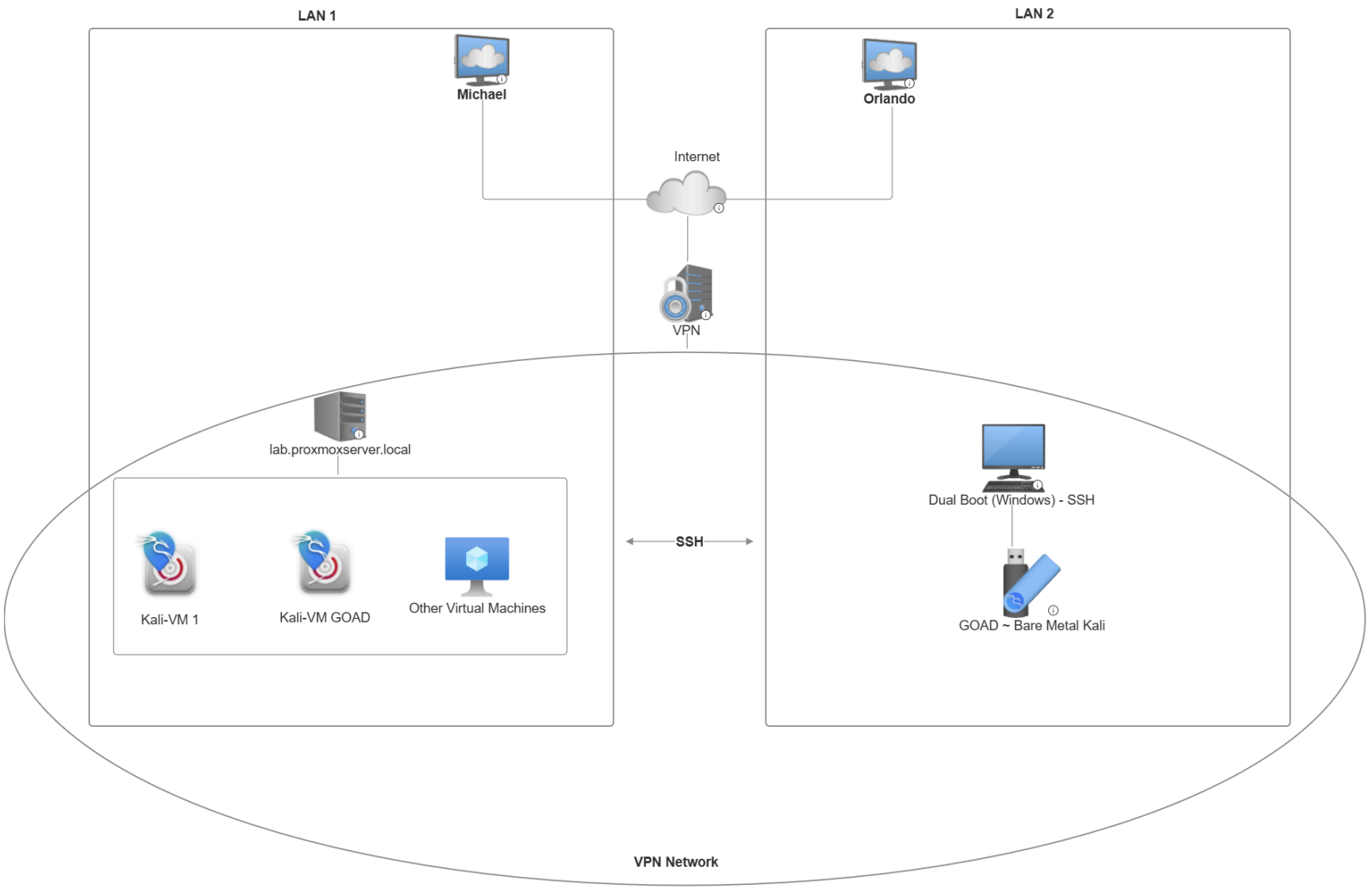
OSCP LAB ~ GOAD Environment

About us:

- Michael .N → https://www.linkedin.com/in/michaelnolk/
- Orlando .C → https://www.linkedin.com/in/orlando-companioni/



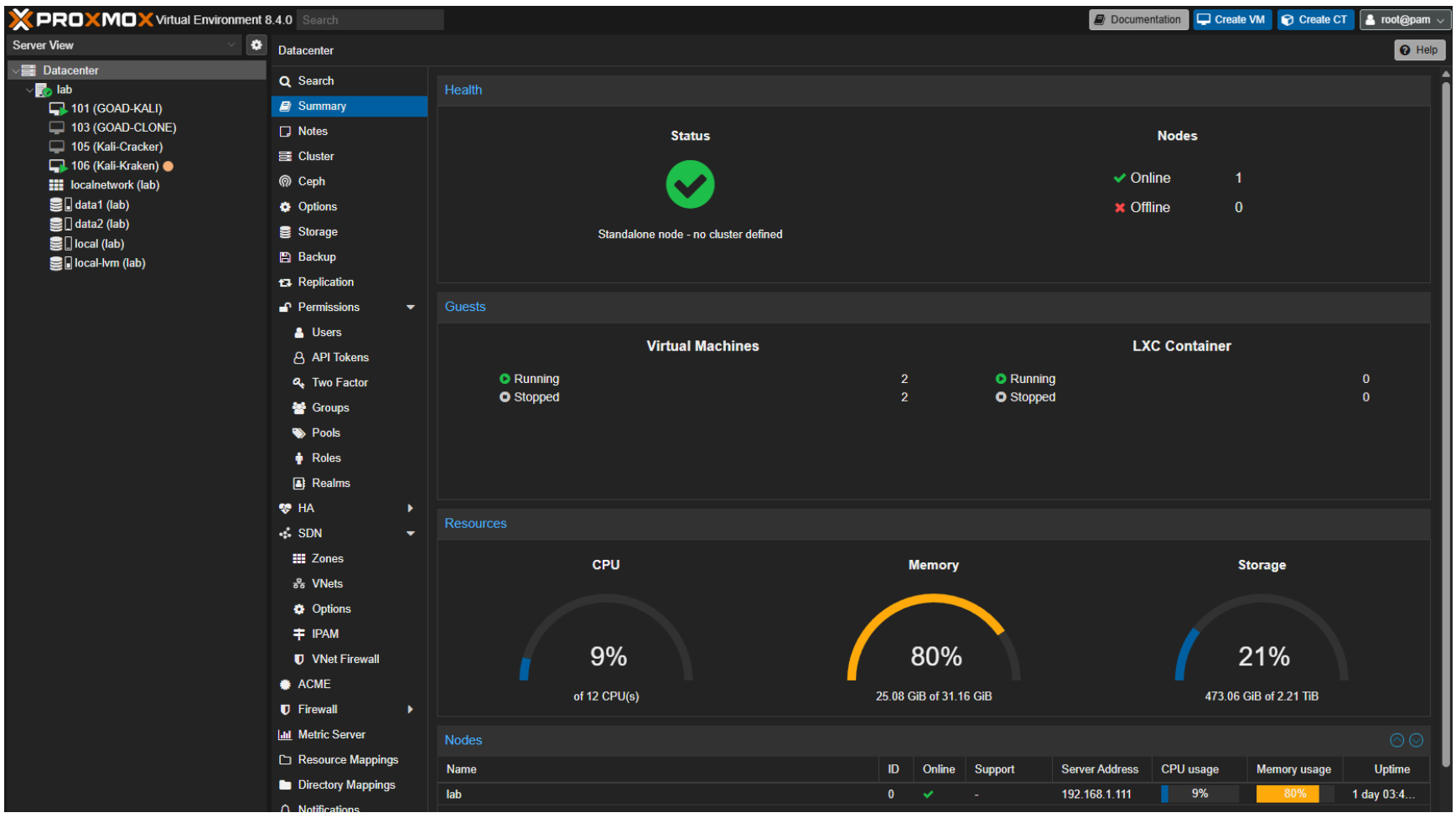
Network & Access Architecture

Our Lab environment required us to be able to access the two machines on which we're hosting on our local network. So we setup a VPN so we can connect remotely through a VPN tunnel to our home-based Proxmox server (`lab.proxmoxserver.local`). This setup is segmented into the following key components:

- **Proxmox Host**: Hosts all virtual machines and services.
- **VPN Server(Third Party)**: Provides encrypted remote access to the lab network.
- **Kali Linux VMs**: Deployed as the primary attack platforms (one per user).
- **GOAD Domain Lab**: Simulates a Windows Active Directory environment with domain controllers, clients, and vulnerable services.
- **Dual Boot System**: We also utilize a local dual-boot machine (Windows/Kali) for direct SSH access and flexibility.

Components of our Lab environment:

Proxmox Server Hardware Specs:

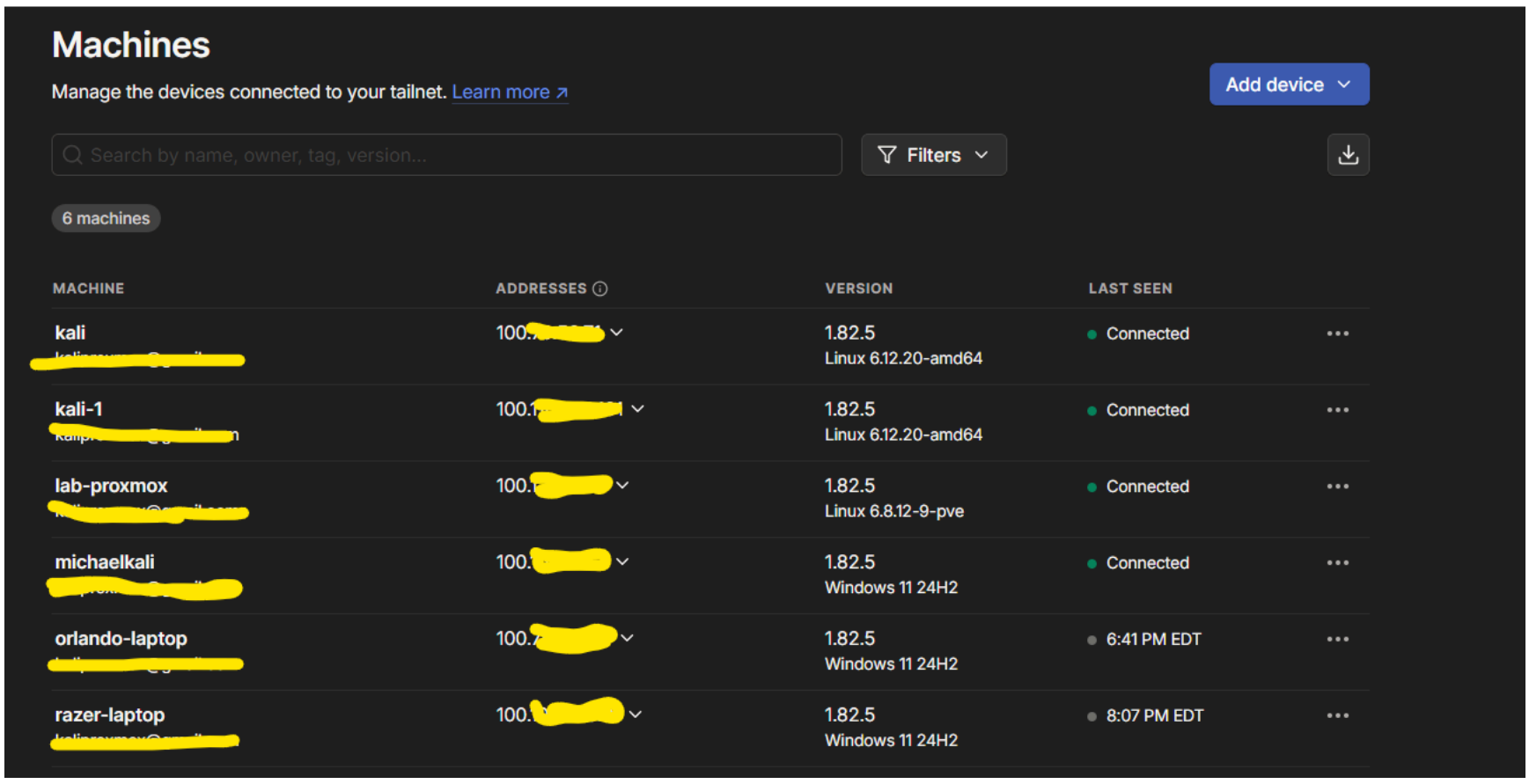


- Proxmox Dashboard.

```
root@lab:~# lshw -short
H/W path          Device           Class          Description
============================================================
                                   system         System Product Name (SKU)
/0                                 bus            PRIME B660M-A D4
/0/0                               memory         64KiB BIOS
/0/42                              memory         32GiB System Memory
/0/42/0                            memory         8GiB DIMM DDR4 Synchronous 3400 MHz (0
/0/42/1                            memory         8GiB DIMM DDR4 Synchronous 3400 MHz (0
/0/42/2                            memory         8GiB DIMM DDR4 Synchronous 3400 MHz (0
/0/42/3                            memory         8GiB DIMM DDR4 Synchronous 3400 MHz (0
/0/52                              memory         288KiB L1 cache
/0/53                              memory         192KiB L1 cache
/0/54                              memory         7680KiB L2 cache
/0/55                              memory         18MiB L3 cache
/0/56                              processor      12th Gen Intel(R) Core(TM) i5-12400F
/0/100                             bridge         Intel Corporation
/0/100/1                           bridge         12th Gen Core Processor PCI Express x1
/0/100/1/0                         display        GA104 [GeForce RTX 3060 Ti Lite Hash R
/0/100/1/0.1                       multimedia     GA104 High Definition Audio Controller
/0/100/6                           bridge         12th Gen Core Processor PCI Express x4
/0/100/6/0        /dev/nvme0       storage        Samsung SSD 980 PRO 500GB
```

- Hardware Specs of the Server.

VPN Setup:


Machines
Manage the devices connected to your tailnet. Learn more
Add device
Search by name, owner, tag, version...
Filters
6 machines
MACHINE
ADDRESSES
VERSION
LAST SEEN
kali
1.82.5
Linux 6.12.20-amd64
Connected
kali-1
1.82.5
Linux 6.12.20-amd64
Connected
lab-proxmox
1.82.5
Linux 6.8.12-9-pve
Connected
michaelkali
1.82.5
Windows 11 24H2
Connected
orlando-laptop
1.82.5
Windows 11 24H2
6:41 PM EDT
razer-laptop
1.82.5
Windows 11 24H2
8:07 PM EDT

Kali-Kraken:


kali@kali: ~
File Actions Edit View Help
hashcat (v6.2.6) starting in benchmark mode
Benchmarking uses hand-optimized kernel code by default.
You can use it in your cracking session by setting the -O option.
Note: Using optimized kernel code limits the maximum supported password length.
To disable the optimized kernel code in benchmark mode, use the -w option.
* Device #1: WARNING! Kernel exec timeout is not disabled.
This may cause "CL_OUT_OF_RESOURCES" or related errors.
To disable the timeout, see: https://hashcat.net/q/timeoutpatch
CUDA API (CUDA 12.2)
* Device #1: NVIDIA GeForce RTX 3060 Ti, 7829/7973 MB, 38MCU
OpenCL API (OpenCL 3.0 CUDA 12.2.149) - Platform #1 [NVIDIA Corporation]
* Device #2: NVIDIA GeForce RTX 3060 Ti, skipped
OpenCL API (OpenCL 3.0 PoCL 6.0+debian Linux, None+Asserts, RELOC, LLVM 18.1.8, SLEEF, DISTRO, PO
CL_DEBUG) - Platform #2 [The pocl project]
* Device #3: cpu-penryn-QEMU Virtual CPU version 2.5+, skipped
Benchmark relevant options:
* --backend-devices=1
* --optimized-kernel-enable
* Hash-Mode 5600 (NetNTLMv2)
Speed.#1.....: 2459.1 MH/s (64.46ms) @ Accel:8 Loops:1024 Thr:512 Vec:1

```
┌──(kali㉿kali)-[~]
└─$ ssh kraken2@100.118.233.101
The authenticity of host '100.118.233.101 (100.118.233.101)' can't be established.
ED25519 key fingerprint is SHA256:nPpZpv7VM/N9u5TXQ0dreeXI67w+pY0zMZd9Han2tWg.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '100.118.233.101' (ED25519) to the list of known hosts.
kraken2@100.118.233.101's password:
Permission denied, please try again.
kraken2@100.118.233.101's password:
Linux kali 6.12.20-amd64 #1 SMP PREEMPT_DYNAMIC Kali 6.12.20-1kali1 (2025-03-26) x86_64

The programs included with the Kali GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Kali GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
```

- We can ssh into it as long as we're connected to our VPN

Bare Metal Kali:

```
┌──(kali㉿kali)-[~]
└─$ sudo lshw -short
[sudo] password for kali:
H/W path             Device          Class          Description
=========================================================
                                     system         OMEN by HP Gaming Laptop 16-wf1xxx (9W3E3UA#ABL)
/0                                   bus            8C77
/0/0                                 memory         128KiB BIOS
/0/4                                 processor      Intel(R) Core(TM) i7-14700HX
/0/4/a                               memory         768KiB L1 cache
/0/4/b                               memory         12MiB L2 cache
/0/4/c                               memory         33MiB L3 cache
/0/5                                 memory         384KiB L1 cache
/0/6                                 memory         256KiB L1 cache
/0/7                                 memory         16MiB L2 cache
/0/8                                 memory         33MiB L3 cache
/0/9                                 memory         384KiB L1 cache
/0/2b                                memory         32GiB System Memory
/0/2b/0                              memory         16GiB SODIMM Synchronous 5600 MHz (0.2 ns)
/0/2b/1                              memory         16GiB SODIMM Synchronous 5600 MHz (0.2 ns)
/0/100                               bridge         Intel Corporation
/0/100/1                             bridge         Raptor Lake PCI Express 5.0 Graphics Port (PEG010)
/0/100/1/0           /dev/fb1        display        AD106M [GeForce RTX 4070 Max-Q / Mobile]
/0/100/1/0.1         card0           multimedia     AD106M High Definition Audio Controller
/0/100/1/0.1/0       input31         input          HDA NVidia HDMI/DP,pcm=3
/0/100/1/0.1/1       input32         input          HDA NVidia HDMI/DP,pcm=7
/0/100/1/0.1/2       input33         input          HDA NVidia HDMI/DP,pcm=8
/0/100/1/0.1/3       input34         input          HDA NVidia HDMI/DP,pcm=9
/0/100/2             /dev/fb0        display        Raptor Lake-S UHD Graphics
/0/100/4                             generic        Raptor Lake Dynamic Platform and Thermal Framework Pro
/0/100/8                             generic        GNA Scoring Accelerator module
/0/100/a                             generic        Raptor Lake Crashlog and Telemetry
/0/100/14                            bus            Raptor Lake USB 3.2 Gen 2×2 (20 Gb/s) XHCI Host Contro
/0/100/14/0          usb1            bus            xHCI Host Controller
/0/100/14/0/4                        bus            USB2.0 Hub
/0/100/14/0/4/1                      input          USB Receiver
/0/100/14/0/4/1/0    input26         input          Logitech M720 Triathlon
```

GOAD Setup:

Setting Up Your New Kali VM

- Since we created a fresh Kali Linux VM, we need to update and initialize the system's libraries and packages to ensure compatibility and stability.

```
sudo apt update
sudo apt upgrade
sudo apt full-upgrade
```

Download the Linux Headers

- **Install Generic Kernel & Headers** (for VirtualBox DKMS modules):

```
sudo apt install linux-image-amd64 linux-headers-amd64
sudo reboot
```

- This Step is essential as if you do not do this step correctly, the rest of the setup will not be possible, since you cannot install your hypervisor.

Install VirtualBox:

- For the GOAD lab the Hypervisor we decided to use in our Kali VM was VirtualBox since that seemed to be the most used and easiest to set up.
  - To install:

```
sudo apt install virtualbox
```

If you get Errors in the installation:

- Go back and try to see what error is being caused; it most likely is something with the Linux headers. Fix the issues, then run this command:

```
sudo apt install --reinstall virtualbox-dkms
sudo dpkg-reconfigure virtualbox-dkms
sudo modprobe vboxdrv
```

Check if VirtualBox works:

```
virtualbox
```

- now we can move onto installing vagrant, and the vagrant plugins.

Vagrant/Docker/pyenv Install

- In this step, we combined three different steps into one since they're very simple and straightforward.

```
sudo apt install vagrant
sudo apt install docker.io
sudo apt install python3-venv   # Or `python<version>-venv` as needed
```

NOTE: We recommend logging into Docker before running the GOAD lab You can do this by running `docker login`

- Next install the vagrant plugins:

```
vagrant plugin install vagrant-reload vagrant-vbguest winrm winrm-fs winrm-elevated
```

Install Ruby gems:

```
sudo gem install winrm winrm-fs winrm-elevated
```

GOAD INSTALL:

- These are the steps we took to install GOAD.

Clone the GOAD Repo:

- Clone the GOAD repo from GitHub

```
git clone https://github.com/Orange-Cyberdefense/GOAD.git
cd GOAD
```

- Since we wanted our provisioning method to be `docker` because we had many issues before with using local provisioning and VM provisioning.

Run goad_docker:

```
sudo ./goad_docker.sh
```

- We recommend running this command with sudo because throughout the install, it will ask for the root password, so to prevent that and to make it a more automated process, use sudo
    - Set the provider to our hypervisor choice by default, it is VMware.

```
set_provider virtualbox
check #check requirments
install
```

- If you don't want the interactive shell to install, you can do it with one command as such, but this caused us to have some issues when we first tried:

```
./goad.sh -t start -p virtualbox -l GOAD -m docker
```

GOAD-DC01: