default driver. Here is what was considered, in preference  
order:  
  • docker: Not healthy: "docker version --format {{.Server.Os}}-{{.Server.Ver  
sion}}" exit status 1: Got permission denied while trying to connect to the Dock  
er daemon socket at unix:///var/run/docker.sock: Get "http://%2Fvar%2Frun%2Fdock  
er.sock/v1.24/version": dial unix /var/run/docker.sock: connect: permission deni  
ed  
  • docker: Suggestion: Add your user to the 'docker' group: 'sudo usermod -aG  
docker $USER && newgrp docker' <https://docs.docker.com/engine/install/linux-po  
stinstall/>  
  • kvm2: Not healthy: /usr/bin/virsh domcapabilities --virttype kvm failed:  
error: failed to get emulator capabilities  
error: invalid argument: KVM is not supported by '/usr/libexec/qemu-kvm' on this  
host  
exit status 1  
  • kvm2: Suggestion: Follow your Linux distribution instructions for configur  
ing KVM <https://minikube.sigs.k8s.io/docs/reference/drivers/kvm2/>  
  • vmware: Not installed: exec: "docker-machine-driver-vmware": executable fi  
le not found in $PATH  
  • podman: Not installed: exec: "podman": executable file not found in $PATH
```

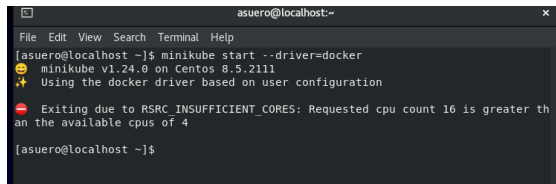
Let's fix it.

-Add my user to the docker group, **sudo usermod -aG docker \$USER && newgrp docker**

-Set up docker driver to take it by default type **minikube start --driver=docker**

-Start docker service, type **sudo systemctl start docker**

I try to open minikube again, and I found a new error

A terminal window titled 'asuero@localhost:~' showing the command 'minikube start --driver=docker' being executed. The output indicates that minikube v1.24.0 is running on Centos 8.5.2111 and is using the docker driver. However, it then fails with an error: 'Exiting due to RSRC_INSUFFICIENT_CORES: Requested cpu count 16 is greater than the available cpus of 4'. The prompt returns to the user at the shell.

```
asuero@localhost:~  
File Edit View Search Terminal Help  
[asuero@localhost ~]$ minikube start --driver=docker  
🐳 minikube v1.24.0 on Centos 8.5.2111  
🔗 Using the docker driver based on user configuration  
❌ Exiting due to RSRC_INSUFFICIENT_CORES: Requested cpu count 16 is greater than the available cpus of 4  
[asuero@localhost ~]$
```

To fix that set up your minikube to 16 cores type **minikube config set cpus 16**

Even though I set up the cpus to 16 it kept saying the same thing, and do not allow me to start minikube...

Source:

<https://docs.docker.com/get-started/overview/#:~:text=Docker%20is%20an%20open%20platform,ways%20you%20manage%20your%20applications.>

<https://searchitoperations.techtarget.com/definition/Google-Kubernetes>