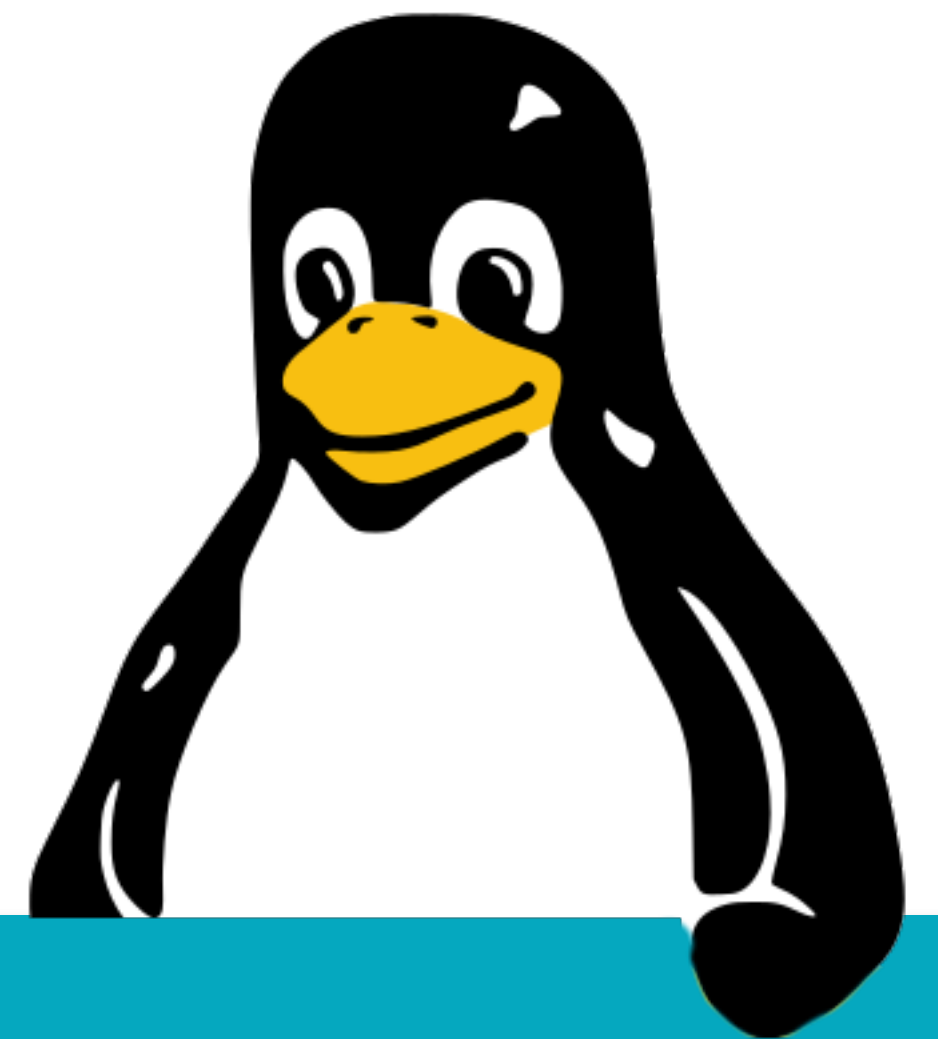


# Linux, day 8



# LAB: Vagrant

# Let's install it!

- We will install Vagrant on our host OS.
  - It will control VirtualBox.
- Go to this site, download for your OS and install.
  - <https://www.vagrantup.com/downloads>
  - Or: "*[yum/apt/brew/winget] install vagrant*"
  - Windows will require a reboot.

# Vagrant configuration files

- Vagrant configurations have a directory tree of files:
  - The main config file
  - And a whole bunch of per-VM files
- You can have multiple of these directory trees!
  - One per project.

# Our first VM

- Open a terminal, or Powershell.
  - Go to your Downloads folder.
  - Make a new directory *"vagrant1"*.
  - *"cd"* into the new *"vagrant1"* directory.
- Run: *"vagrant init debian/buster64"*
  - Check the *"Vagrantfile"*.

# Boot your VM

- Run: "*vagrant up*"
- This will:
  - Download the needed VM image.
  - Setup the VM in VirtualBox.
  - Setup the port forward for SSH.
  - And start the VM!

# Boot your VM

- Booting will take a while. When it's done:
  - "*vagrant ssh*" logs you into the VM.
  - "*vagrant halt*" stops the VM.
  - "*vagrant destroy*" destroys the VM.
- Go ahead and destroy this VM.
  - If you "*vagrant up*" again it's now faster.

# Let's do something cool

- I have provided you with a sample *Vagrantfile*.
  - "*008 - Vagrantfile*"
- In your Downloads folder, make a dir "*vagrant2*".
- "*cd*" into "*vagrant2*".
- Now copy the *008-Vagrantfile* into "*vagrant2*".
  - Rename to "*Vagrantfile*".
  - Yes, a capital V.



# Let's read the Vagrantfile!

- The syntax is more complicated than before!
- It has a number of recognizable blocks.
- Can you figure out what we're doing here?

**Note:** On Intel i-series and Windows 11, you must change the CPU core count to "2".

# Boot the test network

- Run "*vagrant up*" in the "*vagrant2*" directory.
  - This will take longer! Now it's 3 VMs!
  - Afterwards, you can browse to:
    - <http://localhost:8081>
    - <http://localhost:8082>
    - <http://localhost:8083>

# What happened?

- We provisioned our 3 VMs using a shell script.
  - Each with a web server
  - ...and its own "*index.html*".
- You could also use Ansible to provision.

# Challenge



# Challenge!

- Based on my Vagrantfile (with Alpine),
  - Can you make a new Vagrantfile for:
    - One VM, on 192.168.56.33
    - With a port forward of 9080 (host) to 80 (guest).
    - Running *lighttpd*, with the following content?
    - <https://github.com/cloudacademy/static-website-example>

# Made a mistake?

- Mistakes in the post-install script?
  - No need to destroy!
  - Just run "*vagrant provision*".

# Step by step

- The Vagrantfile should have:
  - Not three but one host.
  - An adjusted port forward.
  - "*git*" added via the "*apk add*" command
  - A "*git clone*", with the files copied into *htdocs*.
  - Fix the file permissions for files+dirs in *htdocs*.

# Spoilers!

- Yes... "*008 - VagrantSpoilers*" is the solution.
  - Try it without spoilers first.



# LAB: Docker

# Let's install it on Ubuntu

- Ubuntu is easy.

```
$ sudo apt install -y docker.io
```

```
$ sudo systemctl start docker
```

# Fedora is harder! 🤖

```
$ sudo yum install -y yum-utils
$ sudo yum-config-manager \
--add-repo \
https://download.docker.com/linux/fedora/
docker-ce.repo

$ sudo yum install docker-ce
$ sudo systemctl start docker
```

# A quick test

- Let's see if we can run something!

```
$ sudo docker pull hello-world
```

```
$ sudo docker run hello-world
```

# Our first container

- In Teams you will find "*008 - Docker.tgz*"
  - Copy this to your VM.
- On your VM, go to your Downloads folder.
  - Extract "*008 - Docker.tgz*".
  - This makes "*~/Downloads/docker-alp/*".

# Let's read the *Dockerfile*!

- The syntax looks way different from Vagrant.
- Each line is a step in the build process.
  - You choose a base OS image.
  - You install extra software and sources.
  - And you specify what to run at boot time.

# Building the container

- Run:

```
$ sudo docker build -t tess/demo .
```

```
$ sudo docker run -ti -p 8080:80 tess/demo
```

# Result?

- Use Fedora's browser to visit:
  - <http://localhost:8080>
- Or on the command line:
  - *curl* <http://localhost:8080>



# Looking at Docker

- More info? Debugging? What's running?
  - *docker images*
  - *docker ps*
  - *docker exec -ti \${containerID} /bin/sh*
  - *docker logs \${containerID}*

# Challenge



# Challenge!

- You have made all kinds of Python scripts, right?
- Can you make a container that runs one?
  - Literally, just run your Python script in a container.

# Step by step

- You will need to:
  - Base on a suitable image, like *"python:slim-buster"*.
  - Put your script in the build directory.
  - Set the script as CMD,
  - With Python as ENTRYPOINT.

# Spoilers!

- Yes... "*008 - DockerSpoilers*" is the solution.
  - Try it without spoilers first.

# Closing



# Homework

- Reading:
  - Chapter 11, p. 329-348

# Homework

- Go do:
  - Use the three VMs made by Vagrant (*vagrant1*).
  - Practice SSH between the hosts.



# Homework

- Go do:
  - Use the three VMs made by Vagrant (*vagrant2*).
  - Setup RSync so */var/www/html* is synced,
    - From host 1, to hosts 2 and 3.
    - Make changes to your "*index.html*" and run *rsync*.
  - This does NOT need to go into your *Vagrantfile*.