How to create Kubernetes home lab on an old laptop

itnext.io/how-to-create-kubernetes-home-lab-on-an-old-laptop-1de6cc12c13e

27 August 2021

Photo by on

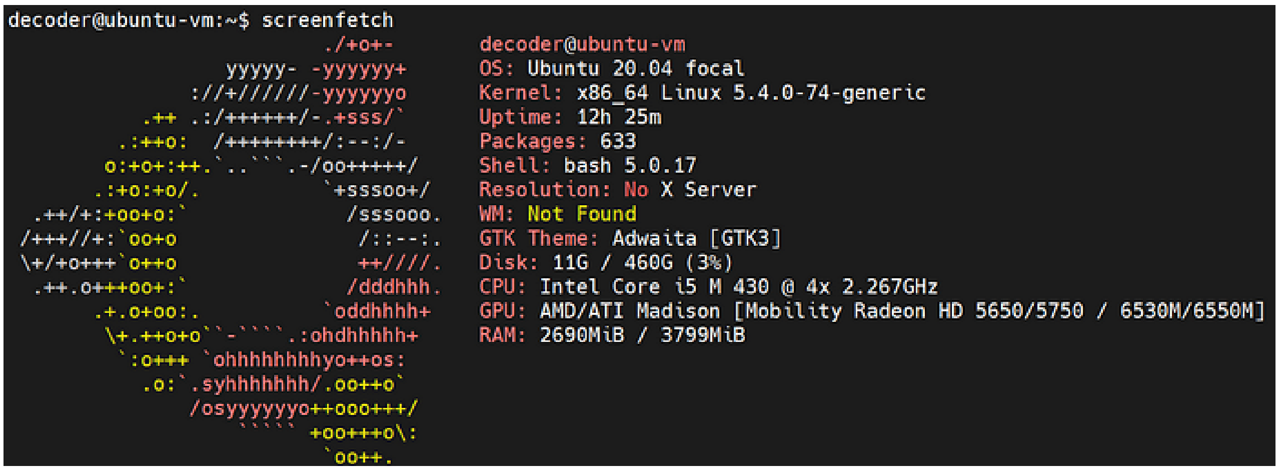
Introduction

Piotr

ITNEXT

I’ve recently “discovered” an old laptop forgotten somewhere in the depths of my basement and decided to create a mini home lab with it where I could play around with Kubernetes.

Here are the specs:

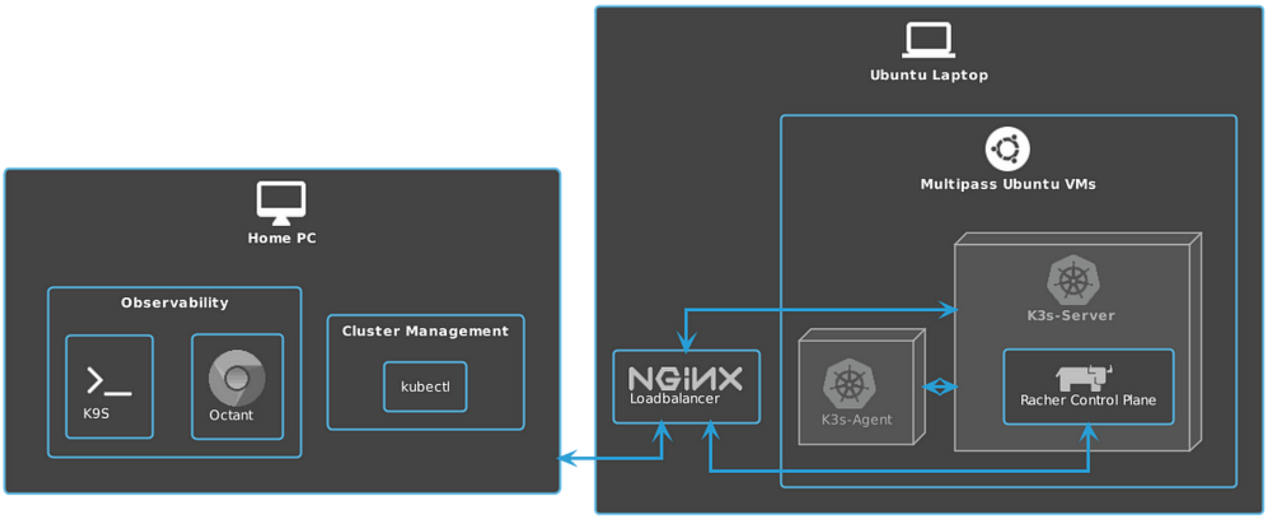


~~~
decoder@ubuntu-vm:~$ screenfetch
                          ./+o+-       decoder@ubuntu-vm
                  yyyyy- -yyyyyy+      OS: Ubuntu 20.04 focal
               ://+//////-yyyyyyo      Kernel: x86_64 Linux 5.4.0-74-generic
           .++ .:/++++++/-.+sss/`      Uptime: 12h 25m
         .:++o:  /++++++++/:--:/-      Packages: 633
        o:+o+:++.`..```.-/oo+++++/     Shell: bash 5.0.17
       .:+o:+o/.          `+sssoo+/    Resolution: No X Server
  .++/+:+oo+o:`             /sssooo.   WM: Not Found
 /+++//+:`oo+o               /::--:.   GTK Theme: Adwaita [GTK3]
 \+/+o+++`o++o               ++////.   Disk: 11G / 460G (3%)
  .++.o+++oo+:`             /dddhhh.   CPU: Intel Core i5 M 430 @ 4x 2.267GHz
       .+.o+oo:.          `oddhhhh+    GPU: AMD/ATI Madison [Mobility Radeon HD 5650/5750 / 6530M/6550M]
        \+.++o+o``-````.:ohdhhhhh+     RAM: 2690MiB / 3799MiB
         `:o+++ `ohhhhhhhhyo++os:
           .o:`.syhhhhhhh/.oo++o`
               /osyyyyyyo++ooo+++/
                   ``````  +oo+++o\:
                          `oo++.
~~~

Laptop spec

In this blog, we are going to go through the whole setup process and end up with a usable Kubernetes setup. The goal is to create a simple but fully-fledged Kubernetes setup that we can quickly spin up and keep alive for some time to run simple workloads and learn/test.

The end architecture would like like this:



Install ubuntu

Start by downloading ubuntu and burning it to a USB. I have used etcher and downloaded ubuntu 20.04.2 LTS from the official page using the torrent link. For torrents download, I recommend qBittorrent but use whatever you are comfortable with.

Once everything is ready, plug in the USB with Ubuntu image to your laptop and follow installation instructions on the Ubuntu page. You can also find a YouTube vid to help you through the process. Here is one https://www.youtube.com/watch?v=K2m52F0S2w8.

Select the following additional packages:

- openssh
- docker

During the "SSH Setup" phase, make sure to select "Allow Password Authentication over SSH". This will allow for a quick login without setting up PKI. Once you are happy with the setup and will want to keep it running, it is easy to disable it and swap to key-based authentication mode.

Enable Wi-Fi [Optional]

After the first boot, I have decided to enable Wi-Fi since I only have one Lan cable. This is optional if you have a dedicated cable for your laptop.

Disable default lid actions

Typically laptops are used as desktop devices and are configured with screen/power saving options. One of such options is powering off or hibernating when the laptop lid is closed. Let's change it to make sure our laptop stays on even when the lid is closed.

Create virtual machines with Multipass

> Multipass is a lightweight VM manager for Linux, Windows and macOS. It’s designed for developers who want a fresh Ubuntu environment with a single command. It uses KVM on Linux, Hyper-V on Windows and HyperKit on macOS to run the VM with minimal overhead. It can also use VirtualBox on Windows and macOS. Multipass will fetch images for you and keep them up to date.
>
> Since it supports metadata for cloud-init, you can simulate a small cloud deployment on your laptop or workstation.

canonical/multipass

Multipass is a lightweight VM manager for Linux, Windows and macOS. It's designed for developers who want a fresh…

github.com

Install k3s server and agent

Since the laptop spec is pretty low, especially on the memory side, we need a Kubernetes distribution that works well on low-end hardware.

There are two choices:

- Rancher
- Mirantis

I have selected k3s as I’m familiar with them and in the end, I want to be able to manage my cluster with Rancher Management.

If you are interested in reading more about the comparison between k3s and k0s here is a good blog:

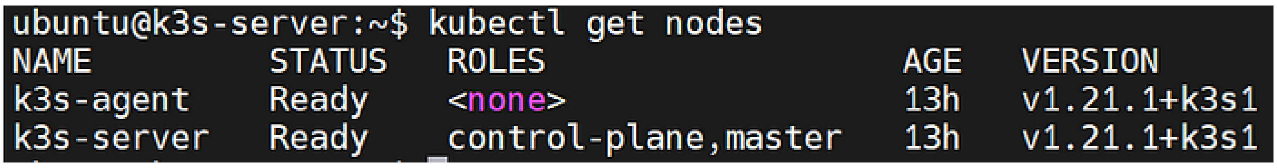
If the k3s config is still not writable, change permissions with

`sudo chmod 644 /etc/rancher/k3s/k3s.yaml`

After installation, let’s make sure that our cluster is up and running

```
# Login to ks3 server vmmultipass shell ks3-server# Check if the nodes are up and runningkubeclt get nodes
```

The output should be similar to this:



```
ubuntu@k3s-server:~$ kubectl get nodes
NAME         STATUS   ROLES                  AGE   VERSION
k3s-agent    Ready    <none>                 13h   v1.21.1+k3s1
k3s-server   Ready    control-plane,master   13h   v1.21.1+k3s1
```

Running nodes

A quick note on taints. Normally we wouldn't want our workloads running on the master node and use taints and tolerations to make sure pods are scheduled only on a worker node. In our case, since the setup is not HA (Highly Available) due to hardware resource constraints, we are going to allow pods scheduling on a master node.

Setup Load Balancer

Our setup so far consists of bare metal "server" with 2 VMS on it. To establish traffic between the host pc (windows 10 in my case) and the Kubernetes cluster, we need to make sure that all requests sent to the server are forwarded to Kubernetes IP or our k3s-server VM.

To do this, we will use nginx and replace the default config with load balancing to our cluster.

https://ubuntu.com/tutorials/install-and-configure-nginx#4-setting-up-virtual-host

```
sudo apt install nginxsudo cat >/etc/nginx/nginx.conf<< EOFevents {}stream {
upstream k3s_servers {  server ${K3S_NODEIP_SERVER}:6443; }server { listen 6443;
proxy_pass k3s_servers; }}EOF
```

Connect to the cluster from your PC

We have a running 2 nodes cluster. Now, let's connect to it from our home PC.

Before connecting from our PC, we need to copy kube config setup from the VM and merge it with our kube config file on the home PC.

I advise upgrading kubectl to the latest version beforehand, follow the installation steps from the .

```
# Login to k3s-server vmmultipass shell k3s-server# Copy content of the kube
configkubectl config view --raw# Create a new file on your machine in the ~./kube
directory and copy contentcat > config_k3s [enter] <paste content>[enter – adds
one line to the file][Ctrl + C – exit]# Merge two config files using KUBECONFIG
variableKUBECONFIG=/root/.kube/config:/root/.kube/config_k3s
```

If you want to learn more about managing kube config, please visit Kubectl context and Configuration on Kubernetes.io

Make sure you swapped to a newly created context.

```
# Check all contextskubectl config get-contexts# Swap to "default" (this is the
name in my case)kubectl config use-context default
```

Now running the **kubectl get nodes** command from your home PC terminal should result in the same output.

Install cluster visualization tools

Finally, let's install 2 of my favourite cluster visualization tools:

- is a VMWare open-source cluster visualizer, running in a browser so no local installation is required.
- is a really amazing console-based cluster visualization tool.

Install Rancher to manage your cluster [Optional]

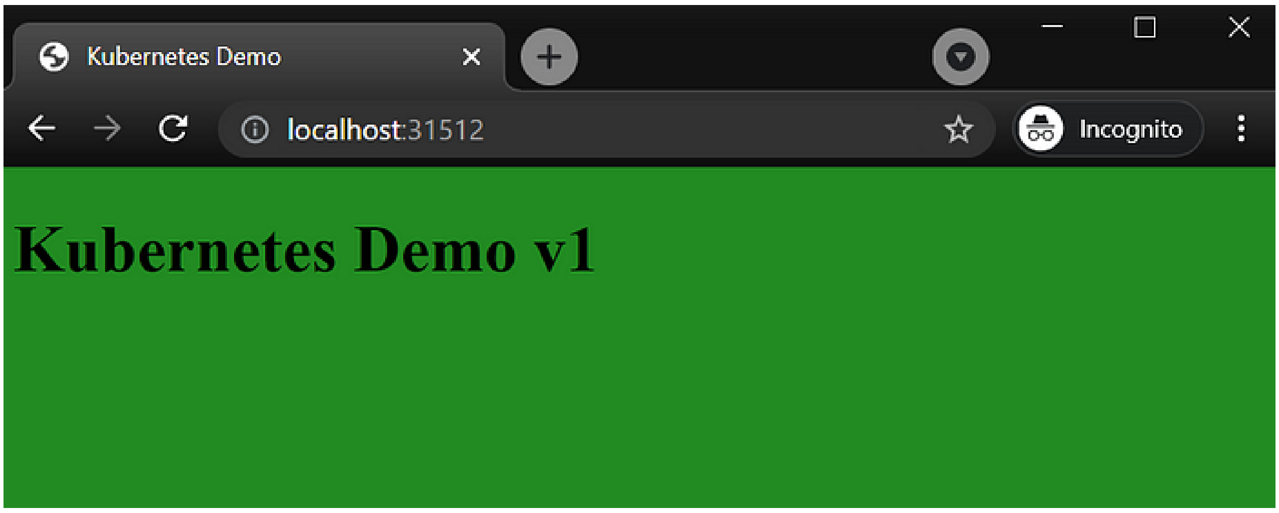
There are many options to install Rancher on any Kubernetes platform. Since this example is for testing purposes, we will follow the installation steps using Rancher in docker.

Run a sample workload

Let's make sure our cluster can accept and serve simple workloads

Since we have exposed a deployment on NodePort, we need to see what port was assigned by Kubernetes:

```
# Check node portkubectl get svc
```



Kubernetes Demo v1

Summary

This installation should serve as an introduction to setting up a home lab even with an old laptop and starting to play around with Kubernetes. There are many things we haven't tried or tested and I encourage you to try on your own and share your experiences.

Additional resources

Here is a list of additional resources and some materials I used when creating this blog.

K3s can be installed much easier using a tool called k3sup, check out the repo from Alex Ellis on GitHub:

alexellis/k3sup

k3sup is a light-weight utility to get from zero to KUBECONFIG with k3s on any local or remote VM. All you need is ssh…

github.com

If you would like to learn how to script some of the manual operations, check out this repo from Sebastiaan van Steenis working in Rancher

superseb/multipass-k3s

Use multipass instances to create your k3s cluster - superseb/multipass-k3s

github.com

If you would like to go through a similar setup, but with video, I highly recommend checking out Techno Tim's YouTube channel.

Finally, if you are interested in learning more about k3s, here is a good overview:

So far this setup was done mostly manually, in the next blogs I will look into automating the rancher deployment using terraform provider for Rancher 2

https://registry.terraform.io/providers/rancher/rancher2/latest/docs

If you encounter any issues along the way, found a bug or simply want to share your ideas, please drop a comment and have fun experimenting with your home lab :)