Vagrant

COMS10012 / COMSM0085

Software Tools

Virtualisation

"Allows your operating system to act as if it were a completely diffrent operating system"

- emulate a different stack
- reproducible build environment
- cost / scalability

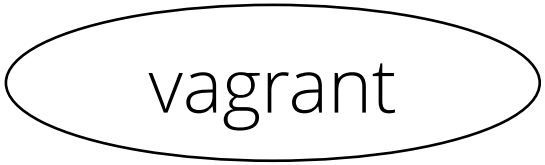
Software

Virtualisation:

VMware,
VirtualBox (Oracle)

Containers:

Docker
Kubernetes



bochs, qemu,
DOSbox, ...

OpenStack, rkt, ...

Installing vagrant

Lab machines (but not seis): installed

From the web: www.vagrantup.com/download

Linux: vagrant recommends *not* using your system's package manager (but Arch seems to work).

Windows: read www.vagrantup.com/docs/installation, you may need to disable Hyper-V.

Vagrant

Host: folder with Vagrantfile (ruby)

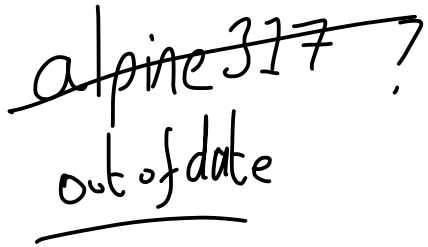
Different providers

ssh access to guest

can share folders between host/guest

Vagrantfile

```
Vagrant.configure("2") do |config|
    config.vm.box = "generic/debian12"
end
```



box repository:
https://app.vagrantup.com/boxes/search

Start the machine

```
$ vagrant up
Bringing machine 'default' up with
'virtualbox' provider...
==> default: Importing base box
'generic/debian12'...
...
==> default: Machine booted and ready!
```

Commands

These commands only work if you're in a directory with a vagrantfile & it will apply to the machine described in that file.

`vagrant up` start machine
`vagrant ssh` log in
`vagrant halt` stop machine
vagrant reload stop+start machine (for config update)
`vagrant destroy` delete machine

All commands require a Vagrantfile in the current directory.

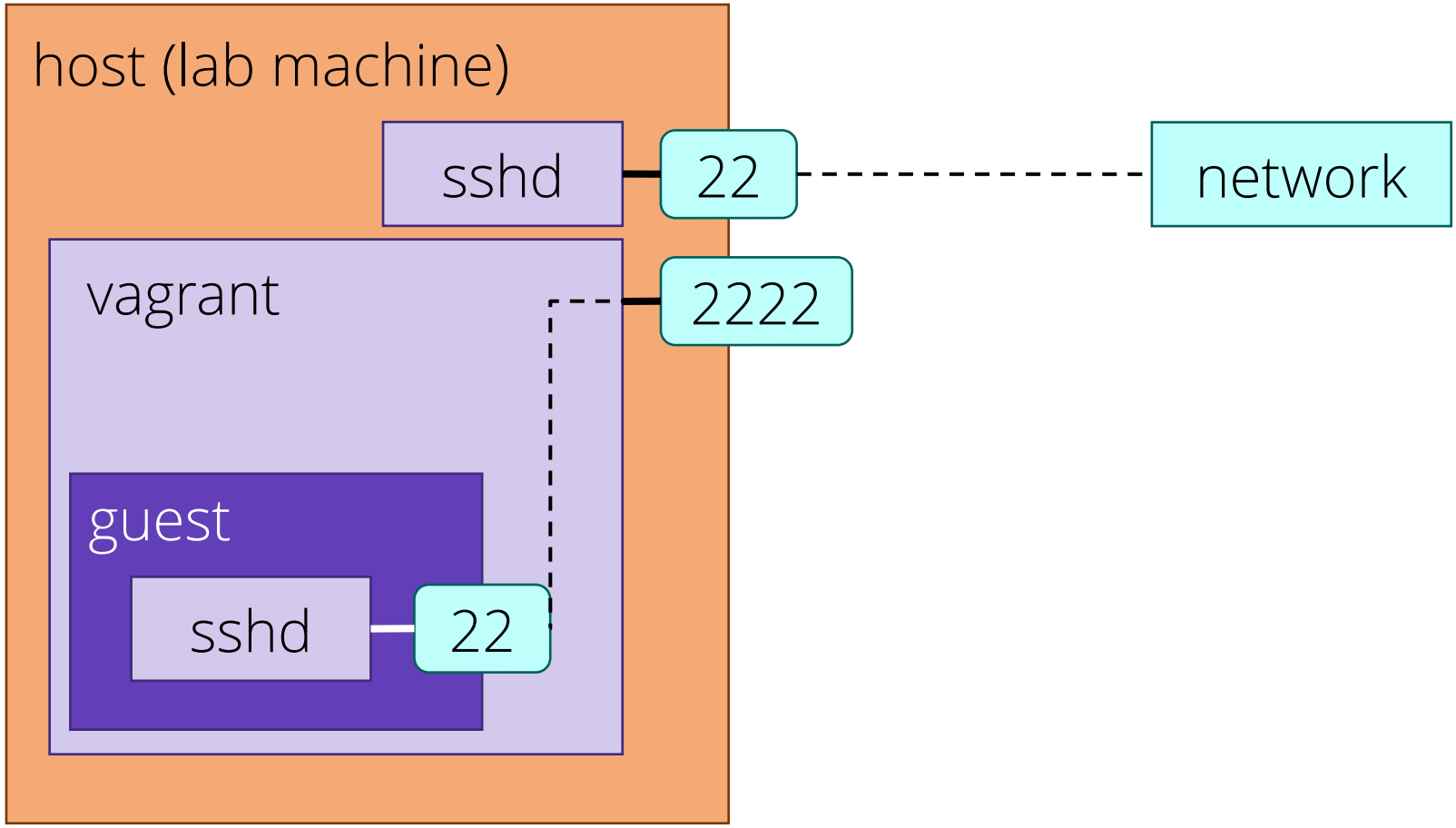
Log in

```
$ vagrant ssh
vagrant@debian12:~$
vagrant@debian12:~$ whoami
vagrant
vagrant@debian12:~$ exit
logout
Connection to 127.0.0.1 closed.
$
```

ssh

```
$ vagrant up
...
==> default: Forwarding ports...
    default: 22 (guest) => 2222 (host)
...
    default: SSH address: 127.0.0.1:2222
    default: SSH username: vagrant
    default: SSH auth method: private key
    default: Vagrant insecure key detected.
Vagrant will automatically replace this with a
newly generated keypair for better security.
    default: Inserting generated public key within
guest...
```

ssh and port forwarding



keys

Remember: if you have a *secret key*, you can ssh in to a machine that has the matching *public key*.

Vagrant box (in repository) has a default public/secret key pair.

When you provision (`vagrant up`) a box, it creates a new key pair – this is more secure, and you can use it with `vagrant ssh`.

Storage

Normal use: virtual machines stored in

- Linux: `~/.vagrant.d`
- Windows: `C:\Users\NAME\.vagrant.d`

Some configuration goes in the `.vagrant` folder in the folder with the Vagrantfile.

Storage – lab machines

VMs are stored in `/tmp` and may not survive host reboots!

Also, they are not on NFS, so not visible from other lab machines.

- This is by design.
- **Treat VMs on lab machines as disposable – back up your data somewhere else!**

Debian Linux

- Common, well-supported Linux distribution.
- Saves us some headaches from previous years' Alpine
- Mostly very similar – but look at slides and exercises if something in a video seems specific to Alpine.

Vagrant

Vagrant is an open-source tool that helps in the management of virtualised development environments. It provides a command line interface to create, configure & manage virtual machines in a consistent & reproducable manner.

Based on a config file called Vagrantfile, it can download & configure disk images, which it calls boxes & call other programs to run them - Vagrant doesn't run the VM itself, we'll use another program called virtualbox for that.

I have just installed Vagrant & virtual box on my laptop.

Creating a Vagrantfile (config)

This will be the config for a Debian Linux (popular Linux distribution)

- Create an empty folder somewhere
- Create a file called Vagratfile in this folder (no extension)
- In this file, input the following:

```
Vagrant.configure("2") do |config|
    config.vm.box = "generic/debian12"
    config.vm.synced_folder ".", "/vagrant"
    config.vm.provision "shell", inline: <<-SHELL
      echo "Post-provision installs go here"
    SHELL
end
```

This is written in Ruby.

- This line selects the virtual machine image (box) to use. List of available boxes in link below
- This sets up a shared folder between the guest (VM) and the host (my laptop)
- This is the most complex line, it refers to how you should provision for the VM, i.e how you set up & configure it when the box is first downloaded

This is a very bare set of provisions which makes it difficult to see what is going on.

It may also look like:

```
config.vm.provision "shell", inline: <<-SHELL
    apt-get update
    apt-get install -y [PACKAGE]
    echo "provisioning complete"
```

- The initial "shell" is a provisioner. It indicates that the provisioner being used is a shell script
- inline: option specifies that the shell script commands are coming from within the Vagrantfile rather than an external script.
- <<-SHELL is a heredocument (heredoc) syntax used in Ruby, allowing you to define a multi-line string. The shell script commands that follow will be part of this multi-line string. The end of this string is denoted by the closing argument, SHELL.

Running Vagrant:

In the directory of the Vagrantfile, do:

Vagrant up

For now, this needs to be done on powershell because WSL doesn't have windows access.

This will take a while the first time you run it.

Take this to the lab on friday!!

- Then you can log into your VM using:

Vagrant ssh

This updates your prompt to: vagrant@debian12:~$

Shared folder doesn't appear, also take this to a lab

Using Vagrant on Lab machines:

Creating VMs on the lab machine network takes up lots of space & would be very slow. Running an OS over a network share causes both bandwidth & latency problems.

Therefore, Vagrant on lab machines store VM instances in the /tmp folder Meaning:

- If you log into a different lab machine your VMs will be gone
- If you restart the same lab machine ... gone
- Your VMs & any files stored in them are not backed up.

Not an issue:

- Any software you want installed should have their install commands added to provisioning script in the Vagrantfile
- Any files you want to keep, put in the shared file (/vagrant)