DevOps Workshop:
for Software Engineers

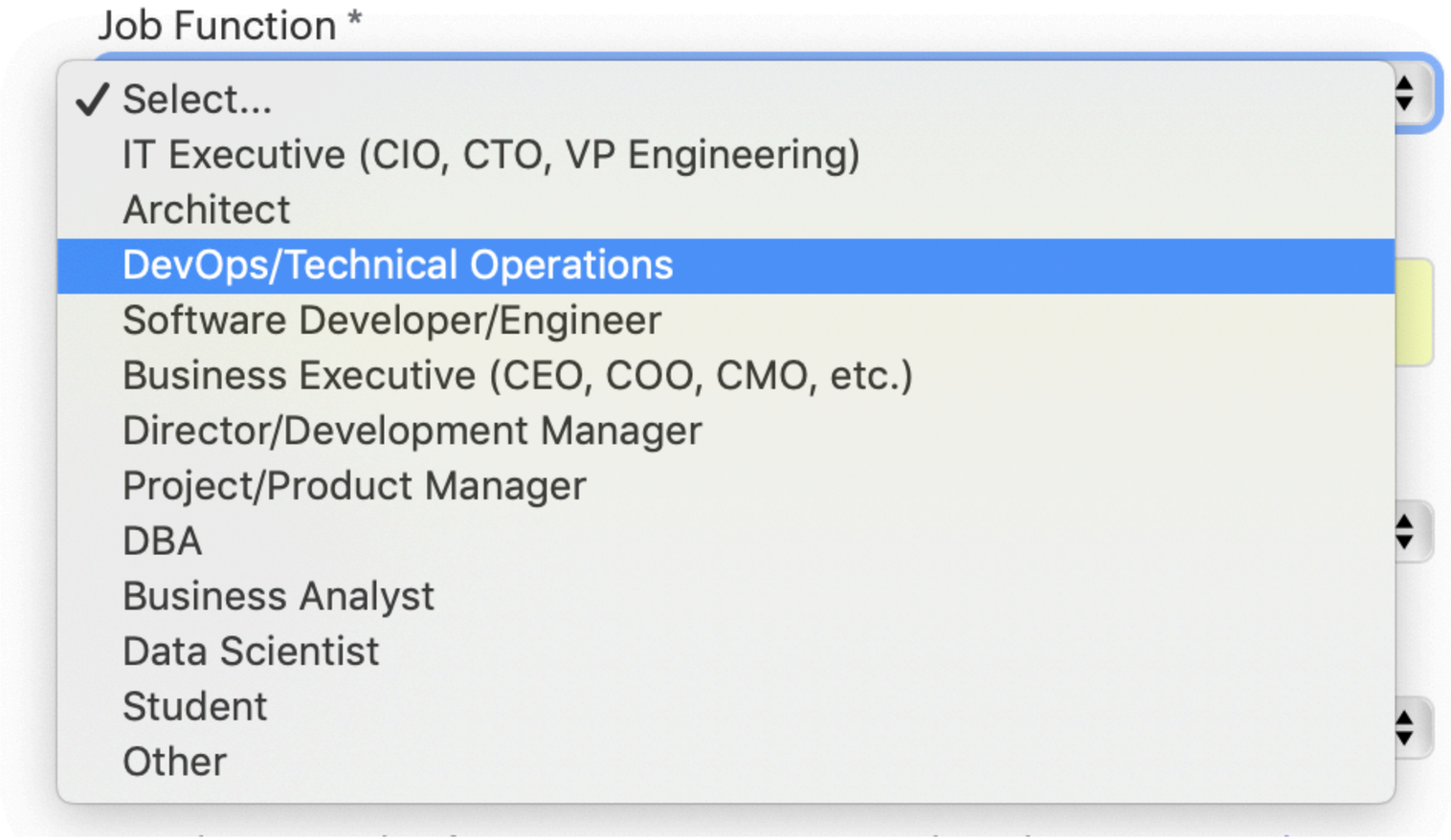
Instructor:
John J Rofrano
Senior Technical Staff Member, DevOps Champion
IBM T.J. Watson Research Center
rofrano@us.ibm.com
@JohnRofrano

Why "for Software Engineers"?

Industry Problem

Select your job function:

From the web site of a major player who should know better



"Zero to Hero" Training

- This training assumes you know nothing about the subject matter
- It does assume that you have a good grasp of software engineering
- I will take you from Zero to Hero on the following topics:
  - DevOps Culture
  - Infrastructure as Code
  - Cloud Native Applications (12-Factor)
  - Test Driven Development
  - Continuous Integration and Continuous Delivery (CI/CD)
  - Docker Containers & Kubernetes

Some Assembly Required

- Tools you will need to complete this lab:
  - Computer running macOS/OS X, Linux, or Windows*
  - Internet Access to download boxes
  - Programmer Editor (Visual Studio Code)
  - GitHub Account
  - PC users must have "VT-x/AMD-V hardware acceleration" turned on in your BIOS for VirtualBox to work.

*** Windows users will need an ssh client**

Coke-a-Cola Wars

...I can't take it anymore

- Pepsi - Coke
- Paper - Plastic
- Windows - Mac
- So many choices
- So little productivity

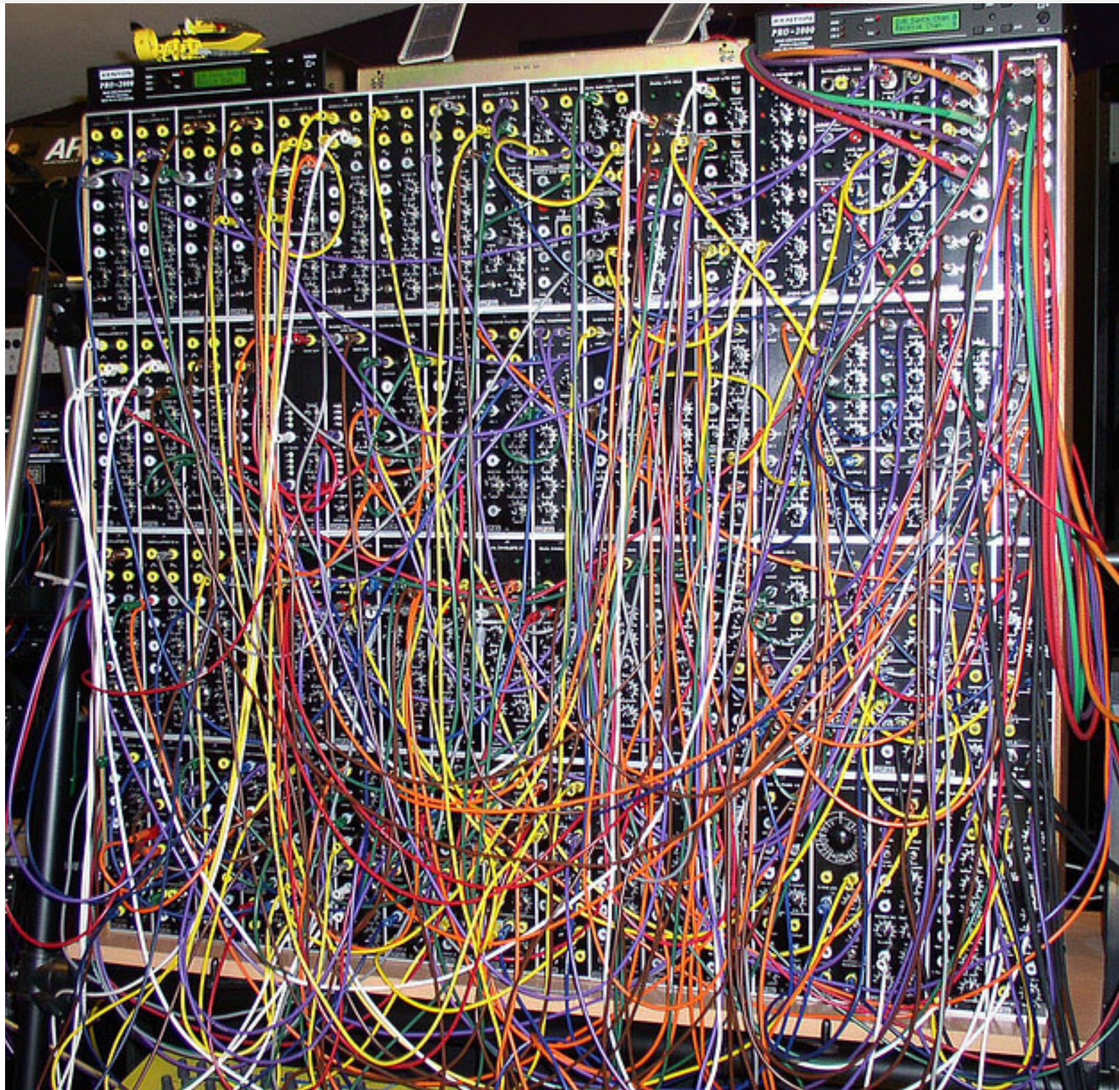
Mac OS

We will use Ubuntu Linux 18.04

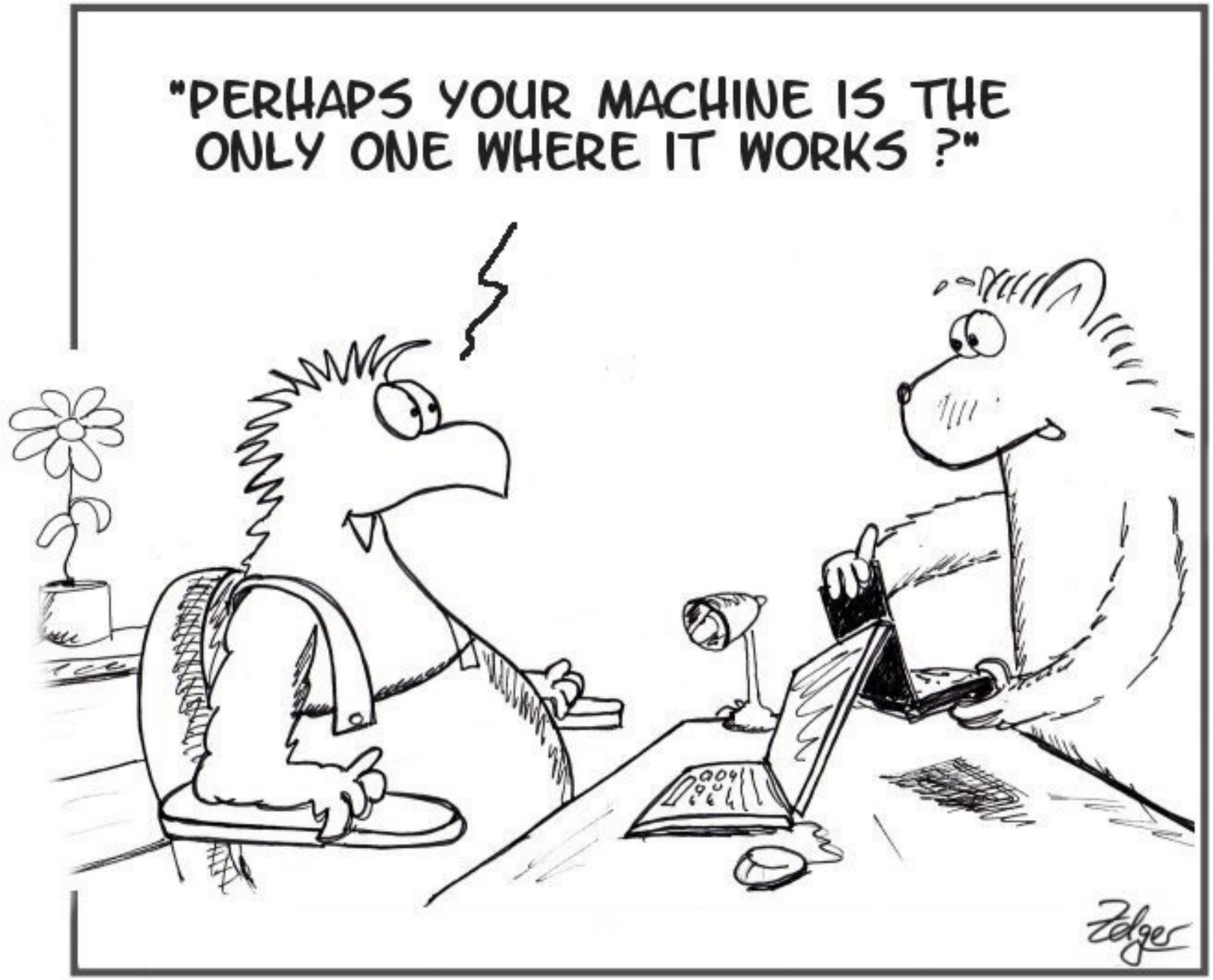
Prerequisite Software for Hands-On Sessions

The Developer On-boarding Problem

- Manually creating local environments for developers to work in is:
  - Time consuming
  - Error prone
  - Inconsistent at best
  - Unreproducible at worst!



The Goal is to Avoid This



It works on my machine

The Wrong Solution

- Carefully and painstakingly create documentation in the form of “runbooks” that give step-by-step instructions on how to recreate the development environment via cut-n-paste
- NOT VERY AGILE !!!
  - We value: Working software over comprehensive documentation

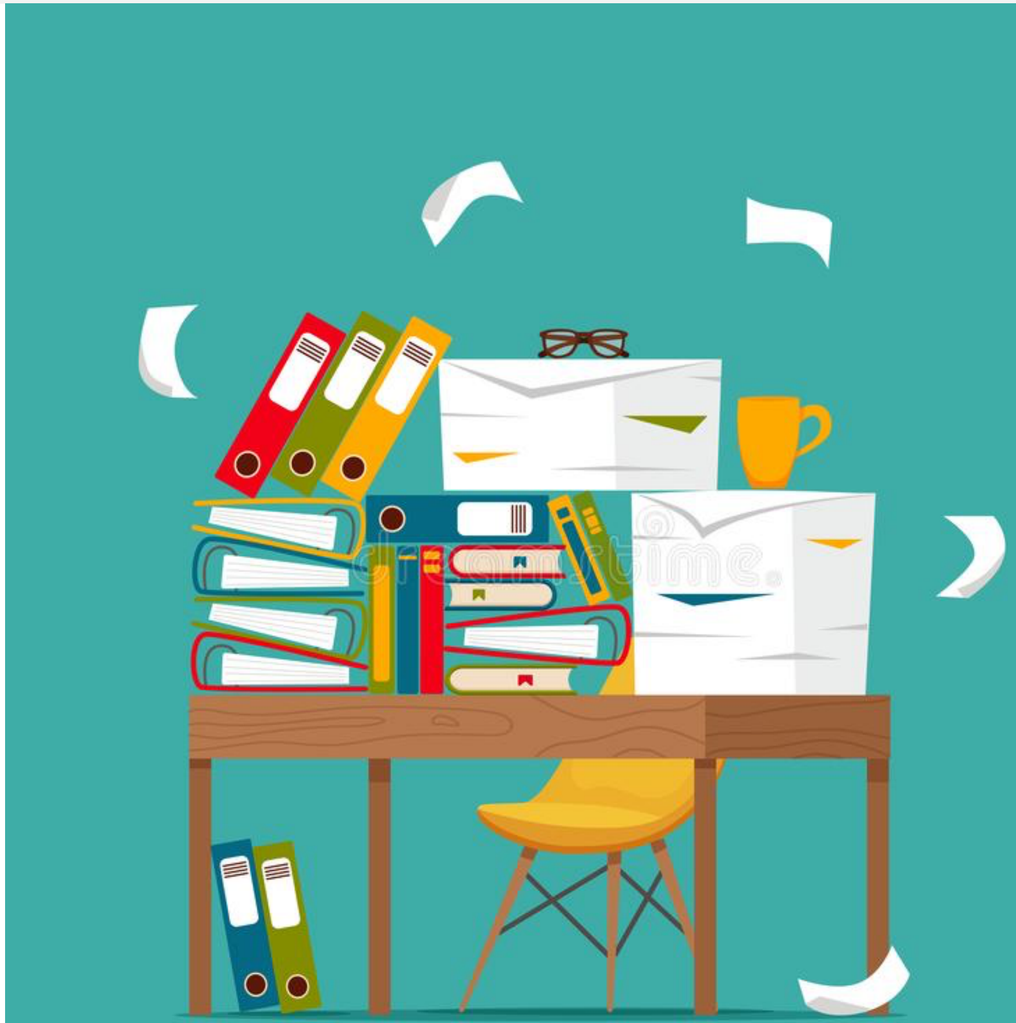


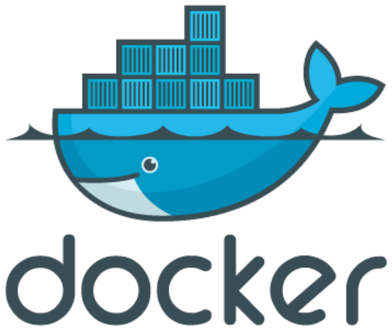
Image: ID 125314060 © Olga Shcherba | dreamstime.com

The Right Solution: Infrastructure as Code

Automate the creation of local development environments right on your laptop or desktop

Use Vagrant to quickly provision complex configurations consistently with repeatability

Use Docker to handle middleware without any installation



Use VirtualBox to create Virtual Machines to develop in

Prerequisite Software Installation Instructions

https://github.com/rofrano/devops-workshop

Cut-n-paste from the README.md

README.md

DevOps Workshop

DevOps Workshop Training that uses Vagrant, VirtualBox, Docker, and Minikube

This repository is part of a 4 Hour DevOps Workshop that gives attendees first hand experience in building a microservice and deploying it as a container on Kubernetes using DevOps practices and methods. It demonstrates how to create a simple RESTful service using Python Flask and SQLAlchemy to access a persistent resource. Its purpose is to show the correct API and return codes that should be used for a REST API. This repo also demonstrates the use of Test Driven Development which us required for setting on a Continuous Integrate and Continuous Delivery Pipeline (CI/CD).

Hands-On

“live session”

macOS Users

If you are using a Mac you can use Homebrew*

Install Homebrew: (https://brew.sh)

```
ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"
```

Install VirtualBox and Vagrant using Homebrew:

```
$ brew install git
$ brew cask install virtualbox
$ brew cask install vagrant
$ brew cask install visual-studio-code
$ brew cask install postman
```

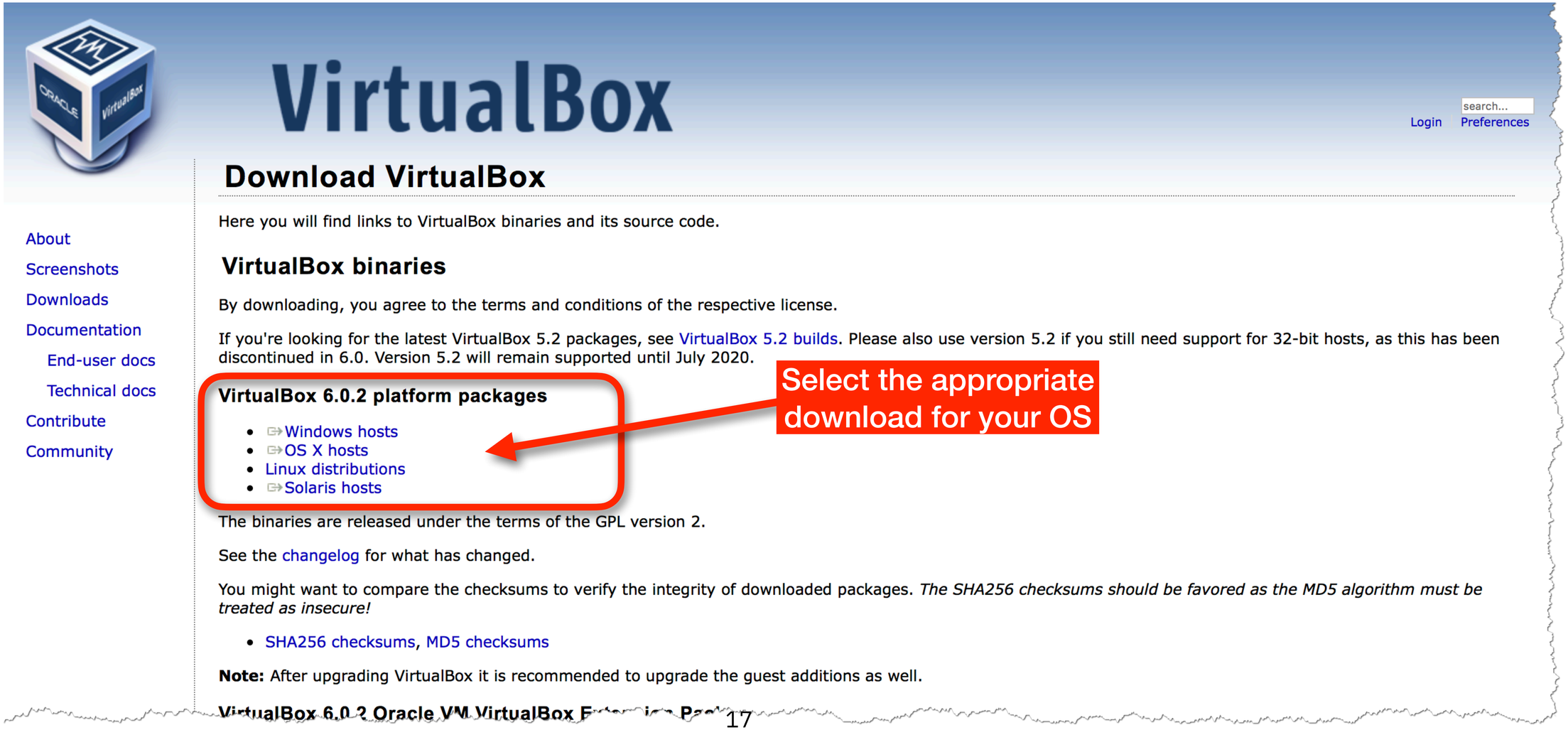
* You must open a terminal (shell) to execute these commands

macOS Users

- If you are using a Mac you can skip the next 6 charts 🙂
- Windows users need to do more configuration... 😞

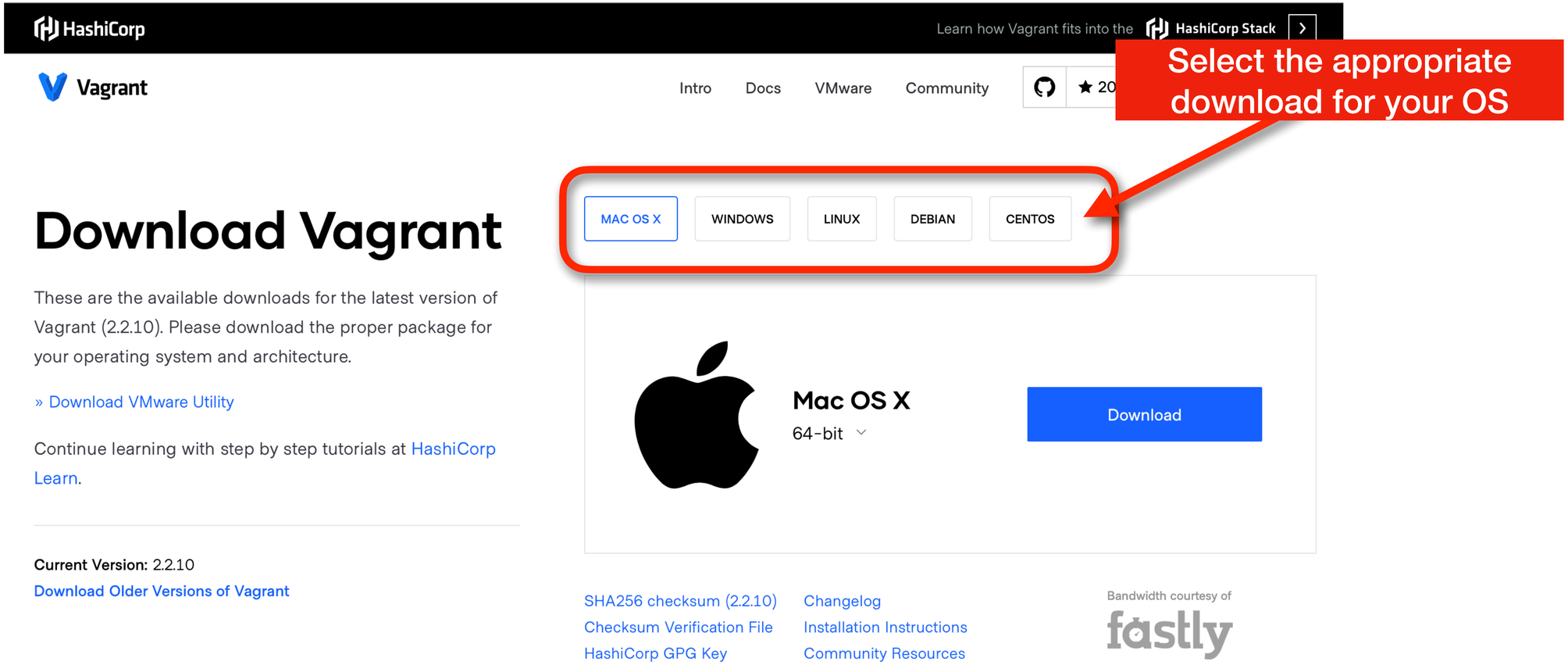
Windows Users Download and Install VirtualBox

https://www.virtualbox.org/wiki/Downloads



Windows Users Download and Install Vagrant

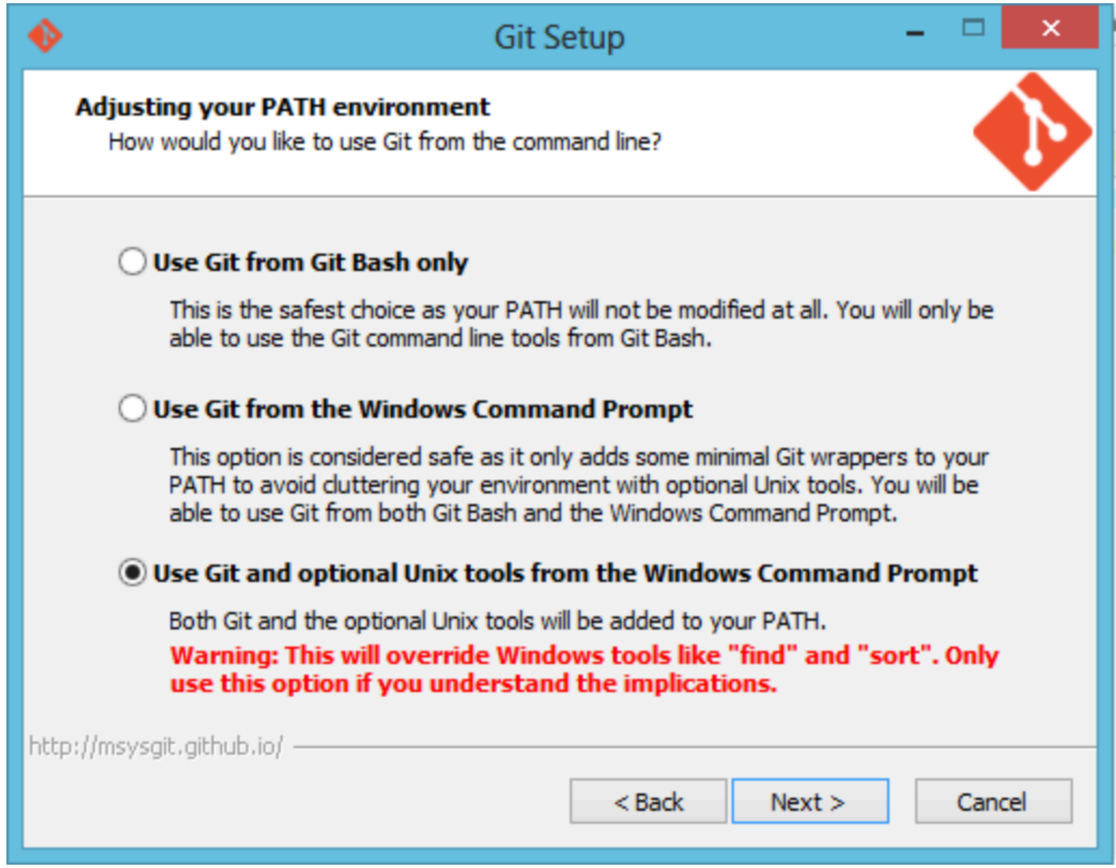
https://www.vagrantup.com/downloads.html



Windows Install SSH

- Vagrant requires an SSH client which Windows doesn't have by default
- The easiest way to solve this is to install git with the optional unix tools from: `http://git-scm.com/downloads`
- Here is a great tutorial:

`http://tech.osteel.me/posts/2015/01/25/how-to-use-vagrant-on-windows.html`



Windows VT-x/VT-d

- VirtualBox requires that Hardware-Assisted Virtualization is enabled in order to work in 64-bit mode.
- Some PC laptop manufacturers disable this by default. ¯_(ツ)_/¯
- If when you try and bring vagrant up you get an error that it cannot load the 64-bit VM then you need to change your BIOS settings to enable VT-x/VT-d
- If you don't know how to do this, you need to consult with your PC documentation because this is different for every manufacturer.
  (... or just buy a Mac 😉)

Install VirtualBox and Vagrant

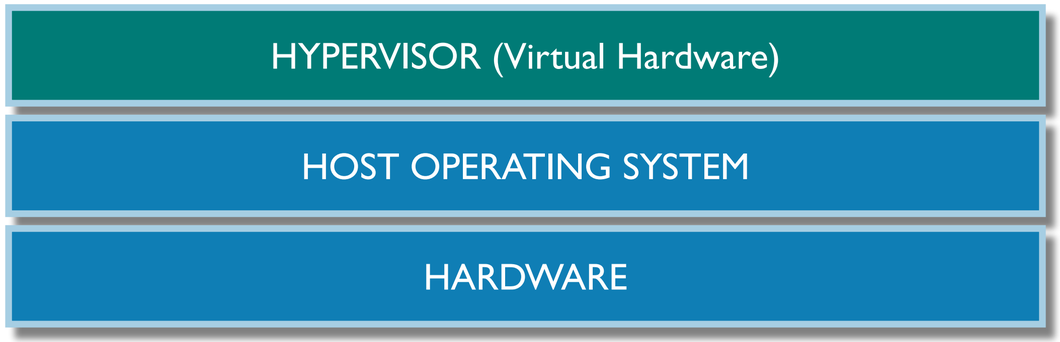
- Once you have the downloads for your OS:
  - Install VirtualBox taking all of the defaults
  - If prompted to download extensions say "Yes"
  - Install Vagrant taking all of the defaults
- You should now have the pre-requisite software installed

AND NOW BACK TO OUR REGULARLY SCHEDULED PROGRAM

Infrastructure as Code

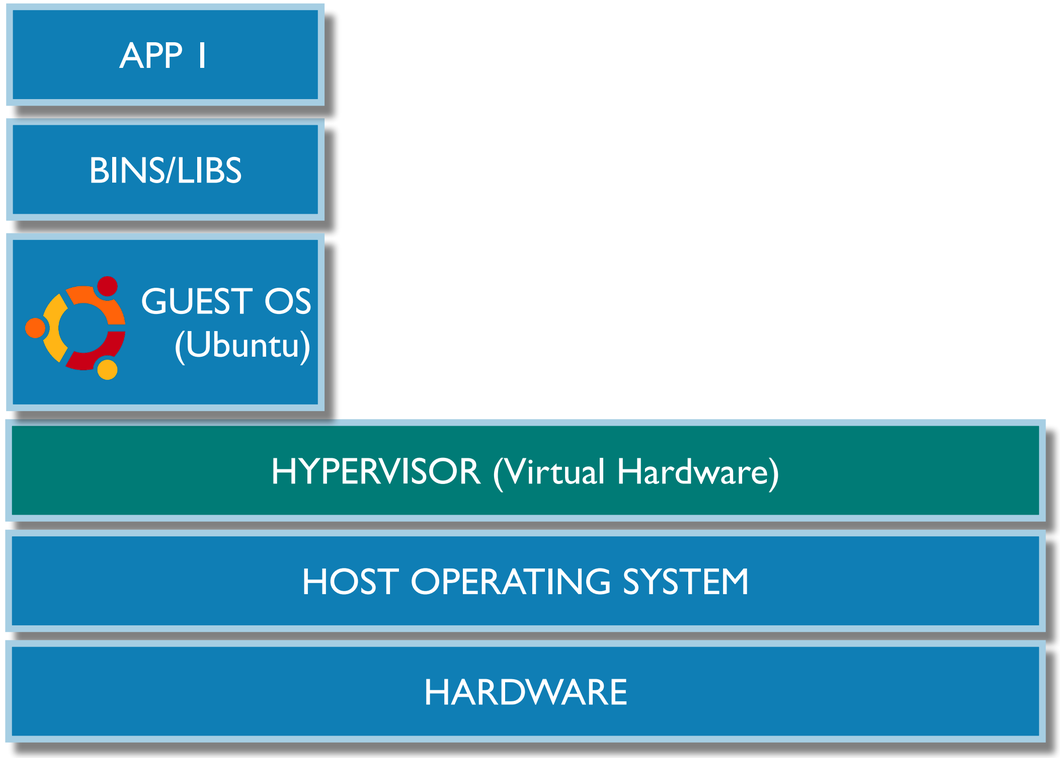
Virtual Machines (a.k.a. Virtual Hardware)

- **Virtual Machines** (VM) are created by emulating computer hardware in software
- The emulation is provided by software called a **Hypervisor**
- Each **Guest OS** thinks it's talking to dedicated computer hardware but it is really talking to the hypervisor that is sharing a much larger system



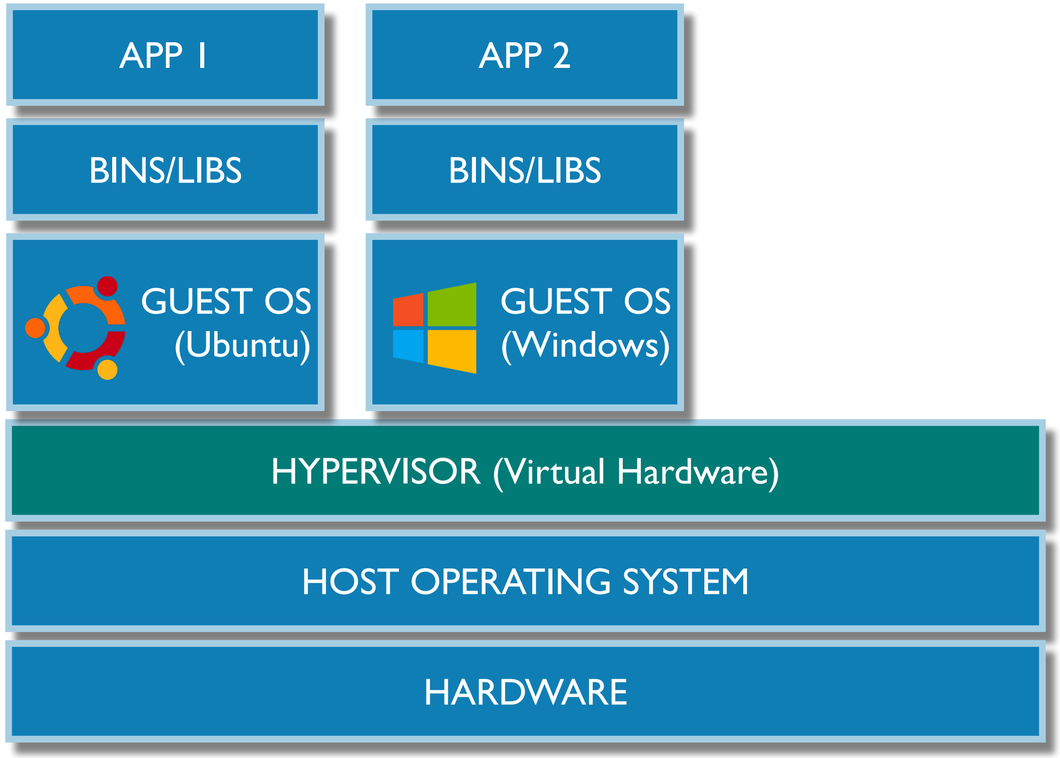
Virtual Machines (a.k.a. Virtual Hardware)

- **Virtual Machines** (VM) are created by emulating computer hardware in software
- The emulation is provided by software called a **Hypervisor**
- Each **Guest OS** thinks it's talking to dedicated computer hardware but it is really talking to the hypervisor that is sharing a much larger system



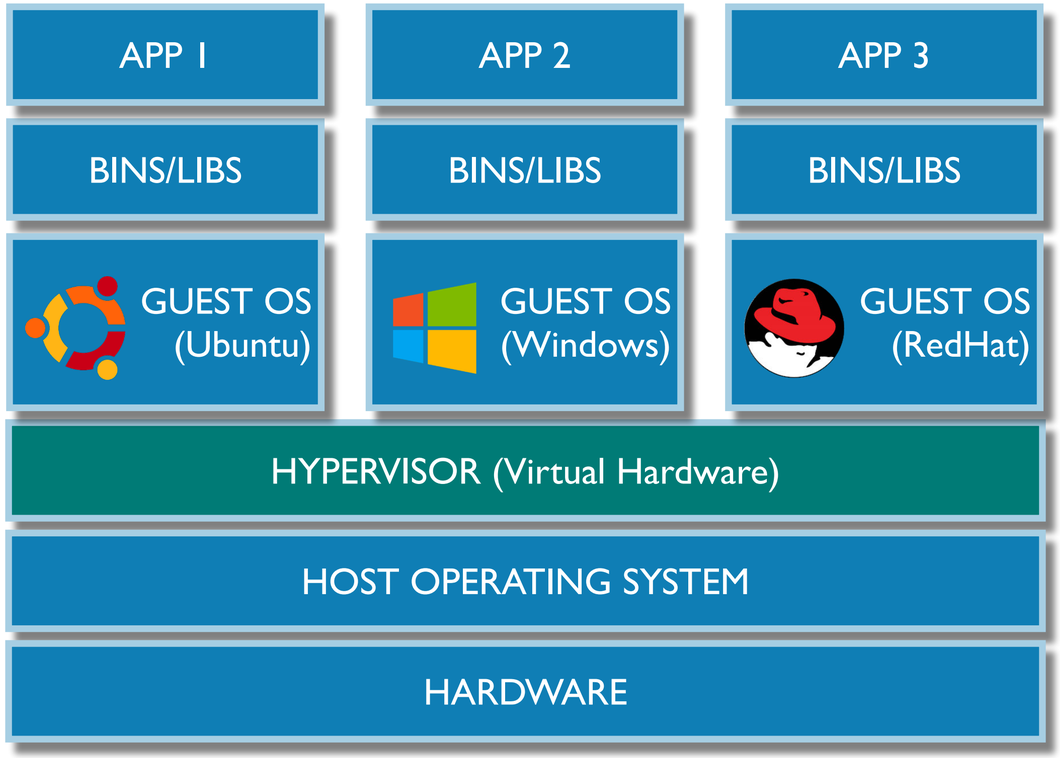
Virtual Machines (a.k.a. Virtual Hardware)

- **Virtual Machines** (VM) are created by emulating computer hardware in software
- The emulation is provided by software called a **Hypervisor**
- Each **Guest OS** thinks it's talking to dedicated computer hardware but it is really talking to the hypervisor that is sharing a much larger system



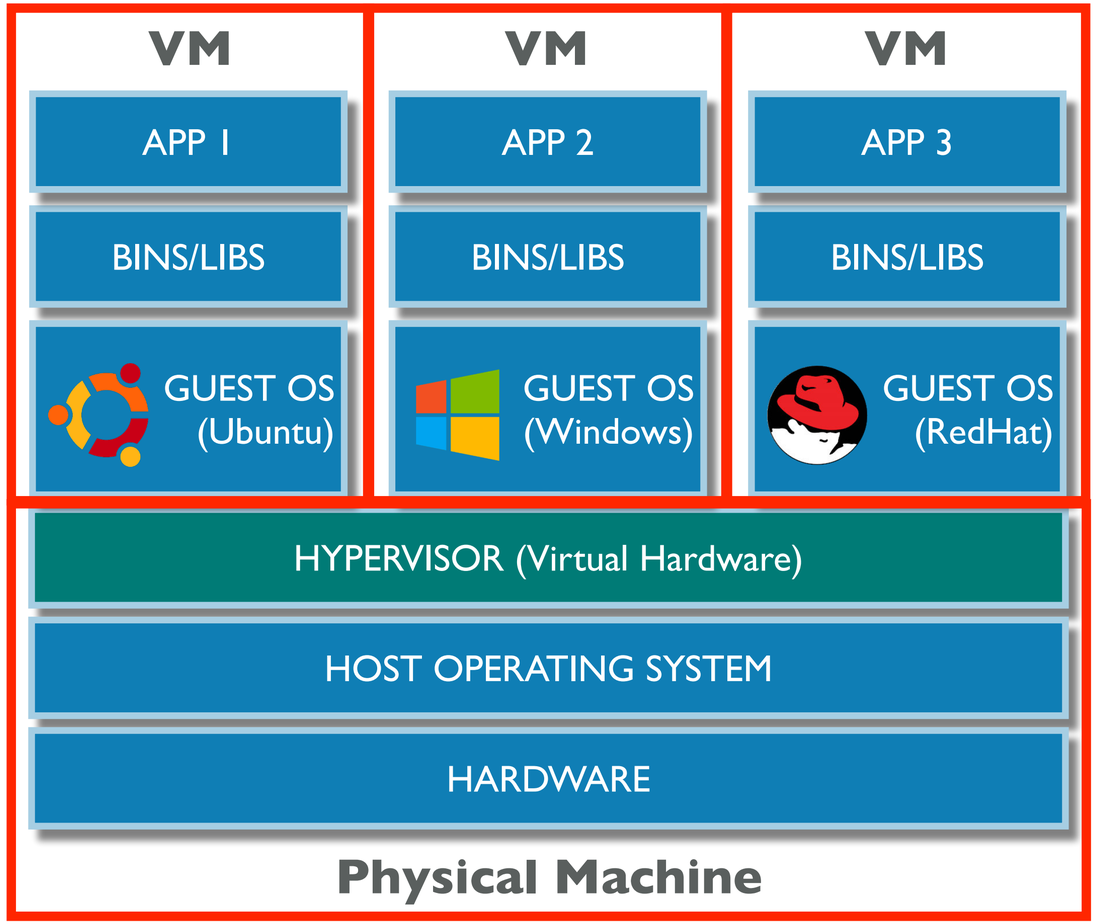
Virtual Machines (a.k.a. Virtual Hardware)

- **Virtual Machines** (VM) are created by emulating computer hardware in software
- The emulation is provided by software called a **Hypervisor**
- Each **Guest OS** thinks it's talking to dedicated computer hardware but it is really talking to the hypervisor that is sharing a much larger system



Virtual Machines (a.k.a. Virtual Hardware)

- **Virtual Machines** (VM) are created by emulating computer hardware in software
- The emulation is provided by software called a **Hypervisor**
- Each **Guest OS** thinks it's talking to dedicated computer hardware but it is really talking to the hypervisor that is sharing a much larger system



What is VirtualBox?

- VirtualBox is a free Hypervisor that runs on OS X, Windows, and Linux
- Similar to VMware Workstation on a PC, or VMware Fusions and Parallels Desktop on a Mac
- Allows you to run your code in a Virtual Machine

What is Vagrant?

- Vagrant is a developers tool for creating lightweight, reproducible and portable virtual environments via command-line
- It supports VirtualBox, VMware, SoftLayer, Amazon AWS and Digital Ocean
- It supports configuration management utilities like Puppet, Chef, etc.
- It supports Docker natively which makes is easy to use Docker containers in VMs

Vagrant is the Puppet Master

- Vagrant controls VirtualBox so you don't have to
- Vagrant also installs software inside of the Virtual Machine
- All of this is controlled by a single file called: `Vagrantfile`
- You start it with one command: `vagrant up`
- This is known as "*Infrastructure as Code*"

Example Vagrantfile

```
Vagrant.configure(2) do |config|
  config.vm.box = "ubuntu/bionic64"

  config.vm.network "private_network", ip: "192.168.33.10"

  config.vm.provider "virtualbox" do |vb|
    vb.memory = "1024"
    vb.cpus = 2
  end

  config.vm.provision "shell", inline: <<-SHELL
    sudo apt-get update
    sudo apt-get install -y git python-pip python-dev
    sudo apt-get -y autoremove
  SHELL

end
```

What’s in This Repo?

- Python 3 starter code
- `Vagrantfile` with:
  - Python 3 environment
  - Redis database in a Docker container
  - Microk8s, Kubectl (<-- Kubernetes)
  - Various Ports (e.g., 8080, 5000) forwarded


Summit of Innovation
Sea of Technical Debt