

All rights, trademarks, and intellectual property belong to their respective rights holders.

Architecting Virtual Machine Labs

Written by: Tony Robinson

Dedication: This is dedicated to IT and computer security people, newbie and veteran alike. I wrote this guide for two reasons: 1) “Published Author” sounds like a great title to have (plus I can cross it off my bucket list) 2) Knowledge sharing. One day, you’re going to inherit the mess of infrastructure and terrible security practices we leave behind. This guide may not have all the answers, but hopefully it will give you a nice head start. I just hope that if this guide helped you in some way, that maybe you’ll repay the kindness to the generation that follows you, offering to guide them as well.

I would like to thank my wife for her patience and unending love, my family for inspiring me to do better, my employer for saying ‘GO WRITE YOU A BOOK’, and finally, several friends who offered to step in and provide feedback and editing when I asked for help. I threw walls of text into a google doc, but you made it awesome.

Table of Contents

[All rights, trademarks, and intellectual property belong to their respective rights holders.](#)

[Architecting Virtual Machine Labs](#)

[Table of Contents](#)

[Purpose of this guide](#)

[A Note About Software Versions](#)

[Prerequisite Knowledge](#)

[Hypervisor and Hardware Considerations](#)

[Introduction to Virtualization](#)

[Introduction to Hypervisors](#)

[What is a Hypervisor?](#)

[Baremetal Hypervisors](#)

[Hosted Hypervisors](#)

[Hardware Considerations](#)

[RAM as a Performance Factor](#)

[Disk I/O as a Performance Factor](#)

[What is seek time?](#)
[CPU Cores and Features as a performance Factor](#)
[Performance is a Vicious Cycle](#)
[Understanding Virtual Networks - Hosted vs. Baremetal Hypervisor Networking](#)
[Hosted Hypervisor Networking - Host-Only, Bridged, and NAT Network Segments](#)
[Bridged Networking](#)
[NAT Networking](#)
[Host-Only Networking](#)
[Virtual Network Adapters and You](#)
[Baremetal Hypervisor Networking - Virtual Switches](#)

[Lab Overview](#)
[Design](#)
[Lab Network Description](#)
[Bridged Network](#)
[Management Network](#)
[IPS 1 and IPS 2 Networks](#)
[AFPACKET Bridging between IPS 1 and IPS 2](#)
[Why All The Trouble?](#)
[VMs, Resource Allocations, and Minimum Hardware Requirements](#)

[Hypervisor Guides](#)
[Setup - Client Hyper-V](#)
[Installation](#)
[Hypervisor Preferences](#)
[Server Settings](#)
[User Settings](#)
[Creating the Virtual Switches](#)
[Creating the First VM, pfSense](#)
[Adding a New VM](#)
[Initial VM Settings](#)
[Installing pfSense](#)
[Final VM Settings](#)
[Network Configuration](#)
[Web Configurator - Initial Setup](#)
[Making Checkpoints](#)
[pfSense Summary](#)
[What's Next?](#)

[Your Turn](#)
[Kali Linux VM](#)
[Siem VM](#)
[IPS VM](#)

[Metasploitable 2](#)

[Port Mirroring and MAC spoofing](#)

[So... What Now?](#)

[Setup - VirtualBox](#)

[Installation](#)

[Hypervisor Preferences](#)

[Creating the first VM, pfSense](#)

[Adding a New VM](#)

[Initial VM Settings](#)

[Installing pfSense](#)

[Final VM Settings](#)

[Network Configuration](#)

[Web Configurator - Initial Setup](#)

[Take a Snapshot](#)

[pfSense Summary](#)

[What's Next?](#)

[Your turn](#)

[Kali Linux VM](#)

[SIEM VM](#)

[IPS VM](#)

[Promiscuous Mode](#)

[Metasploitable 2](#)

[So... What Now?](#)

[Setup - VMware Fusion Pro](#)

[Installation](#)

[Hypervisor Preferences](#)

[Creating the First VM, pfSense](#)

[Adding a New VM](#)

[Installing pfSense](#)

[Final VM Settings](#)

[Network Configuration](#)

[Web Configurator - Initial Setup](#)

[Take a Snapshot](#)

[pfSense Summary](#)

[What's Next?](#)

[Your Turn](#)

[Kali Linux VM](#)

[Siem VM](#)

[IPS VM](#)

[Metasploitable 2](#)

So... What Now?

Setup - VMware Workstation Pro

[Installation](#)

[Hypervisor Preferences](#)

[Virtual Networks](#)

Creating the First VM, pfSense

[Adding a New VM](#)

[Installing pfSense](#)

[Final VM Settings](#)

[Network Configuration](#)

[Web Configurator - Initial Setup](#)

[Take a Snapshot](#)

[pfSense Summary](#)

[What's Next?](#)

Your Turn

[Kali Linux VM](#)

[Siem VM](#)

[IPS VM](#)

[Metasploitable 2](#)

So... What Now?

Setup - VMware VSphere Hypervisor(ESXi)

[Installation](#)

[Accessing ESXi](#)

Hypervisor Setup

[Licensing](#)

[Resolving Some Interface Bugs](#)

[Networking and Virtual Switches](#)

[Creating Virtual Switches](#)

[Port Groups](#)

[Adding Port Groups via the ESX Web Interface](#)

[Using the Windows vSphere Client to work around ESXi Web Interface Bugs](#)

Creating the First VM, pfSense

[Adding a New VM](#)

[Installing pfSense](#)

[Final VM Settings](#)

[Network Configuration](#)

[Web Configurator - Initial Setup](#)

[Take a Snapshot](#)

[pfSense Summary](#)

[What's Next?](#)

Your Turn

[Kali Linux VM](#)

[Siem VM](#)

[IPS VM](#)

[Metasploitable 2](#)

So... What Now?

pfSense Firewall Rules and Network Services Guide

Network Configuration - Segmentation and Firewall Config

[Firewall Rules for the WAN Network](#)

[Firewall Rules for the Management Network](#)

[Firewall Rules for the IPS Network](#)

Network Configuration - Core Network Services

[NTP](#)

[DHCP](#)

[DNS Resolver](#)

[Squid Proxy](#)

Defense in Depth for Windows Hosted Hypervisors

[Unbinding Network Protocols on Windows Virtual Adapters](#)

[Using Windows Firewall to Limit Exposure of Windows Hypervisor Hosts](#)

Automated Patching for Linux Lab VMs

[updater.sh](#)

Remote Lab Management

[Windows Remote Access](#)

[Persistent Static Routes](#)

[Windows SSH and SCP Software](#)

[Generating an SSH key in Windows using PuTTYgen](#)

[Using MRemoteNG - Connection Files](#)

[Using MRemoteNG - PuTTY Saved Sessions](#)

[Enabling Key-Based Authentication in Linux/Unix systems](#)

[Key Copy Method 1: echo append to authorized_keys](#)

[Key Copy Method 2: using vi](#)

[Key Copy Method 3: SCP](#)

[Making sure it worked](#)

[How to use Key-Based Authentication with WinSCP](#)

Remote Access for Linux and OSX

[Static Routes in Linux and OSX](#)

[Adding Routes to Linux with the ip Command](#)

[Adding Routes to OSX/BSD with the route command](#)

[Making Static Routes Persistent](#)

[Linux and BSD Route Persistence via /etc/rc.local](#)
[OSX Route Persistence on Hosted Hypervisors](#)
 [flightcheck.sh](#)
[OSX route persistence for Baremetal Hypervisors](#)
 [flightcheckBM.sh](#)
[The ssh and scp terminal Applications](#)
[iTerm2 and Terminator](#)
[Generating ssh keys using ssh-keygen](#)
[The alias Command](#)
[Enabling Key-Based Authentication in Unix/Linux Systems](#)
 [Key Copy Method 1: echo append to authorized_keys](#)
 [Key Copy Method 2: using vi](#)
 [Key Copy Method 3: SCP](#)
 [Making Sure it worked](#)
 [Using key-based authentication with the SCP command](#)
[How to Enable SSH on Kali Linux](#)
[Enabling, and securing root SSH](#)
 [Adding your SSH public key to root's authorized_keys file](#)
 [Disabling password authentication entirely via sshd_config](#)
[Network Design Factors When Working with Baremetal Hypervisors](#)
 [Prereqs](#)
 [Creating static routes](#)
 [Creating Firewall Rules](#)
 [Dealing with DHCP](#)
 [Jump Boxing](#)
 [Using a Raspberry Pi as a Jump Box](#)
 [Installing the Raspian Image to your Raspberry Pi](#)
 [Configuring Raspian](#)
 [Creating a Jump Box VM](#)
 [Other Physical Jump Boxes](#)
 [Preparing Your Jump Box for Service](#)
 [Configuring Static DHCP Allocations](#)
 [Enabling Key-Based Authentication for your Jump Box](#)
 [Windows](#)
 [Linux/OSX/BSD](#)
 [Adding Static Routes to your Jump Box](#)
 [Adding Firewall Rules and SSH tunnels to allow access to the VM lab networks](#)
 [I Can Still Access the pfSense WebConfigurator with my Management Workstation](#)
 [I Have Lost Access to the pfSense WebConfigurator UI](#)
 [TCP Forwarding and You](#)

[Windows SSH Tunnels](#)

[Linux/BSD/OSX SSH Tunnels](#)

[Testing your Dynamic Tunnels with FoxyProxy](#)

[Testing Your Forward Tunnels](#)

[Windows](#)

[Linux/OSX/BSD](#)

[What? How?](#)

[Closing Note on Jump Boxing](#)

[Key-Based Authentication](#)

[IDS/IPS Installation](#)

[Installing and configuring Snort \(via Autosnort\)](#)

[Installing and configuring Suricata \(via Autosuricata\)](#)

[Testing your IPS Bridge](#)

[Splunk Installation](#)

[Initial setup \(server installation\)](#)

[\(Optional\) Requesting and Implementing a Splunk Dev License](#)

[Universal Forwarder Setup](#)

[Splunk TA for Suricata](#)

[Hurricane Labs Add-On for Unified2](#)

[Starting The Forwarder + Persistence](#)

[Testing Splunk and the Universal Forwarder](#)

[Generating The Test Battery](#)

[Verifying Results with Snort](#)

[Verifying Results with Suricata](#)

[In Your Own Image](#)

[Visions of What Might Be](#)

[Malware Analysis Lab](#)

[Penetration Testing Lab](#)

[IT/OPs Lab](#)

[Summary](#)

[What Have We Learned Today?](#)

[Epilogue: We Need You \(Now More than Ever\)](#)

Purpose of this guide

This guide is designed to teach you about virtualization and how to build out the virtual machine lab environment that is easy to maintain, portable, relatively well secured, and flexible enough to accommodate IT and security students that need an environment to practice their trade. The goal is to teach you how to build the baseline network and get familiar with using a hypervisor of your choice. The initial network and VM (Virtual Machine) design we will be working on together can easily be expanded upon, or swapped out to support various roles, such as:

- Testing and/or developing new systems administration tools
- Learning the ropes for offensive security tools for red team
- Practicing with detection and response tools for blue team
- Providing a safe, secure environment to perform reverse engineering, malware analysis and/or exploit development with reasonably good security protections in place

This guide is not meant to be read from front to back. If you do this, you are going to get really bored, and notice a lot of repetition. Think of this book as a “Choose your own adventure” novel; this book covers how to produce a robust virtual machine lab environment across five different hypervisors. Unless you’re crazy, a VM enthusiast, or a researcher (or some combination), the likelihood that you will want or need to read all five hypervisor setup guides is going to be pretty low. Keeping that in mind, here are my recommendations:

Read all the chapters up to, and including the “Hypervisor Guides” chapter in order to develop a better understanding of the skills you’ll want and need to create your own lab, better understand how virtualization works in general, hardware recommendations, and finally, understand what you are building, before you pick a hypervisor and actually start building your lab environment. There are then five chapters detailing how to perform initial setup and configuration of five unique hypervisors:

- Oracle VirtualBox
- Microsoft Client Hyper-V
- VMware Workstation Pro
- VMware Fusion Pro
- VMware ESXi

These chapters instruct you on how to acquire and install the hypervisor of your choosing, configure the hypervisor and VMs to support the virtual machine lab environment I will teach you how to build, and perform initial installation and setup of those virtual machines. Choose a hypervisor that suits your budget and your goals, and follow the setup instructions.

After performing the setup and configuration tasks for the hypervisor of your choice, you are then meant to finish configuring the virtual machines in the lab environment, as well as the hypervisor host or management workstation you will be using to access your virtual machines. Each of the hypervisor setup guides has a section entitled “So... What now?” that will guide you

on recommendations on what tasks need to be done to finish making the lab functional (e.g. the IDS and Splunk installation chapters), as well as what supplemental chapters to consider reading (e.g. enabling remote access for the lab VMs, hardening hosted hypervisors on Windows, network design factors for baremetal hypervisors, etc.) for a much better experience when utilizing your VM lab.

A Note About Software Versions

Writing books for security and/or most IT disciplines is a daunting task. The moment you put ink to paper, the information contained in the book deprecates. You see this a lot with textbooks where there are multiple revisions that need to be written to discuss updates in the material.

I'll make mention of what software version for both hypervisors and operating installation ISOs I used throughout the guides, but don't be obsessed over using the exact same version I used when I made these guides. These are merely the software versions I had available to me while writing this book. As a security practitioner, I always recommend updating your software when updates are available, and using the most current software version available. This includes your hypervisors and OS distributions.

If you're from the future and using future versions of hypervisors, and future versions of operating systems, you may notice that some configuration settings may or may not be in the exact place a screen capture I made said it was going to be in. Button colors and styles may have changed, radio buttons may now be checkboxes, etc. This is because UI (User Interface) developers may or may not have changed exactly where a given configuration setting is. This is just a fact of life when it comes to new software releases. Sometimes they do it because they can't leave well enough alone, or sometimes they just want to make the user experience (UX) better.

So if you're panicking because a given configuration setting has moved, or that a checkbox isn't in the location indicated by my screen captures or instructions, don't panic. This is the first rule of any IT related discipline. The second rule is that software changes. Sometimes arbitrarily, sometimes for the better. The third rule is to consult the documentation. Maybe the config option has migrated to a new menu location, or maybe it was integrated as part of another, related configuration setting. Consult the product patch notes, documentation included with the software, and/or online knowledgebase/forums for the product to find out where the configuration setting lives now. The goal of this book isn't to mindlessly instruct you to click here, open this menu, and check these boxes, it is also for you to understand WHAT the configuration settings that you are modifying do, and WHY I am telling you to modify them so that if and when you want to experiment, make changes, and add or remove features to your lab, you have the knowledge and proper understanding what the settings do as opposed to what buttons you need to press.

Prerequisite Knowledge

Here are some things you should have a basic to intermediate grasp of in order to get the most out of this guide:

- **TCP/IP Networking**
 - Understand what an IP address is as well as how to configure an IP address, default gateway and/or DNS servers in Windows as well as Linux/Unix-based Operating Systems. (e.g. network adapter properties in Windows, ifconfig, ip link, ip route, route, /etc/resolv.conf, etc.)
 - Basic understanding of RFC1918 addressing
 - Basic understanding of subnetting
 - Understand the Open Systems Interconnection (OSI) model and/or TCP/IP networking models
 - A basic understanding on how stateful firewalls operate, and on what criteria they typically block (e.g. IPv4/IPv6 addresses, transport protocol (tcp/udp/icmp) and port/icmp code (e.g. port 80/tcp, port 123/udp, icmp type 0))
 - Familiarity on ports and protocols (e.g. is it TCP or UDP) for common network services (e.g. FTP, HTTP/HTTPS, NTP, SSH, DNS, etc.)
- **Familiarity with Operating Systems and their Installation Procedures**
 - Familiarity with installing Linux/Unix/Windows operating systems is absolutely necessary. If you know what an ISO image is, and how to boot from one, this will help you progress a bit faster
 - Familiarity with the Unix/Linux/Windows command line (e.g. “shells”) is key; most of the lab is based on Linux and BSD systems that will NOT have a GUI installed. You’ll need to know how to navigate the shell to navigate the file system, edit configuration files, utilize CLI tools, and run shell scripts.
 - You should know how to perform various network troubleshooting commands from the command line on Linux/OSX/BSD systems, as well as Windows systems (e.g. ping, netstat, wget, curl, ipconfig, ifconfig, etc.)
 - You should be familiar with at least one command line text editor that Linux/OSX/BSD systems use (e.g. vim, nano, emacs, etc.), or if you insist on writing/editing your config files on Windows, then copying them to lab VMs, you should be familiar with the dos2unix command to prevent Linux systems from failing to parse config files properly
 - Familiarity with SSH and SCP for remotely managing and copying files to Linux/OSX/BSD systems will be extremely helpful
- **Virtualization**
 - If you are familiar with the basics of virtualization, it will make a lot of this guide much easier to understand.

There are numerous resources out there for learning the basics I’ve listed. If you understand most of the points listed above, you shouldn’t have too hard of a time following along. If you don’t... things might be difficult at first, but the only way they will get easier is if you struggle

along the way. If you are not against reading a good book or two, the publishing company No Starch Press produces a variety of high quality reference books to help you gain a better understanding of various subjects; of particular note are the books “The Linux Command Line”, “TCP/IP Guide”, “Network Know-How”, and “Practical Packet Analysis”.

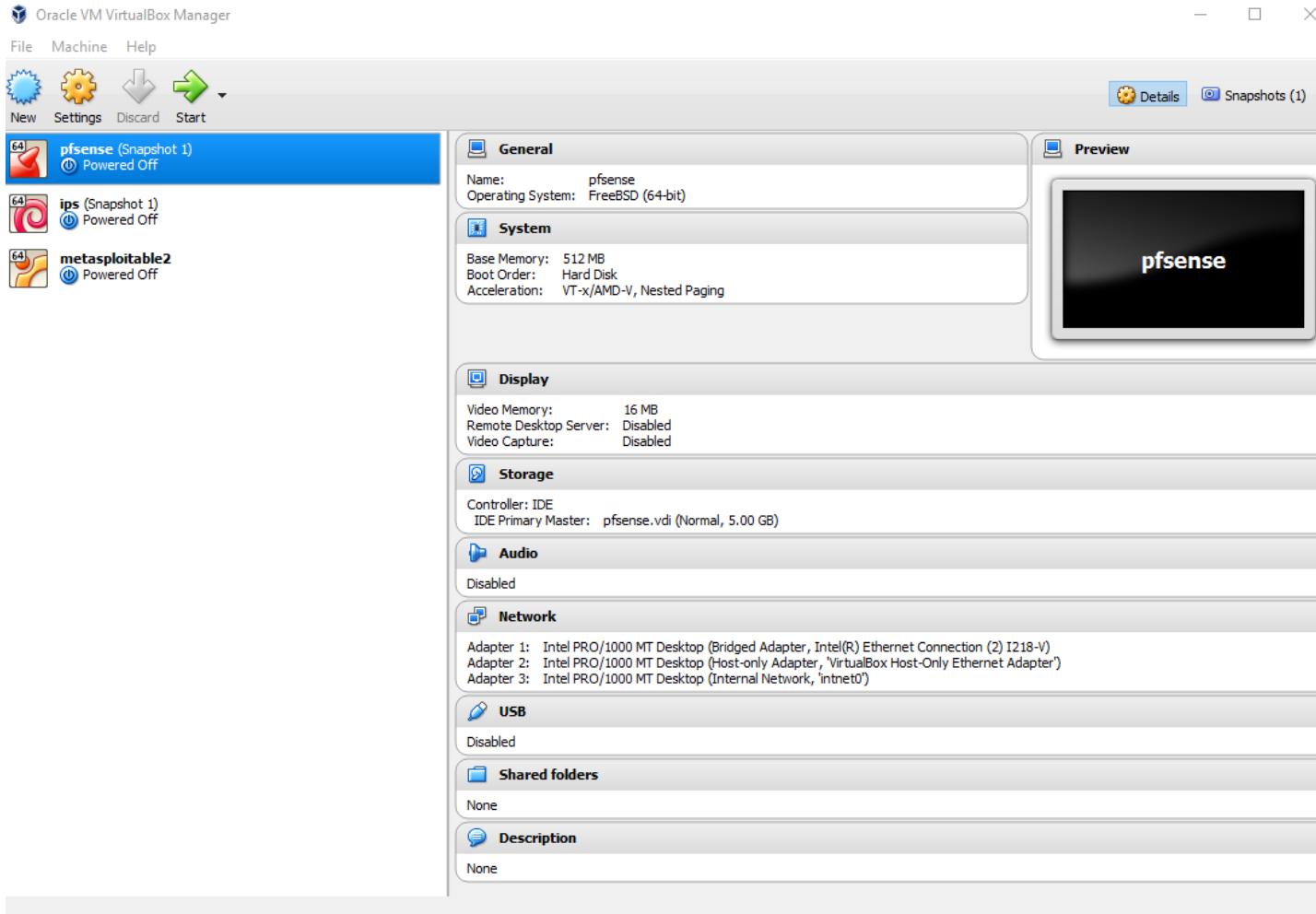
If you’re looking for free resources, the website cybrary.it has a great collection of free videos. Take a look at some of the videos produced for COMPTIA certifications, specifically the Linux+, A+ and Network+ coursework, as they have direct application to what we’re doing here, and will teach enough of the basics to allow you to progress a bit easier. Additionally, Zed Shaw has created a series of guides collectively called “Learn Programming The Hard Way”. He has a nice crash course introduction to the command line for various operating systems here: <https://learnpythonthehardway.org/book/appendixa.html>

Hypervisor and Hardware Considerations

Before you get started with the hypervisor setup guides, it helps to know some of the basics of virtualization. This is going to involve you learning about hardware resources hypervisors need, how VMs actually work, virtual networking, and finally, understand how your hypervisor choices will influence your lab’s network design. Before we dive into the hypervisor guides, I’m going to give you the crash course on what virtualization is, different types of hypervisors, how virtual networking works on different hypervisors, illustrate how the lab network we will be created, then instructing you on how to recreate the lab network and VMs on a variety of hypervisor platforms.

Introduction to Virtualization

Virtualization is the concept of taking one physical computer and splitting its resources into separate chunks/containers in order to host smaller, independent “virtual” systems using those reserved resources. For example, if you have a physical machine with 2 CPU cores, 4GB of RAM, and 40GB of disk space, you could, using special software called a hypervisor, allocate a portion of these resources to create a container to host a virtualized computer. You could install any OS you have licensing for and have it using that chunk of disk, CPU and memory you set aside. These virtual machines, for all intents and purposes, are independent computers. You can host many virtual machines on a single physical computer, with the only limitations being the amount of resources the host computer has, and how many resources the virtualized computers (known as virtual machines, or VMs) require to run.



In the screen capture above for example, there are three VMs listed. The VM that is highlighted is named “pfSense”. In this case, I set aside 512MB of ram, 5GB of hard drive space, and 1 virtual CPU to install and run the pfSense OS (Operating System) in a VM.

Virtualization became a big deal because it allows companies to run more services, and do more things with less physical hardware; not to mention it's easier for developers, researchers, and IT professionals to have a testing environment full of virtualized systems for testing compatibility, patch deployments, and a ton of other tasks that would have required a lot of physical hardware. Not only that, virtualization software supports “snapshots” or the ability to capture the status of a virtualized system at a certain point in time, and restore the virtual machine to that state at will. This means that test cases, malware analysis and several other scenarios that make a significant number of changes to the system's operating status can be undone in mere moments, restoring the VM to a known good configuration state with the click of a button.

Introduction to Hypervisors

Now that you know what a VM is, let's talk about hypervisors. There are a wide variety of them to choose from. Your first hurdle is choosing which option is right for you, and that depends on a number of different factors.

What is a Hypervisor?

A hypervisor is software that is used to create, manage, and run, virtual machines. Hypervisors are responsible for resource allocation (CPU, disk, RAM), network configuration, virtual hardware allocation, snapshot management, and controlling the current operating status of VMs that they are responsible for managing. There are two classes of hypervisors recognized today: baremetal hypervisors, and hosted hypervisors.

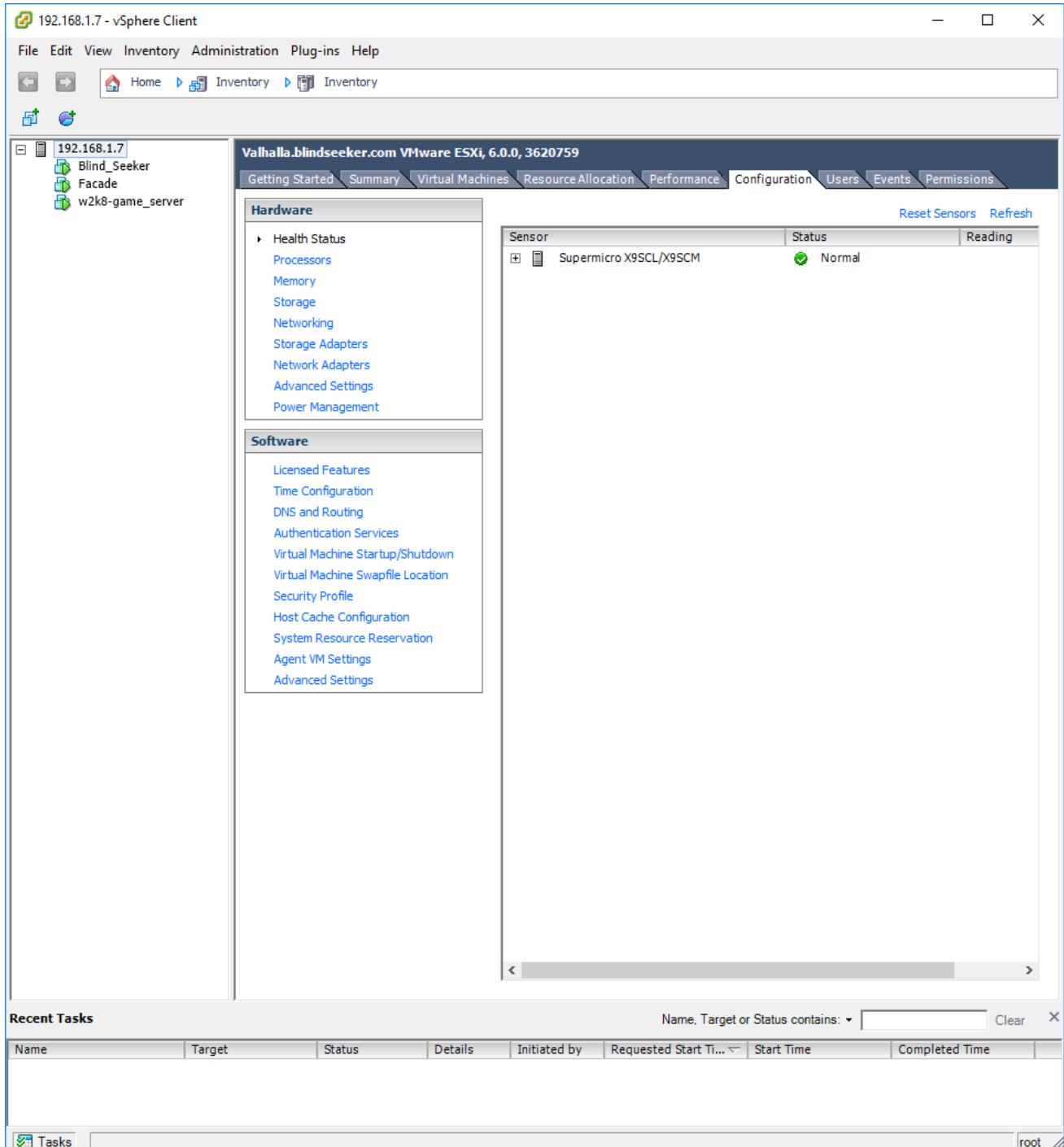
Baremetal Hypervisors

A baremetal hypervisor is installed directly on to server hardware (in IT/sysadmin circles, known as "big iron" or the "bare metal" - hence the name for this class of hypervisor). Baremetal Hypervisors usually feature a very lightweight operating system with minimal functionality in addition to their hypervisor functionality. In fact, in most cases, baremetal hypervisor installation can be done on an SD card or USB drive, leaving the internal hard drives and/or Solid State Drives (SSDs) 100% dedicated to the virtual machines.

Once the baremetal hypervisor is installed, booted, and given a network configuration, typically it is managed remotely over the network through a web interface, API, or through some sort of an application installed on a workstation you use to manage the hypervisor remotely. For sake of simplicity, throughout this guide, I will refer to this system as a management workstation. All the major configuration aspects for a baremetal hypervisor are done from the management console/web interface, using a management workstation.

While baremetal hypervisors tend to be very lean, they are packed with advanced features with more of a focus towards enterprise environments that require those features for a production environment where downtime means lost revenue. These are features such as fault tolerance, availability/failover solutions, advanced networking, etc. Since baremetal hypervisors are commonly used in enterprise networks all over the world, it would definitely benefit you to learn how to use them in lab environment, and make yourself familiar with how they operate. However, be aware that some baremetal hypervisors are EXTREMELY picky about what hardware they work with (VMware vSphere Hypervisor for example, is notorious for being extremely picky with what hardware it will detect and use). Even then, even if you can actually get the hypervisor to recognize your system's hardware, not all of your hardware's features may be supported. For example, integrated network ports on a system motherboard may not be supported, or a built-in RAID controller may not be recognized by the hypervisor. This is something you either have to deal with, find supported hardware, or find another baremetal hypervisor that isn't so picky about what hardware it supports. Popular baremetal hypervisors

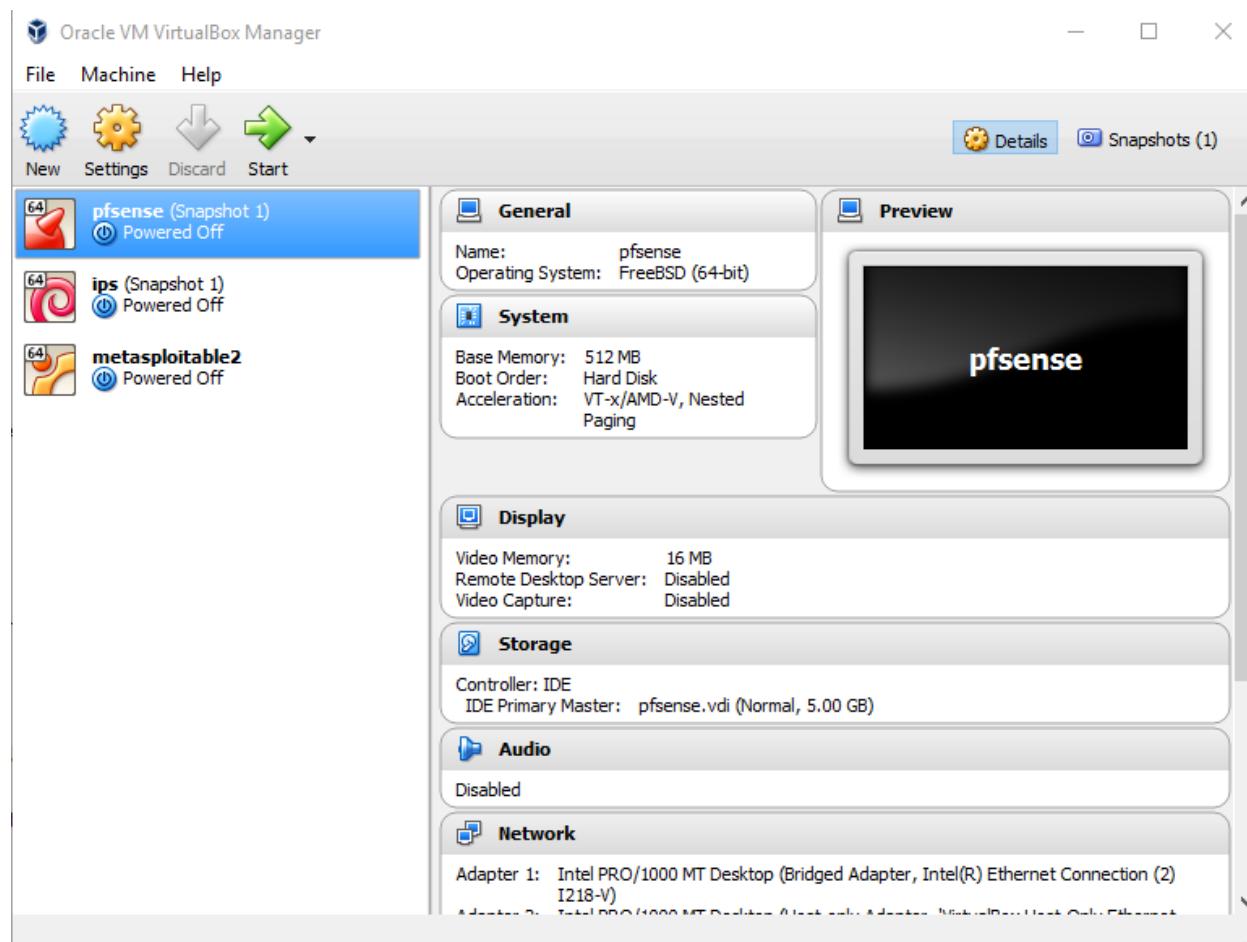
include Microsoft's Hyper-V (server edition), Citrix Xenserver, Proxmox, and VMware vSphere Hypervisor (also known as ESXi).



The illustration is VMware vSphere Client, which is used to manage ESXi baremetal hypervisors remotely. The newest versions of ESXi also feature a built-in web interface for system management as well.

Hosted Hypervisors

Hosted hypervisors differ from baremetal hypervisors in that they are an application that is installed on top of an already installed operating system, such as a Linux, OSX, BSD or Windows. Hosted hypervisors are applications that are “hosted” by the installed operating system, thus the name. Typically, some sort of a console is installed as a part of the hosted hypervisor to allow users logged on to the workstation to manage the configuration locally, though some hosted hypervisors have remote management features that can also be used. Popular hosted hypervisors include Oracle VirtualBox, VMware Workstation, VMware Fusion, and Microsoft Client Hyper-V.



The illustration depicted above is Oracle VirtualBox. VirtualBox is a free, multi-platform hosted hypervisor.

Hardware Considerations

What hardware do you have available to run a VM lab? Do you plan on using a spare desktop you had lying around? Your primary workstation? A laptop? An actual server? The hardware you have available to dedicate to your lab environment will determine how far you can take your lab, and what type of hypervisor you will be using (e.g. baremetal vs. hosted). In this section, we

will be discussing how RAM, Disk I/O, number of CPU cores, CPU featureset, and motherboard BIOS dictate how well your hypervisor and VMs will perform.

RAM as a Performance Factor

All operating systems, applications, utilities and VMs all require some amount of system memory to perform their tasks. Obviously, the more you have, the more applications and VMs with their own applications you can run concurrently. While there are some design factors in hypervisors that allow RAM that is not being used in one VM to be allocated to another VM when it is needed, you should never rely on these technologies, because it could result in overextending your RAM, which leads to swapping/paging to disk, which results in “disk thrashing”, which leaders to disk performance problems, further exacerbating your performance problems. The only solutions to a lack of RAM are to reduce the number of running applications and services to decrease the demand, or to increase the amount of RAM installed on your system and/or the amount of RAM allocated to the affected VMs.

Disk I/O as a Performance Factor

If at all possible, regardless of baremetal or hosted hypervisor, you should consider using SSDs where possible. SSDs are a relatively new storage medium. They have virtually no seek time and read/write performance that is orders of magnitude faster than your traditional magnetic platter based hard drives.

These performance factors are a big deal if you are doing multiple disk intensive things across your VMs at the same time. This means that the VMs will be fighting one another for I/O (input/output) access to read and write data from the same disk, which directly impacts the performance of your VMs, and if you’re running a hosted hypervisor, also affects the performance of the OS you are hosting the hypervisor on. This means that the more VMs you have on the same disk doing disk intensive tasks, the longer they all have to wait and contend for disk access to read and write their data to the hard drive.

If you don’t have an SSD or didn’t budget for one, don’t fret too much. Here are a few recommendations that might help you squeeze performance out of your disks and allow your VMs to coexist peacefully.

- If possible, consider installing more than one disk to running your VMs. Baremetal hypervisors benefit by being able to spread disk I/O across multiple disks. Hosted hypervisors benefit by being able to dedicate one of the disks for OS operations, and the other for VM storage and operations.
- If possible, consider putting your disks into a RAID1 (minimum of 2 disks) or RAID5 (minimum of 3 disks) array to protect against drive failure and possibly improve disk I/O.
- If you are using a baremetal hypervisor, consider installing it on a separate disk, USB drive, or SD card (if possible), in order to fully dedicate your disk drives solely to VM storage.

- If you are using a hosted hypervisor, minimize the amount of resource-intensive intensive tasks and applications you have running on the host OS (aside from the hypervisor of course).
- Do everything possible to avoid paging/swapping to disk on your VMs. This occurs when VMs need more RAM than the system has physically available; the VM will start using the disk like RAM. This should be avoided at all costs, even if you are using SSDs since paging/swapping causes intense wear and tear on all drive types platter or SSD (this is also known as disk thrashing). The key to preventing this is to ensure you have plenty of RAM installed, and that you have allocated enough RAM to your VMs. This means you have to monitor system performance on a regular basis.

What is seek time?

Seek time is the time delay associated with traditional spinning platter-based hard drives that is required for them to spin up and move tiny arms over portions of the disk platter (called read/write heads) to retrieve the data a system requires.

CPU Cores and Features as a performance Factor

All services and applications have processing and calculations that need to be done to achieve some sort of input or output, or manage some aspect of system operation. All of these applications and services need some slice of time on the CPU to have their calculations performed so that they perform their functions. Hypothetically, the more CPUs/cores you have available, the more tasks a system can perform at once. Whether that is multiple tasks for a single application, or multiple tasks for multiple processes the idea is supposed to be that more cores are better. Each VM has at least 1 virtual CPU (tied to the physical CPUs on your system) attached to it, with the ability to allocate more within the limits of how many physical CPUs/cores you have installed on your PC, laptop, or server.

Each VM has services and applications that need CPU time to run calculations to manage system services and input/output for running applications, just like any other physical system. The more intense the applications and services are in terms of CPU processing required, the more time and utilization those applications take to finish their calculations. This means that other applications and services need to wait for time on the CPU to perform their calculations. This in turn results in applications and services (even the entire OS in some cases) becoming unresponsive during extremely CPU intensive tasks.

Like with RAM utilization, the only solutions here are to reduce the number of running applications or services, deal with the factors that are causing them to take so much CPU time, or increase the number of CPUs/cores available.

In addition to the number of cores your system has available to handle CPU load, it is also important to confirm that both your CPU and/or system motherboard include support for virtualization features. Intel and AMD processors have virtualization extensions (Intel:VT-x AMD:AMD-V) that are built-in CPU features. Most of the time you can simply use your favorite

search engine, and search for your processor name and the results will return links to the CPU manufacturer that usually include a spec sheet page that confirms whether or not virtualization features are included in that particular CPU model. Some of the low-end CPUs do NOT include virtualization extensions in order to cut costs, so be aware of this!

In addition to checking the CPU specs, you also need to confirm that the motherboard BIOS supports virtualization. If you are using a prebuilt system from a large PC manufacturer (e.g. Dell, HP, etc.) then usually it's as easy as searching for your PC or laptop's model name on the manufacturer's support website to find and download a system manual, or spec sheet to confirm the features the system supports. However, if you're like me and you're into building your own PCs, try visiting the motherboard manufacturer's website, search for the model of motherboard you wish to use, then download and review the motherboard documentation to confirm that virtualization is a supported BIOS feature.

Performance is a Vicious Cycle

You should be seeing a pattern at this point. Don't have enough RAM? That leads to resource contention for memory and swapping/paging to disk. Don't have enough disk I/O? That leads resource contention for apps to read/write to and from disk. Don't have enough CPU/cores? That leads to resource contention for calculations. All of these can feed on one another and result in poor system stability and performance as a whole.

Keep an eye on CPU, disk and RAM performance metrics on a regular basis for your hosted and/or baremetal hypervisors, and adjust accordingly. Windows systems have task manager, while most Unix-like operating systems (That is, Linux, OSX, BSD) have several performance measuring utilities that can be ran from the command line such as top, free, iostat and iotop. Most baremetal hypervisors have their own performance graphs that measure the performance of the hypervisor as a whole, as well as the performance of individual VMs in terms of disk, RAM, and CPU utilization.

Understanding Virtual Networks - Hosted vs. Baremetal Hypervisor Networking

Lets discuss how networking is done on both hosted and baremetal hypervisors, because this is a key factor in how you access your lab VMs, and how our lab will be built on baremetal and hosted hypervisors.

Hosted Hypervisor Networking - Host-Only, Bridged, and NAT Network Segments

In most hosted hypervisors, virtual networks are typically divided into special network segments that all serve a different function. There are usually three types of network segments that VMs get connected to. "NAT" network segments, "Bridged" network segments, or "Host-Only"

network segments. Some hosted hypervisors allow you create additional “custom” network segments, but they usually fit into one of these three categories.

Bridged Networking

Bridged network segments are used to share the host system’s network card to directly connect to the same network as the host operating system. In this case, VMs that connect to the bridged network segment act as though they are directly connected to the same physical network as the host system. From a network perspective, they look exactly like any other system connected to your physical network. They can have their own IP address on the physical network to which they are attached, and would respond to network requests like any other system on that physical network.

NAT Networking

If you have an understanding on how Network Address Translation/Port Address Translation (NAT/PAT) works on a regular/home network for connecting multiple machines to the internet through a single publically routable IP address, then NAT network segments are exactly how you’d imagine them: The host operating system makes connections to external resources on behalf of the VMs connected to the NAT network. VMs connected to a NAT network share the host system’s IP address. All outbound traffic appears to be coming from the host system, and any services you set up on the VM for external access (through port forwarding) appear to be hosted on the host system as well.

For the rest of you who have no idea what I’m talking about, VMs that get added to a NAT network have their own internal network addressing scheme (IP address range, subnet mask, default gateway, etc.). The default gateway for this private network that does that NAT functions is usually a special virtual network card or IP address attached to the host operating system. Any traffic not destined for other hosts in the NAT network gets routed to this address. The host operating system then sends out the request on behalf of the system on the NAT network, using its network connection. The responses (or lack thereof) are then relayed back to the VM in the NAT network. NAT networks can be used to share the host operating system’s network connection and make it appear as though the connection requests are coming directly from the host operating system, instead of directly exposing the virtual machines to the network. Some hosted hypervisors also allow you to setup port forwarding so that connections to the host OS on certain ports get forwarded to a specific VM’s IP address in the NAT network.

In general, I tend to avoid using NAT network segments that most hypervisors provide. The only time I generally use NAT networks for network connectivity is if for some reason, bridged networking isn’t working properly. For instance, a bridged VM isn’t getting an IP address, or network connectivity is otherwise broken (maybe there is some sort of MAC address filtering, preventing network access, etc.). I also tend to avoid using NAT network segments because with most hosted hypervisors, the port forwarding functionality that NAT networks are supposed to provide is either buried in the user interface, or requires modifying configuration files buried somewhere in the hosted hypervisor’s installation files.

Host-Only Networking

Host-only networks are network segments that, by definition, do not get any sort of network connectivity to the outside world. VMs on these network segments can only communicate with hosts on the same host-only virtual network, and/or the hypervisor itself.

Virtual Network Adapters and You

The NAT and Host-Only networks allow the hypervisor's host OS to communicate with VMs inside of their respective networks through the use of virtual network adapters. The hypervisor creates a virtual network card and attaches it to the hypervisor host. To the host OS running the hypervisor, this virtual network card is just like any physical network card, and can be configured like one. Virtual network adapters allow your hypervisor host to connect to VMs in these networks, and interact with their network services.

Some hypervisors actually allow you to disable creation of the virtual network adapter for these network segments. In the case of a NAT network, this would create a network that uses the host system for external network connectivity, but the host would not be able to directly connect to VMs behind the NAT IP address; you'd have to configure port forwarding to access any of the network services for the VMs in that NAT network. In the case of host-only networks, this creates a "private" internal network in which the VMs could only connect to one another, but the host OS would not be able to connect (except through the hypervisor's console so that you could still install and configure VM on this network). Usually this is done to provide some degree of isolation between the hypervisor host and the VMs, and provide better network segmentation.

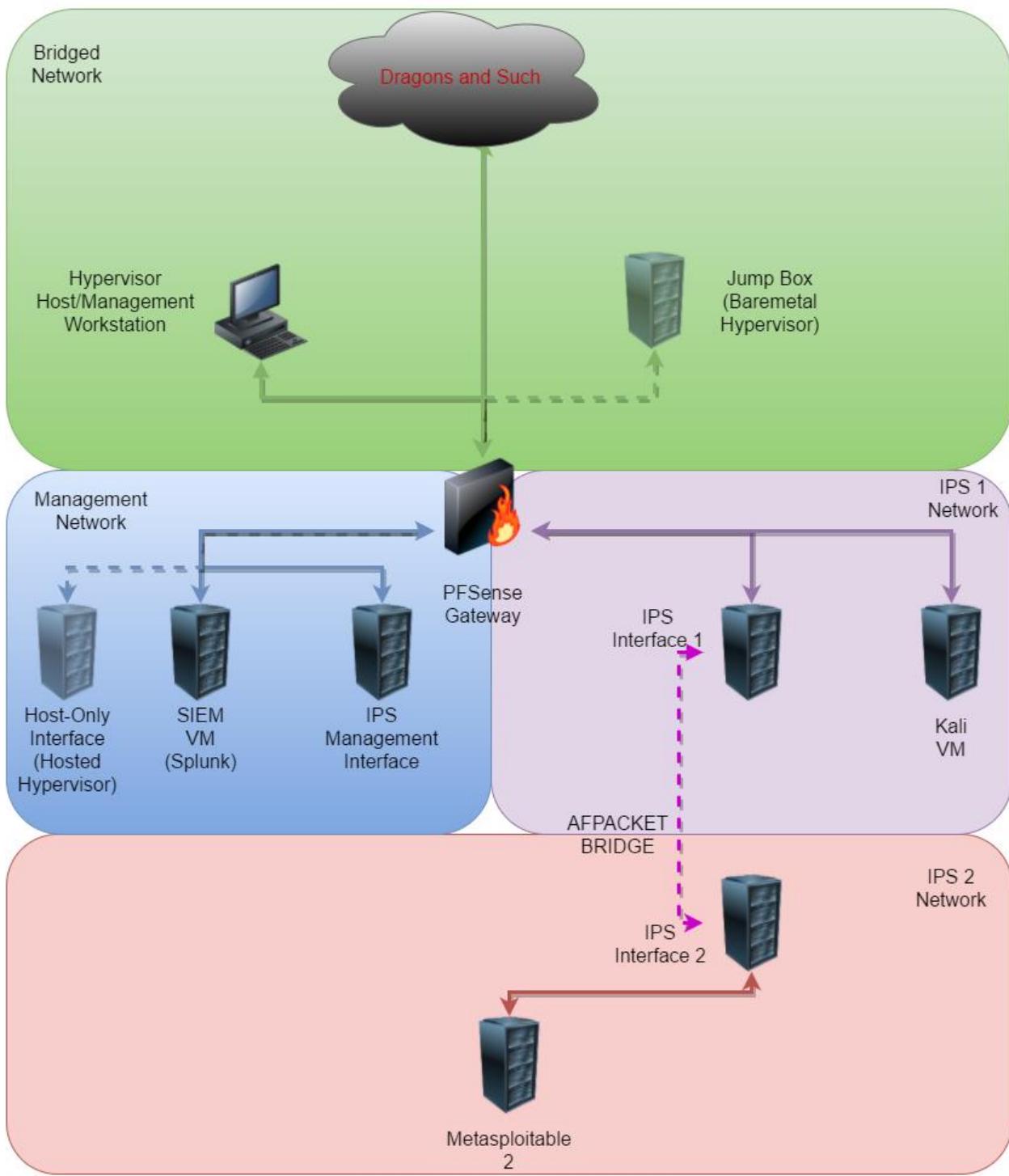
Baremetal Hypervisor Networking - Virtual Switches

Most baremetal hypervisors implement virtual networking via virtualized network switches (vswitches). You don't really have the concept of host-only, NAT and/or bridged networks the same way you do in hosted hypervisors. I'm going to use VMware's vSphere Hypervisor (also known as ESXi) as an example here, since most baremetal hypervisors (but not all) implement their virtual networking in roughly the same manner.

With ESXi, you simply have virtual switches that have a number of ports you can attach virtual machines to, so that they are on the same Layer 2 network. Those vswitches can also be attached to the server's physical network cards, allowing virtual machines attached to that switch to interact with the rest of the network through that network card. In this way, it emulates how most professional grade network switches have designated "uplink" ports to connect the switch to the rest of a larger enterprise network, not unlike how a hosted hypervisor's bridged network operates. Alternatively, you could choose not to uplink to the server's physical network connection and the hosts attached to that vswitch would be considered isolated and only able to talk to machines attached to that particular vswitch, not unlike how a hosted hypervisor's host-only network operates.

Lab Overview

In this chapter we will review a network diagram of the VM lab we will be building together, discuss resource allocations for our VMs, and discuss how our virtual network lab will be laid out, regardless of what hypervisor you use.



Lab Network Description

We have 5 virtual machines in our network, and 4 distinct network segments. We're using a pfSense VM with three network interfaces to route traffic and provide security for the bridged, management, and both IPS networks. pfSense is a very popular, flexible and easy to use BSD-based firewall distro that, in addition to acting as a network firewall, can provide a variety of other network services. Our pfSense VM will handle routing traffic between network segments as necessary, network segmentation and security through a series of firewall rules, and core network services for our lab environment, including DHCP, DNS, and NTP services. Having pfSense sit between all of these network segments leads to why the network was designed the way it was. Each of the network segments in our lab has a particular function and reason for being there, so let's talk about how they're set up and why.

Bridged Network

The bridged network is like an upstream internet connection that connects our VMs to the local physical network the hypervisor is connected to, and (eventually) the internet. Firewall rules we put on the pfSense VM will be set up to prevent local hosts from accessing the lab VMs, and will prevent the lab VMs from communicating while the local hosts on the physical network, while still allowing internet access as necessary. If you're running a baremetal hypervisor, you will also be setting up firewall rules to allow your management workstation, or a dedicated "jump box" to access the VM lab systems.

Management Network

The management network acts as a trusted, safe and secure network. This network is where our SIEM (Security Intrusion Events Manager) VM will be, and one of the three network interfaces we will be using for IPS (Intrusion Prevention System) VM (I will explain why the IPS VM has 3 interfaces in just a moment), and one of the three firewall interfaces for our pfSense VM. Our IPS VM will be running either Snort (<https://snort.org>) or Suricata (<https://suricata-ids.org>) for network inspection, while our SIEM VM will be running Splunk (<https://www.splunk.com>) to collect and allow us to query generated alerts from our IPS system.

IPS 1 and IPS 2 Networks

The IPS 1 network hosts a Kali Linux VM, and one of our three network interfaces connected to the IPS VM, while the IPS 2 network hosts the third of 3 network interfaces for our IPS VM, as well as the Metasploitable 2 VM. Kali Linux is a very popular and easy to use penetration testing distro that is loaded with offensive security tools and scripts, while Metasploitable 2 is a very vulnerable VM. Metasploitable has been traditionally used to introduce red teamers to penetration testing and exploitation with a tool called The Metasploit Framework. In our case, we're using Kali and Metasploitable 2 for the express purpose of testing our IPS system and making sure that these two VMs on the two network segments can communicate with one another.

AFPACKET Bridging between IPS 1 and IPS 2

Wait, so, if IPS vswitch 2 is NOT connected to the pfSense VM, how does Kali reach it? Remember how I mentioned our IPS VM has three network interfaces? One is connected to the management network, while the other two are connected to both the IPS 1 and IPS 2 networks. We will be connecting the IPS 1 and IPS 2 networks together through our IPS VM, using a technology available in both Snort and Suricata called AFPACKET bridging.

Why All The Trouble?

AFPACKET bridging allows the two network interfaces of the IPS VM, connected to IPS 1 and IPS 2 networks to be fused together, allowing communication between the two network segments, so long as the IPS VM is powered on, and the Suricata or Snort IPS service is running. For instance, if you wanted your lab to serve as a malware analysis lab and wanted to ensure complete isolation to the internet. If you place your VMs in the IPS 2 network, you could then turn off the IPS VM, or disable the Snort/Suricata service, and VMs in the IPS 2 network have no external access to the internet or any of the other network segments (even the IPS 1 network) whatsoever. Conversely, if you are host pentesting lab with your VMs hosted on the IPS 2 network, and you have a bunch of lab systems you wanted to update, then isolate again, you can turn on the IPS VM, bridge the two networks together, and get your network access from the pfSense interface connected to the IPS 1 network. After you're done downloading updates, turn off the IPS VM, and you have an isolated lab environment again. This network design is called "fail-closed networking".



This is how it works in a nutshell.

Fail-closed networking operates on the idea if that you would rather have the network be secure and unable to communicate with external networks, as opposed to fail-open, and have no protection/isolation at all. The advantages for our lab in implementing a fail-closed network are pretty great. At a moment's notice, our lab VMs can have access to the internet, and at a moment's notice, if there is some sort of a catastrophic issue, the VMs in the IPS 2 network can be isolated quickly. This allows us the ability to switch between the equivalent of a host-only and

bridged network in an instant. To put it bluntly, it's the secret sauce behind how our lab can be modified to accommodate a variety of needs for IT and/or information security training needs.

VMs, Resource Allocations, and Minimum Hardware Requirements

We will be using at least 5 virtual machines as a part of the lab network we will be building together.. Here are the recommended resource allocations for the VMs::

- pfSense: 512MB RAM, 5GB Disk, 1 cpu/core
- Kali Linux:4GB RAM, 80GB Disk, 1 cpu/core
- SIEM (Ubuntu 16.04 Server): 4GB RAM, 80GB Disk, 1 cpu/core
- IPS (Ubuntu 16.04 Server): 2GB RAM, 80GB Disk, 1 cpu/core
- Metasploitable 2: 512MB RAM, 10GB Disk, 1 cpu/core

If you add up all the disk and RAM requirements, we're looking at 255 GB of disk space to host the VMs (note that this does NOT account for space required for VM snapshots), and 11GB of RAM. As such, I would recommend a box with a minimum of 16GB of RAM, 4 CPU cores (with Intel's VT-x or AMD's AMD-V virtualization technology supported by the CPU and motherboard), and 500GB of disk space, at a minimum. As with most hardware requirements, more is always better.

The resource allocations I've recommended for the VMs ensure that each of the VMs performs well. The hardware recommendations guarantee that you have room to create an additional VM or two, as well as store snapshots of your VMs, operating system files, hypervisor files, and OS installation ISOs.

If you are really in a squeeze and need to fit more VMs on the same disk, or you have less than 16GB of RAM to work with here are some things you can do to try and spread your hardware a bit further. Keep in mind that pushing your system too hard, or spreading your system resources too thin can lead to poor performance.

- Since this is a simple lab environment, you may consider cutting the RAM allocations on the SIEM and Kali Linux VMs from 4GB to no less than 2GB. This has the potential to reduce the max RAM consumption across your VMs from 11GB to 7GB, but graphical tools you use in the Kali Linux VM may be a little sluggish, and take a little longer to load.
- You could decrease the disk space allocation on the SIEM, IPS and Kali Linux VMs from 80GB each, to 60GB each for the IPS and SIEM VMs, and 40GB for the Kali Linux VM. This has the potential to save you 20-80GB of disk space, but means you will have to closely monitor disk usage in your VMs.

Hypervisor Guides

If you've made it this far, you know what we're building, you have the hardware to build it, and you may even have a hypervisor in mind that you want to use. I have created a guide on how to set up 5 different hypervisors to support the lab network that we want to create. Which one you choose is up to you. Please note that any software requirements specified below are in addition to the hardware recommendations we discussed earlier (e.g. RAM, Disk, number of Cores, Virtualization Extension support, etc.). Here are your hypervisor choices:

[Microsoft Client Hyper-V](#): A hosted hypervisor by the Microsoft Corporation for users of Windows 8.1 or Windows 10. You must be running Professional, Enterprise, Ultimate or Education edition, and the operating system must be 64-bit.

[Oracle VirtualBox](#): A hosted hypervisor by the Oracle Corporation. Available on practically every operating system out there. 64-bit OS is recommended.

[VMware Fusion Pro](#): A hosted hypervisor by the VMware Corporation made specifically for Apple OSX. This guide specifically requires VMware Fusion Pro edition.

[VMware Workstation Pro](#): A hosted hypervisor by the VMware Corporation for Microsoft Windows and Linux systems. 64-bit OS recommended.

[VMware vSphere Hypervisor\(ESXi\)](#): A baremetal hypervisor by the VMware Corporation. Extremely picky about hardware compatibility, but free and very robust. Support for new and/or exotic systems may not be guaranteed.

I have provided step-by-step instructions on how to create your VM lab on the five hypervisors listed above. Each hypervisor setup guide has the following sections:

- Initial installation of your hypervisor
- Setup and customization of the hypervisor
- Detailed step-by-step guide on creating your first VM, pfSense
 - OS installation
 - Initial network setup
 - Initial setup of the WebConfigurator (the pfSense web interface)
 - Taking your first snapshot
 - Redirecting you to the pfSense Firewall rules and Network Services guide to fully set up the pfSense VM including guidance on:
 - Setting up the firewall rules for the Bridged, Management and IPS networks
 - Setting up core network services
 - DHCP
 - DNS
 - NTP

- Squid Proxy (optional)
- Guidelines to follow for creating 3 out of the remaining 4 VMs
- Instructions on how to acquire and set up Metasploitable 2 as necessary
- Recommendations on remaining chapters to read to create a fully functional lab environment such as:
 - Defense in Depth for Windows Hosted Hypervisors
 - Remote Lab Management
 - Network Design Factors When Working with Baremetal Hypervisors
 - IDS/IPS Installation
 - Splunk Installation

So without further adieu, choose a hypervisor, and get started!

Setup - Microsoft Client Hyper-V

Note: Please be aware that this guide was written using Client Hyper-V running on Windows 10 Professional, anniversary edition. If you are running Client Hyper-V on Windows 8.x, you may notice some slight differences, but the instructions should more or less be the same.

I've been using Client Hyper-V as my hosted hypervisor of choice recently because it's convenient (I primarily run Windows for playing video games and other applications), the hypervisor's featureset is robust, and the price is right (read: free) if you're running the right version of Windows. This guide will instruct you on how to create the our lab environment on Client Hyper-V, one step at a time.

Installation

The only downside to running Client Hyper-V, is that you have to be running a premium version of Windows in order to get the option to install it. Specifically, Client Hyper-V is an additional feature you can install for the following Windows operating systems:

- Windows 8 (or 8.1) Pro 64-bit
- Windows 8 (or 8.1) Enterprise 64-bit
- Windows 10 Enterprise
- Windows 10 Professional
- Windows 10 Education

Note: While not explicitly stated on the "Windows 10 Hyper-V System Requirements" page on the MSDN (https://msdn.microsoft.com/virtualization/hyperv_on_windows/quick_start/walkthrough_compatibility), you'll need a 64-bit OS to be able to address all the memory we will need to use for our lab. Therefore, you'll want to use a 64-bit version of Windows 10.

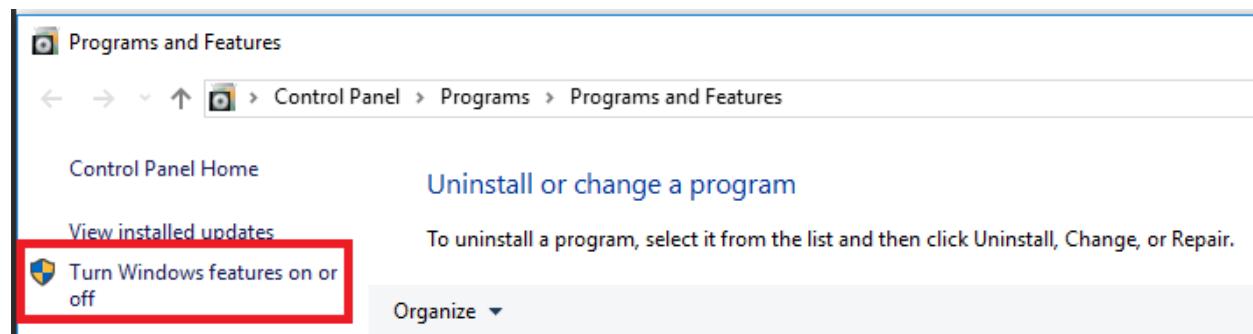
So in addition to requiring one of the premium editions of Windows 8 or Windows 10, There are certain specific hardware requirements that Client Hyper-V needs in order to run. To make it easy however, you can open up a command prompt and run:

systeminfo.exe

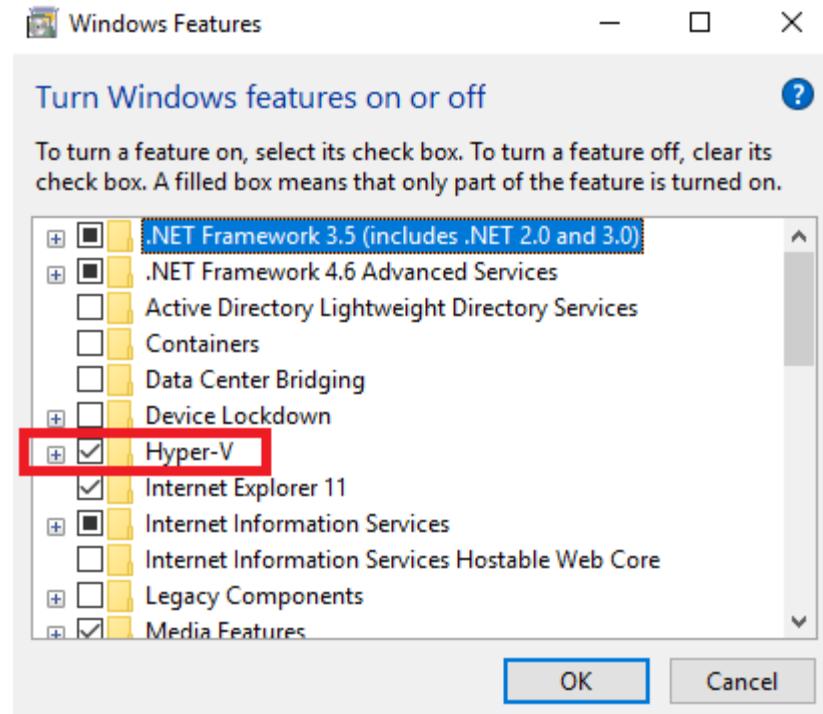
```
Hyper-V Requirements:          VM Monitor Mode Extensions: Yes
                           Virtualization Enabled In Firmware: Yes
                           Second Level Address Translation: Yes
                           Data Execution Prevention Available: Yes
```

The last thing the command will output is a section called “Hyper-V Requirements”: This will tell you whether or not your hardware can run Client Hyper-V. If all the options come back “Yes” you should be good to go. Otherwise, double check that your motherboard has virtualization options enabled and that your CPU supports the required virtualization options as well.

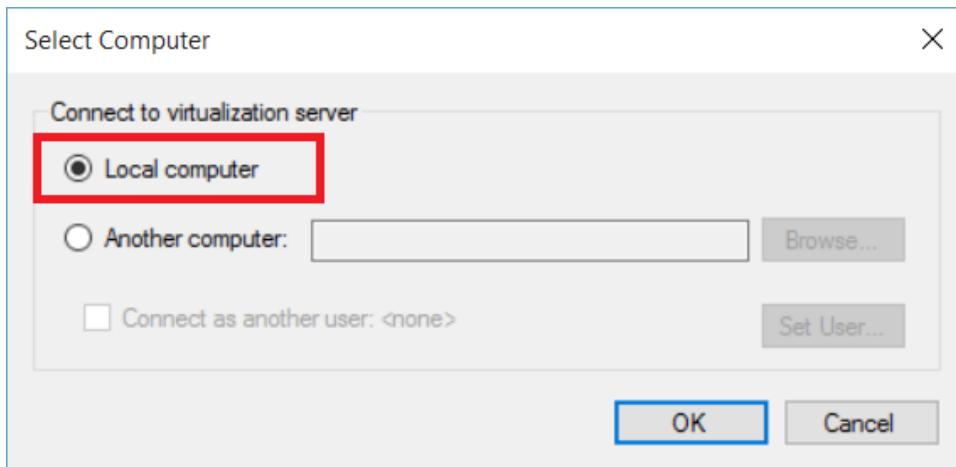
To install Client Hyper-V, navigate to the “Programs and Features” menu, and select the “Turn Windows Features on and off” option.



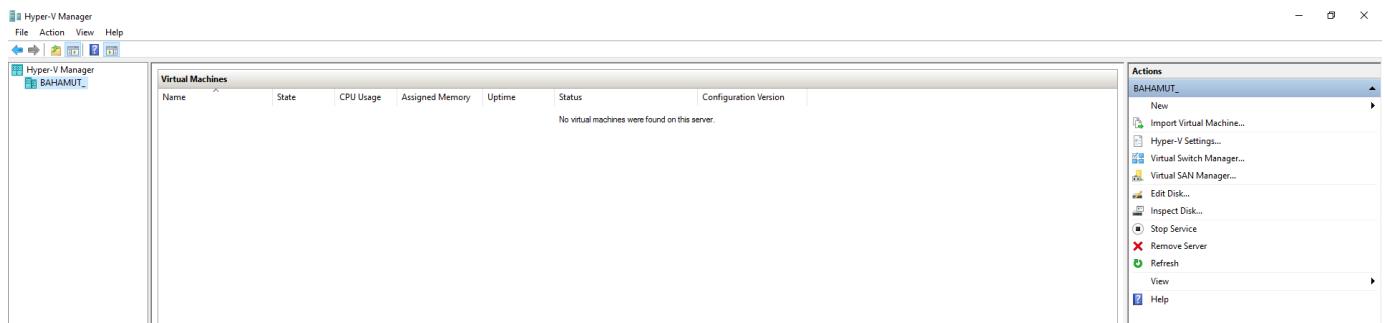
In the next window. Locate the “Hyper-V” feature, and click the checkbox so that a checkmark appears in the box, then click OK. Windows will go through enabling Client Hyper-V and installing the management tools/console for managing the hypervisor. After the installation is complete, you will need to reboot your system.



After you have rebooted, run the “Hyper-V Manager” application. The system will ask you if there is a remote server you wish to connect to, or if you would like to connect locally. Choose the option to connect to the local computer.

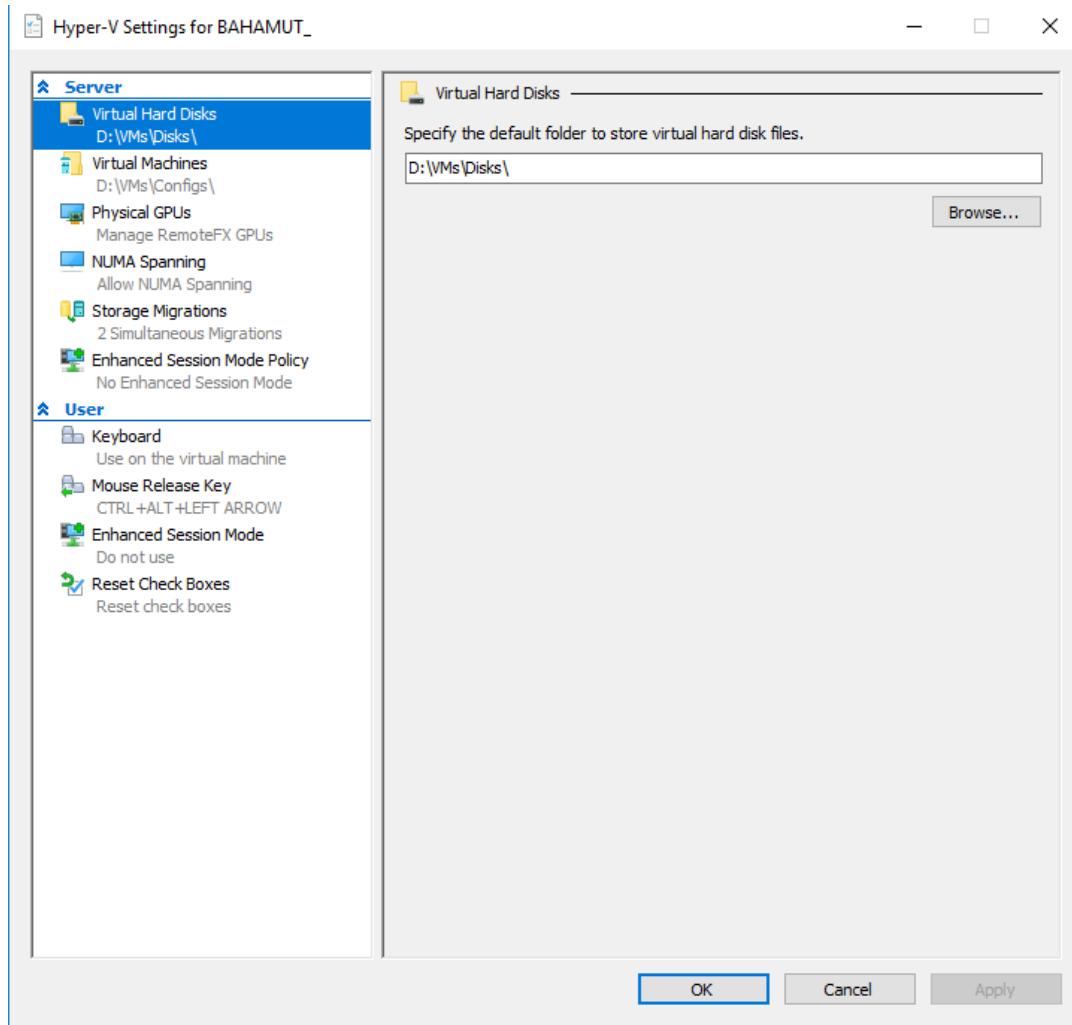


You should be greeted by a screen that looks like this:



Hypervisor Preferences

So now that we have Client Hyper-V installed, let's do some customization. On the right pane of the Hyper-V Manager interface labeled "Actions", click on "Hyper-V Settings..." to bring up the Hyper-V Settings menu for your system.



You'll notice that on the left pane, there are two subsections: Server and User. The Server settings affect how the Hyper-V server operates, while the User section determines how the client interacts with VMs on the Hyper-V console.

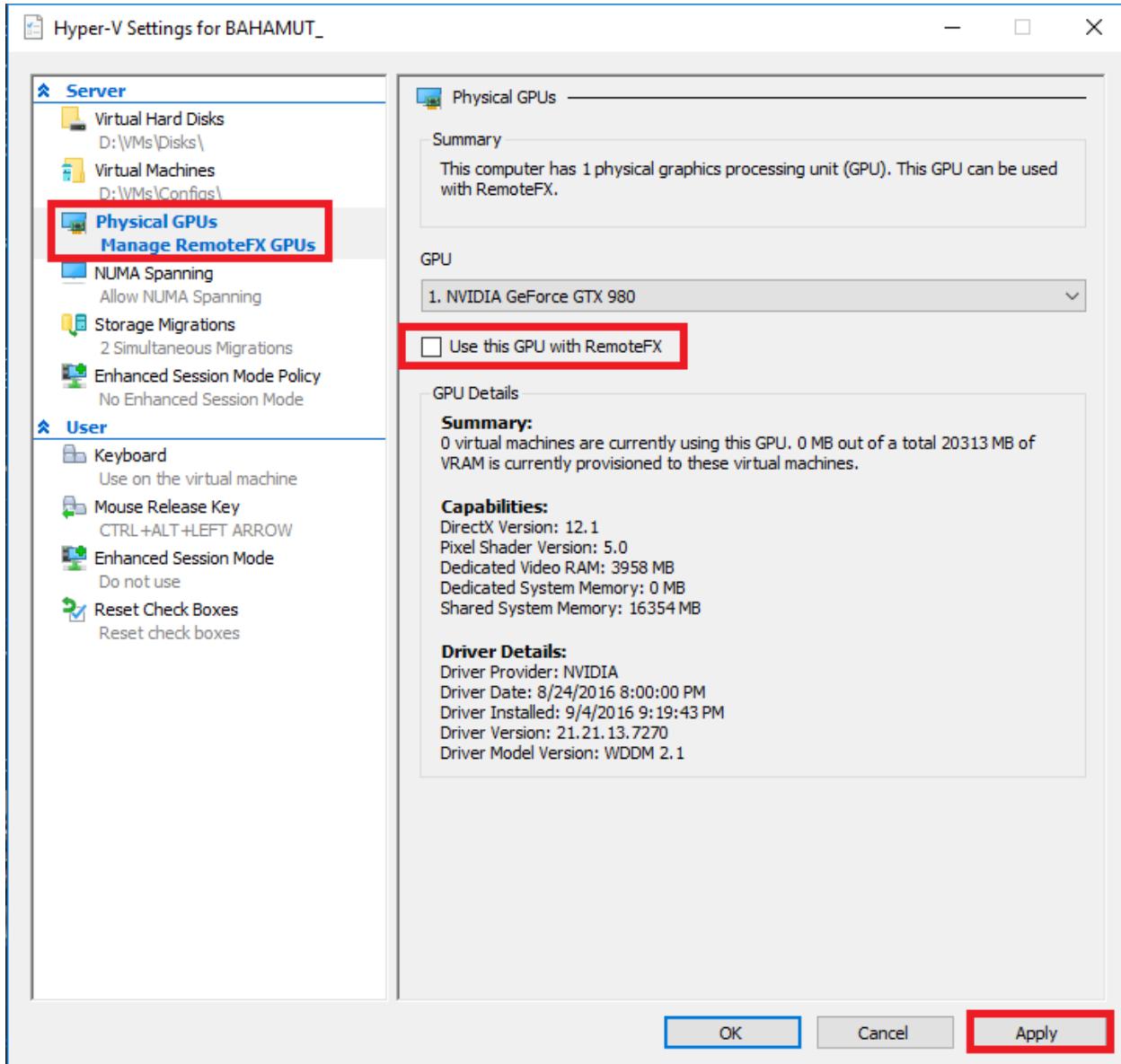
Server Settings

The first server setting, "Virtual Hard Disks" Determines where on the system the virtual hard drives, known as vhd (virtual hard drive) or vhdx (virtual hard drive extended) files in Hyper-V terminology, will be stored. The files are the containers that Hyper-V uses for installing virtual machine operating systems and the files for that operating system.

In the hardware considerations portion of this guide, we discussed the importance of disk I/O as a performance factor, but lets recap here. If you have a second physical hard drive installed on your system, and you are using a hosted hypervisor, it is a good practice to place your virtual machine files on a separate hard drive that the host operating system is NOT installed on. This is in order to maximize the amount of disk I/O available to both your VMs, and the host operating system. For example, If you have two physical hard drives installed in Windows formatted as the C: and D: drives, and you installed Windows on the C: drive (usually the default setting), you would want to store the VM files on the D: drive to make sure that your VMs and the host OS are not competing for disk I/O, impacting the performance of both the host OS and the VMs you wish to run. In the case of my system, "BAHAMUT", I chose to locate the VHD files under D:\VMs\Disks. If you change this setting, be sure to click the Apply button in the bottom right corner of the window.

The second server setting, "Virtual Machines" defines where the configuration files and folders for each of your VMs should be stored. While these files aren't as big as the VHD files, generally speaking, its good practice to store the virtual machine files in the same location as the VHDs for organizational purposes. In my case, I chose to store the virtual machine files in D:\VMs\Configs. Note that Hyper-V creates a subdirectory for each VM in the Configs and stores the actual virtual machine configuration files in that subdirectory.

The next server setting "Physical GPUs" determines whether or not virtual machines created will have direct hardware access to any installed GPU units installed on the physical host system. There should be no reason to have this functionality enabled, since none of our VMs have the means to utilize GPU hardware like that, so you'll want to uncheck the "Use this GPU with RemoteFX" checkbox, then click the Apply button.



I left the “NUMA Spanning” and “Storage Migrations” server settings alone. Since they won’t really have an effect on our environment. NUMA spanning is more for massive multi-CPU systems with tons of memory. It allows the system to use memory that isn’t necessarily local to the CPU. Storage migration is the ability to move a number of VMs while the VMs you want to migrate are still up and running. It’s a useful feature for enterprise environments, but a feature you won’t have to worry about too often in a lab environment. The only time you’ll be moving VMs is if you get a hardware upgrade, and chances are, you can spare the downtime in your lab if that’s the case.

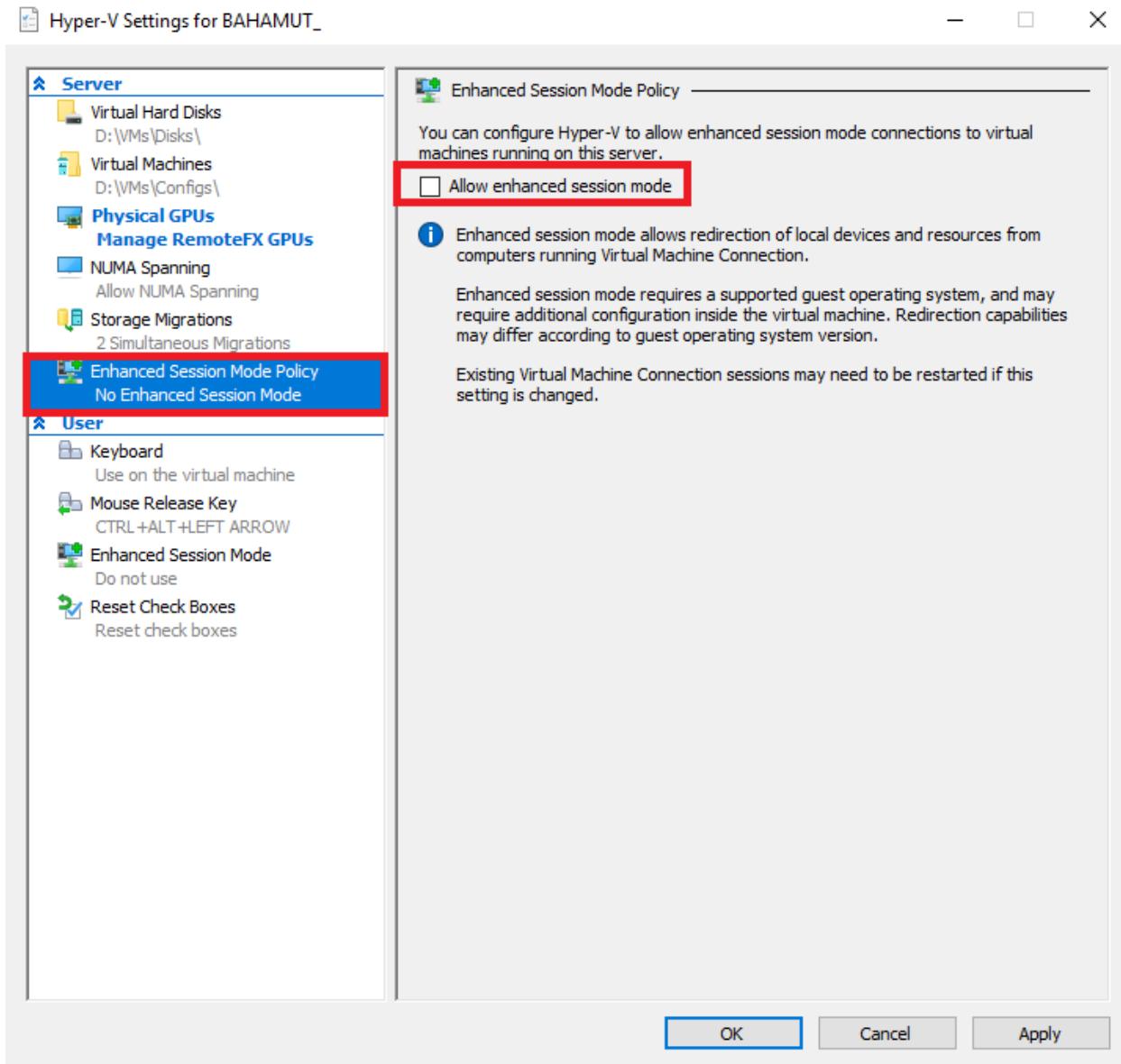
The final server setting, “Enhanced Session Mode Policy” governs whether or not extra features of the vmconnect application will be available or not. The vmconnect application is what you use

to interact with your VMs in Hyper-V. If you're familiar with how Microsoft's Remote Desktop Protocol application works, vmconnect operates in a very similar manner.

In the RDP application, there is a tab called "Local Resources" that allows you to select what things, if anything, you want to have available when you connect to a remote computer. You can have locally connected drives, printers, sound devices, etc., and have them behave as though they're connected to the system you are connecting to remotely. Enhanced Session Mode is essentially the same thing, except for the vmconnect application. If enhanced session mode is allowed, the vmconnect client will be able to share client resources on the VM the user connects to.

As always, there are restrictions. Enhanced Session Mode only applies to Windows VMs that have Remote Desktop Protocol enabled. This makes sense, seeing as how it is practically the same thing. . Since our lab is entirely Linux/BSD based, this has absolutely no bearing on us whatsoever. However, since the possibility exists that Windows VMs may be added to your lab network later as you need them, so I would highly recommend unchecking the "Allow enhanced session mode" checkbox, then clicking the Apply button to disable it.

You need to be very careful about features that make resources from your hypervisor host available to VMs in your lab, because this has a habit of breaking segmentation, security boundaries, and isolation between the VM host and the VMs themselves. For instance, if you had enhanced session mode enabled, and shared drives from your host system to the VM, there is a chance that your hypervisor host could become infected with malware, or active samples could escape your lab network, limiting your ability to effectively contain it. This is why I recommend disabling enhanced session mode entirely, as a precaution.



If you want to find out more about NUMA spanning check out this MS technet article:
<https://blogs.technet.microsoft.com/pracheta/2014/01/22/numa-understand-it-its-usefulness-with-windows-server-2012/>.

If you're interested in learning more about Storage Migration, you can read this MS technet article:
[https://technet.microsoft.com/en-us/library/hh831656\(v=ws.11\).aspx](https://technet.microsoft.com/en-us/library/hh831656(v=ws.11).aspx)

If you'd like to learn more about Enhanced Session Mode, go here:
<https://technet.microsoft.com/en-us/windows-server-docs/compute/hyper-v/learn-more/use-local-resources-on-hyper-v-virtual-machine-with-vmconnect>

User Settings

The Hyper-V user settings are fairly self-explanatory, but for sake of completeness, let's go over them.

The “Keyboard” setting determines how the system should respond to special key combinations when you are using the keyboard while interacting with a VM through the console. This setting determines whether or not the host OS or the VM should interpret the keyboard shortcut you are sending.

The “Mouse Release Key” setting defines what key combination should be entered in order to release control of the mouse from a virtual machine. Most hypervisors (hosted or baremetal) have a way to allow the user of the hypervisor to “attach” to the console of a VM they are running. This would be the equivalent of attaching your keyboard, mouse, and monitor directly to the VM. If you’re familiar KVM (Keyboard, Video, Mouse) switches operate, its like that. When you interact with the VM console in most hypervisors, (e.g. click anywhere in the console), the VM takes over the keyboard and mouse, and assumes that all button clicks are intended for the VM. This setting controls what key combination releases the mouse (and keyboard in most cases) from the VM console so that you can interact with other applications on the hypervisor host.

The final user setting is for configuring whether or not the client will use Enhanced Session Mode, if it is available. If you followed the previous instructions, you should have already disabled enhanced session mode under the server settings. For the sake of completion, you'll want to make sure that the “Use enhanced session mode” checkbox is unchecked under the user settings section, then click Apply.

The final option “Reset Check Boxes” allows you to restore all of the settings (Both Server and User settings) we configured here back to defaults if needed.

Virtual Switches

So now that we have the hypervisor configured, we need to set up the network infrastructure. Hyper-V uses virtual switches to signify different virtual network segments. These virtual switches come in three varieties: External, Internal, and Private.

Virtual Switch Types

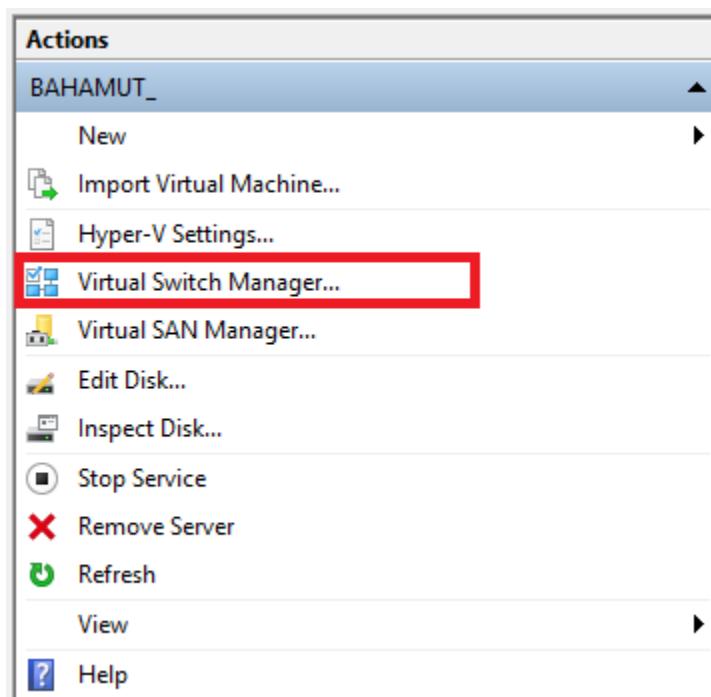
External virtual switches allow you to use your physical host system’s network connection as an uplink port to connect any VMs connected to the External virtual switch to the same physical network your host system is connected to. This would allow remote systems on your network to interact with VMs hosted on your system. External virtual switches are similar in functionality to “bridged” network segments that other hosted hypervisors use.

Internal virtual switches create a standalone virtual network with no connection to the local network, or the internet. Hosts connected to an Internal virtual switch can only communicate with other hosts on that virtual switch, including the hypervisor host. The host system is able to talk to VMs on the Internal virtual switch through a virtual network card the hypervisor creates that is attached to the virtual switch. This virtual network card allows it the hypervisor host to communicate with VMs connected to the internal network switch over the network. As you might have guessed, Internal virtual switches are similar to “host-only” network segments.

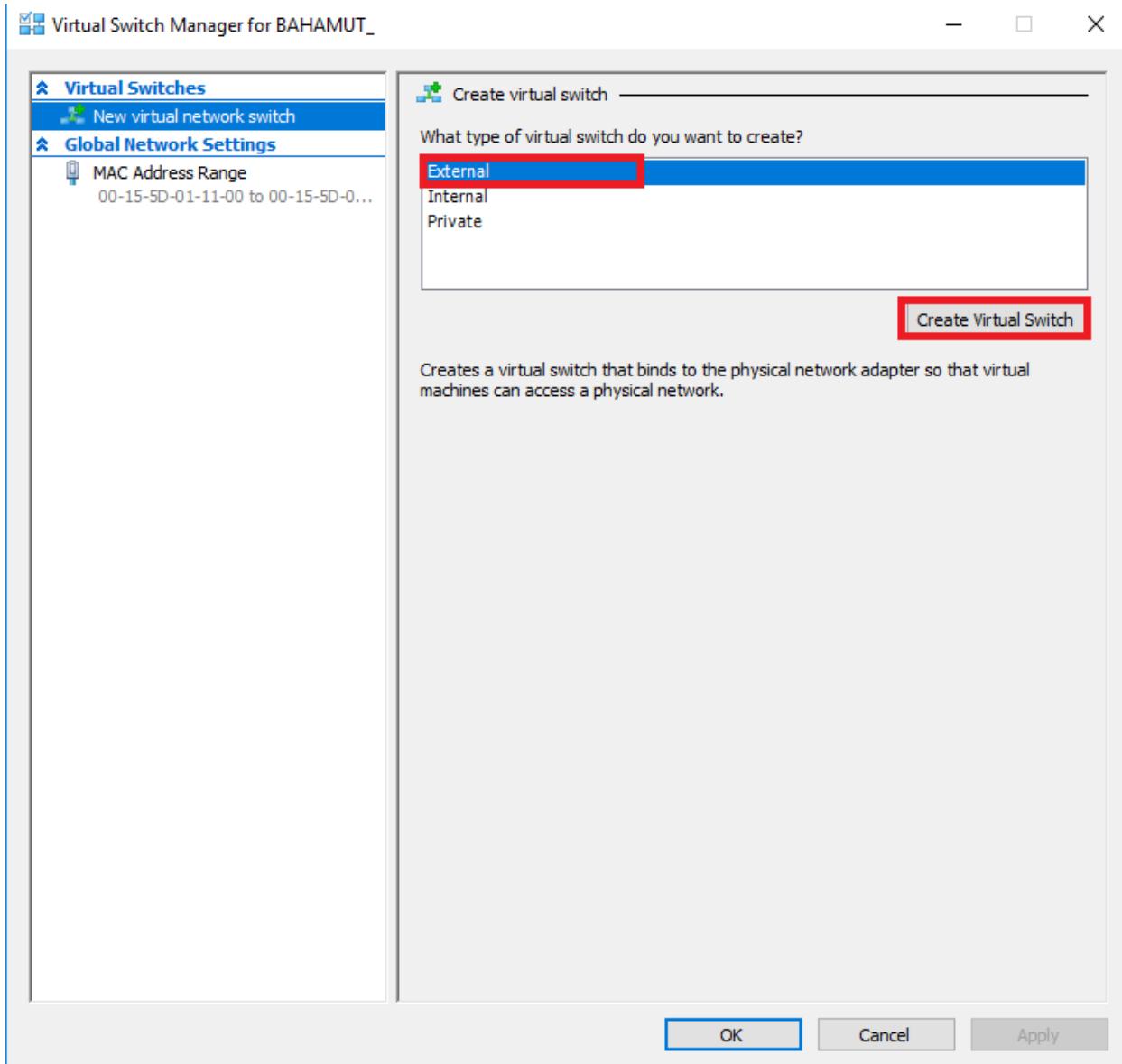
Private virtual switches are very similar to Internal virtual switches, in that they create a standalone network that only the hosted virtual machines can use to talk to one another directly. The main difference between an Internal and Private virtual switch is that the host system does not get a virtual network card, and therefore cannot communicate with VMs on a Private virtual switch directly.

Creating Virtual Switches Using the Virtual Switch Manager

So now that you know the difference, between the three types of virtual switches, we need to create 4 of them: 1 External, 1 Internal, and 2 Private switches. In the Hyper-V Manager main screen, on the right pane labeled “Actions, click on “Virtual Switch Manager...” to get started.



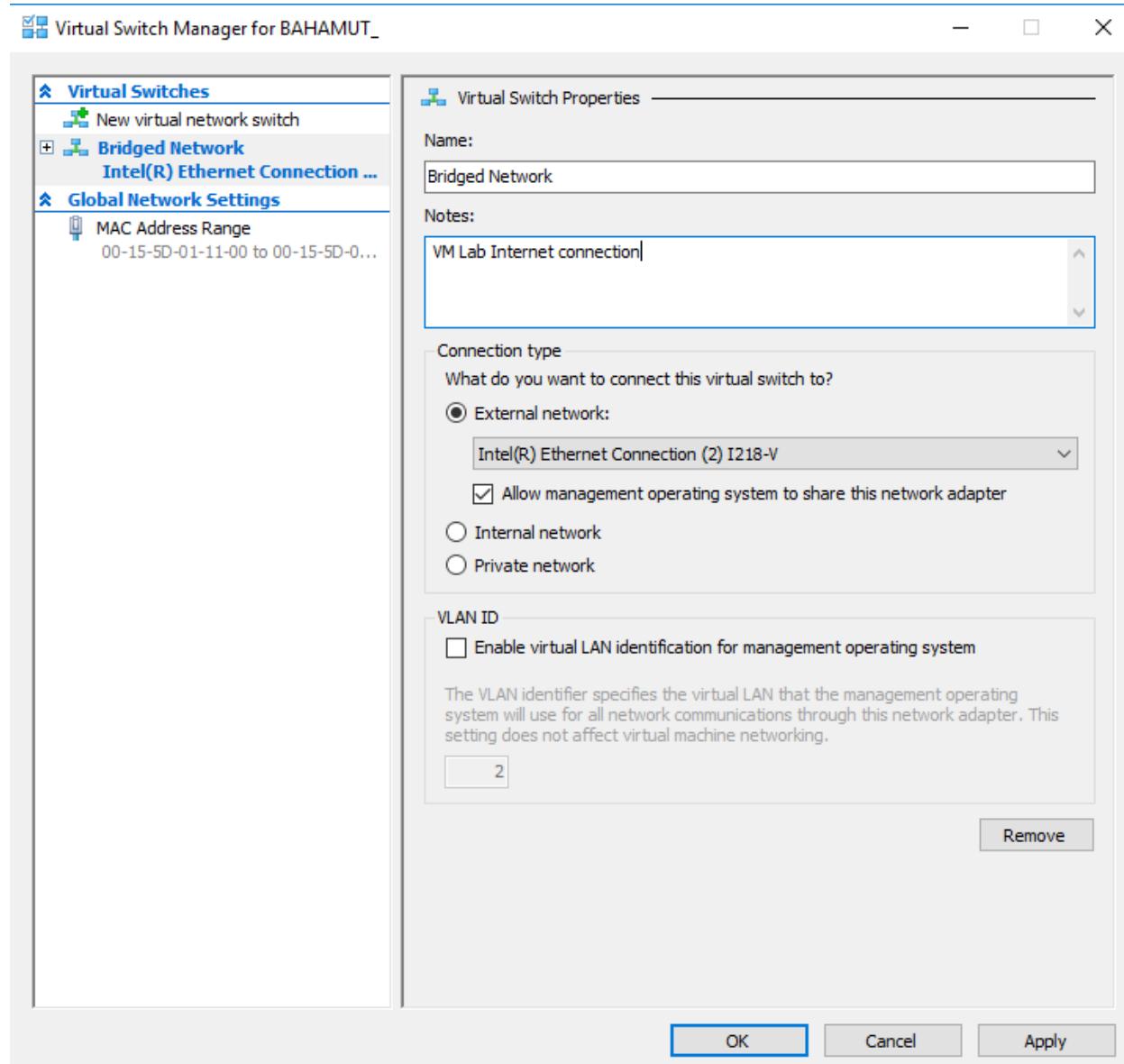
A new window titled “Virtual Switch Manager for [system name]” will appear. On the right pane, you will see the label “Create Virtual switch” the a window that has the choices External, Internal, and Private. Click on External so that it is highlighted, and click the “Create Virtual Switch” button.



The screen will update with the new virtual switch you just created. Under virtual switch properties, there are input boxes titled "Name:" and "Notes:". This External virtual switch is our Bridged network in the network diagram, I would name it something simple like "Bridged Network". As for notes, be sure to input notes to remind you of what function the switch serves. In my case I entered "VM lab internet connectivity".

Underneath this, you will see the section titled "Connection type". If you made the wrong type of switch and need to change its type, you can do so here. Note that under external network, there is a drop-down selection. This drop-down contains a list of all the network cards installed on the physical host. This determines which network card VMs attached to this External switch will use for access to the physical network (and/or internet). Underneath the drop-down list, is a checkbox "Allow management operating system to share this network adapter". If you want the

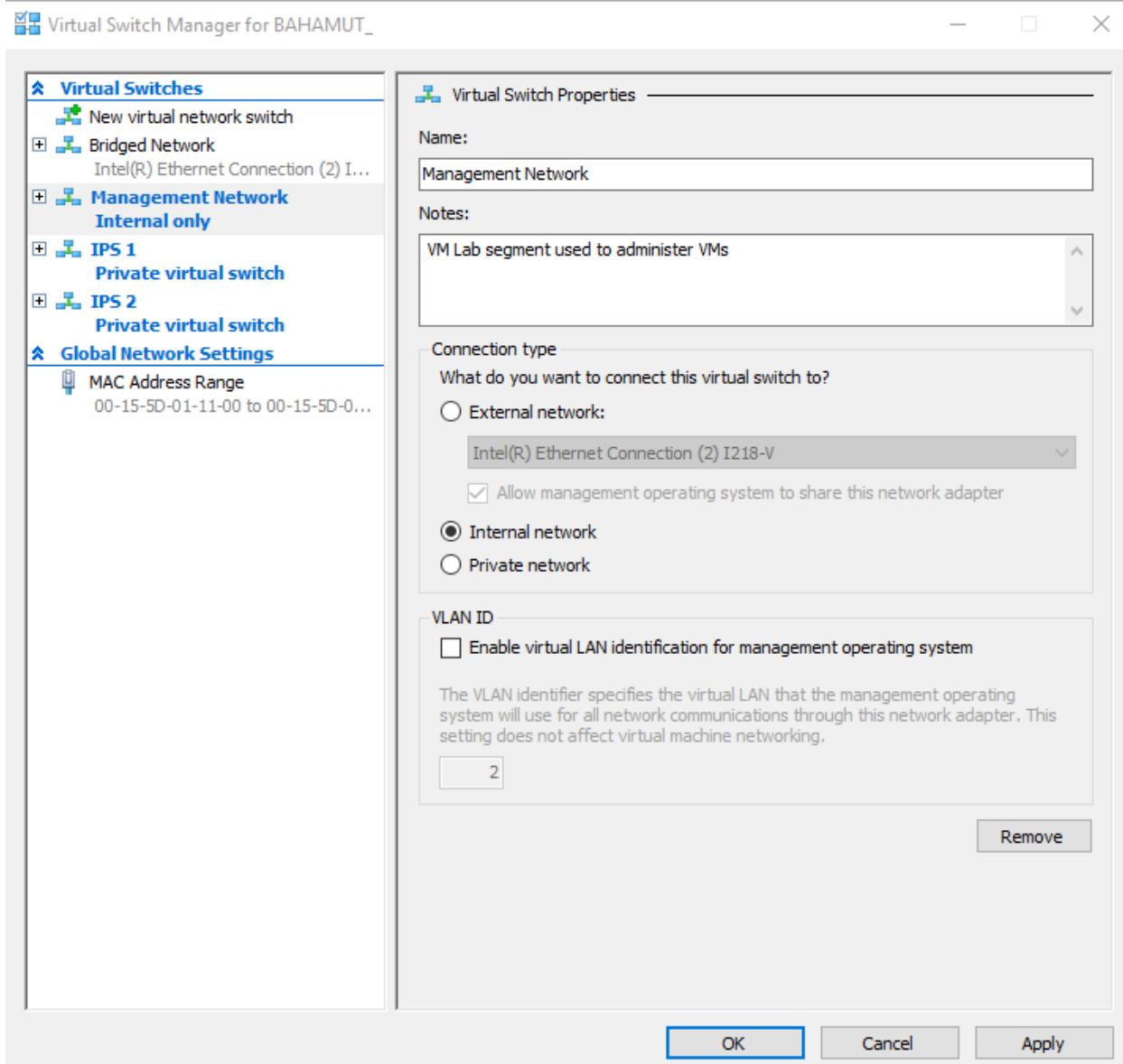
host system to be able to access the physical network through the adapter you selected, this box must be checked.



After you are done configuring the “Bridged Network” virtual switch, click on “New virtual network switch” on the left pane to return to the previous screen. Create the following:

Create one Internal virtual switch with the name “Management Network” and a description “VM lab segment used to administer VMs.”

Create two Private virtual switches with the names “IPS 1” and “IPS 2”. “IPS 1” should have the description “Connects firewall to IPS”. “IPS 2” should have the description “Connects IPS to vulnerable VMs”. When you are done you should have four virtual switches. The Virtual Switch Manager interface should look something like this:



Click the Apply button at the bottom right of the window to have Hyper-V create the new virtual switches, then click OK.

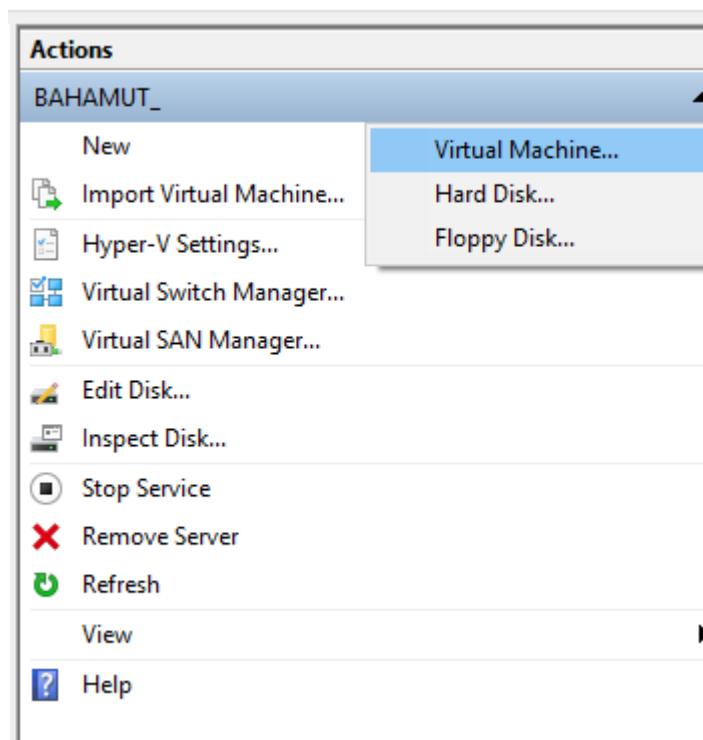
Creating the First VM, pfSense

pfSense is the keystone holding this entire configuration together. Personally, it's my favorite firewall distro due to ease of use, the amount of functionality it includes out of the box, combined with a plugin/add-on system for additional functionality. If you have the CPU, RAM, and disk, pfSense can easily be converted into a so-called "Next-Generation" firewall.

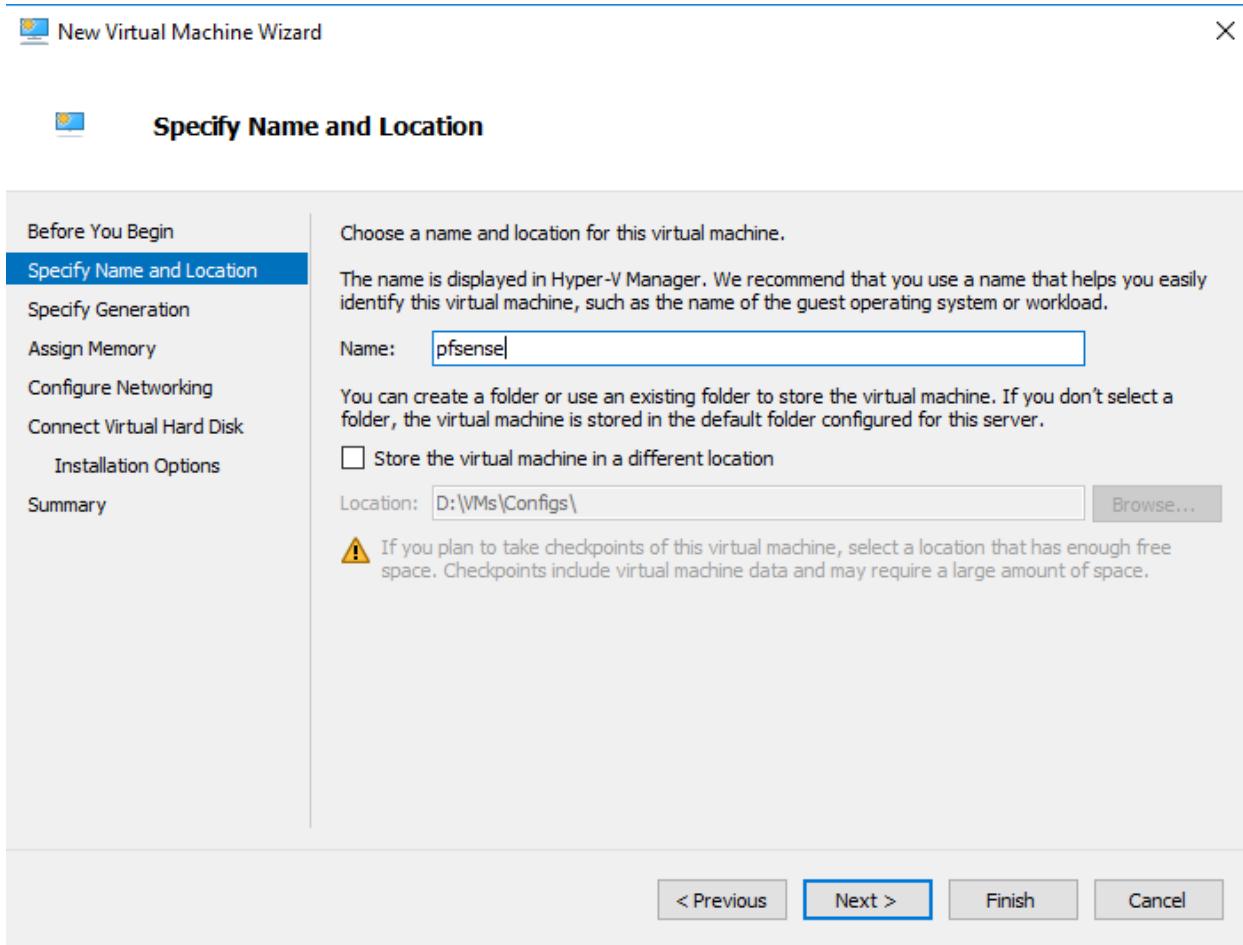
Make your way to <https://www.pfsense.org/download/>. Download the latest installation ISO for the amd64 architecture. While you're at it, you may want to download a compression utility. The pfSense maintainers distribute pfSense as a ISO image file compressed with gzip. This means that we'll need to decompress the ISO file at some point. On Windows, I prefer 7-Zip (<http://www.7-zip.org/>) as my compression utility of choice for decompressing files, since 7-Zip can handle zip, gzip, rar, and 7z files (among others) easily.

Adding a New VM

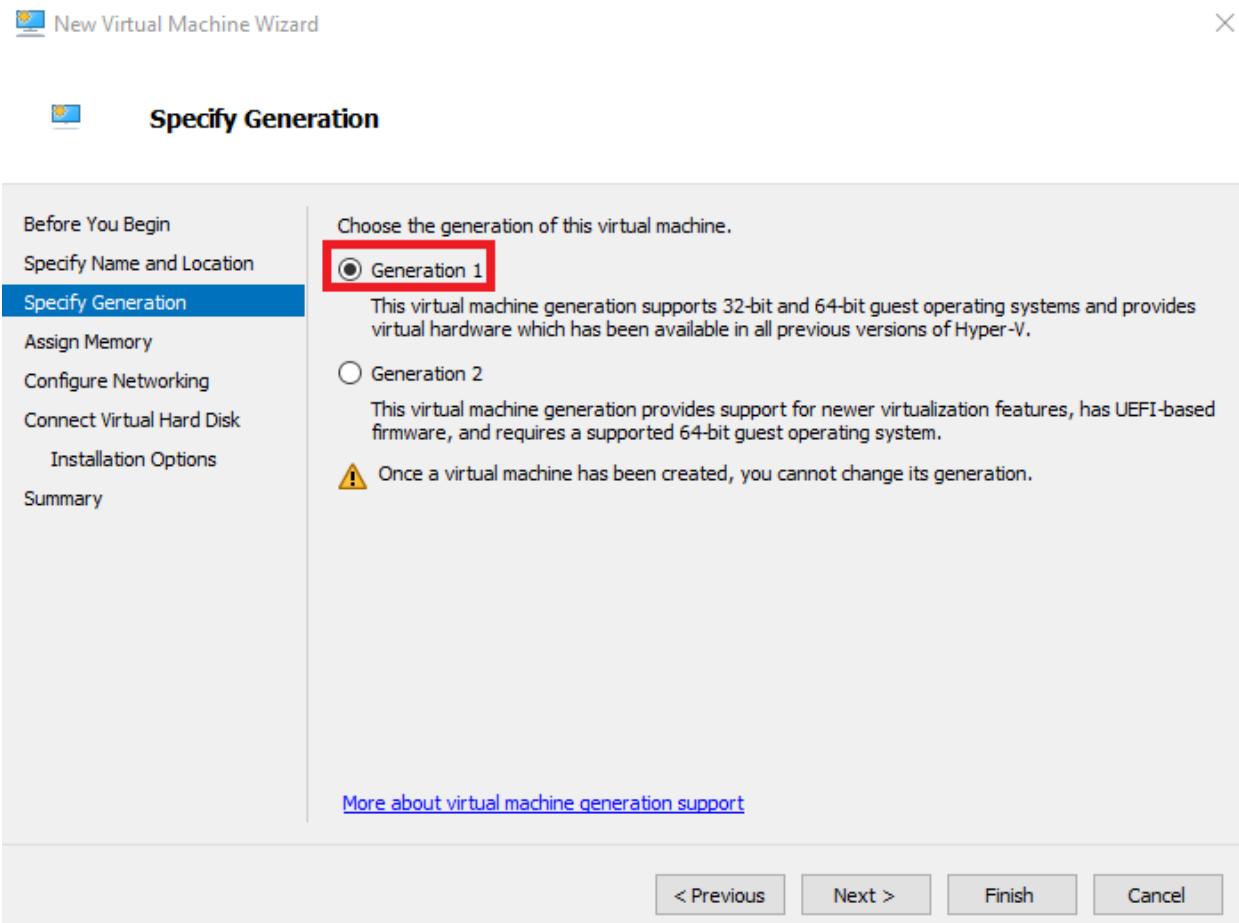
On the Hyper-V Manager window, on the left pane titled “Actions”, click “New” and a drop-down menu appears. Click on “Virtual Machine...” in the drop-down menu to start the New Virtual Machine Wizard.



The first screen has us specify a name for our new VM. We'll keep it simple and name it “pfSense”. This screen also has an option to allow you to store the VM files in an alternate location if you need to. In my case, I'm fine with it being stored in D:\VMs\Configs. After naming the VM pfSense, click Next.

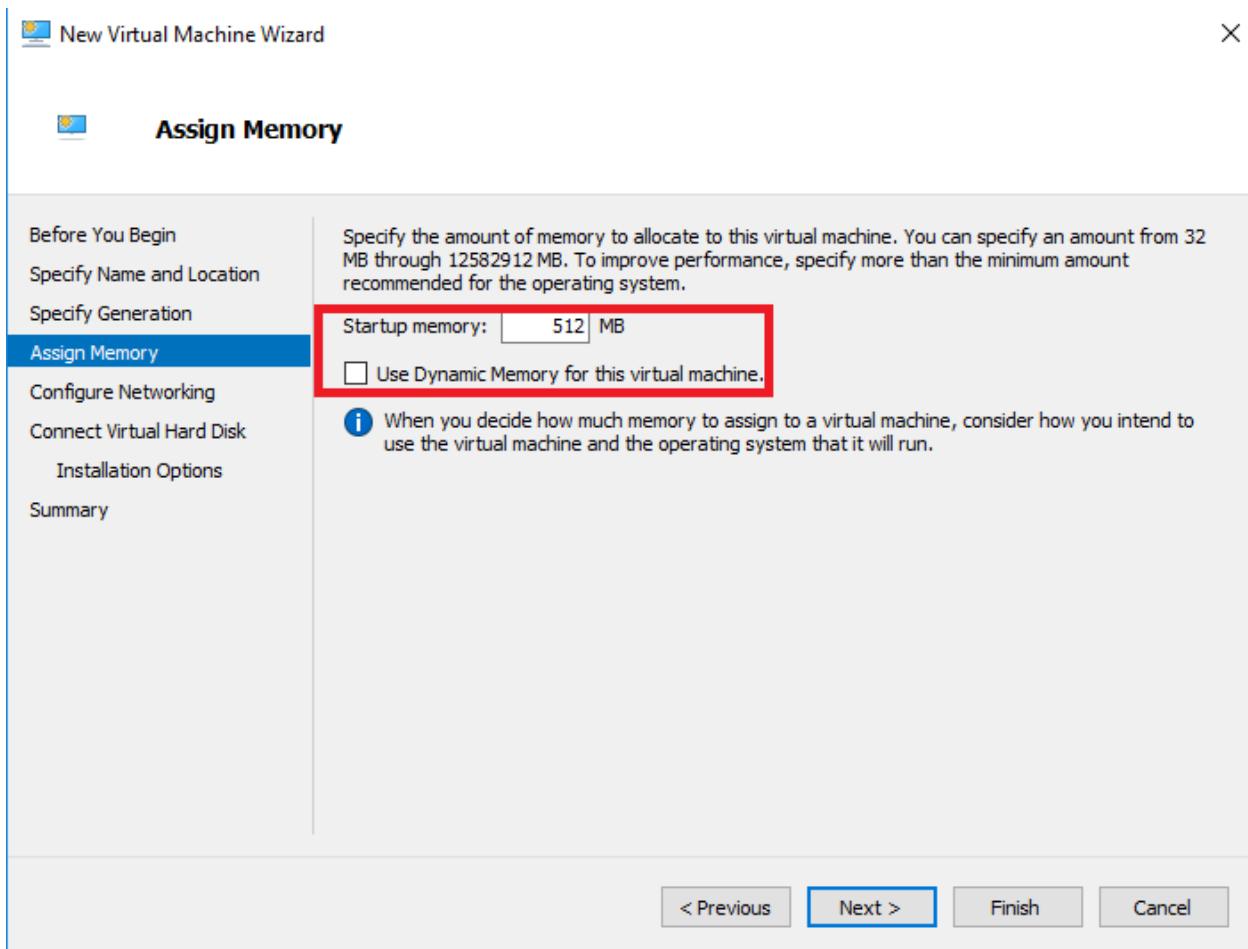


The next screen has us specify whether or not the virtual machine is a Generation 1 or Generation 2 VM. Without going into too many details, our VM will be running pfSense, which is based on FreeBSD. Hyper-V only supports FreeBSD as generation 1 VMs, so that's the type of VM we'll be creating. Click on the Generation 1 radio button, then click Next.

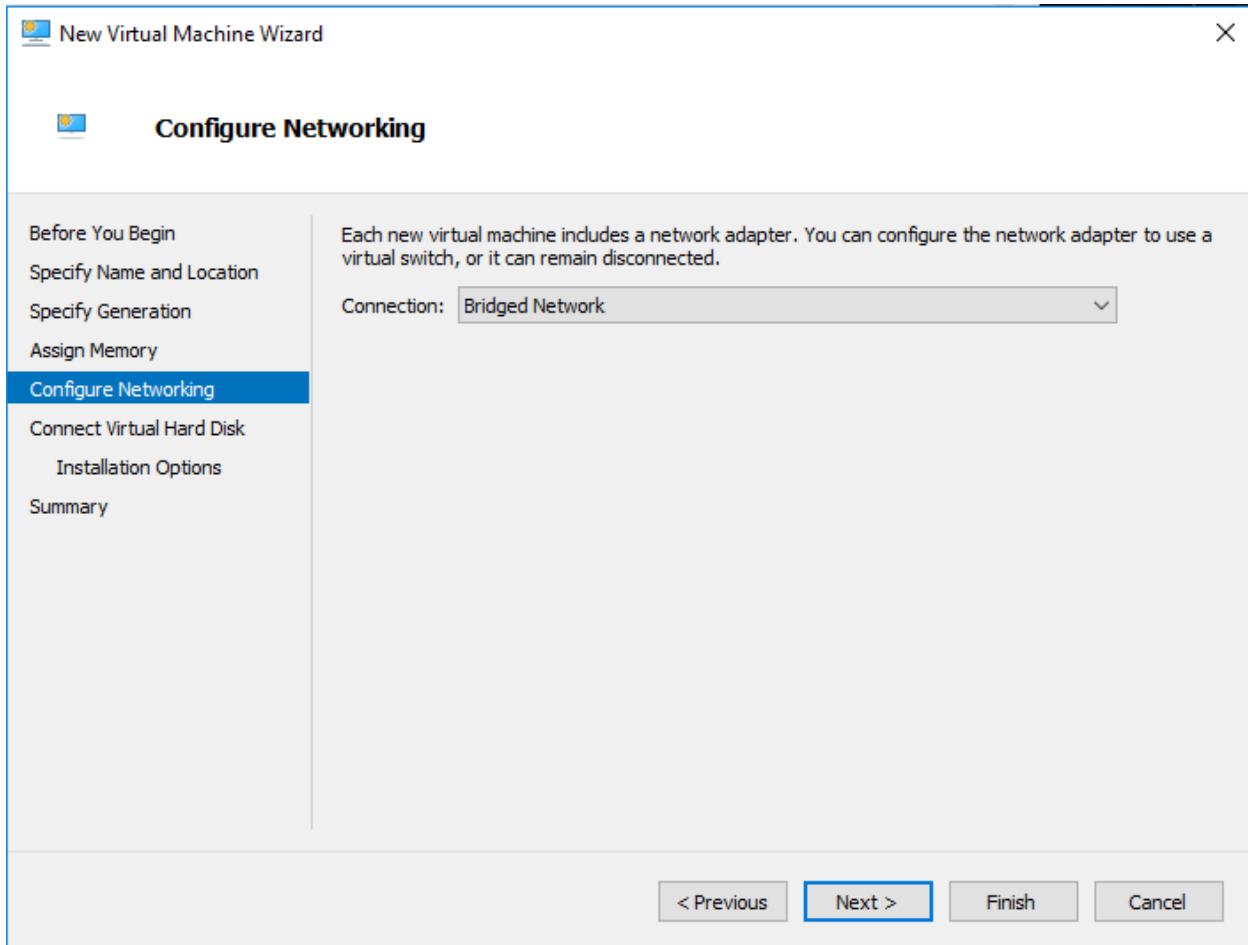


Note: If you want to find out more about Generation 1 vs. Generation 2 VMs (including more OS support information) visit this page: <https://technet.microsoft.com/windows-server-docs/compute/hyper-v/plan/should-i-create-a-generation-1-or-2-virtual-machine-in-hyper-v>.

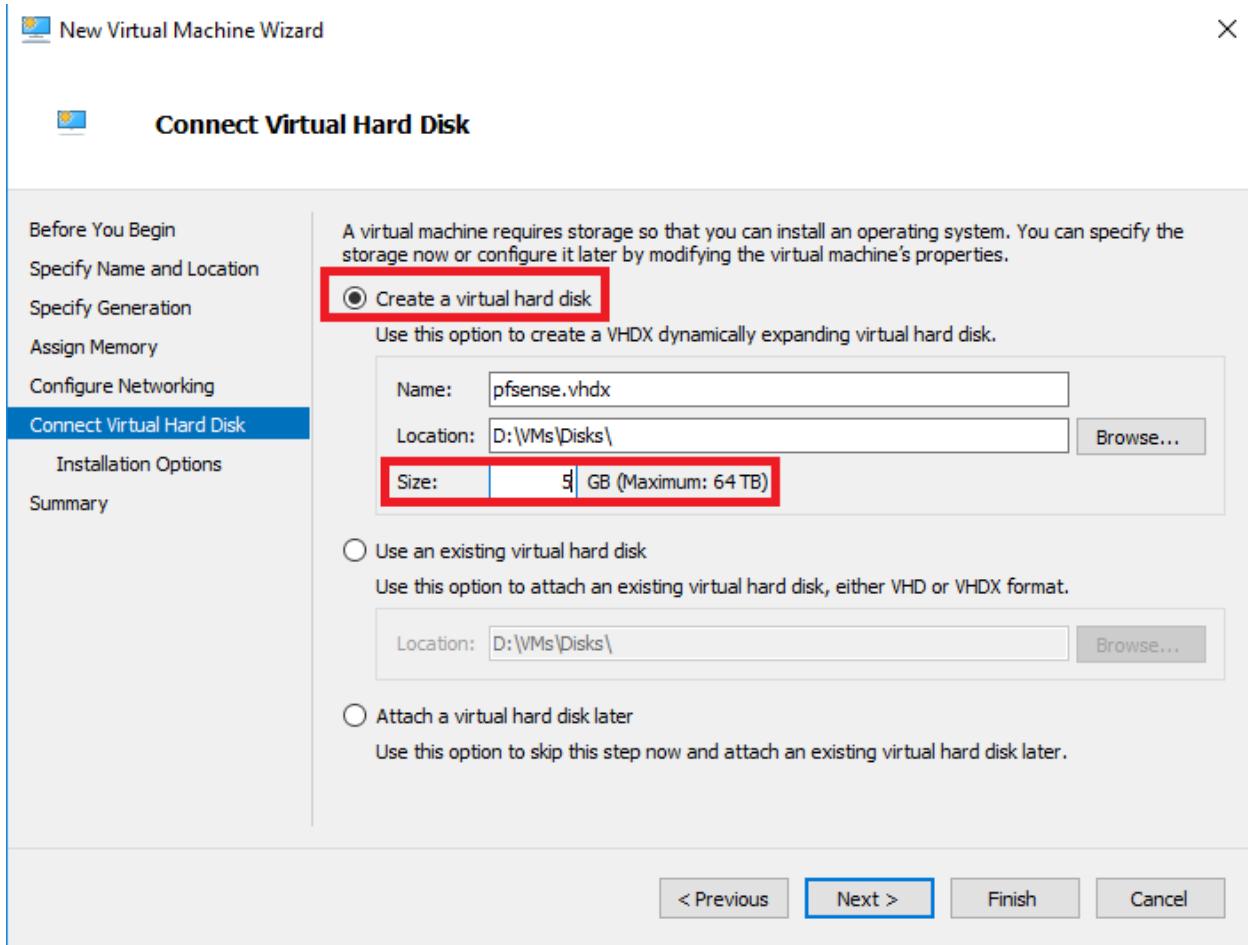
The next screen has us specify how much memory we want to allocate to our VM. Enter “512” in the input box labeled “Startup memory:” and uncheck the “Use Dynamic Memory for this virtual machine.” checkbox. Dynamic memory allows the virtual machine to allocate more RAM to itself as it sees fit. If you have a bit more RAM to play with, you could experiment with this feature, but for now, we’re going to leave it disabled. When you are finished, click Next.



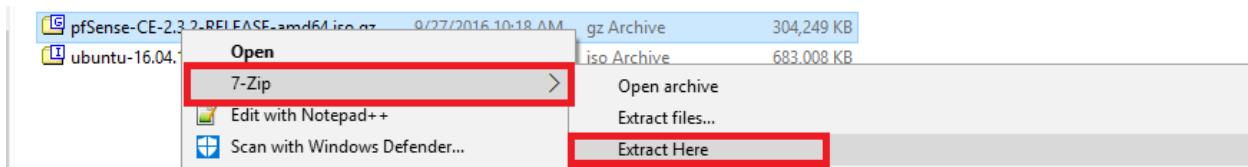
The next screen has us configure networking. Eventually we will have 3 network interfaces in total for this VM, but for now, select “Bridged Network”. We will add the other network connections in just a moment. Afterwards, click Next.



The next screen has us allocate disk space to create a virtual hard disk for to install our VM's OS and files. Select the "Create a virtual hard disk" radio button. Unless you have a compelling reason, the vhdx filename and location should not need to be changed. The "Size:" input box should be changed to "5" to allocate 5GB of disk space to this virtual machine. After altering the size of the virtual hard disk, click Next.

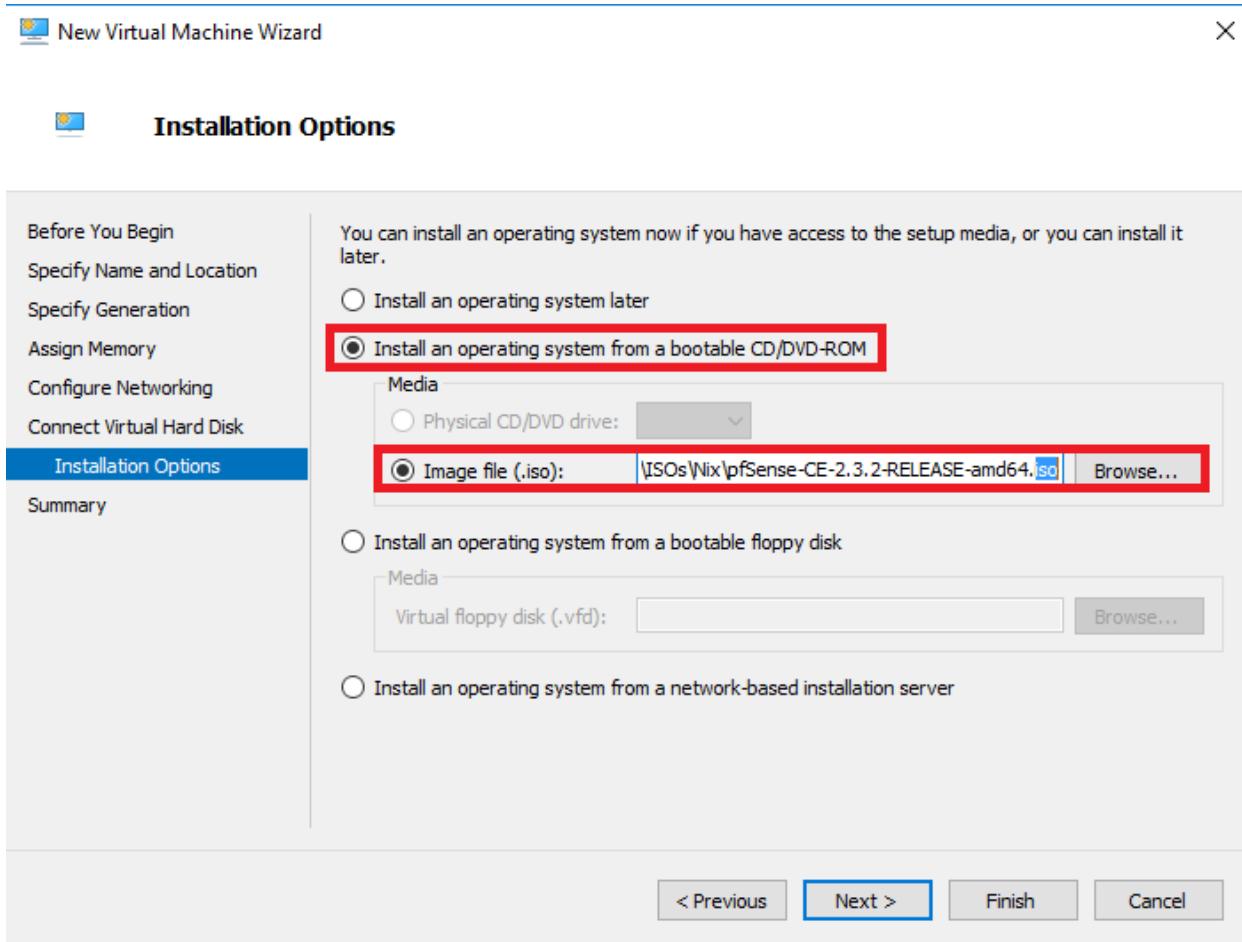


Before we move on to the next screen, if you haven't already downloaded your pfSense ISO I would suggest doing so now; the instructions are at the start of this section. If you downloaded 7-Zip, be sure to extract the ISO now as well. The current version of pfSense as of writing this is version 2.3.2, and the current ISO file name is "pfSense-CE-2.3.2-RELEASE-amd64.iso.gz". We need to decompress this ISO so that we can use it on the next screen. On your host, open file explorer, and navigate to where you download the pfSense ISO file. Right click on the file and choose the "7-Zip > Extract Here". This will make the file "pfSense-CE-2.3.2-RELEASE-amd64.iso" appear in the directory, right next to the original gz file.

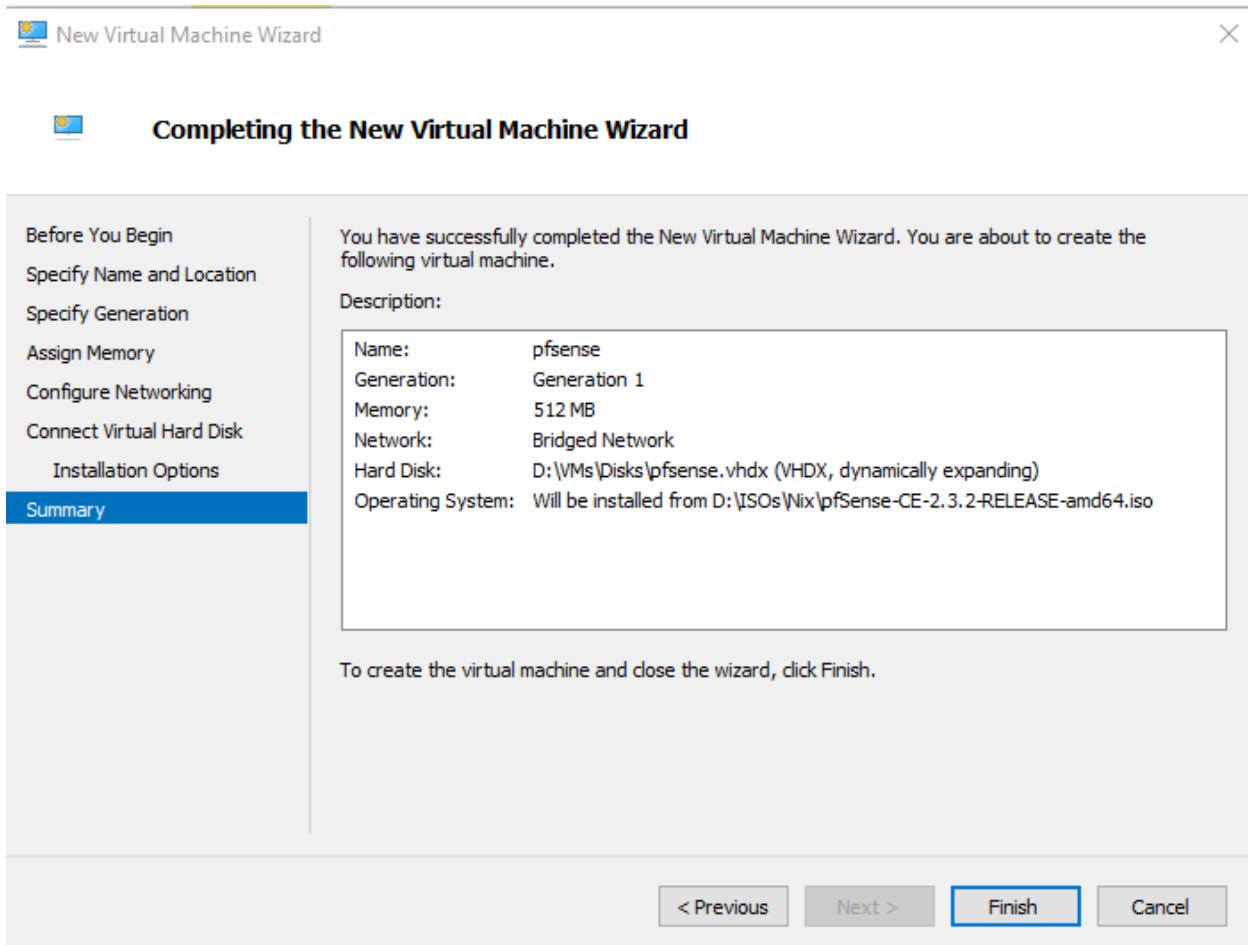


On the next screen in the New Virtual Machine Wizard, labeled "Installation Options", Click the radio button "Install an operating system from a bootable CD/DVD-ROM". This will make the "Media" portion of the screen active. Click on the radio button labeled "Image file (.iso):" then click browse to open the file explorer and locate the "pfSense-CE-2.3.2-RELEASE-amd64.iso"

file. In my case, I stored all my Unix/Linux ISO files in D:\ISOs\Nix in order to make them easier to find. After you have select the pfSense ISO, click Next.



The final screen verifies the settings we used to create our first VM. Confirm that your VM's configuration looks similar to what is presented on this screen, then click Finish.

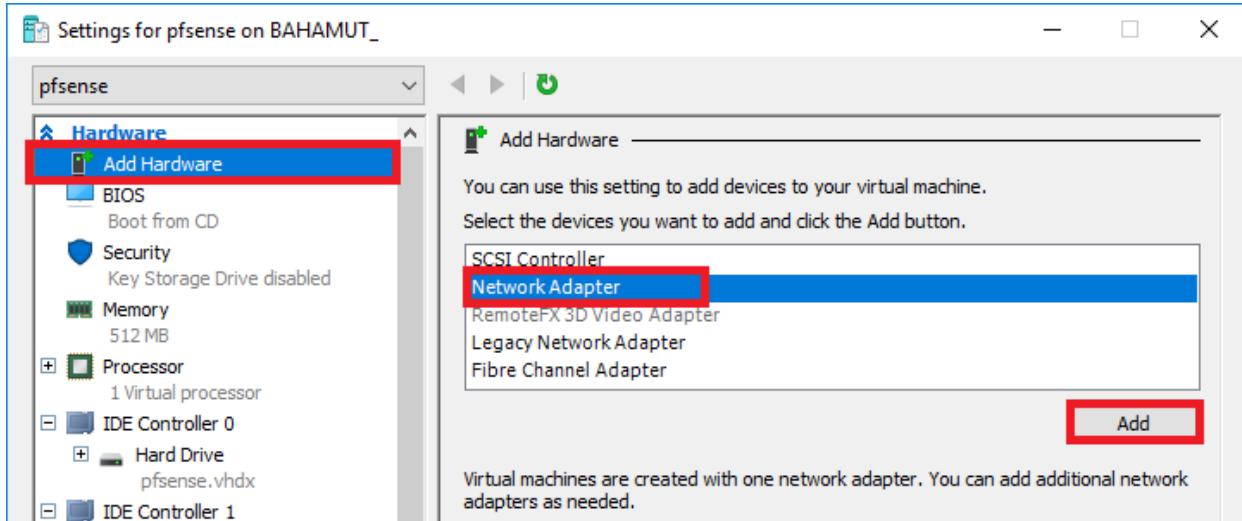


Initial VM Settings

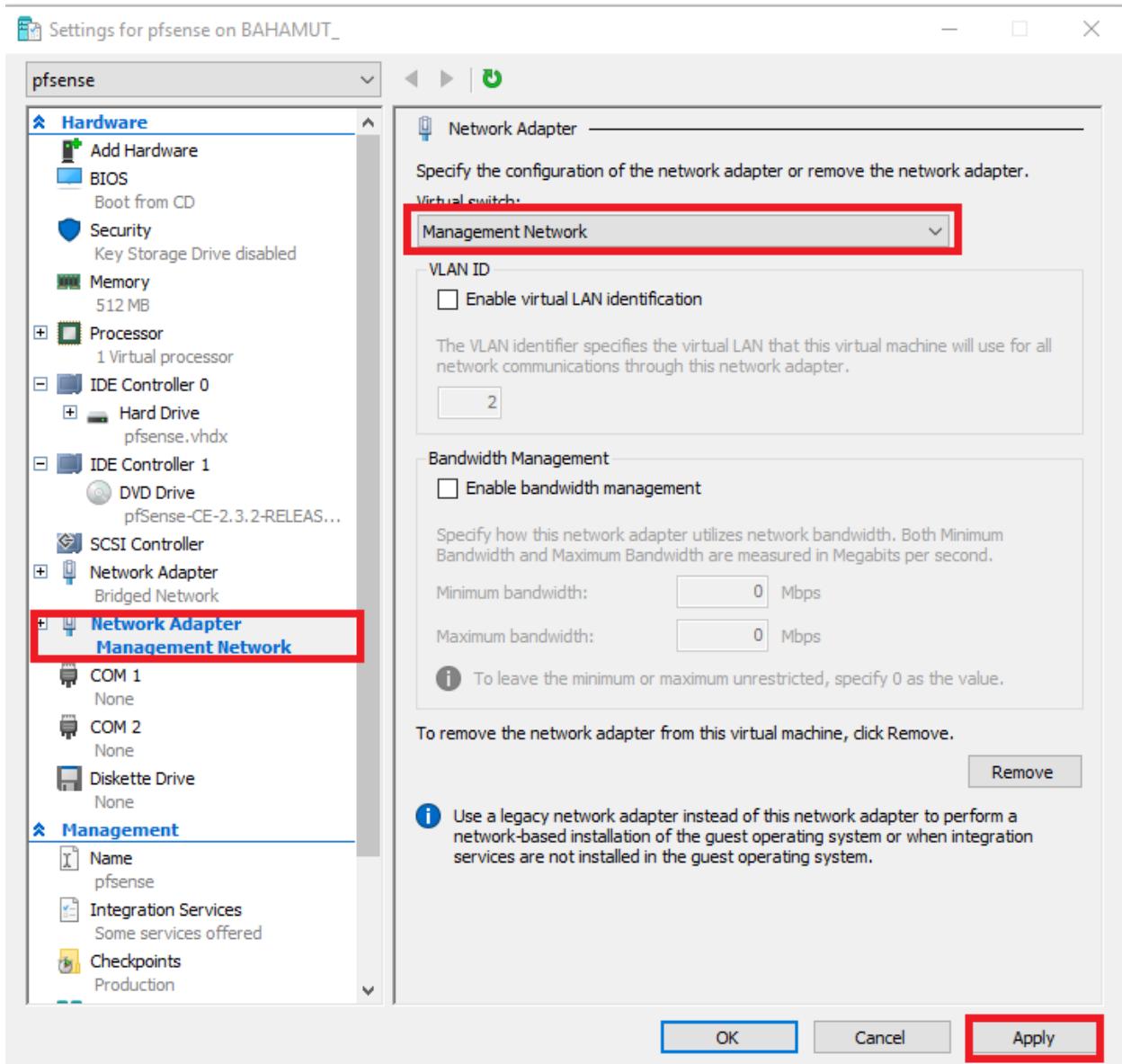
So now we have our first VM created. However, before we start it to install its operating system, there are a few more adjustments we have to make. In the Hyper-V Manager, under the “Virtual Machines” pane, right click on pfSense and select “Settings” to enter the VM’s setting menu.

The screenshot shows the Hyper-V Manager interface with the “Virtual Machines” pane. A table lists the VMs, showing “pfsense” as the current selection. A context menu is open over the “pfsense” row, with “Settings...” selected and highlighted in blue. Other options in the menu include Connect..., Start, Checkpoint, Move..., Export..., Rename..., Delete..., and Help.

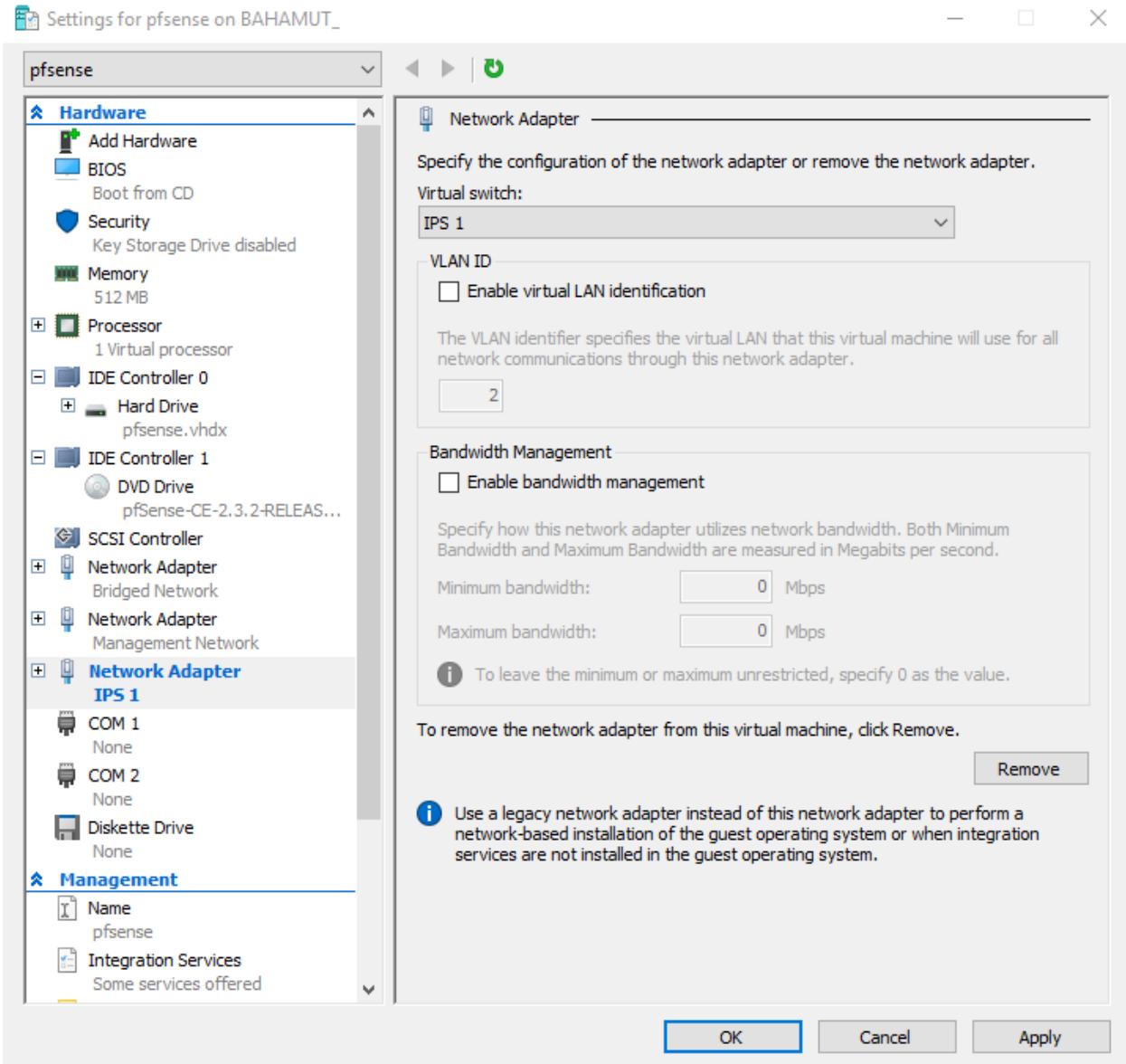
In the settings window for pfSense, The first thing we're going to do is add new hardware. Specifically, we have to add two more network adapters. Under "Hardware" in the left pane, click on "Add Hardware" to highlight, then on the right pane, click "Network Adapter" so that it is highlighted, then click the Add button.



A new network adapter should be added to your VM underneath the first network adapter we connected to the "Bridged Network" when we created this virtual machine. You should automatically be brought to this new network adapter to configure it. On the right side of the screen, on the drop-down menu labeled "Virtual Switch:" select "Management Network" then click Apply.

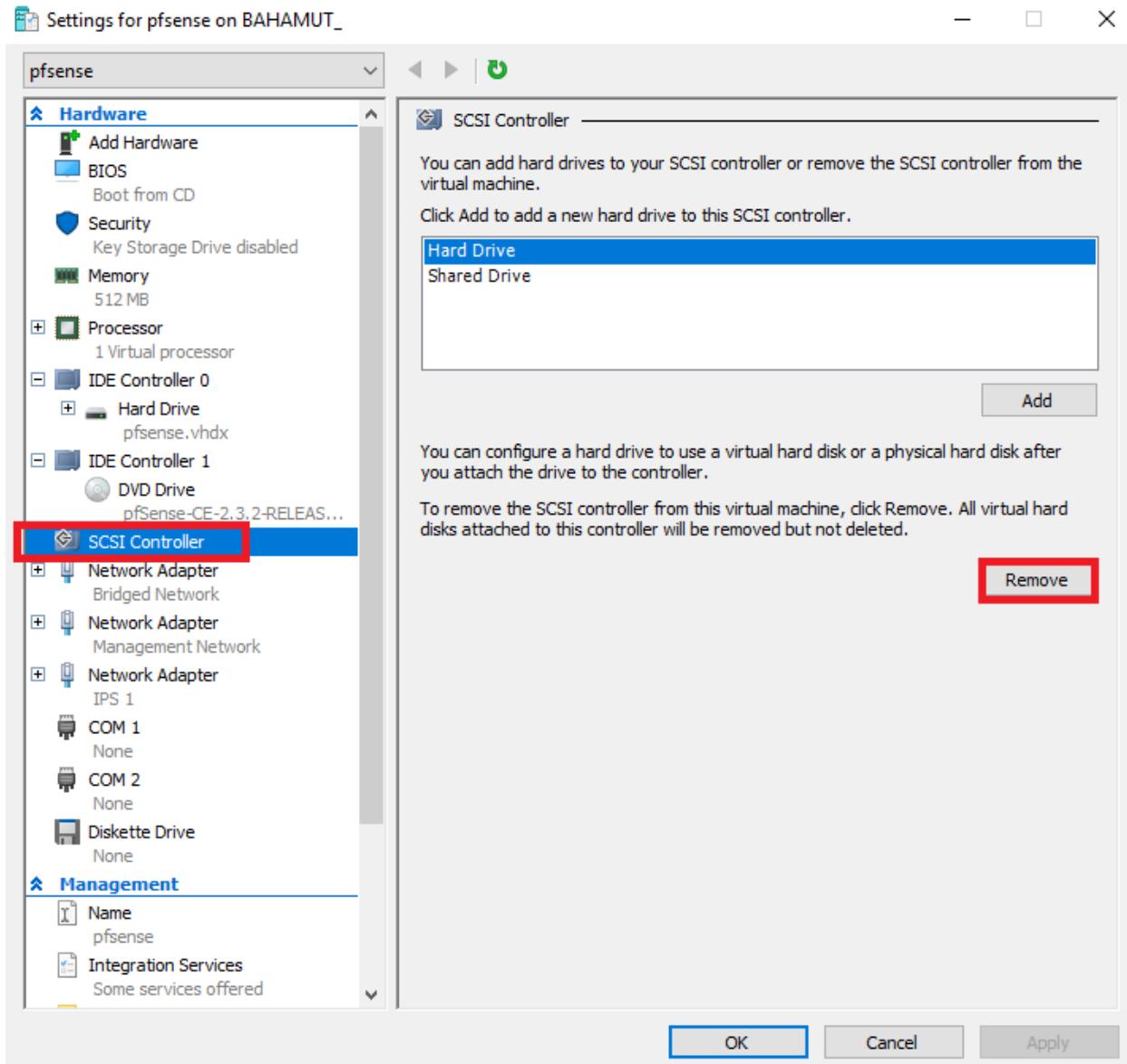


Click on “Add Hardware” again to highlight and select “Network Adapter” and the Add button again. This time we’re going to select “IPS 1” for the virtual switch drop-down. Click Apply once more.

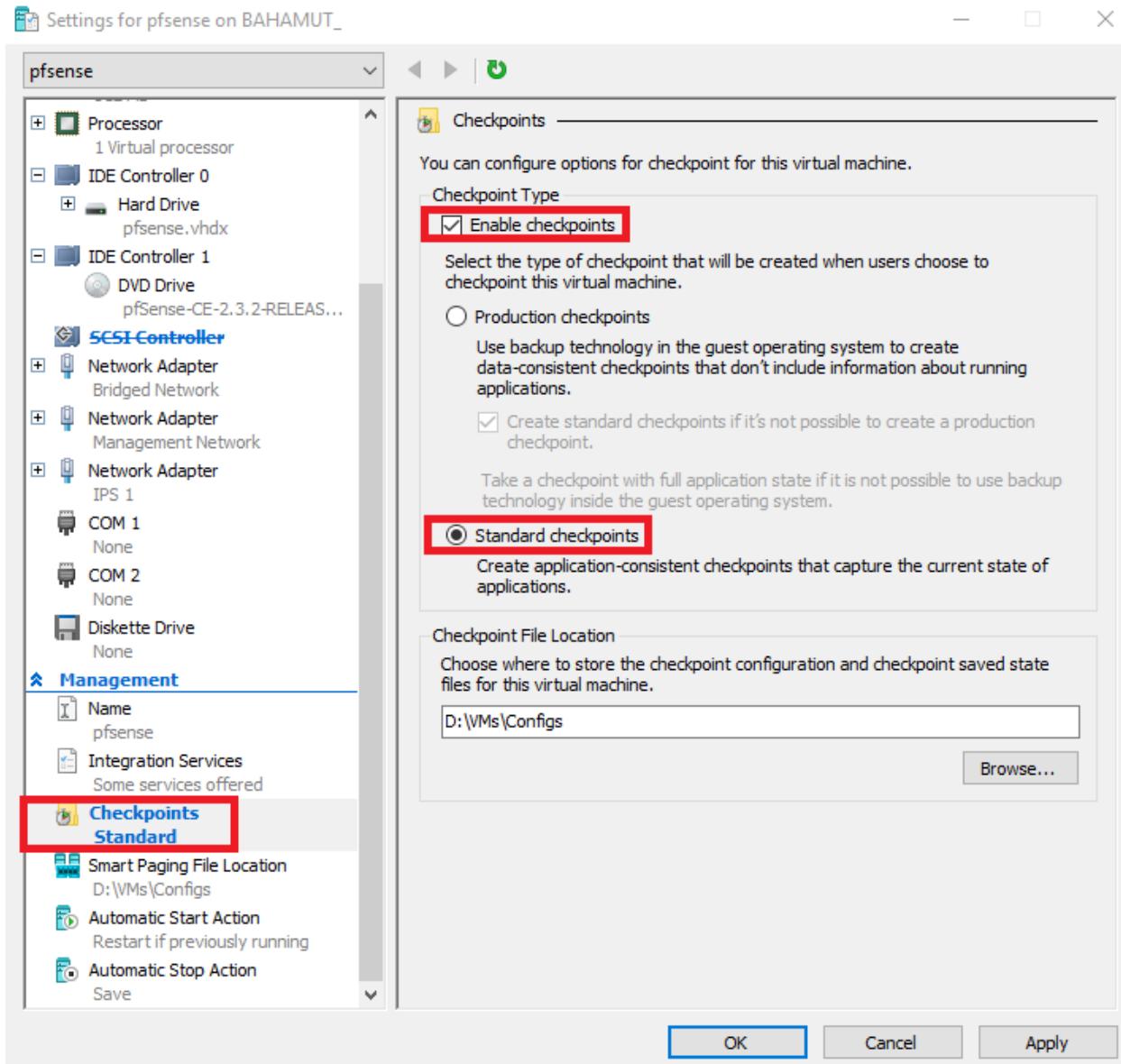


At this point, our pfSense VM should have three network cards attached to three different virtual switches.

Next, click on “SCSI Controller” under the “Hardware” list on the left pane, and click the Remove button that appears. We’re removing the SCSI controller because our VM has no use for it. Afterwards, click Apply.



Scroll down on the left pane, under "Management" and click on "Checkpoints". Under the section labeled "Checkpoint Type", ensure the "Enable checkpoints" checkbox is checked, click on the "Standard checkpoints" radio button, then click Apply.

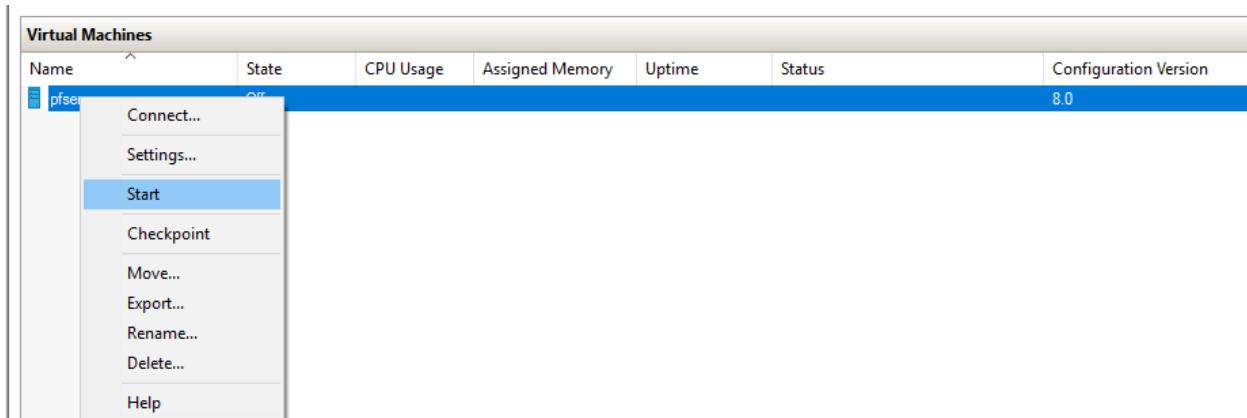


Without going too deep into it, checkpoints are essentially Hyper-V's terminology for snapshots, or a way to restore the VM back to a particular configuration in an instant. I was never able to get production checkpoints to work with pfSense, presumably because it's not supported. We will be using standard checkpoints for all of our VMs from here on out.

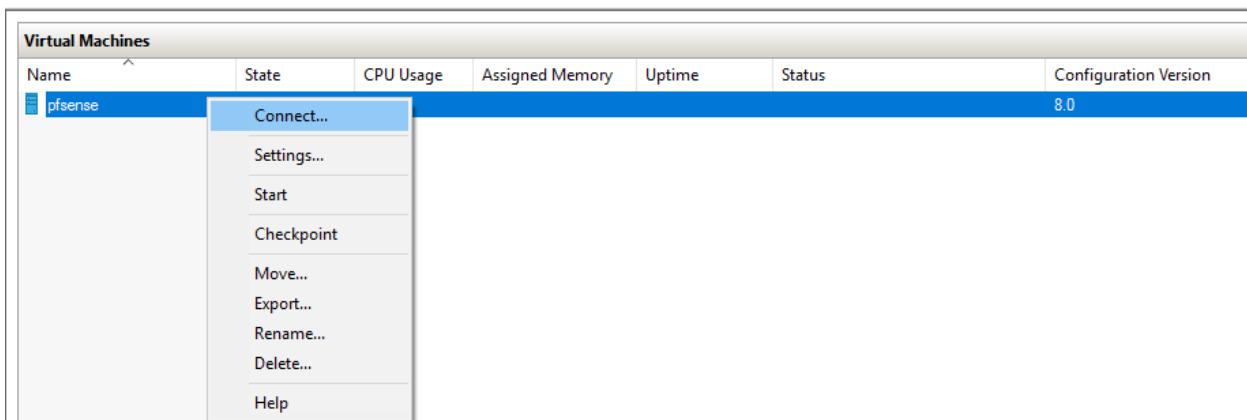
Note: If you are running Client Hyper-V on Windows 8.x, you may not have a choice between "Production" and "Standard" Checkpoints. You can disregard this section. Production checkpoints were added in with the Windows 10 version of Client Hyper-V. If you want to find out more about the differences between production and standard checkpoints, go here: <https://technet.microsoft.com/en-us/windows-server-docs/compute/hyper-v/manage/choose-between-standard-or-production-checkpoints-in-hyper-v>.

Installing pfSense

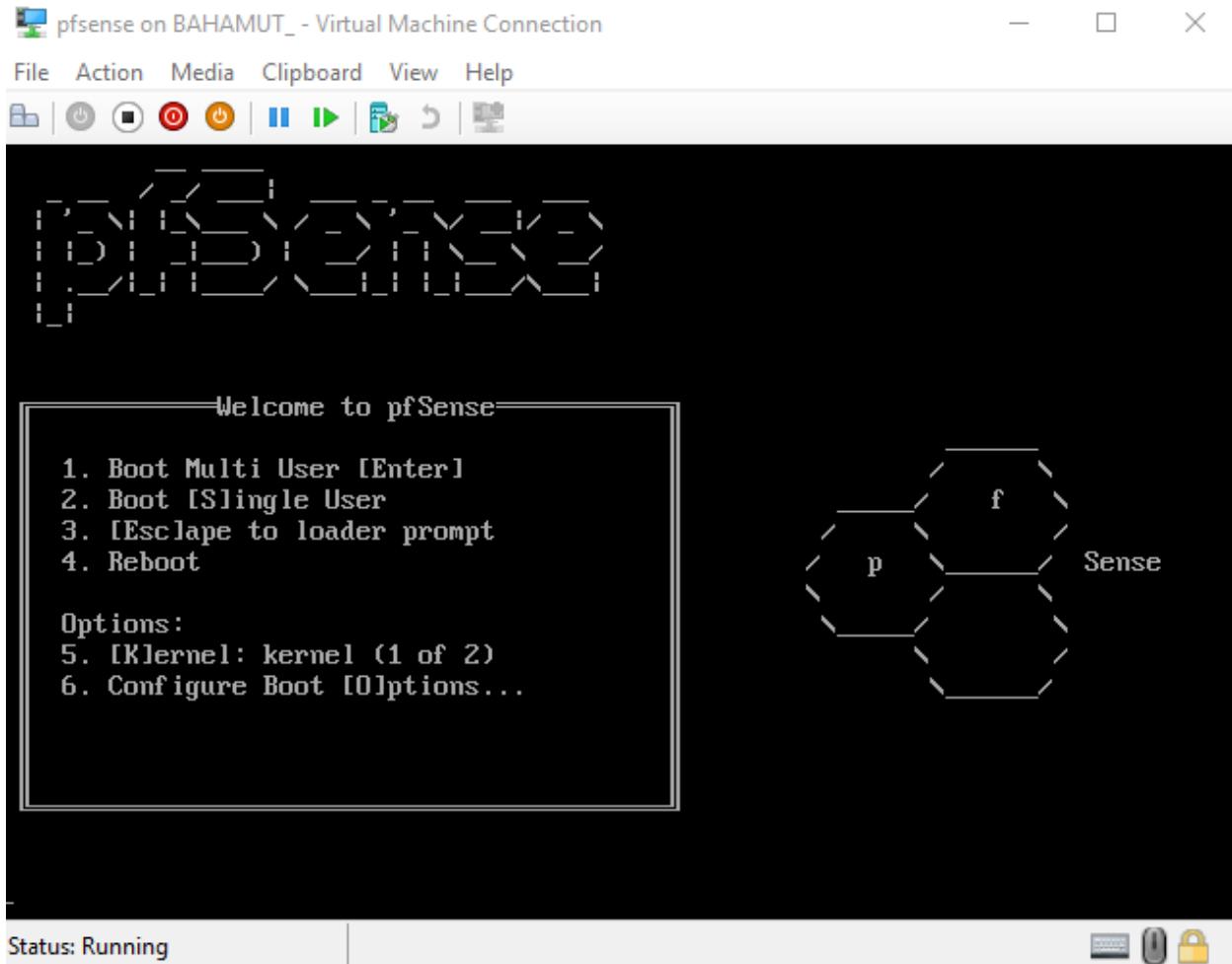
At this point, the VM is ready to be powered on to have the operating system installed. pfSense is really easy to install if you've installed any operating system's before. Power on the VM by right clicking it and selecting "Start" from the menu.



You should see the VM's state in the Hyper-V Manager console change from "Off" to "Running". Right click on the VM again, and select "Connect..." to connect a console session to the virtual machine.

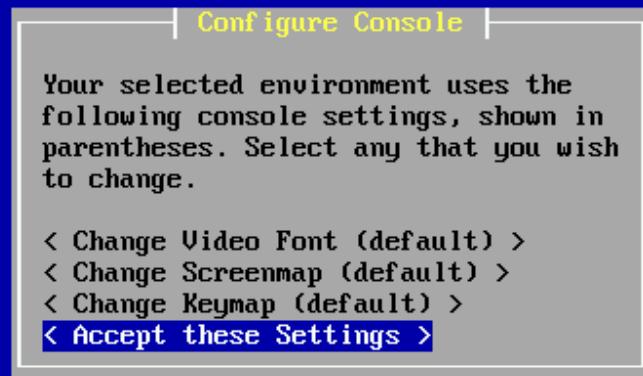


Installing pfSense is very straightforward if you've installed any OS from optical media or USB before, it should be fairly easy to figure out. You'll want to select the option 1 "Boot Multi User [Enter]" by hitting the enter key.



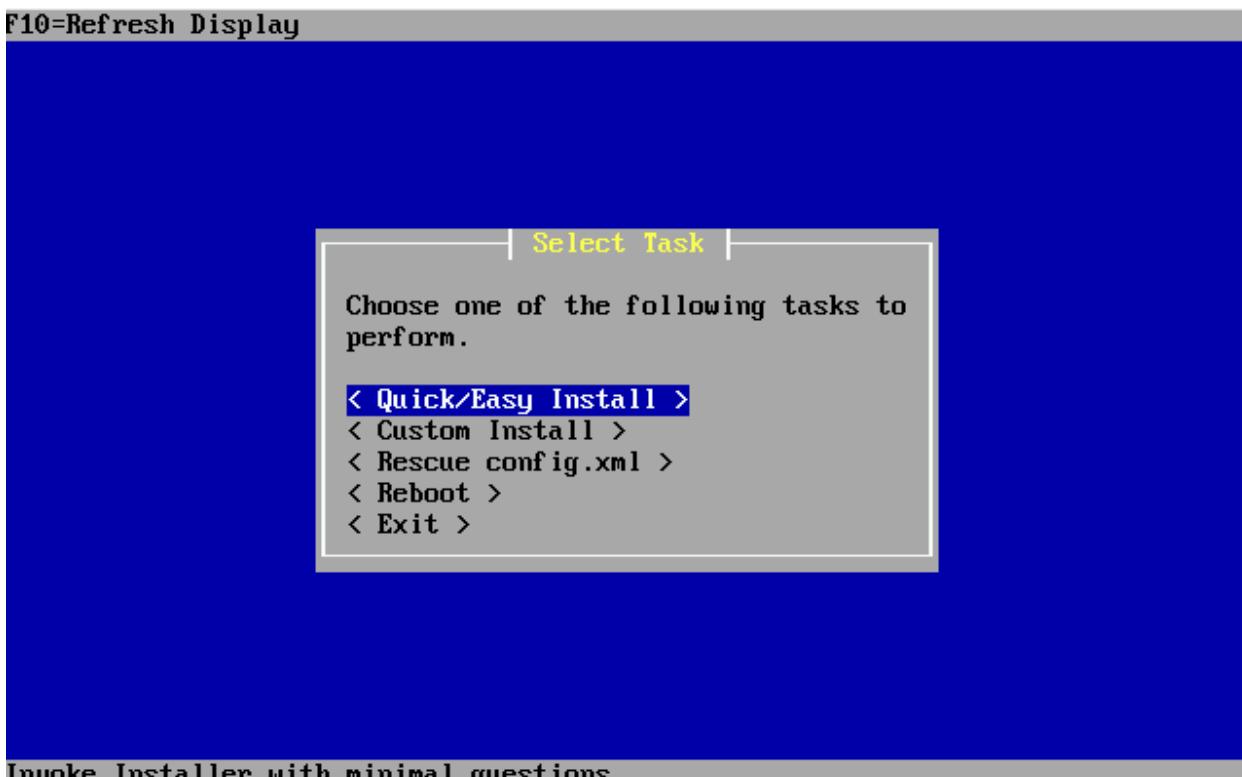
Let pfSense boot up, and the system should automatically run the installer. Adjust your video, screenmap, and keymap settings as necessary, then select “< Accept these Settings >”.

F10=Refresh Display



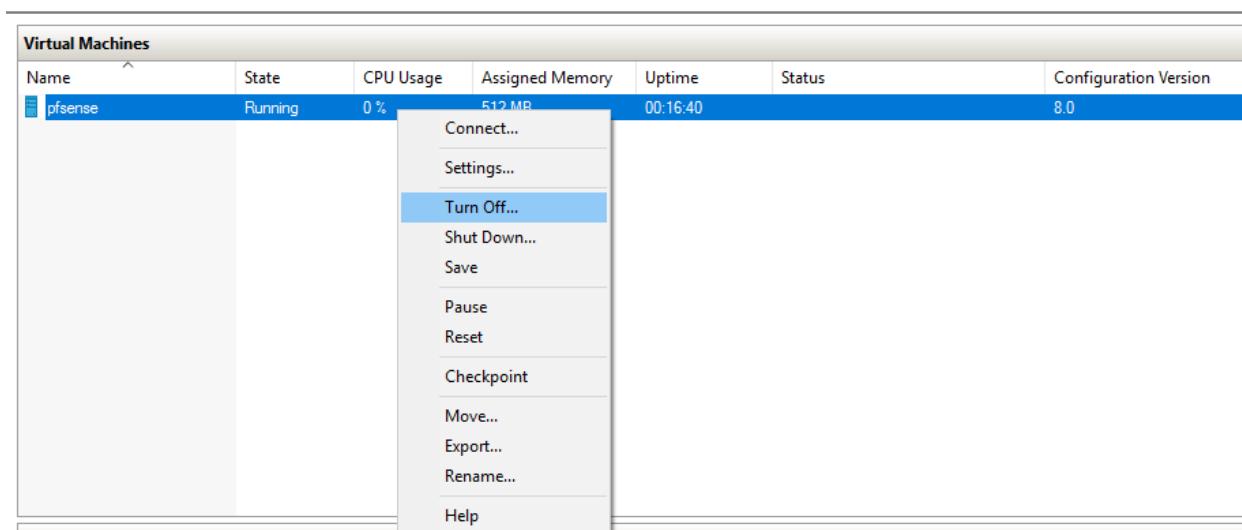
On the next screen, select “< Quick/Easy Install >” and let pfSense do all the heavy lifting. The next screen will inform you that the install will erase the contents of the hard disk. Since our virtual disk is already empty, this doesn’t matter in the least. Select OK, and let the installer run.

F10=Refresh Display



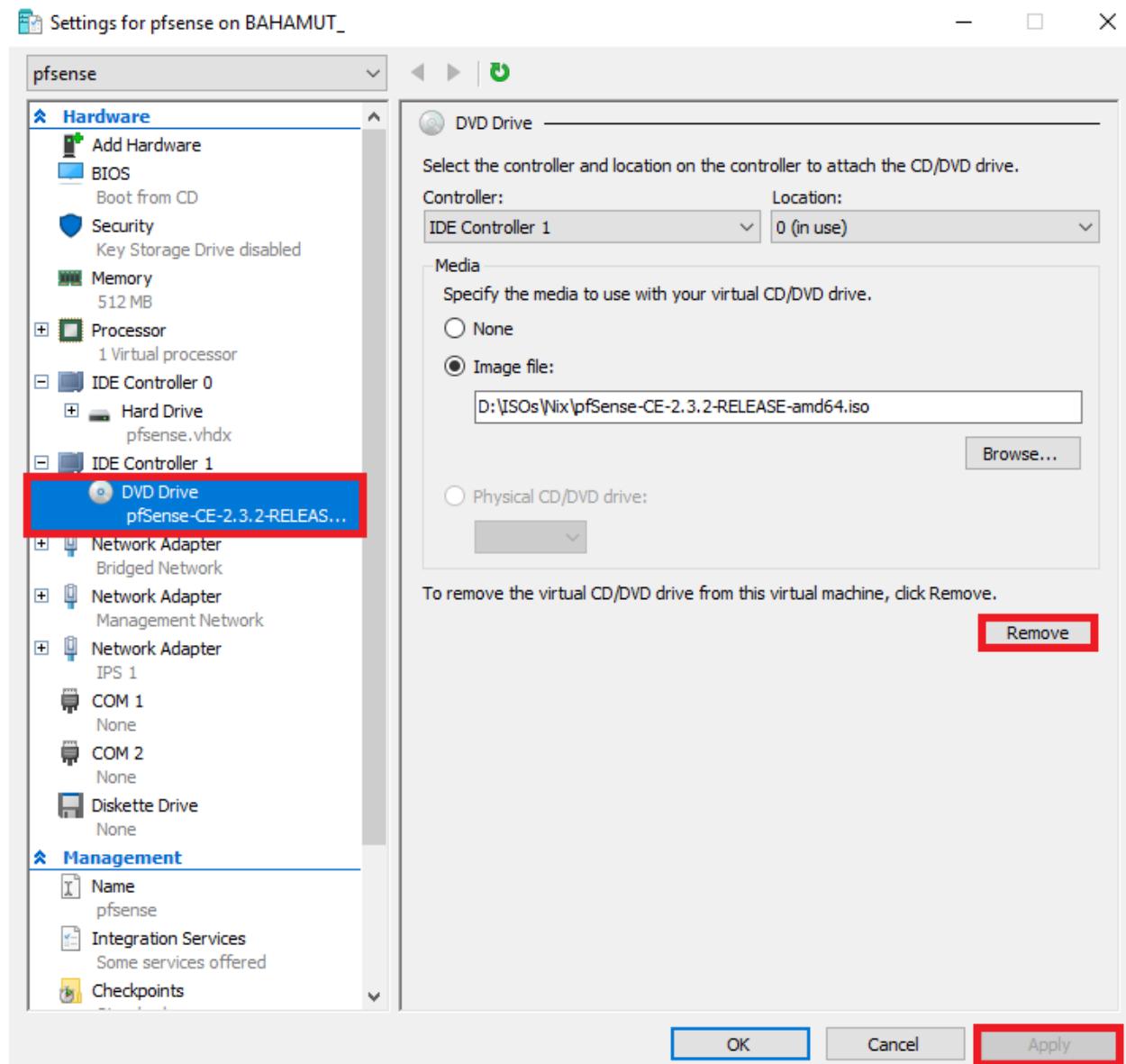
Invoke Installer with minimal questions

The installer will go through and install pfSense to the virtual hard disk. The installer will ask if you want to install a standard or embedded kernel. Be sure to select “< Standard Kernel >”. Finally, the installer informs you to reboot the machine to boot from the hard drive. The installation is done, however, we’re not going to reboot. In the Hyper-V Manager console, right click on the VM, and choose “Turn Off”. There are some minor modifications we want to make before we boot to the hard drive and begin configuring pfSense.



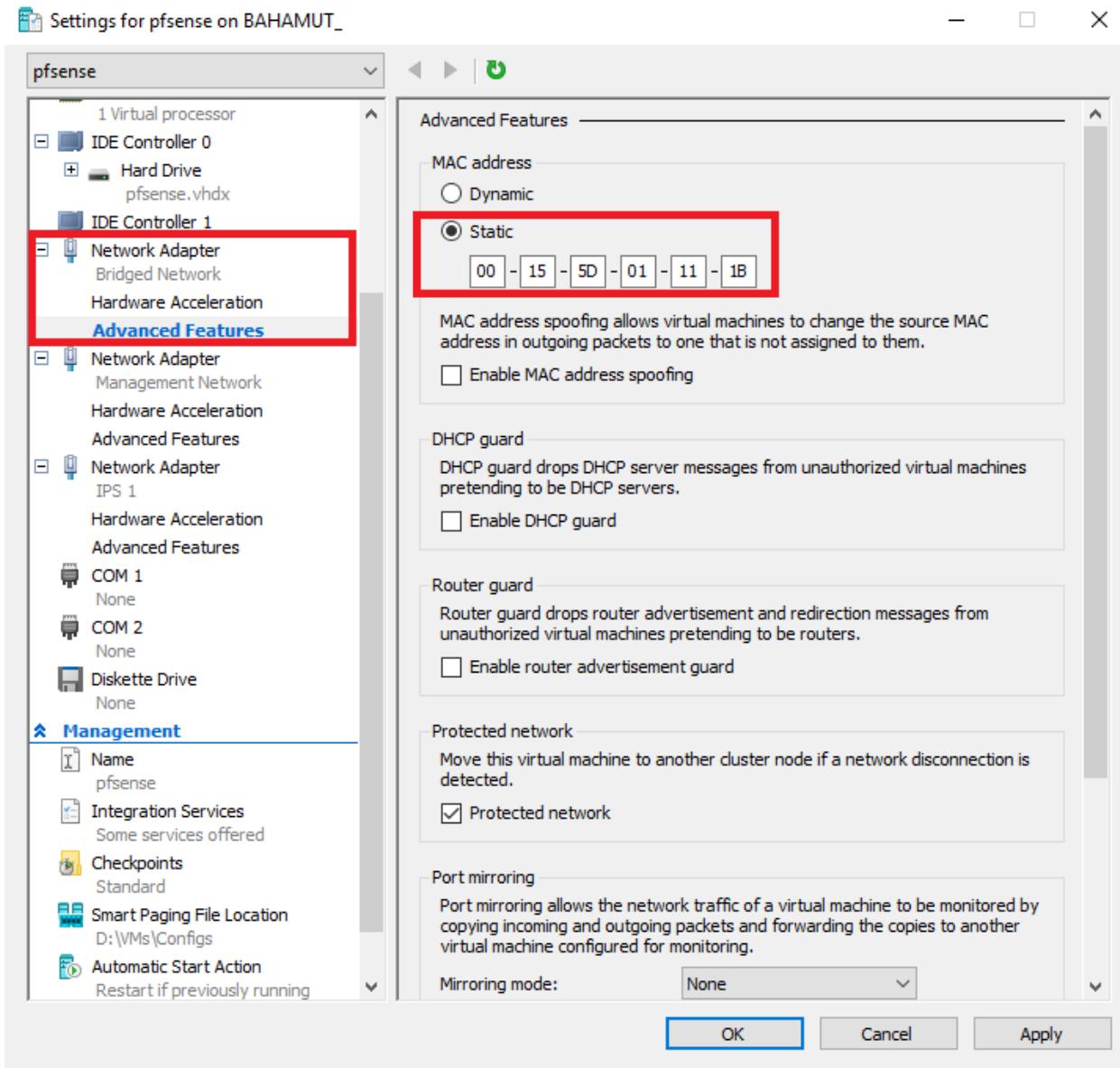
Final VM Settings

There is one last configuration change we want to make to the pfSense VM before we boot it up to finish configuring it. Make sure that the VM has been turned off and that the State reads as “Off” in the Hyper-V Manager. Next, right click on the VM in the Virtual Machines pane, and select “Settings...” to bring up the settings menu. On the left pane under “Hardware”, select “DVD Drive”. If you don’t see it, double click on “IDE Controller 1” to bring it up under the IDE controller, then click on it. On the “DVD Drive” screen, select the Remove button to uninstall the virtual DVD drive entirely, then click Apply.



Before closing the settings menu for the pfSense VM, We need to record the MAC addresses for each of our network adapters. Under “Hardware”, on the left pane, look for the network adapter labeled “Bridged Network.” Next to the adapter, you’ll notice a “+” sign. Either, click on

the symbol, or double click on “Network Adapter” to display “Hardware Acceleration” and “Advanced Features” under the network adapter. Click on “Advanced Features” to bring up the Advanced Features screen. Under the section labeled “MAC address”, make sure to document or write down the MAC address displayed.



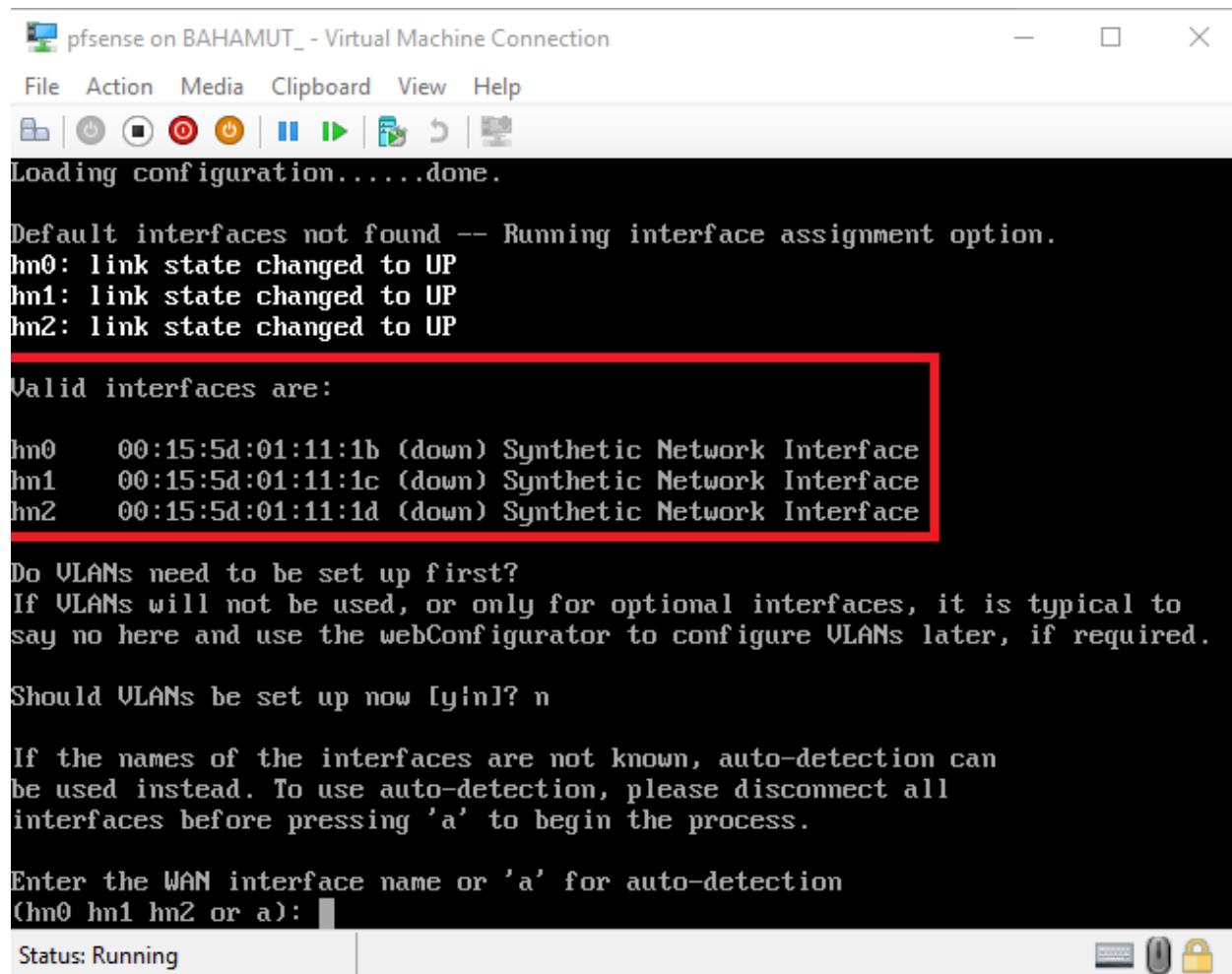
Repeat this process for the Network Adapters labeled “Management Network” and “IPS 1”.

Note: In the illustration above, I opted to change the MAC address setting from “Dynamic” To “Static”. You are not required to do this. When you first boot a VM in Hyper-V, the hypervisor will automatically generate a MAC address for you that begins with 00-15-5D (the OUI for “Microsoft”). If you select “Static”, You can overwrite the MAC address that Hyper-V assigns to the interface, to one of your choosing.. You can use this to define a MAC address with a custom OUI. You can use this to feature to fool some malware and/or applications that attempt to check

the MAC address to identify whether or not a system is a VM or not. **Keep this in mind for systems in which you plan on doing malware analysis.** For the time being however, I recommend leaving the “MAC address” radio button set to “Dynamic”, and recording the MAC address that appears.

Network Configuration

Now, we need to set up networking in the VM itself. In Hyper-V Manager, start the VM if it isn’t already running, then connect to the console. Wait for the VM to finish booting. You’ll be greeted by the “Assign Interfaces” wizard. The wizard will start by asking if you want to set up VLANs. Say no, because our virtual lab will NOT be using VLANs. Next, the wizard will ask you to define which interface should be assigned to what role. There should be three network interfaces displayed:



The screenshot shows a Windows application window titled "pfsense on BAHAMUT_ - Virtual Machine Connection". The menu bar includes File, Action, Media, Clipboard, View, and Help. Below the menu is a toolbar with icons for file operations. A message box displays the text: "Loading configuration.....done." followed by a series of interface status messages: "Default interfaces not found -- Running interface assignment option.", "hn0: link state changed to UP", "hn1: link state changed to UP", and "hn2: link state changed to UP". A red box highlights the text "Valid interfaces are:" followed by a list of three synthetic network interfaces: "hn0 00:15:5d:01:11:1b (down) Synthetic Network Interface", "hn1 00:15:5d:01:11:1c (down) Synthetic Network Interface", and "hn2 00:15:5d:01:11:1d (down) Synthetic Network Interface". Another red box highlights the question "Do VLANs need to be set up first?". The response "If VLANs will not be used, or only for optional interfaces, it is typical to say no here and use the webConfigurator to configure VLANs later, if required." is shown below. The next prompt "Should VLANs be set up now [y\ln]? n" is followed by a note about auto-detection. The final prompt "Enter the WAN interface name or 'a' for auto-detection (hn0 hn1 hn2 or a):" is shown at the bottom, with a status bar indicating "Status: Running".

When we performed the Initial VM Settings configuration, I had you statically configure the MAC addresses for each of the network adapters - Bridged, Management and IPS 1. Your goal right now is determine which network adapter (hn0, hn1, and hn2) corresponds to the network adapter configuration in Hyper-V.

For example, in the illustration above, hn0 with a MAC address of 00:15:5d:01:11:1b has the MAC address of the network adapter connected to the “Bridged Network”. The 1c MAC address of hn1 corresponds to the network adapter attached to the “Management Network” and 1d MAC address of hn2 corresponds to the network adapter attached to “IPS 1”. For our lab, the Bridged Network is our connection to the outside world, and therefore will serve as the “WAN” interface in pfSense. In my case, that means that hn0 is the “WAN” interface. The hn1 interface is connected to the “Management Network” and will serve as the “LAN” interface, and finally, hn2 connected to the “IPS 1 Network” will serve as the “OPT1” interface.

```
Valid interfaces are:

hn0      00:15:5d:01:11:1b (down) Synthetic Network Interface
hn1      00:15:5d:01:11:1c (down) Synthetic Network Interface
hn2      00:15:5d:01:11:1d (down) Synthetic Network Interface

Do VLANs need to be set up first?
If VLANs will not be used, or only for optional interfaces, it is typical to
say no here and use the webConfigurator to configure VLANs later, if required.

Should VLANs be set up now [y|n]? n

If the names of the interfaces are not known, auto-detection can
be used instead. To use auto-detection, please disconnect all
interfaces before pressing 'a' to begin the process.

Enter the WAN interface name or 'a' for auto-detection
(hn0 hn1 hn2 or a): hn0

Enter the LAN interface name or 'a' for auto-detection
NOTE: this enables full Firewalling/NAT mode.
(hn1 hn2 a or nothing if finished): hn1

Enter the Optional 1 interface name or 'a' for auto-detection
(hn2 a or nothing if finished): hn2
```

The setup script will ask you to confirm your interface choices. If you are satisfied, type “y” to let pfSense configure the interfaces and initial system settings. When everything is done, you’ll be dropped into a menu system.

Note: If for some reason the “Assign Interfaces” wizard was not run automatically when you first booted your pfSense VM, or you made a mistake and you need to correct it, option 1 on the pfSense main menu, “Assign Interfaces” Will run the wizard, and/or allow you to correct interface assignment errors as necessary.

Now that we have assigned the network interfaces, we have to configure IP addresses for these interfaces. In most cases, the WAN interface will automatically get an IP address from the device on your physical network that provides DHCP services. If this is not happening, you may have to troubleshoot your physical network. This is beyond the scope of our guide. As for the

LAN and OPT1 networks, we have to manually set the IP address, subnet mask, and DHCP scopes for these networks. Select option 2 in the pfSense main menu to get started.

```
*** Welcome to pfSense 2.3.2-RELEASE (amd64 full-install) on pfSense ***

WAN (wan)      -> hn0      -> v4/DHCP4: 192.168.1.21/24
LAN (lan)      -> hn1      ->
OPT1 (opt1)    -> hn2      ->

0) Logout (SSH only)          9) pfTop
1) Assign Interfaces          10) Filter Logs
2) Set interface(s) IP address 11) Restart webConfigurator
3) Reset webConfigurator password 12) PHP shell + pfSense tools
4) Reset to factory defaults 13) Update from console
5) Reboot system              14) Enable Secure Shell (sshd)
6) Halt system                15) Restore recent configuration
7) Ping host                  16) Restart PHP-FPM
8) Shell

Enter an option: [
```

The configuration wizard starts by asking for the interface the settings should be changed for. We have to go through the process two times, for the LAN and the OPT1 interface individually. Here are the settings I recommend:

LAN IP: **172.16.1.1**

LAN Subnet bit count: **24**

LAN upstream gateway address: <empty>

LAN IPv6 address: <empty>

Do you want to enable the DHCP server on LAN? **y**

LAN DHCP start address: **172.16.1.10**

LAN DHCP end address: **172.16.1.254**

Do you want to revert to HTTP as the webConfigurator protocol? **N**

OPT1 IP: **172.16.2.1**

OPT1 Subnet bit count: **24**

OPT1 upstream gateway address: <empty>

OPT1 IPv6 address: <empty>

Do you want to enable the DHCP server on OPT1? **y**

OPT1 DHCP start address: **172.16.2.10**

OPT1 DHCP end address: **172.16.2.254**

Do you want to revert to HTTP as the webConfigurator protocol? **n**

The wizard asks whether or not IPv6 should be configured. Say no, because that's a can of worms we're not going to deal with here. The wizard will also ask if would like to use DHCP

for the LAN and OPT1 networks. Respond with yes. For the LAN network, enter 172.16.1.10 as start address, and 172.16.1.254 as end address. For the OPT1 network, enter 172.16.2.10 as start address, and 172.16.2.254 as end address.

The reason we choose 1.10 and 2.10 as the DHCP start addresses is to reserve a few IP addresses for static DHCP allocations. Static DHCP allocations allow you to configure a DHCP server to always serve the same IP address when requested from a particular MAC address. We'll talk about this a little bit more later. For now, we should be done mucking around in the pfSense CLI. From here on out, we get to use the webUI to configure pfSense.

Note: If you are using 172.16.1.0/24 or the 172.16.2.0/24 network ranges in your physical network, choose another network range to assign to the LAN and OPT1 interfaces to avoid network conflicts.

Note: When you are configuring the LAN and OPT1 interface IP addresses, the configuration script will ask "Do you want to revert to HTTP as the webConfigurator protocol?" **Always say no to this.** This allows the web UI to accept HTTP logins, and makes your firewall credentials vulnerable to sniffing over the network.

Your pfSense main menu should look something like this when you're all done:

```

*** Welcome to pfSense 2.3.2-RELEASE (amd64 full-install) on pfSense ***

WAN (wan)      -> hn0          -> v4/DHCP4: 192.168.1.21/24
LAN (lan)      -> hn1          -> v4: 172.16.1.1/24
OPT1 (opt1)    -> hn2          -> v4: 172.16.2.1/24

0) Logout (SSH only)          9) pfTop
1) Assign Interfaces          10) Filter Logs
2) Set interface(s) IP address 11) Restart webConfigurator
3) Reset webConfigurator password 12) PHP shell + pfSense tools
4) Reset to factory defaults   13) Update from console
5) Reboot system               14) Enable Secure Shell (sshd)
6) Halt system                 15) Restore recent configuration
7) Ping host                   16) Restart PHP-FPM
8) Shell

Enter an option: 

```

If so, let's move on to setting up pfSense from the webConfigurator. Before doing so however, you must configure the vEthernet adapter attached to the "Management Network" on your Windows host. If you haven't already, you'll want to visit the "[Unbinding Network Protocols on Windows Virtual Adapters](#)" at a minimum. This document will guide you through configuring the vEthernet adapter on the "Management Network" with an IP address as well as unbinding network protocols on this adapter to increase the security of your hypervisor host. You may also want to consider "[Using Windows Firewall to Limit Exposure of Windows Hypervisor Hosts](#)" to further enhance the security of the Windows host when interacting with your lab VMs.

webConfigurator - Initial Setup

On the Windows Client Hyper-V host, open your favorite web browser (I prefer Firefox - <https://www.mozilla.org/en-US/firefox/new/>) and navigate to <https://172.16.1.1> (or the IP address you assigned to the LAN interface of your pfSense system). The default credentials for access to the web interface are admin/pfsense. If this is your first time logging in, pfSense takes you through a nice little setup wizard. I'll highlight some of the important things to take note of, and/or change as necessary:

Set the primary and secondary DNS servers you plan on using. I typically use 8.8.8.8 (Google public DNS) and 4.2.2.2 (Level 3 public DNS). If you're using this lab at work, your workplace may have restrictions and you may have to use their DNS servers instead.

Primary DNS Server	8.8.8.8
Secondary DNS Server	4.2.2.2

Uncheck the checkbox next to "Block private networks from entering via WAN" rule. Uncheck the checkbox next to "Block Bogen Networks" on this page as well.

RFC1918 Networks

Block RFC1918 Private Networks **Block private networks from entering via WAN**

When set, this option blocks traffic from IP addresses that are reserved for private networks as per RFC 1918 (10/8, 172.16/12, 192.168/16) as well as loopback addresses (127/8). This option should generally be left turned on, unless the WAN network lies in such a private address space, too.

The other default settings on the first time setup wizard should be fine. Please note that you will be asked to change the password for the admin account as a part of the setup wizard. Document the password, and keep it somewhere safe.

Our next order of business is to restrict access to the web interface of the firewall to the machine with the IP address 172.16.1.2. On the menu bar at the top of the page, select Firewall > Rules. Click on LAN to modify the firewall policy for the LAN interface. Click the “Add” button with the arrow facing up. This will add the firewall rule to the top of the firewall policy to where it will evaluated first. The “Edit Firewall Rule” page is fairly straightforward. I’ve highlighted the options to be aware of.

Edit Firewall Rule

Action Pass

Choose what to do with packets that match the criteria specified below.
Hint: the difference between block and reject is that with reject, a packet (TCP RST or ICMP port unreachable for UDP) is returned to the sender, whereas with block the packet is dropped silently. In either case, the original packet is discarded.

Disabled Disable this rule
Set this option to disable this rule without removing it from the list.

Interface LAN

Choose the interface from which packets must come to match this rule.

Address Family IPv4

Select the Internet Protocol version this rule applies to.

Protocol TCP

Choose which IP protocol this rule should match.

Source

Source Invert match. Single host or alias 172.16.1.2 /

Display Advanced Display Advanced

Destination

Destination Invert match. Single host or alias 172.16.1.1 /

Destination port range HTTPS (443) From Custom To Custom
Specify the destination port or port range for this rule. The "To" field may be left empty if only filtering a single port.

Extra Options

Log Log packets that are handled by this rule
Hint: the firewall has limited local log space. Don't turn on logging for everything. If doing a lot of logging, consider using a remote syslog server (see the [Status: System Logs: Settings](#) page).

Description pfSense strict anti-lockout

When you are done, click the Save icon at the bottom of the page. This will take you back to the previous page. A yellow dialogue box with a green button called “Apply Changes” will appear. Click this button to apply this new firewall rule. Next, we want to disable the default anti-lockout rule. Next, let’s navigate to Firewall > Aliases. On the Firewall Aliases page, click on “IP”, then click Add. Create an alias with the following settings:

Properties	
Name	RFC1918 The name of the alias may only consist of the characters "a-z, A-Z, 0-9 and _".
Description	An alias for all RFC1918 networks A description may be entered here for administrative reference (not parsed).
Type	Network(s)

Network(s)	
Hint	Networks are specified in CIDR format. Select the CIDR mask that pertains to each entry. /32 specifies a single IPv4 host, /128 specifies a single IPv6 host, /24 specifies 255.255.255.0, /64 specifies a normal IPv6 network, etc. Hostnames (FQDNs) may also be specified, using a /32 mask for IPv4 or /128 for IPv6. An IP range such as 192.168.1.1-192.168.1.254 may also be entered and a list of CIDR networks will be derived to fill the range.
Network or FQDN	10.0.0.0 / 8 10.x.x.x RFC 1918 networks Delete
	172.16.0.0 / 12 172.16.x.x RFC 1918 networks Delete
	192.168.0.0 / 16 192.168.x.x RFC 1918 networks Delete

Save + Add Network

Click Save to be brought back to the previous page, then click “Apply Changes”. We just created an alias for RFC1918 networks (local networks that are not routable through the public internet). This will come in handy later for creating firewall rules around these networks. Next, navigate to System > Advanced. Click to fill in the checkbox next to the option “Disable webConfigurator anti-lockout rule”, and click Save on the bottom of the page.

<input checked="" type="checkbox"/> Anti-lockout	<input checked="" type="checkbox"/> Disable webConfigurator anti-lockout rule
--	---

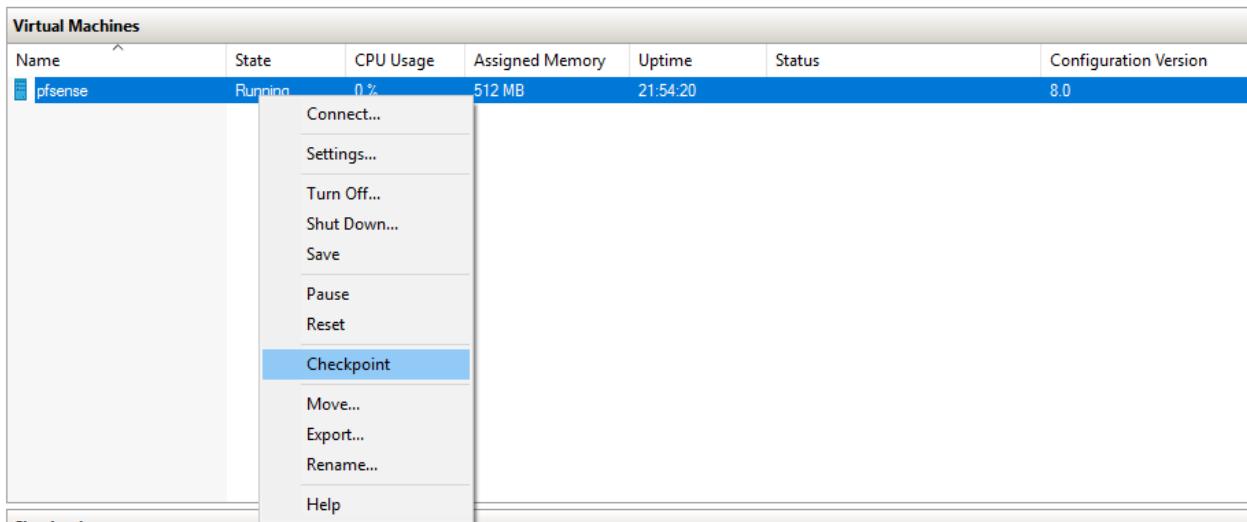
When this is unchecked, access to the webConfigurator on the LAN interface is always permitted, regardless of the user-defined firewall rule set. Check this box to disable this automatically added rule, so access to the webConfigurator is controlled by the user-defined firewall rules (ensure a firewall rule is in place that allows access, to avoid being locked out!) Hint: the “Set interface(s) IP address” option in the console menu resets this setting as well.

Making Checkpoints

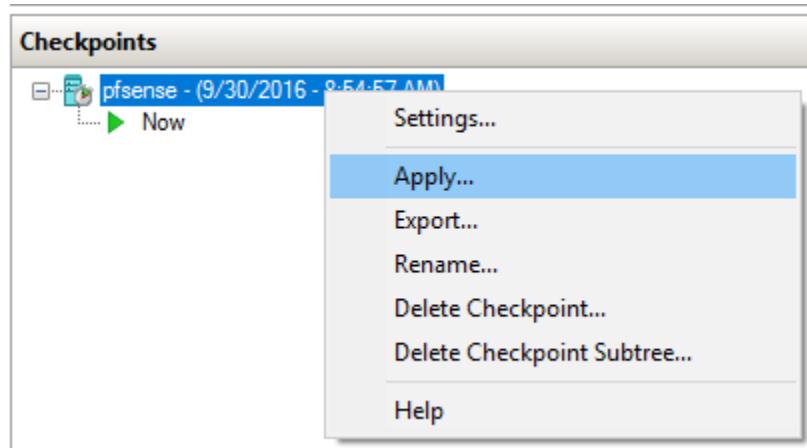
Snapshot management is an important part of any virtual lab. In Hyper-V terminology, snapshots are known as checkpoints. Since we’ve hit a pretty important milestone of getting the pfSense in a somewhat operational state, and mostly configured network-wise, the next step would be to create a snapshot so that if we make a mistake in the configuration, you have a known good, somewhat functional baseline to fall back to.

I’m going to show you how to create a snapshot for your pfSense VM. Before doing so, ensure that in the VM’s settings that checkpoints are enabled, and that “Standard” checkpoints have

been selected. In the Hyper-V Manager, under the Virtual Machines pane, right click on pfSense, and choose the Checkpoint option.



That is all there is to it. If you notice under the “Checkpoints” pane you now have a new checkpoint available. Underneath that check point you should see a green triangle and the word “Now”. If at any point you needed to revert back to this checkpoint, simply right click on the checkpoint you just made, and select “Apply” to restore the checkpoint.



pfSense Summary

At the end of all this, you should have a pfSense VM with the following settings:

- 512MB of RAM
- 5GB of disk
- 1 Virtual CPU
- 3 Network interfaces:
 - 1 Bridged Network (WAN interface)

- 1 Management Network (LAN interface)
- 1 IPS 1 Network (OPT1 interface)

pfSense should be configured as followed:

- The WAN interface should be the Hyper-V network adapter connected to the Bridged Network. This interface should automatically get an IP address via DHCP. If this isn't happening, get some help with whoever administers your local network, or start troubleshooting
- The LAN interface should have an IP address of 172.16.1.1.
- The LAN network should be 172.16.1.0 with a /24 (255.255.255.0) subnet mask
- DHCP should be configured for this network with a scope of 172.16.1.10-172.16.1.254
 - 172.16.1.3-172.16.1.9 are available for setting static DHCP mappings
- The OPT1 interface should have an IP address of 172.16.2.1
- The OPT1 network should be 172.16.2.0 with a /24 (255.255.255.0) subnet mask
- DHCP should be configured for this network with a scope of 172.16.2.10-172.16.2.254
 - 172.16.2.2-172.16.2.9 are available for static DHCP mappings
- 172.16.1.2 should have a firewall rule configured to allow it access to the firewall over HTTPS; this is the anti-lockout firewall rule
- You should have an alias for RFC1918 networks configured for use with the firewall policy.
- You should have at least one good checkpoint you can revert to if you need to redo a step and somehow got lost/confused

What's Next?

Now that you have an operational pfSense VM, your next steps are to read the [pfSense Firewall Rules and Network Services Guide](#). Make sure the firewall policies match those illustrated in the segmentation guide, and make sure that the pfSense VM is hosting all the services mentioned in the core network services guide to make your firewall fully functional and ready to handle your lab network. Once you have verified that your firewall policies and services are properly configured, **SNAPSHOT THE VM BEFORE** moving on, and trying your hand at setting up the remaining VMs.

Your Turn

The pfSense VM required a lot of custom configuration, both in Hyper-V, as well as in the VM itself once the OS had been installed. Now that we have done that together, I'm going to have you create the rest of the virtual machines yourself -- with the exception of the Metasploitable 2 VM since it's a special case for reasons you will see momentarily. Below are a set of spec sheets to help you create the remaining virtual machines for the lab on your own. Think of them as checklists you should be able to run through on your own.

All of our Linux-based VMs are built on Debian Linux based distros (Both Ubuntu and Kali Linux are derived from Debian), so setup and configuration of all of our lab VMs should be very similar.

Kali Linux VM

Kali Linux is a popular penetration testing distro. Popular because hackers and script kiddies are lazy, and practically everything you need for performing a penetration test or joining anonymous is installed by default. Kali is going to be our loud, obnoxious attacker that we're going to use as a noise generator for the express purpose of generating events on our IPS VM. I want you to perform the following tasks:

- Download Kali Linux 64-bit here: <https://www.kali.org/downloads/>
- Create a VM with the following settings:
 - Name the vm "kali"
 - Make the vm "Generation 1"
 - Allocate 4GB of RAM
 - Uncheck the "Use Dynamic Memory for this virtual machine." feature
 - Connect the VM's network adapter to the "IPS 1" virtual switch
 - Allocate 80 GB of space for the vhdx file
 - Under "Installation Options" select the "Install from a bootable CD/DVD-ROM" radio button, then select the "Image file (.iso)" option, and browse to the Kali Linux ISO you downloaded
- Once the VM is created, adjust the following settings:
 - 1-2 virtual CPUs (virtual CPUs can be adjusted under Settings > Hardware > Processor and adjust "Number of Virtual Processors:" to 2)
 - Remove the SCSI Controller
 - If you are running Windows 10, change the "Checkpoint"s option (Under Settings > Management > Checkpoints) to "Standard"
- Install Kali Linux
 - **Note:** If you configured the firewall on the pfSense VM, and decided to install and enable squid proxy services, during the apt package retrieval portion of the installer, you will be ask if your network uses a proxy server, and asked to enter the address of the proxy server as well as a username and password (if necessary). You will need to enter `http://172.16.2.1:3128` as the proxy server address. There is no username or password required to use the squid proxy.
- After the install is completed, shut down, or turn off the VM and adjust the following settings:
 - Remove the DVD drive located under IDE Controller 1
 - Under Network Adapter > Advanced Features in the MAC address section, document the MAC address for this network adapter.
- Log in to the pfSense web UI, navigate to Services > DHCP Server, and add a static mapping to the OPT1 interface DHCP, for the MAC address of adapter 1 for the Kali Linux VM; assign it the IP address 172.16.2.2, making sure to enter a description that tells you which VM this static mapping belongs to!
- Power on the virtual machine and do the following:

- Verify that the static mapping for the VM's IP address worked; open a terminal/shell and run `ifconfig -a | less` and verify that eth0 or whatever the network adapter's name is in the VM was given the IP address you configured its static mapping for in pfSense.
 - If the VM does not have an ip address, run the `dhclient` command to have the VM request an IP address from the DHCP server.
- Verify the virtual machine can reach the internet.
 - In a terminal/shell run the command `ping -c 5 www.google.com`
 - This will confirm DNS can resolve domain names, and that the VM can reach the internet
- Update/patch the VM
 - In a terminal/shell, run the following, and depending on your network connection, etc., be prepared to wait:
 - `export DEBIAN_FRONTEND=noninteractive; apt-get -q update; apt-get -y -q dist-upgrade`
- Create a checkpoint for the VM

Now you should have a functional Kali Linux VM. At this point, the virtual machine should be fully operational, and you should be able to control it from the VM's console. However, if you want to enable and configure SSH access to this host, I have configured a guide for doing so. Jump to "[How to Enable SSH on Kali Linux](#)" to enable the ssh service for this VM.

SIEM VM

SIEM is shorthand for Security Intrusion Events Manager. This is fancy security nomenclature for "log aggregator"; we will be having our IPS VM log its events here. We're going to be running Splunk on this VM. Splunk is a commercial program for managing logs on a pretty large scale. By default, and with no licensing, Splunk only allows you to collect or "index" 500MB worth of logs per day however, there are /ways/ around this that we'll talk about later. I want you perform the following tasks to set up the SIEM VM:

- Download the latest Ubuntu Server 64-bit LTS release. In our case, this is Ubuntu Server 16.04.1, which can be found here: <http://www.ubuntu.com/download/server>
- Create a VM with the following settings:
 - Name the VM "siem"
 - Make the VM "Generation 1"
 - Allocate 4GB of RAM
 - Uncheck the "Use Dynamic Memory for this virtual machine." feature
 - Connect the VM's network adapter to the "Management Network" virtual switch
 - Allocate 80 GB of space for the vhdx file
 - Under "Installation Options" select the "Install from a bootable CD/DVD-ROM" radio button, then select the "Image file (.iso)" option, and browse to the Ubuntu Server ISO you downloaded
- Once the VM is created, adjust the following settings:

- 1-2 virtual CPUs (virtual CPUs can be adjusted under Settings > Hardware > Processor and adjust “Number of Virtual Processors:” to 2)
 - Remove the SCSI Controller
 - If you are running Windows 10, change the “Checkpoints” option (Under Settings > Management > Checkpoints) to “Standard”
- Install Ubuntu Server
 - Be sure to install the command line utilities and SSH Server role when asked. You should not need to install any other roles or features for this server.
 - **Note:** If you configured the firewall on the pfSense VM, and decided to install and enable squid proxy services, during the apt package retrieval portion of the installer, you will be asked if your network uses a proxy server, and asked to enter the address of the proxy server as well as a username and password (if necessary). You will need to enter http://172.16.1.1:3128 as the proxy server address. There is no username or password required to use the squid proxy.
- After the install is completed, shut down, or turn off the VM and adjust the following settings:
 - Remove the DVD drive located under IDE Controller 1
 - Under Network Adapter > Advanced Features in the MAC address section, document the MAC address for this network adapter.
- Log in to the pfSense web UI, navigate to Services > DHCP Server, and add a static mapping to the OPT1 interface DHCP, for the MAC address of adapter 1 for the SIEM VM; assign it the IP address 172.16.1.3, making sure to enter a description that tells you which VM this static mapping belongs to!
- Power on the Virtual Machine and do the following:
 - Verify that the static mapping for the VM’s IP address worked; open a terminal/shell and run `ifconfig -a | less` and verify that eth0 or whatever the network adapter’s name is in the VM was given the IP address you configured its static mapping for in pfSense.
 - If the VM does not have an ip address, run the `dhclient` command to have the VM request an IP address from the DHCP server.
 - Verify the virtual machine can reach the internet.
 - In a terminal/shell run the command `ping -c 5 www.google.com`
 - This will confirm DNS can resolve domain names, and that the VM can reach the internet
 - Update/patch the VM
 - In a terminal/shell, run the following, and depending on your network connection, etc., be prepared to wait:
 - `export DEBIAN_FRONTEND=noninteractive; apt-get -q update; apt-get -y -q dist-upgrade`
 - You will need to be the root user to run these commands. Run `sudo su -` and enter your user’s password to gain access to a root shell in order to run these commands.
 - Create a checkpoint for the VM

IPS VM

This VM is going to be responsible for running the AFPPACKET bridge between the IPS 1 and IPS 2 virtual switches. Perform the following tasks to install the IPS VM:

- If you installed the SIEM VM already, you should already have Ubuntu Server downloaded. If not, download the latest Ubuntu Server 64-bit LTS release. In our case, this is Ubuntu Server 16.04.1, which can be found here:
<http://www.ubuntu.com/download/server>
- Create a VM with the following settings:
 - Name the VM “IPS”
 - Make the VM “Generation 1”
 - Allocate 2GB of RAM
 - Uncheck the “Use Dynamic Memory for this virtual machine.” feature
 - Connect the VM’s first network adapter to the “Management Network” virtual switch
 - Allocate 80 GB of space for the vhdx file
 - Under “Installation Options” select the “Install from a bootable CD/DVD-ROM” radio button, then select the “Image file (.iso)” option, and browse to the Ubuntu Server ISO you downloaded
- Once the VM is created, adjust the following settings:
 - 1-2 virtual CPUs (virtual CPUs can be adjusted under Settings > Hardware > Processor and adjust “Number of Virtual Processors:” to 2)
 - Remove the SCSI Controller
 - Under “Add New Hardware”, We need to add two more Network Adapters. For now, **DO NOT** connect either of these network adapters to a virtual switch; leave them as “Not connected” for right now
 - The only interface that should be connected to a virtual switch right now should be the first network adapter that was connected to the “Management Network”
 - If you are running Windows 10, change the “Checkpoints” option (Under Settings > Management > Checkpoints) to “Standard”
- Install Ubuntu Server
 - The installer will reach a section titled “Configure the network” and ask you which network card is the primary network interface for this system
 - Usually, these interfaces are labeled eth0, eth1, and eth2, but may be labeled differently in your case.
 - In almost all cases, the first network adapter in Client Hyper-V that is added to the VM will line up with the first network interface listed on the “Configure the network screen (This is almost always eth0). Hit Enter, and wait for it to try and configure the interface via DHCP
 - The installer will try to get an IP address for the network interface via DHCP. If this is not successful, select the <Go Back> option, and select one of the other two network interfaces, and try again

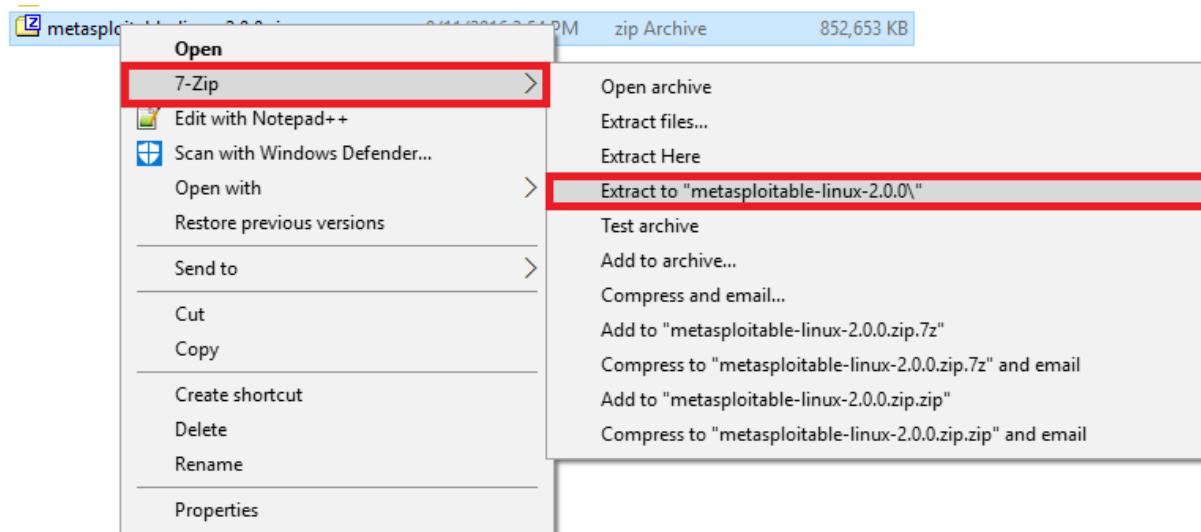
- If none of the interfaces work, verify the pfSense VM is running, connected to the “Management Network”, and that the DHCP service is enabled, then try again until DHCP configuration is successful.
- Be sure to install the command line utilities and SSH Server role when asked. You should not need to install any other roles or features for this server.
- **Note:** If you configured the firewall on the pfSense VM, and decided to install and enable squid proxy services, during the apt package retrieval portion of the installer, you will be asked if your network uses a proxy server, and asked to enter the address of the proxy server as well as a username and password (if necessary). You will need to enter `http://172.16.1.1:3128` as the proxy server address. There is no username or password required to use the squid proxy.
- After the install is completed, shut down, or turn off the VM and adjust the following settings:
 - Remove the DVD drive located under IDE Controller 1
 - Select the second and third network adapters, and connect them to the “IPS 1” and “IPS 2” networks respectively.
 - Under Network Adapter > Advanced Features in the MAC address section, document the MAC address for all three network adapters on this VM.
- Log in to the pfSense web UI, navigate to Services > DHCP Server, and add a static mapping to the OPT1 interface DHCP, for the MAC address of the network adapter attached to the “Management Network” virtual switch; assign it the IP address 172.16.1.4, making sure to enter a description that tells you which VM this static mapping belongs to!
- Power on the Virtual Machine and do the following:
 - Verify that the static mapping for the VM’s IP address worked; open a terminal/shell and run `ifconfig -a | less` and check the mac addresses of all three interfaces to verify which interface is connected to the “Management Network”
 - Verify that the VM has an IP address on the “Management Network” (172.16.1.4) and *only* on that network.
 - If the VM does not have an IP address, run the `dhclient -i [interface connected to the management network]` to have the VM request an IP address from the DHCP server.
 - If for some reason the VM has an IP address from the 172.16.2.0/24 network, you can run `ifdown [interface name]` to fix this.
 - Verify the Virtual Machine can reach the internet.
 - In a terminal/shell run the command `ping -c 5 www.google.com`
 - This will confirm DNS can resolve domain names, and that the VM can reach the internet
 - Update/patch the VM
 - In a terminal/shell, run the following, and depending on your network connection, etc., be prepared to wait:
 - `export DEBIAN_FRONTEND=noninteractive; apt-get -q update; apt-get -y -q dist-upgrade`

- You will need to be the root user to run these commands.
Run `sudo su` - and enter your user's password to gain access to a root shell in order to run these commands.
- Create a checkpoint for the VM

Metasploitable 2

Setting up Metasploitable 2 on Hyper-V is slightly more complex than the just creating a virtual machine. In order to install metasploitable 2, you'll need a special powershell script in order to convert the VMware VMDK disk file that it is distributed as into a VHD file that Hyper-V can make sense of.

First and foremost, go and download metasploitable 2 from <https://sourceforge.net/projects/metasploitable/files/Metasploitable2/metasploitable-linux-2.0.0.zip/download>. Once the zip file has been downloaded, you'll need a compression utility to unzip the zip file. If you followed the instructions for decompressing the pfSense ISO earlier, you should have 7-Zip installed. Once downloaded, I moved the zip file to D:\VMs, where the rest of my Hyper-V VMs are stored. We'll need to extract the zip file to get access to the VMDK. Right click on the zip file and select 7-Zip > Extract to "metasploitable-linux-2.0.0\"



Next, open your favorite web browser, and visit <https://www.microsoft.com/en-us/download/details.aspx?id=42497>. Since Microsoft has a habit of occasionally reorganizing their website, and breaking links, You want to look for the "Microsoft Virtual Machine Converter" (Specifically, I'm using version 3.0, but as always, use the latest version available). Download the .msi file and install it.

Please note, that depending on your system configuration, you may need to open a powershell.exe shell as administrator and run Set-ExecutionPolicy Bypass in order to run the

powershell module this application relies on. For more information, please review this MS Technet article: <https://technet.microsoft.com/en-us/library/hh849812.aspx>

Once everything is installed, open an elevated powershell prompt (right click on the icon and select “Run as Administrator”) and run the following command: Import-Module ‘C:\Program Files\Microsoft Virtual Machine Converter\MvmcCmdlet.psd1’

If everything ran correctly you should get a new prompt back with no errors. The following command will extract the VMDK file from D:\VMs\metasploitable-linux-2.0.0\Metasploitable2-Linux\Metasploitable.vmdk, and dump the converted VHDX file to D:\VMs\Disks

```
ConvertTo-MvmcVirtualHardDisk -SourceLiteralPath D:\VMs\metasploitable-linux-2.0.0\Metasploitable2-Linux\Metasploitable.vmdk -VhdType DynamicHardDisk -VhdFormat vhdx -destination D:\VMs\Disks\
```

Adjust the drive letters and directory paths as necessary for your Windows host. If everything was executed properly, you should be greeted with this in the powershell prompt, as it converts the disk format into something Hyper-V can use:

Administrator: Windows PowerShell

```
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

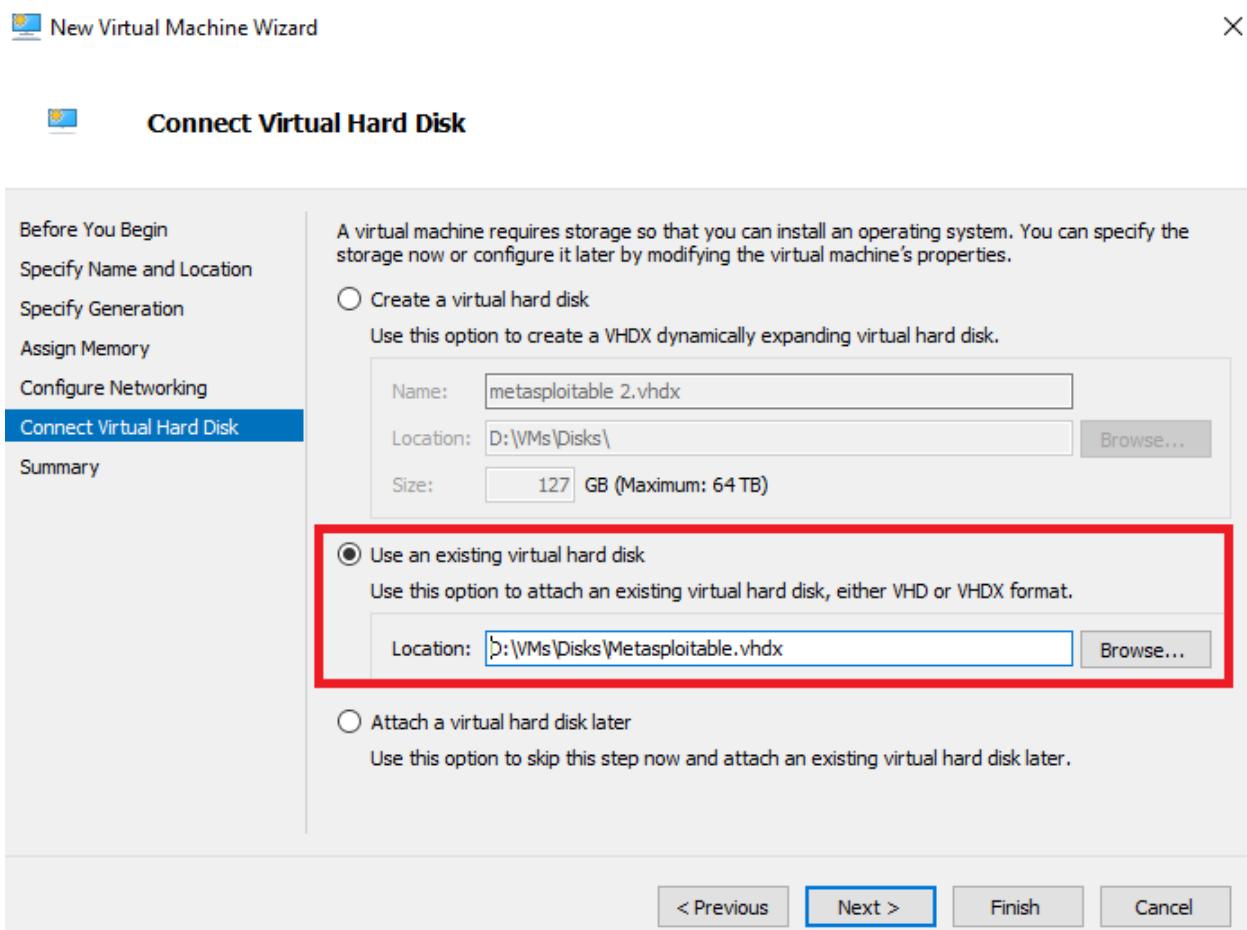
Converting drive D:\VMs\metasploitable-linux-2.0.0\Metasploitable2-Linux\Metasploitable.vmdk to dynamic VHDX.
Copying data from source to destination.
[ooooooooooooooo]

Destination           Source
-----              -----
D:\VMs\metasploitable-linux-2.0.0\Metasploitable.vhdx D:\VMs\metasploitable-linux-2.0.0\Metasploitable2-Linux\Metasp...

PS C:\WINDOWS\system32> ConvertTo-VmcVirtualHardDisk -SourceLiteralPath D:\VMs\metasploitable-linux-2.0.0\Metasploitab
e2-Linux\Metasploitable.vmdk -VhdType DynamicHardDisk -VhdFormat vhdx -destination D:\VMs\Disks\
```

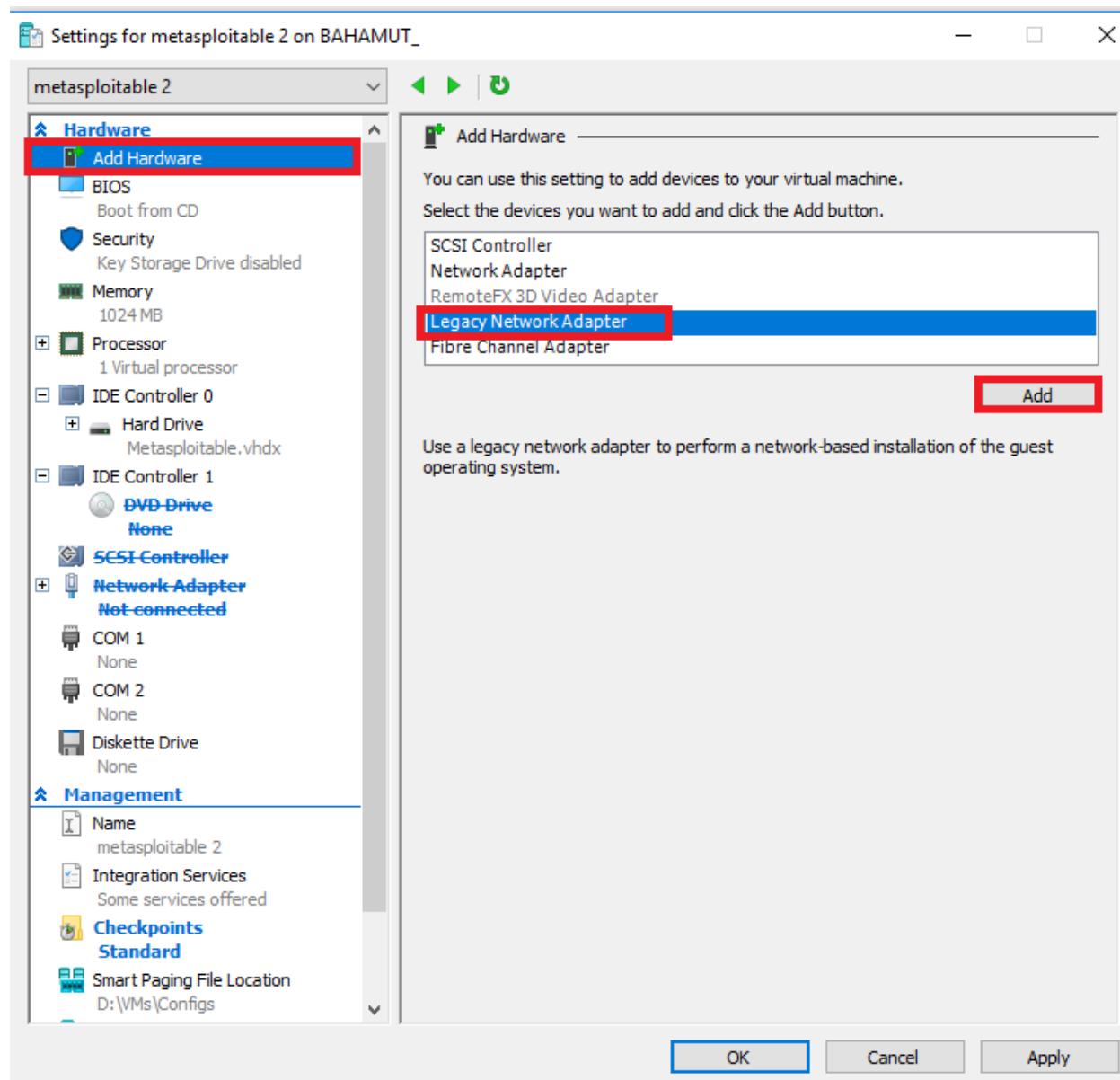
Now that we have the VHDX, we need to create the VM to run it. Open Hyper-V Manager and create a new VM, just like with the previous VMs. Give it the following settings:

- Name: Metasploitable 2
- Generation: 1
- RAM: 512MB
- Uncheck “Use Dynamic Memory for this virtual machine.”
- Leave the network adapter in its default state (Not Connected)
- On the “Connect Virtual Hard Disk” screen, click the “Use an existing virtual hard disk” radio button, then browse to D:\VMs\Disks\Metasploitable.vhdx (or the location your Metasploitable vhdx is located)

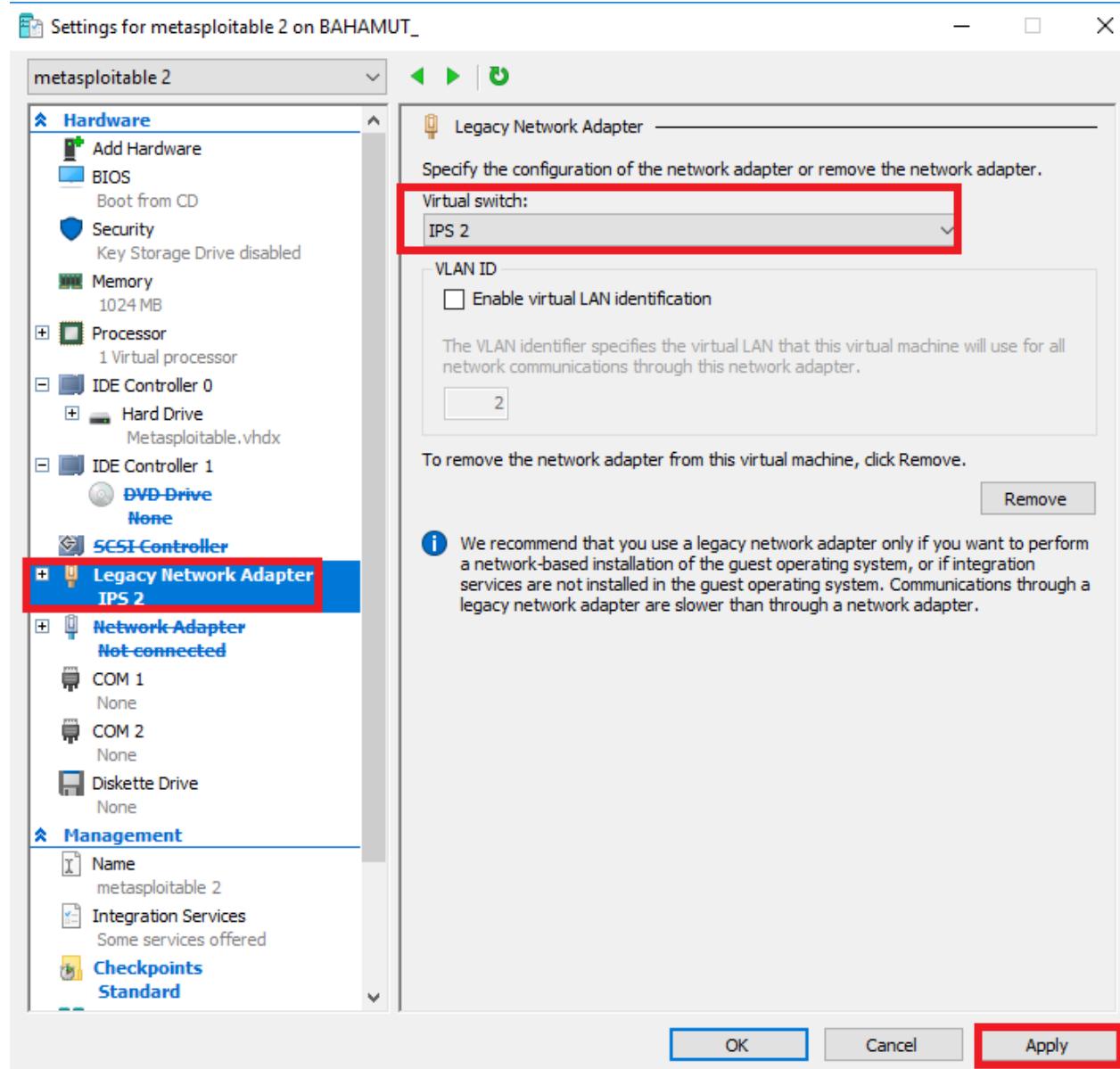


The installer will skip the installation options screen, so click Finish to create the VM. In the Hyper-V Manager, open the settings for the “Metasploitable 2” VM and make the following changes:

- Remove the SCSI Controller
- Remove the DVD Drive
- Remove the Network Adapter
- Reconfigure Checkpoints from Production to Standard
- Choose “Add New Hardware” add the “Legacy Network Adapter”
-



Click on the “Legacy Network Adapter” and make sure it is connected to the “IPS 2” virtual switch. When you’re all done, click apply to apply all of these changes.



The reason we’re installing the “Legacy Network Adapter” is because metasploitable 2 is very, *VERY* old. It’s based on Ubuntu 8.04, which as of writing this, was released over 8 years ago. It doesn’t have the drivers for the standard Hyper-V network adapter available. However, the legacy network adapter works just fine, so we’re going to use that for this VM instead.

Now, power on the VM. We only need to make sure that it reaches the login prompt. You will not be able to reach the internet as of right now. Once you verify that the VM boots properly, turn it off, then enter the VM’s settings again. Select “Advanced Features” under the legacy network

adapter, and make sure to record the MAC address, so that we can create a static DHCP mapping for the “Metasploitable 2” virtual machine under OPT1 DHCP in pfSense.

At this point, the VM should be bootable. Feel free to power it on to make sure there aren’t any problems. If you’re greeted with a login prompt, then this indicates everything is working fine. Power down the virtual machine and create a checkpoint.



The screenshot shows a terminal window with a black background and white text. At the top, there is a decorative header consisting of various symbols like asterisks and underscores. Below the header, the text reads:

```
Warning: Never expose this VM to an untrusted network!
Contact: msfdev[at]metasploit.com
Login with msfadmin/msfadmin to get started
metasploitable login: _
```

Note: This VM is located on “IPS 2” network, which we haven’t bridged to “IPS 1” network with the “IPS” VM yet. So in spite of having a static DHCP mapping, it has no way of reach out to the “pfSense” virtual machine for an IP address yet. We are going to fix this soon.

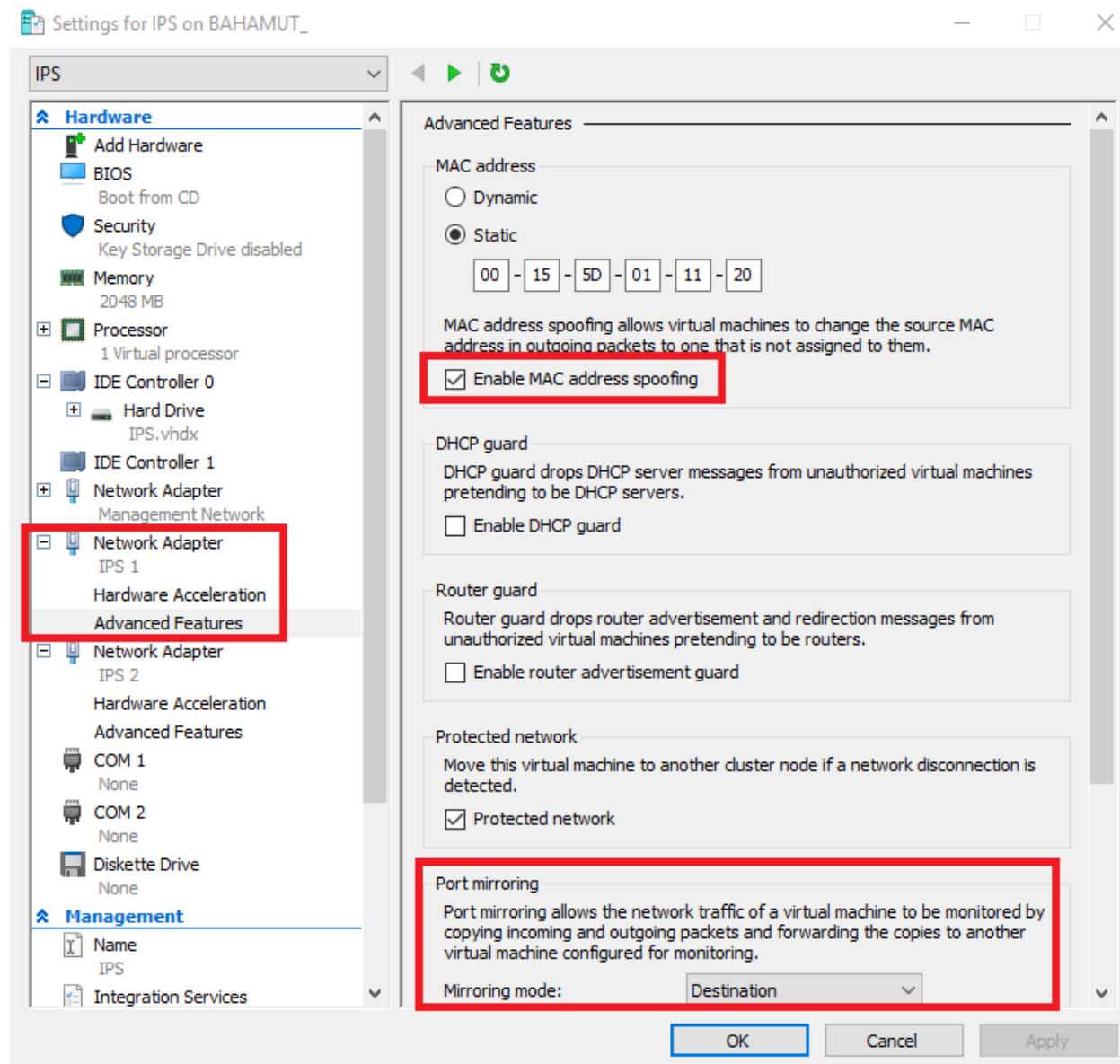
Port Mirroring and MAC spoofing

Mirroring network traffic in Hyper-V requires you have to specify whether or not a VM’s network adapter is a source or a destination for mirroring network traffic. The system that will be collecting network traffic must have its network adapter specified as a destination, and the hosts we want to collect traffic from must be specified as sources.

In order for the AFPACKET bridge we will be implementing to work properly, the network adapters of all the VMs connected to the “IPS 1” and “IPS 2” networks must be defined as port mirroring sources, except for the network adapters connected to our IPS VM, which must be specified as port mirroring destinations. In addition to that, we also have to enable MAC address spoofing on the IPS VM for the traffic to be passed correctly. We will walk through configuring the IPS VM as a destination, and the pfSense VM as a source together, then I will have you configure the remaining VMs on your own.

Configuring the IPS VM as a Port Mirroring Destination

In Hyper-V Manager, open up the Settings menu for the “IPS” VM. Click on the network adapter connected to the “IPS 1” virtual switch. Double click on it, or click the “+” sign to bring up the “Advanced Features” menu. In the “MAC address” section, click the “Enable MAC address spoofing” checkbox. In the section labeled “Port mirroring”, Select “Destination” in the “Mirroring mode:” drop-down.

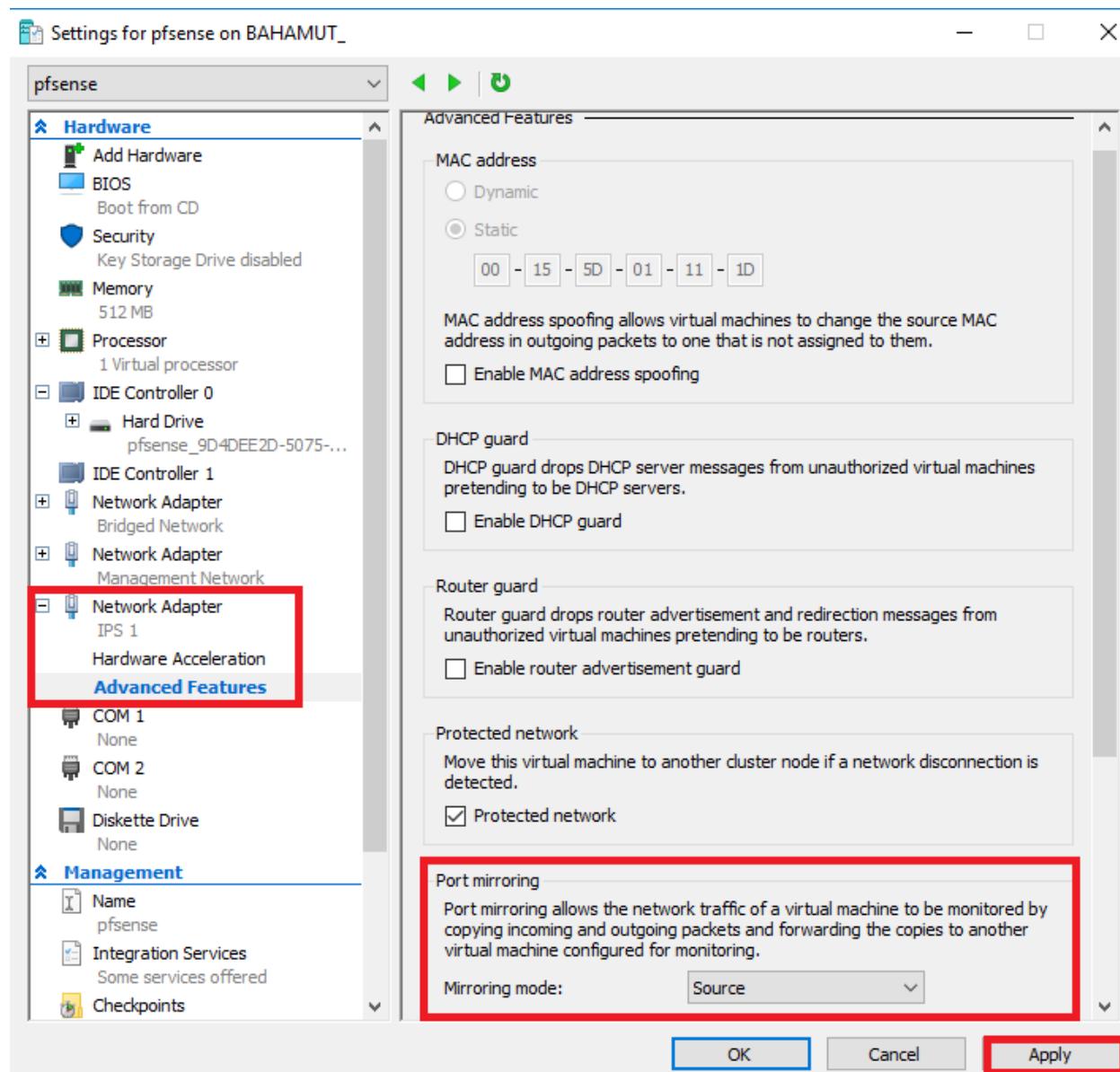


Repeat this process for the “IPS 2” Interface (enable MAC address spoofing, and set the adapter as a port mirroring destination), then click Apply. At this point, we have the “IPS” VM set

up and ready to go, but now, we have to configure all the remaining VMs connected to the “IPS 1” and “IPS 2” networks as sources to send their traffic to the IPS VM’s network adapters.

Configuring the pfSense VM as a Port Mirroring Source

In the Hyper-V Manager, right click on pfSense and choose settings to open the VM’s settings. Double click on the Network Adapter connected to the IPS 1 virtual switch, or click the “+” sign to display the “Advanced Features” menu. Under the “Port mirroring” section, select “Source” in the “Mirroring mode.” drop-down. Apply these changes.



Port Mirroring for the Remaining VMs

Now that you understand how to configure port mirroring, you will need to configure the network adapters on the Kali Linux and Metasploitable 2 VMs as port mirroring sources. Simply follow the same directions I laid out above for the pfSense VM to configure the remaining VMs as port mirroring sources. After you have completed this task, all that's left is to install and configure Snort or Suricata on the IPS VM to pass traffic between the "IPS 1" and "IPS 2" virtual switches.

Note: Please be aware that if you choose to customize your lab network, and add additional VMs to the "IPS 1" or "IPS 2" networks, that in order for those VMs to utilize the AFPACKET bridge provided by the IPS VM, each new VM will need to be specified as a port mirroring source.

Next Steps

The Client Hyper-V configuration is all but done at this point. However, you're not quite done yet. Here is a checklist of tasks to complete:

- If you haven't completed the chapter "[Defense in Depth for Windows Hosted Hypervisors](#)", I would very highly suggest doing so in order to harden your hypervisor host.
- While not strictly necessary, you may also want to complete the "[Remote Lab Management](#)" guide. Specifically, the sections related to [Windows Remote Access](#), [Enabling SSH on Kali Linux](#), and if you're lazy like me, The section on [Securing root SSH](#) access. This will allow you to remotely manage all of your lab VMs, as well as finish the IPS and Splunk setup guides much more easily than through the Hyper-V console.
- You still need to install IDS/IPS software on the IPS VM to get a functioning AFPACKET bridge. Check out the "[IPS Installation Guide](#)" to learn how to do this with either Snort or Suricata as your IPS software of choice.
- The SIEM VM needs to have Splunk installed and configured. You'll need to complete the "[Splunk Installation Guide](#)".
- Do you want some ideas on where to take your lab? Check out the chapter "[In Your Own Image](#)", for some tips on how to mold your VM lab to better suit your needs

Setup - VirtualBox

Note: Please be aware that this guide was written using VirtualBox version 5.1.6, with Windows 10 as the host Operating System. The host OS should not matter since the virtualbox graphical interface is fairly consistent across operating systems. For sake of sanity, we're not going to cover VBoxManage (a command line VirtualBox management interface) in this guide.

This guide will instruct you on how to create the our lab environment on Oracle's VirtualBox hosted hypervisor. This guide will teach you how to install and customize VirtualBox, then you will create your first VM, and finally will instruct you on how to create and configure the remaining VMs for your lab environment

Installation

VirtualBox's biggest draws are that its 100% free, and that its multi-platform; VirtualBox can be installed on practically every major operating system out there today, and can virtualize practically any operating system today, provided you meet the hardware requirements. Open your favorite web browser and navigate to <https://www.virtualbox.org/wiki/Downloads> and download the latest version of VirtualBox for the host OS you are using, then run the installer.

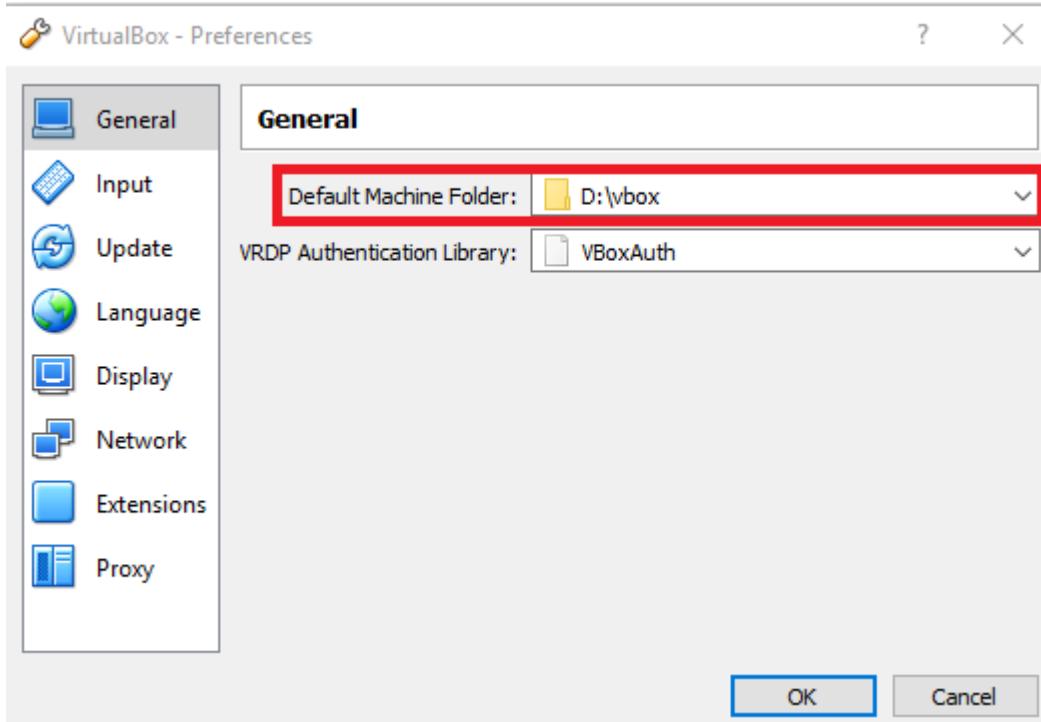
Note: Linux and BSD hosts can install VirtualBox through their respective package manager (e.g. yum, apt-get, pkg, etc.) For instructions on how to install VirtualBox on Linux via yum or apt-get visit https://www.virtualbox.org/wiki/Linux_Downloads. For instructions on how to install VirtualBox on FreeBSD, visit <https://www.freebsd.org/doc/handbook/virtualization-host-virtualbox.html>.

No matter what operating system you choose to run VirtualBox on, the installation is usually straightforward; simply proceed with the default installation options and reboot the host as necessary.

Hypervisor Preferences

So now that we have VirtualBox installed, let's get some general settings sorted before we create our first virtual machine. Open VirtualBox, then open the VirtualBox preferences menu. This can be done on Windows and Linux by clicking File > Preferences. On OSX, Click on VirtualBox > Preferences.

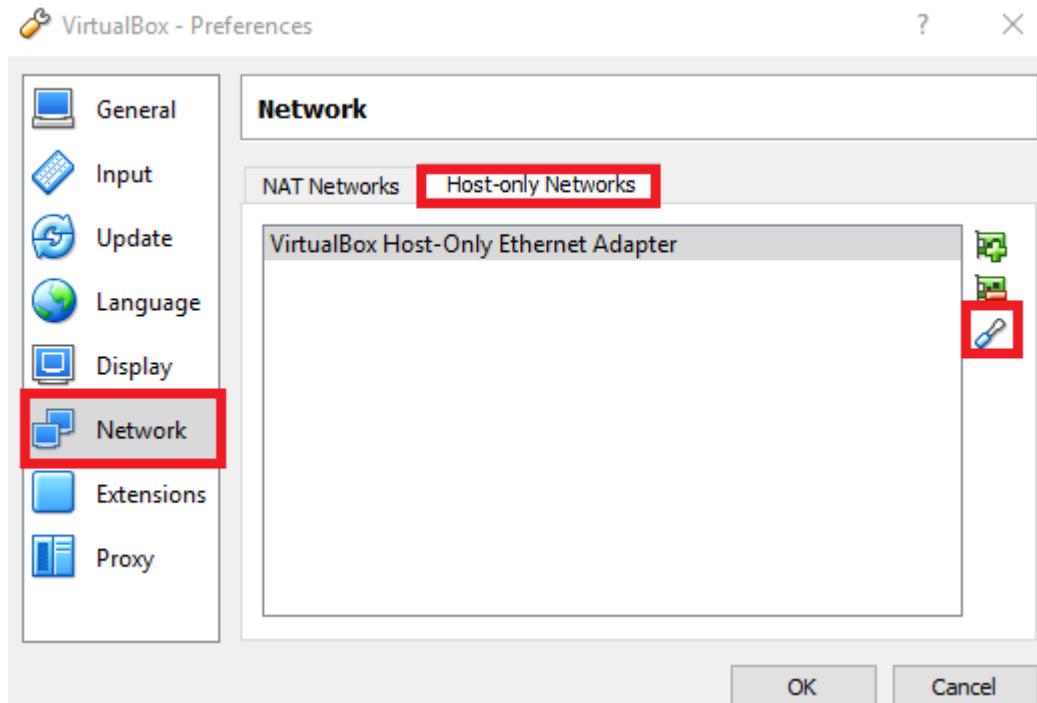
You should be dropped into the “General” screen by default. Pay attention to the setting “Default Machine Folder:”; If you have multiple drives or partitions and you wanted to have your virtual machine files stored on a particular drive, create a folder on that drive and edit the default path by clicking the drop-down arrow, and selecting “Other...” VirtualBox will then open an explorer window for you to navigate to the folder you want to use to store your VM files and folders. In the case of my setup, I chose to have my VirtualBox files stored in D:\vbox.



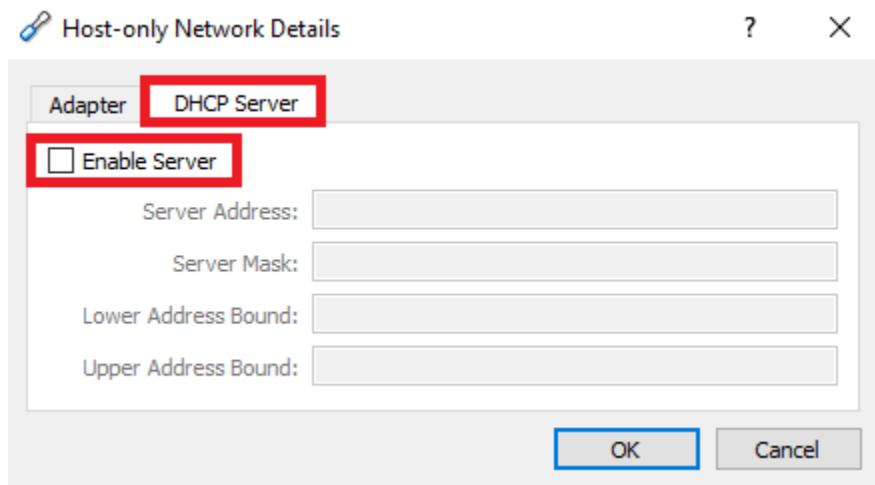
As discussed in the “Hardware Considerations” section, storing your VMs on a separate physical drive is generally a good idea to ensure that your VMs are NOT competing for disk I/O with the host operating system.

The next order of business is to disable DHCP for the host-only network, since we want the pfSense VM to handle DHCP services. To do this, click on “Network”, then click on the “Host-

only Networks” tab in the new pane that appears. If you are running Windows, VirtualBox creates your first host-only network, and names it “VirtualBox Host-Only Ethernet Adapter”. If you are running Linux, OSX or BSD, you will first need to click the network card icon with a “+” sign on it to the right of the “Host-Only Networks” pane to create your system’s first host-only network. In this case, this first host-only network will be named “vboxnet0” on non-Windows operating systems. Ensure the host-only network is highlighted, then click third icon on the right in the window that looks like a screwdriver.



In the next window, click on the DHCP Server tab, then uncheck the “Enable Server” Checkbox. Click OK to exit.



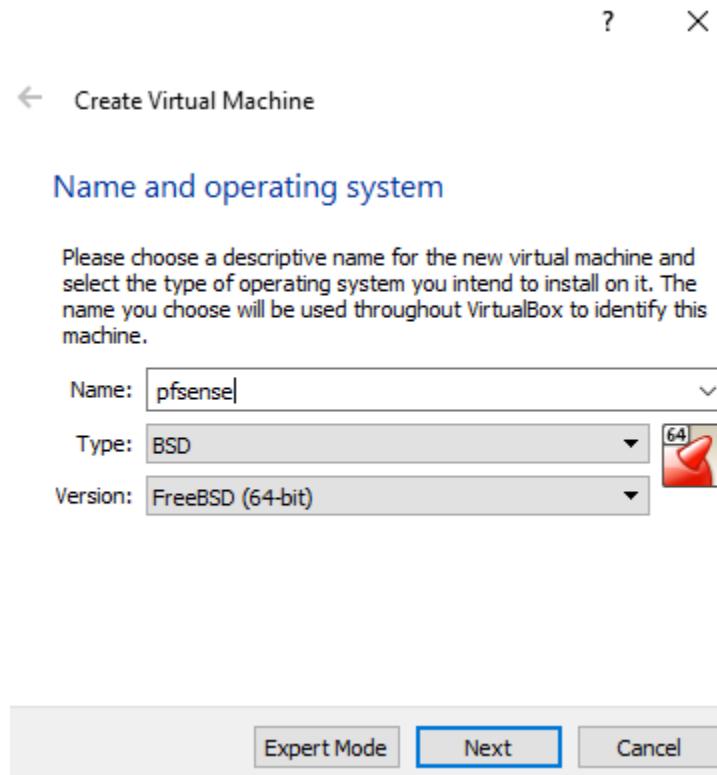
Creating the first VM, pfSense

pfSense is the keystone holding this entire configuration together. Personally, it's my favorite firewall distro due to ease of use, the amount of functionality it includes out of the box, combined with a plugin/add-on system for additional functionality. If you have the CPU, RAM, and disk, pfSense can easily be converted into a so-called “Next-Generation” firewall.

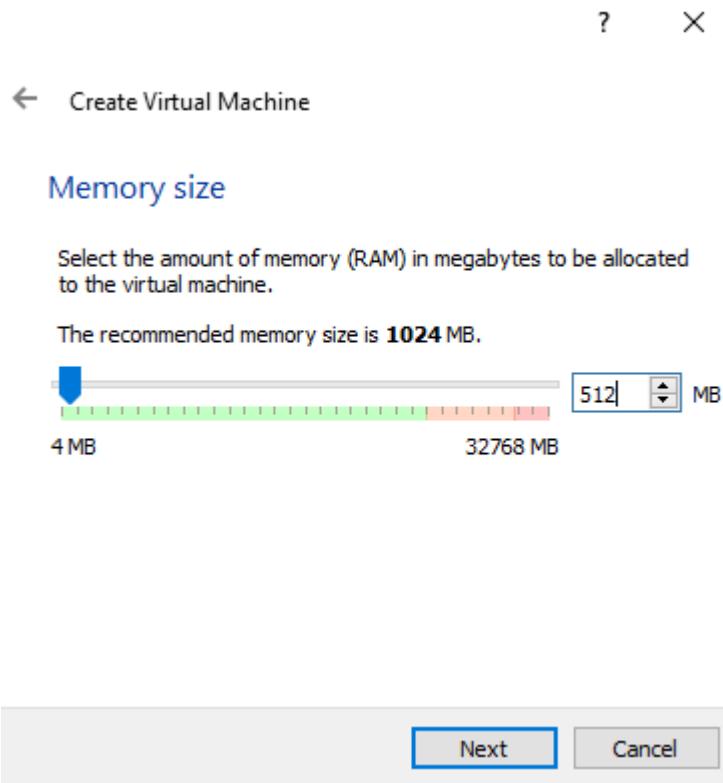
Make your way to <https://www.pfsense.org/download/>. Download the latest installation ISO for the amd64 architecture. While you're at it, you may want to download a compression utility. The pfSense maintainers distribute pfSense as a ISO image file compressed with gzip. This means that we'll need to decompress the ISO file at some point. On Windows, I prefer 7-Zip (<http://www.7-zip.org/>) as my compression utility of choice for decompressing files, since 7-Zip can handle zip, gzip, rar, and 7z files (among others) easily. Linux, BSD, and OSX usually include the gzip or gunzip command line utilities for decompressing gzipped files.

Adding a New VM

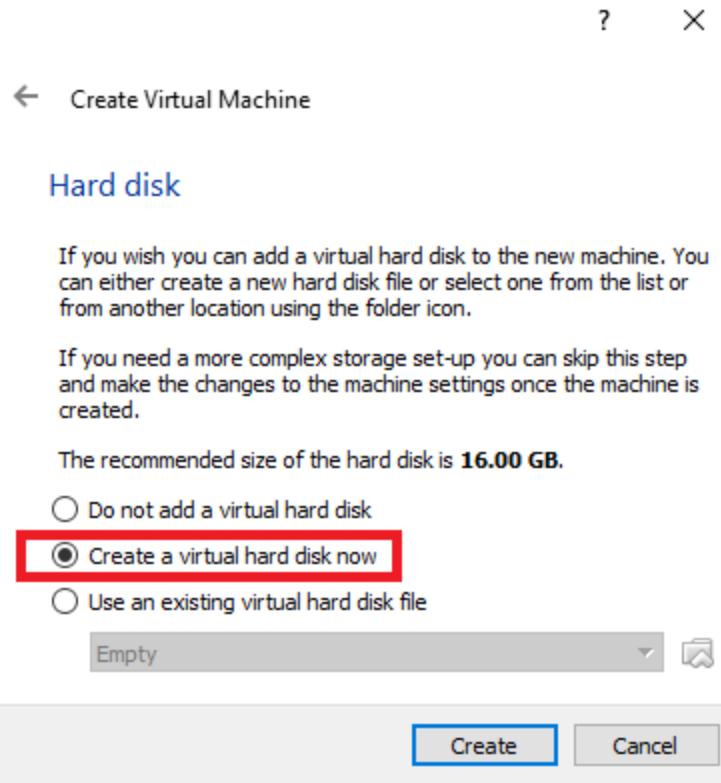
In VirtualBox, click the “New” icon to be taken through the Create Virtual Machine wizard. The first page will have you choose a name for your new VM, as well as assigning the type of OS you’ll be installing and the version. In our case, I named the VM pfSense. pfSense is BSD derived, and is based on FreeBSD, so I chose BSD as the type, and FreeBSD (64-bit) as the Version. Click the Next button to continue.



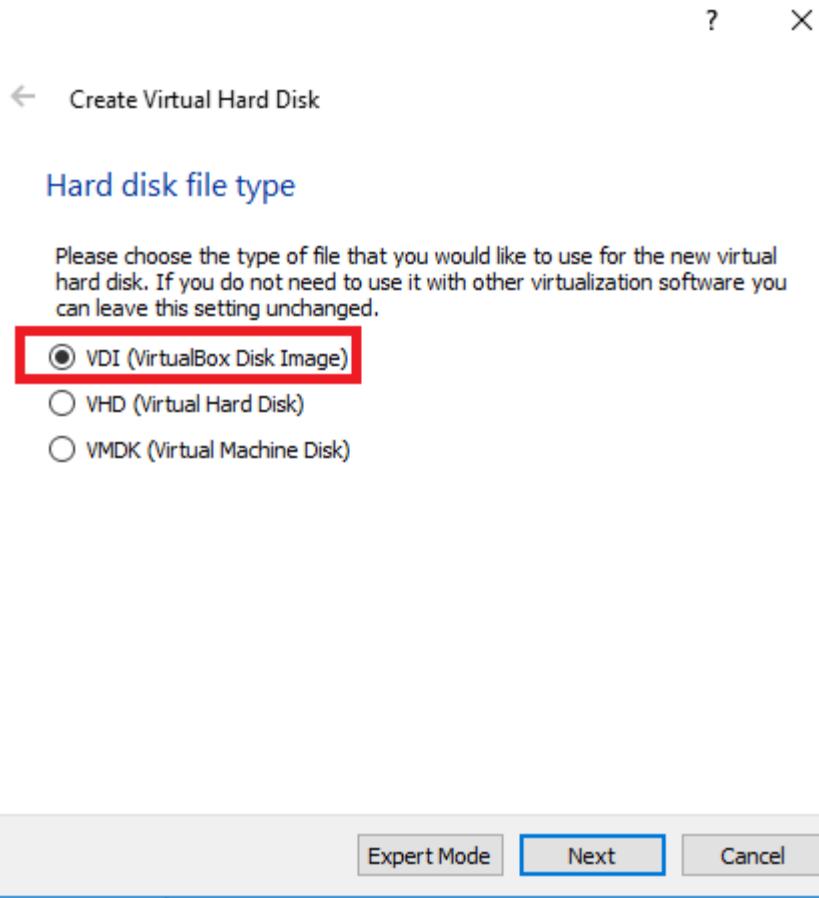
On the next screen, you'll be asked how much RAM you wish to allocate to this VM. 512MB is sufficient for this VM. After inputting 512MB, click Next.



The next screen is where we allocate hard drive space for the new VM. Click the “Create a virtual hard disk now” radio button, then click Create.

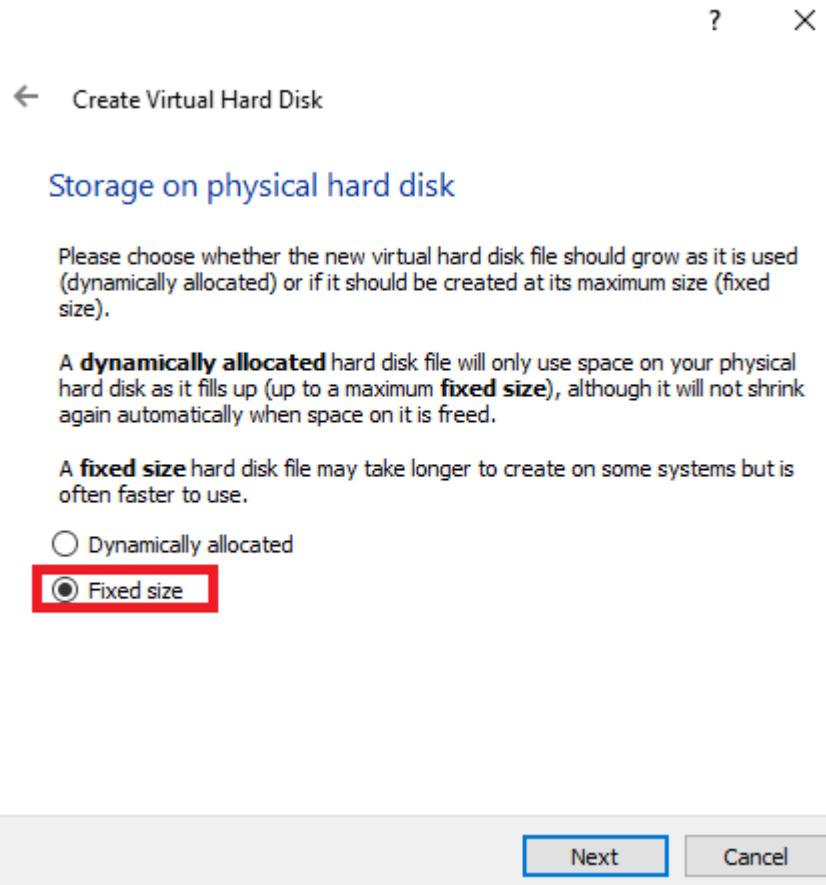


On the next screen, select the “VDI (VirtualBox Disk Image)” radio button and click Next.

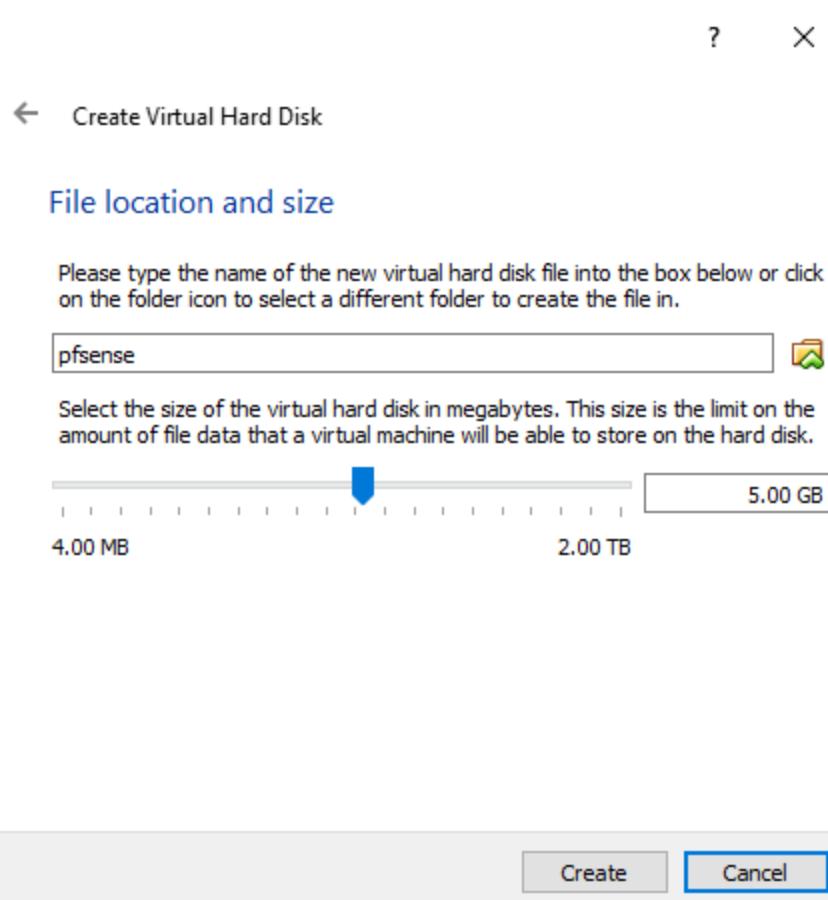


This screen asks if you want to use a fixed or dynamically allocated disk. Dynamic allocation allows the VDI file to expand in file size up to the maximum size we configure for it. This may lead to some I/O and/or CPU penalties as the file allocation is expanded, and more disk space is utilized in the VM. Choosing fixed means that the full amount of space we request for the VM

is immediately allocated to the corresponding VDI file. Click the “Fixed size” radio button, then click Next.

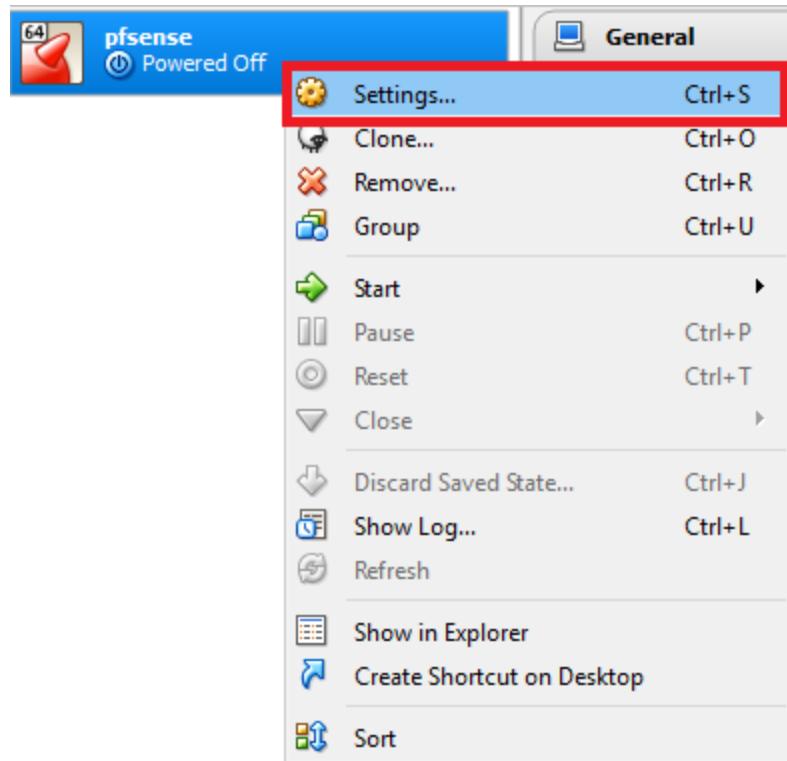


On the next screen, we allocate disk space that we'll be using to install pfSense to our VDI container. 5GB is what I recommend, but if you plan on utilizing more features, adjust the the size of your VM's virtual hard disk accordingly. Click Create and wait for VirtualBox to write the initial VDI file.

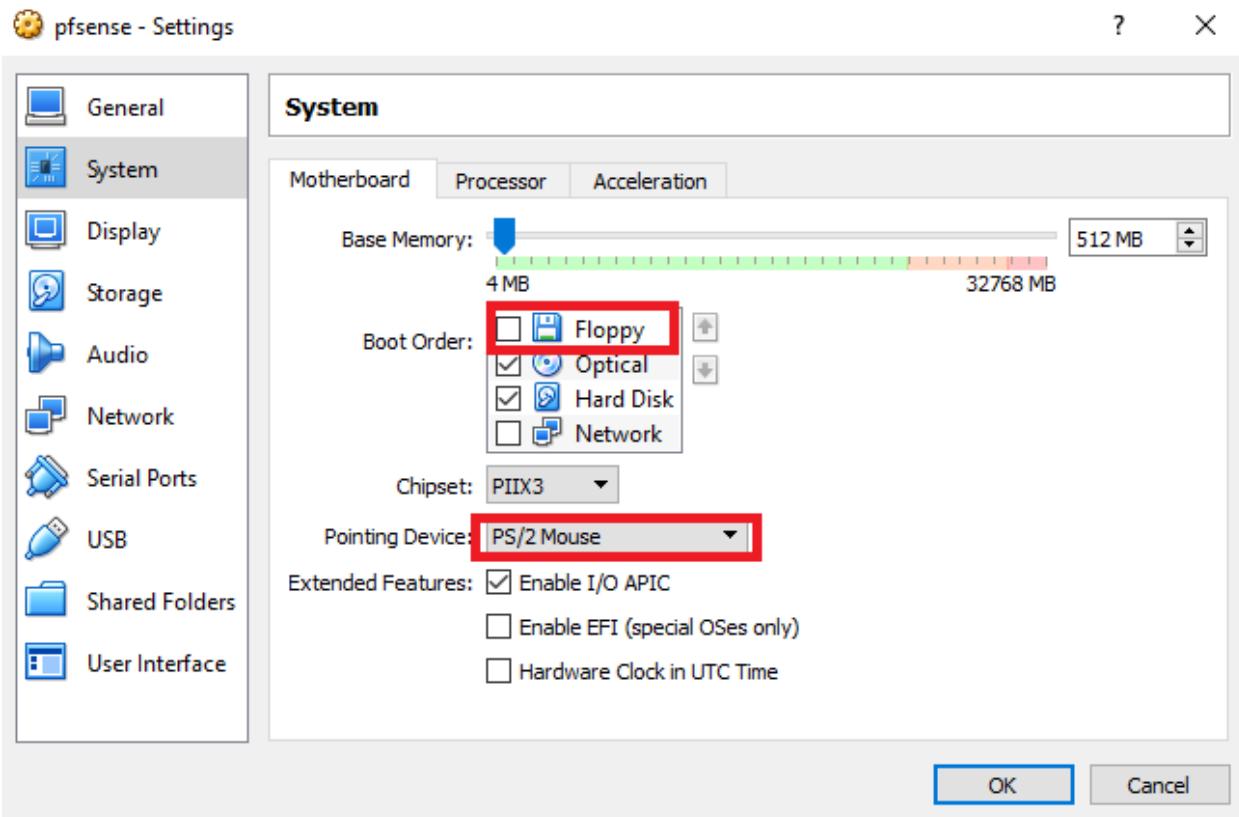


Initial VM Settings

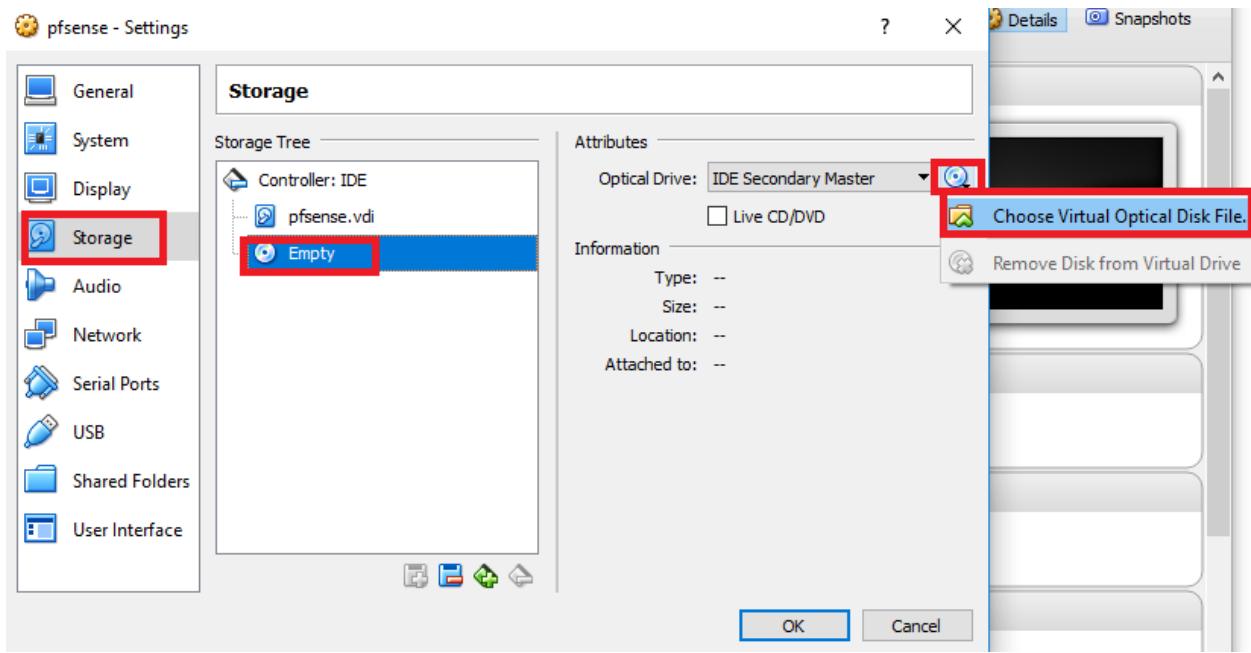
At this point, there should be a VM in the VirtualBox menu called “pfSense” listed in the main menu. It should be displayed as “Powered Off”. Before we power it on, there are a number of configuration changes we need to make. Right click on “pfSense”, and select “Settings...”.



The settings menu for the VM will be displayed. Let's begin by clicking on "System". This causes a new pane to appear with a variety of system hardware settings. On the "Motherboard" tab, In the "Boot Order" selection, uncheck the "Floppy" checkbox, then navigate to the "Pointing Device:" drop-down menu, and make sure that "PS/2 Mouse" is selected.

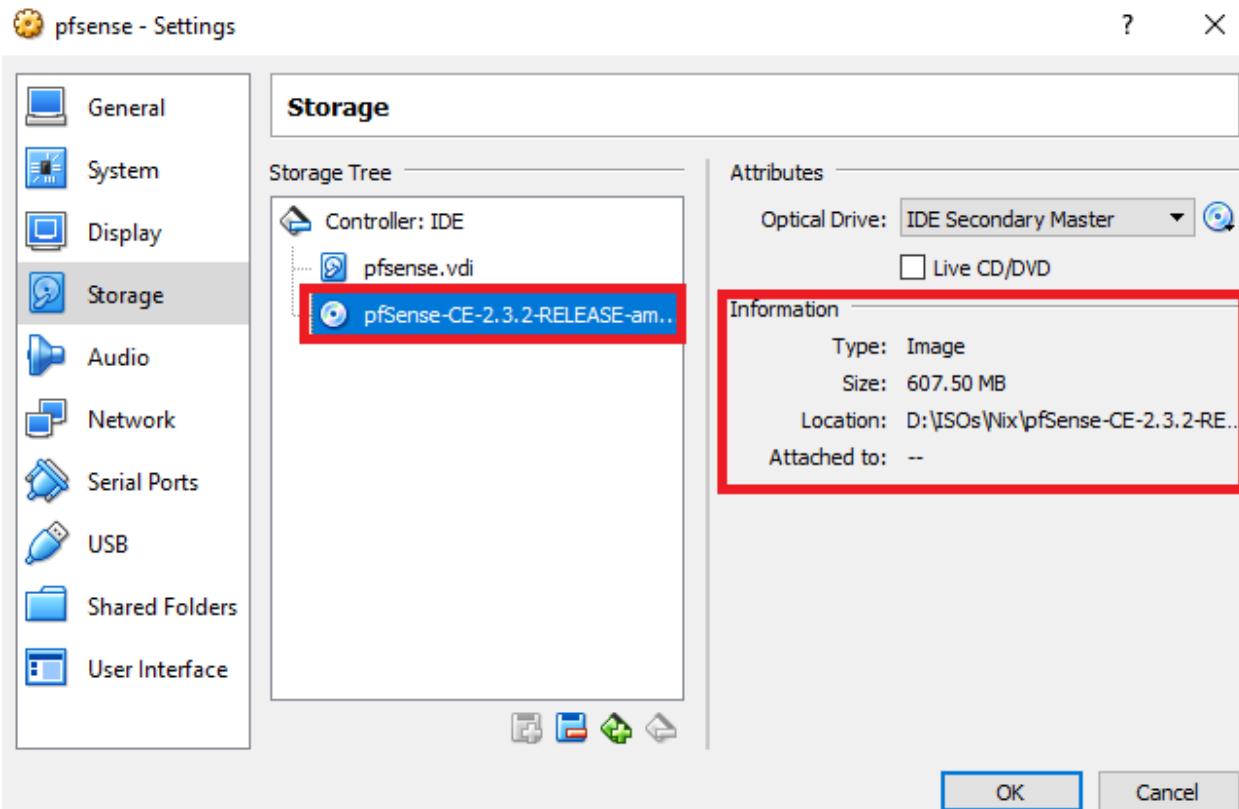


Next, click on “Storage”, then click on the icon that looks like a CD under the “Storage Tree” pane. Then, under the Attributes portion of the screen, click on the small icon that looks like a CD. On the drop-down menu that appears, then click “Choose Virtual Optical Disk File..”. This will cause an explorer window to pop up. Navigate to the directory where your pfSense ISO is located and double click on the .iso file to select it.

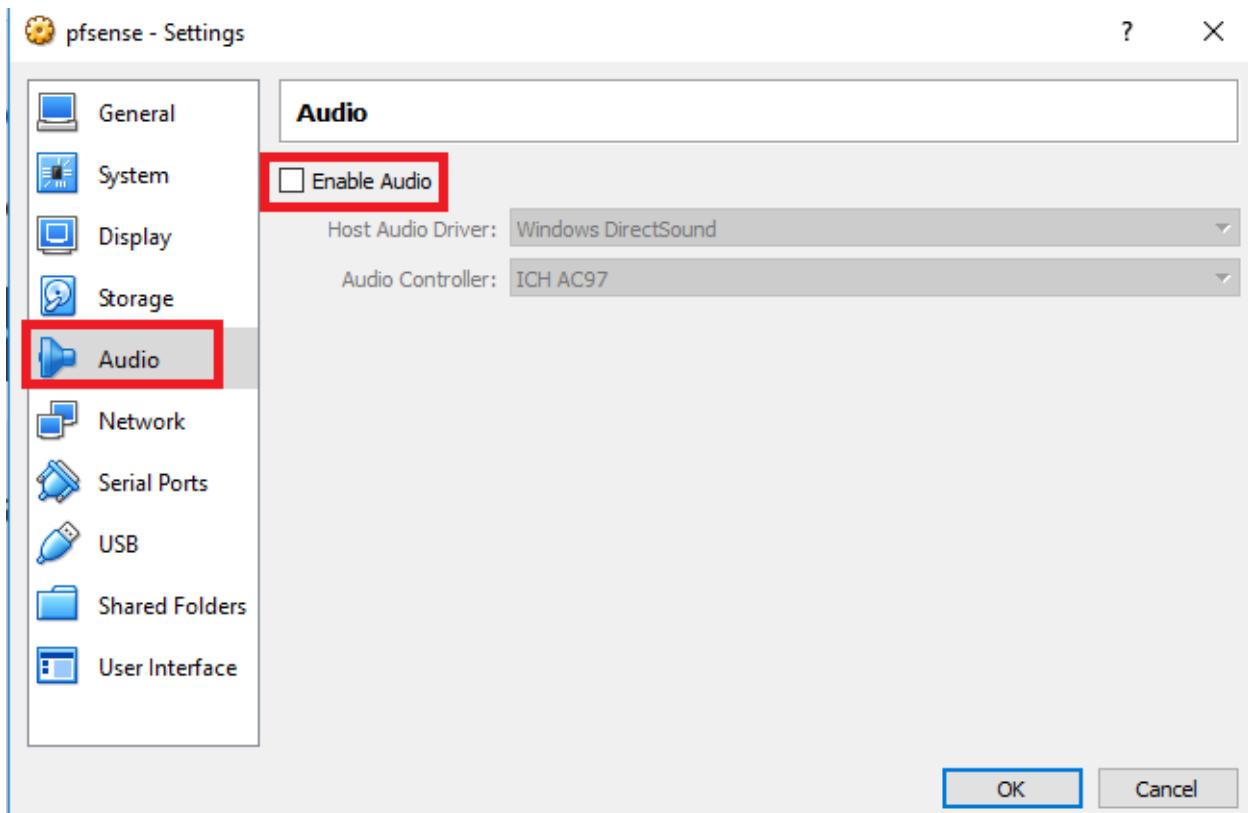


Note: VirtualBox is NOT capable of reading the pfSense ISO while it is compressed (.gz). You will need to decompress it first. As stated previously, I recommend using 7-zip on Windows, or the gzip command on Linux, BSD, or OSX to decompress the ISO.

The reason we are doing this is that we need to tell the VM to boot from this ISO file (that we're treating as though it is a virtual CD/DVD drive) in order to install an operating system to the VDI container. The "Storage" pane should look like this after finding and selecting the pfSense ISO:

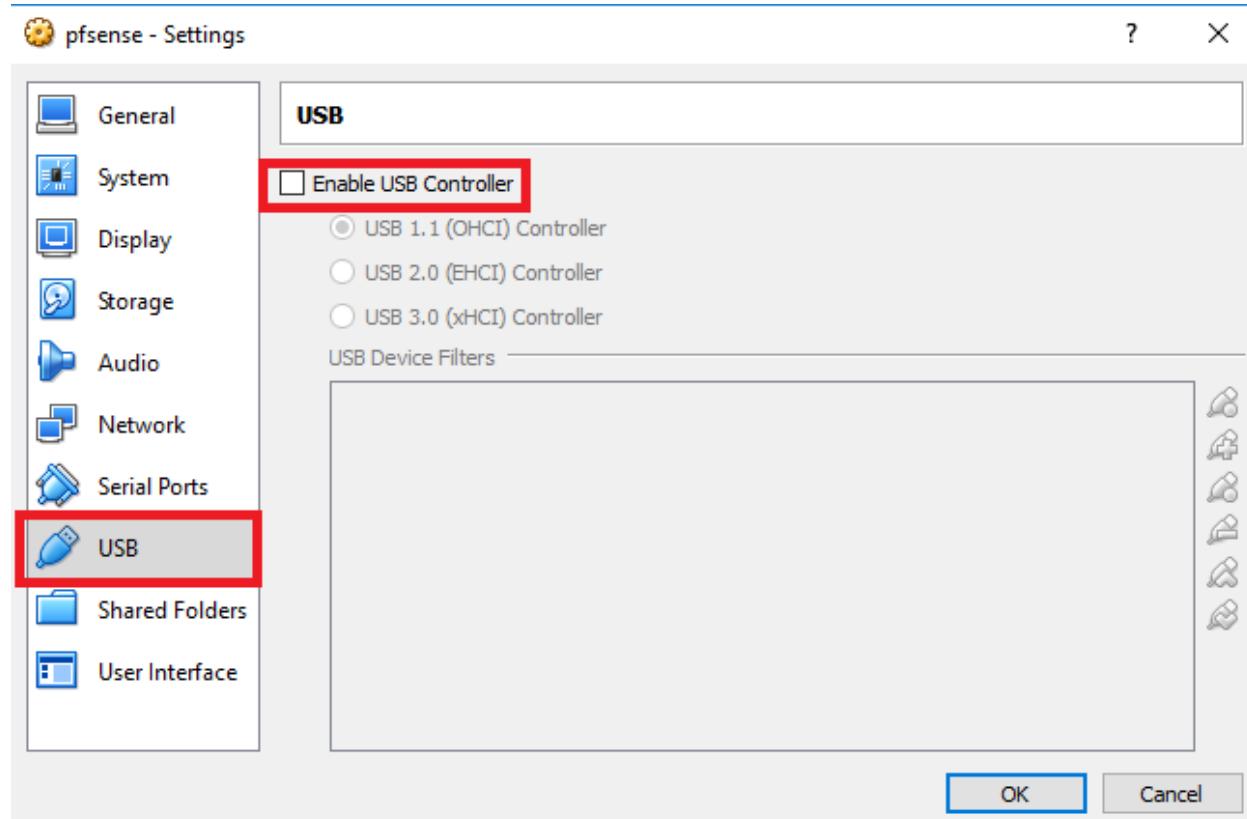


Next, click on “Audio”, and uncheck the “Enable Audio” checkbox.



Next, click on “USB” (OSX users should click on “Ports”, then click on the the “USB” tab, instead). Uncheck the “Enable USB Controller” checkbox to disable USB support for the VM. We are disabling the audio and USB controllers for our VM in order to reduce possible attack

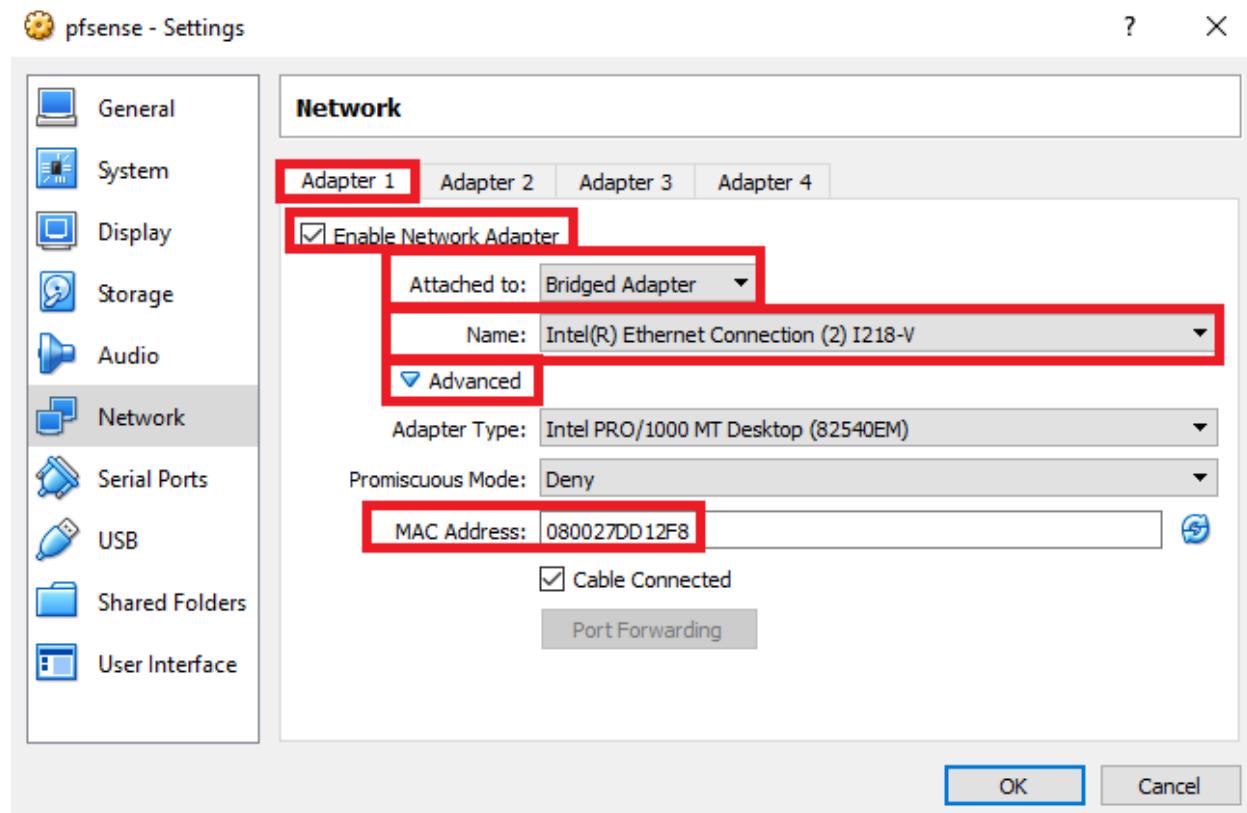
surface for virtual machine escapes, as well as ensure that host and VM are as isolated from one another as reasonably possible.



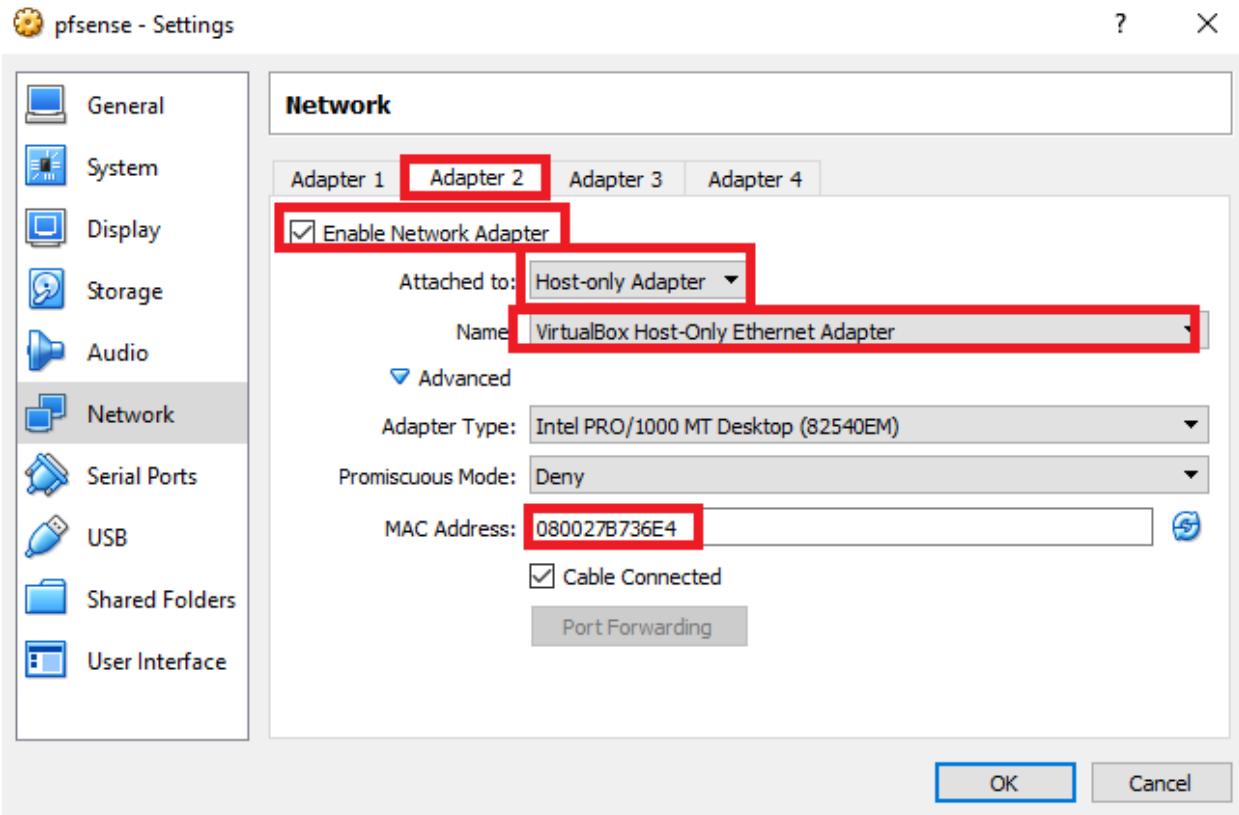
Note: Virtual machine escapes or “VM escapes” are a specific type of exploit that an attacker will attempt to use in order to gain some sort of execution or access to resources on the hypervisor host itself. While VM escapes that result in code execution on the hypervisor host are rare, there are other means that attackers or malware could potentially abuse to attack the hypervisor host itself, such as through shared folders (file system folders that both the host and the VM can read/write to) or through guest extension software packages (special software packages installed on the VM itself that enable extra functionality, or access to host system resources). Therefore it is important to make your lab VMs as lean as reasonably possible in order to avoid compromising the hypervisor host system.

Finally, click on “Network”. You will be greeted by a window pane that has multiple tabs. The first tab is labeled “Adapter 1”. First, check that the “Enable Network Adapter” checkbox is checked. On the “Attached to:” drop-down, make sure that “Bridged Adapter” is selected. Under the “Name:” drop-down, choose the physical network card you want to have virtual machine

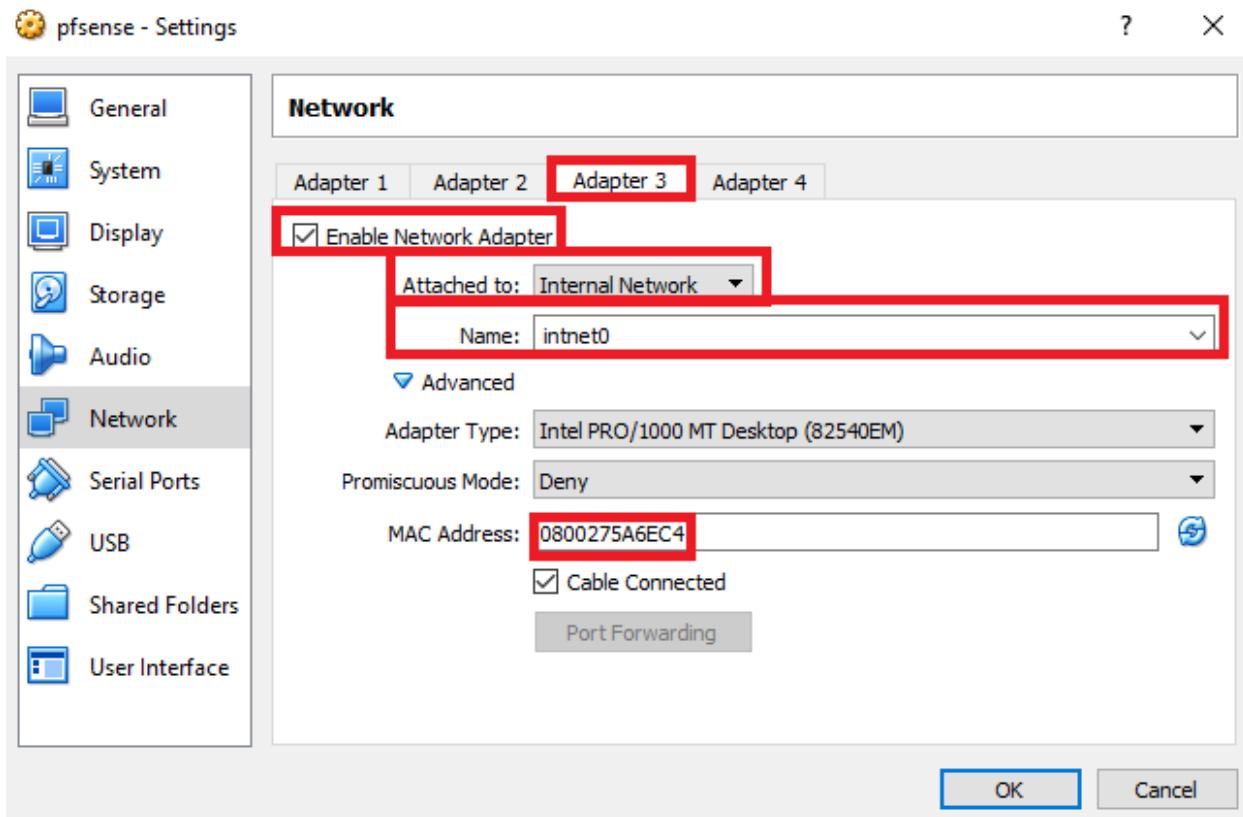
bridge to. Select the network interface you typically use on the host to connect to the internet. Click on the “Advanced” arrow to display more network adapter options. One of the fields that gets displayed is labeled “MAC Address:”. Document the MAC address displayed.



Click on the “Adapter 2” tab, and enable the network adapter if it is disabled, just like with “Adapter 1”. Next, set “Attached to:” to “Host-Only Adapter”, and “Name:” should default to “VirtualBox Host-Only Ethernet Adapter” on Windows, or “vboxnet0” on non-Windows hosts. Just like with “Adapter 1”, click on “Advanced”, and record the MAC address of this interface.



Finally, click on the “Adapter 3” tab. Enable the adapter, and set “Attached To:” to “internal network”. The default internal network name will be “intnet” on Windows, or “intnet0” on non-Windows hosts. The default internal network name (in either case) is perfectly fine for this network adapter, just make sure you document the internal network name somewhere. Make sure to record the internal network name for this adapter for later reference. Be sure to click “Advanced” and document the MAC address of this network adapter as well. Click OK to exit the settings menu.



Installing pfSense

Back at the main menu, click on the “pfSense” entry to highlight it, then click the big green start arrow to boot up the VM. A console window should appear that will allow you to interact with the VM as though you had a keyboard, mouse, and monitor directly attached to it. By default, if you click on the window, the virtual machine will take control of the mouse and keyboard. To detach the mouse and keyboard from the VM console, you have to enter a special key or keyboard combination called a hotkey. In Windows and Linux, this is the right CTRL key, while on OSX, this is the left Command key.

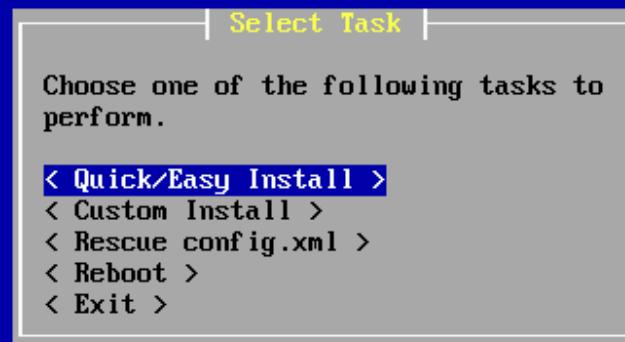
Installing pfSense is very easy. Let the VM boot up, and the system should automatically run the installer. Adjust your video, screenmap, and keymap settings as necessary, then select “<Accept these Settings >”.

F10=Refresh Display



On the next screen, select “< Quick/Easy Install >” and let pfSense do all the heavy lifting. The next screen will inform you that the install will erase the contents of the hard disk. Since our virtual disk is already empty, this doesn’t matter in the least. Select OK, and let the installer run.

F10=Refresh Display



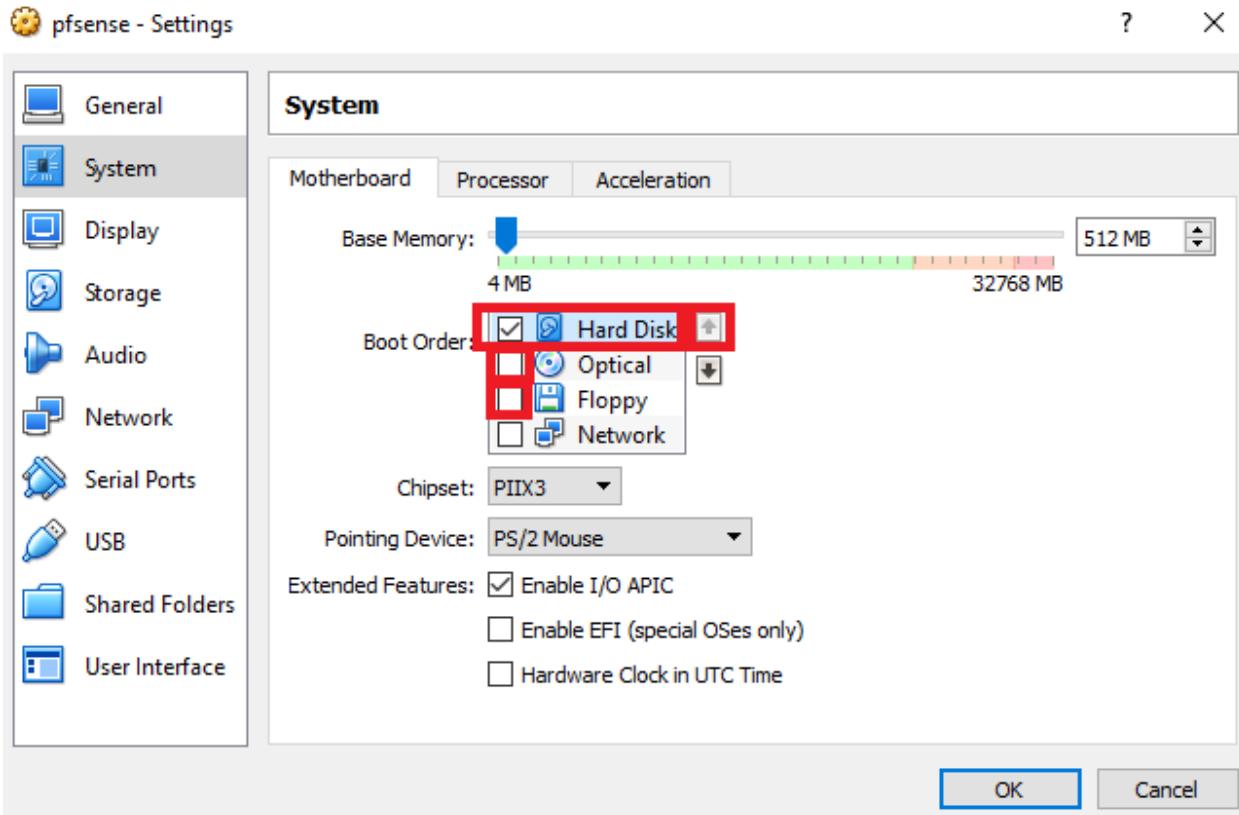
Invoke Installer with minimal questions

The installer will go through and install pfSense to the virtual hard disk. The installer will ask if you want to install a standard or embedded kernel. Make sure to select “< Standard Kernel >”. Finally, the installer informs you to reboot the machine to boot from the hard drive. The installation is done, however, we’re not going to reboot. Instead, shut the VM down. Click “Machine”, in the VM’s console window menu, and clicking “ACPI Shutdown”.

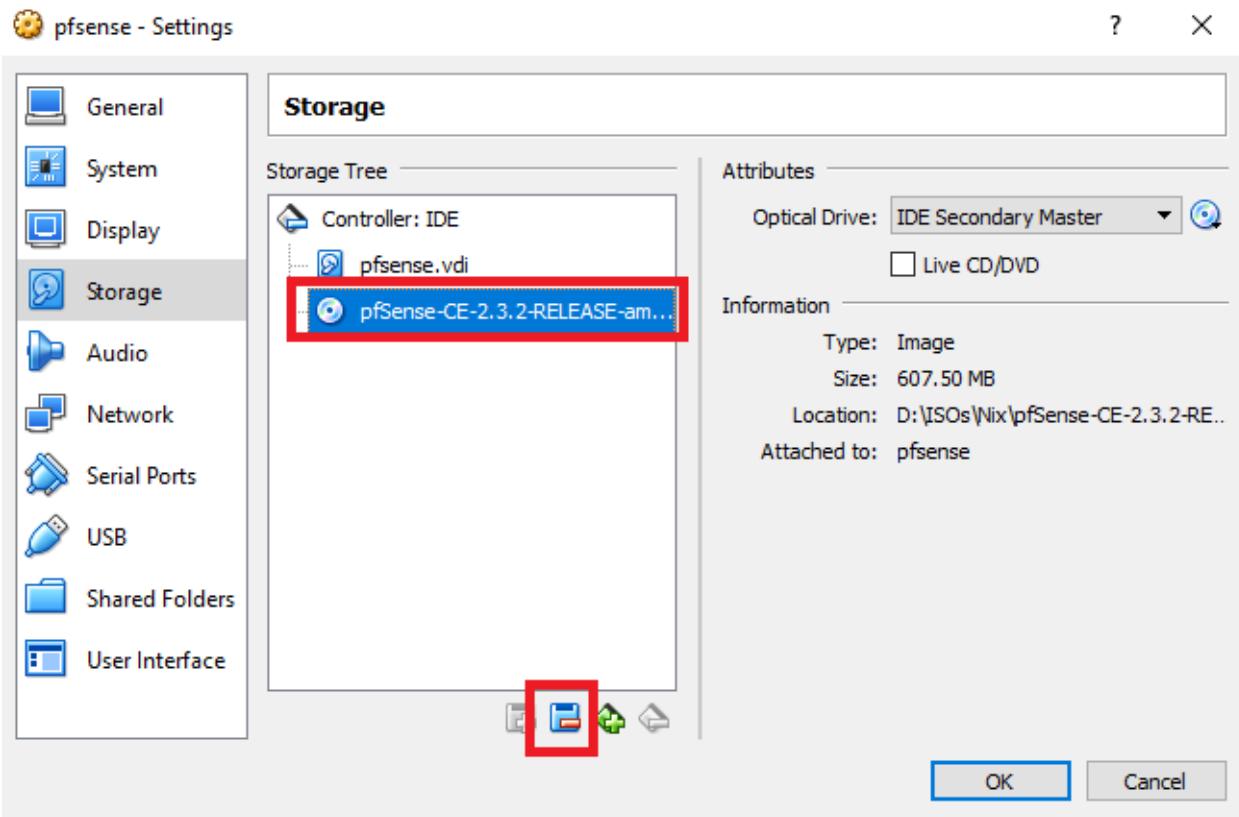
Final VM Settings

We have to modify a few more settings of the VM before we start configuring pfSense properly. On the VirtualBox main menu, right click on pfSense and go back to the “Settings...” menu. Select “System”, then ensure the the “Motherboard” tab is selected. We need to modify the “Boot Order” options.

First, uncheck the checkbox next to “Optical”, then make sure that “Hark Disk” is the only checked option remaining. Finally, highlight “Hard Disk” then click the up arrow to the right of the “Boot Order” list, until “Hard Disk” is at the top. This change ensures that the operating system installed on the hard drive is the first and ONLY bootable device for this VM.

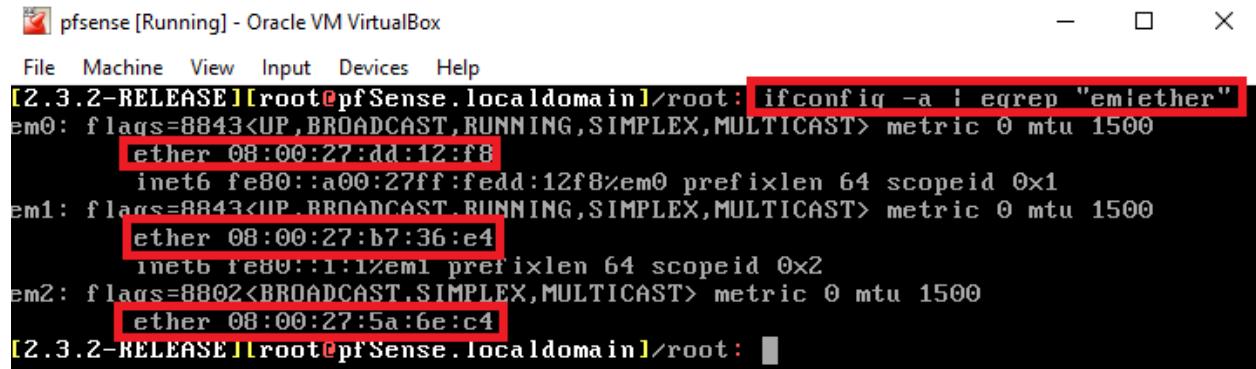


Next, click “Storage”. Under the “Storage Tree” pane, click on the icon that looks like a CD, then on the bottom of the pane, click on the blue icon with a red minus sign on it. This will uninstall the virtual CD/DVD drive for this VM. This change ensures that the virtual hard drive is the only storage media available for this VM now. Click OK to exit the “Settings” menu.



Network Configuration

On the VirtualBox main menu. Click on the “pfSense” VM to highlight it, then click the start arrow to boot the VM. Once the console appears, wait a moment for the VM to finish booting, and you should be greeted by the pfSense main menu. First things first, we have to correlate which interfaces in the VM correspond to adapter 1, adapter 2, and adapter 3 in the VirtualBox network settings. In the pfSense main menu, select option 8 to open up a shell. If you're not familiar with the command line interface in Linux/Unix, don't worry; we won't be here very long. Type the command `ifconfig -a | egrep "em|ether"`. What this command does is run `ifconfig -a`, a command that shows all the information for all of the network interfaces installed that the OS recognizes. The pipe symbol “|” is taking the output from the `ifconfig` command and “piping” it as input to the `egrep` command. We are telling `egrep` to process the output from `ifconfig` and ONLY show us lines that contain “em” OR “ether”. If you did it correctly, the output should look something like this:



```
[2.3.2-RELEASE][root@pfSense.localdomain]# ifconfig -a | egrep "em|ether"
em0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
    ether 08:00:27:dd:12:f8
    inet6 fe80::a00:27ff:fedd:12f8%em0 prefixlen 64 scopeid 0x1
em1: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
    ether 08:00:27:b7:36:e4
    inet6 fe80::1:1%em1 prefixlen 64 scopeid 0x2
em2: flags=8802<BROADCAST,SIMPLEX,MULTICAST> metric 0 mtu 1500
    ether 08:00:27:5a:6e:c4
[2.3.2-RELEASE][root@pfSense.localdomain]#
```

In BSD/Linux terms, the MAC address is referred to as “ether”, shorthand for ethernet address. What we need to do is compare the ether addresses displayed here to figure out which interface in the VM (em0, em1, and em2) maps to which adapter (adapter 1, 2, and 3) in VirtualBox. This is why I had you document the MAC addresses for adapter 1, 2, and 3 earlier when we created the VM; We need to know which em interface corresponds to which adapter number in the VirtualBox network settings so we can make sure the interfaces in the VM are assigned and configured correctly.

We need to know which interface in pfSense corresponds to which adapter so we can designate the WAN (bridged network), LAN (Host-Only Network) and OPT1 (Internal Network) properly in pfSense. In my case, the ether address for em0 is 00:00:27:dd:12:f8. This corresponds to the MAC address assigned to adapter 1 in VirtualBox (000027dd12f8-- exactly the same, without the colon separators), em1 corresponds to adapter 2, and em2 corresponds to adapter 3. Once you have figured out which interface corresponds to which VirtualBox adapter, type “exit” to leave the shell and go back to the pfSense main menu.

Select option 1, “Assign Interfaces” to run the interface assignment wizard. To manage its network services and firewall policies, pfSense uses internal network interface descriptors (WAN, LAN, OPT1, etc.). Assigning the interfaces defines which of them (em0, em1, em2, etc.) corresponds to which of pfSense’s internal descriptors.

```
0) Logout (SSH only)
1) Assign Interfaces
2) Set interface(s) IP address
3) Reset webConfigurator password
4) Reset to factory defaults
5) Reboot system
6) Halt system
7) Ping host
8) Help
9) pfTop
10) Filter Logs
11) Restart webConfigurator
12) PHP shell + pfSense tools
13) Update from console
14) Enable Secure Shell (sshd)
15) Restore recent configuration
16) Restart PHP-FPM
```

The WAN interface needs to be assigned to the bridged network adapter (Adapter 1, the Bridged network). The LAN interface needs to be assigned to our Host-Only adapter (Adapter 2, the Management network), and finally, OPT1 needs to be assigned to our internal network adapter (Adapter 3, the IPS 1 network). In my case, em0 became the WAN interface, em1 the LAN interface, and em2 was mapped to the OPT1 interface. After confirming that the settings are correct, you should automatically be returned to the pfSense main menu.

```
Enter the WAN interface name or 'a' for auto-detection
(em0 em1 em2 or a): em0

Enter the LAN interface name or 'a' for auto-detection
NOTE: this enables full Firewalling/NAT mode.
(em1 em2 a or nothing if finished): em1

Optional interface 1 description found: OPT1
Enter the Optional 1 interface name or 'a' for auto-detection
(em2 a or nothing if finished): em2

Enter the Optional 2 interface name or 'a' for auto-detection
( a or nothing if finished):

The interfaces will be assigned as follows:

WAN -> em0
LAN -> em1
OPT1 -> em2
```

Now that we have assigned the network interfaces, we have to configure IP addresses for these interfaces. In most cases, the WAN interface will automatically get an IP address from the device on your physical network that provides DHCP services. If this is not happening, you may have to troubleshoot your physical network. This is beyond the scope of our guide. As for the LAN and OPT1 networks, we have to manually set the IP address, subnet mask, and DHCP scopes for these networks. Select option 2 in the pfSense main menu to get started.

```
0) Logout (SSH only)          9) pfTop
1) Assign Interfaces          10) Filter Logs
2) Set interface(s) IP address 11) Restart webConfigurator
3) Reset webConfigurator password 12) PHP shell + pfSense tools
4) Reset to factory defaults 13) Update from console
5) Reboot system              14) Enable Secure Shell (sshd)
6) Halt system                15) Restore recent configuration
7) Ping host                  16) Restart PHP-FPM
8) Shell
```

The configuration wizard starts by asking for the interface the settings should be changed for. We have to go through the process two times, for the LAN and the OPT1 interface individually. Here are the settings I recommend:

LAN IP: **172.16.1.1**

LAN Subnet bit count: **24**

LAN upstream gateway address: <empty>

LAN IPv6 address: <empty>

Do you want to enable the DHCP server on LAN? **y**

LAN DHCP start address: **172.16.1.10**

LAN DHCP end address: **172.16.1.254**

Do you want to revert to HTTP as the webConfigurator protocol? **N**

OPT1 IP: **172.16.2.1**

OPT1 Subnet bit count: **24**

OPT1 upstream gateway address: <empty>

OPT1 IPv6 address: <empty>

Do you want to enable the DHCP server on OPT1? **y**

OPT1 DHCP start address: **172.16.2.10**

OPT1 DHCP end address: **172.16.2.254**

Do you want to revert to HTTP as the webConfigurator protocol? **n**

The wizard asks whether or not IPv6 should be configured. Say no, because that's a can of worms we're not going to deal with here. The wizard will also ask if would like to use DHCP for the LAN and OPT1 networks. Respond with yes. For the LAN network, enter 172.16.1.10 as start address, and 172.16.1.254 as end address. For the OPT1 network, enter 172.16.2.10 as start address, and 172.16.2.254 as end address.

The reason we choose 1.10 and 2.10 as the DHCP start addresses is to reserve a few IP addresses for static DHCP allocations. Static DHCP allocations allow you to configure a DHCP server to always serve the same IP address when requested from a particular MAC address. We will talk about this a little bit more later. For now, we are done mucking around in the

pfSense CLI. From here on out, we will be using the webConfigurator to manage and configure our pfSense VM.

Note: If you are using 172.16.1.0/24 or the 172.16.2.0/24 network ranges in your physical network, choose another network range to assign to the LAN and OPT1 interfaces to avoid network conflicts.

Note: When you are configuring the LAN and OPT1 interface IP addresses, the configuration script will ask “Do you want to revert to HTTP as the webConfigurator protocol?” **Always say no to this.** This allows the web UI to accept HTTP logins, and makes your firewall credentials vulnerable to sniffing over the network.

Your pfSense main menu should look something like this when you’re all done:

```
*** Welcome to pfSense 2.3.2-RELEASE (amd64 full-install) on pfSense ***

WAN (wan)      -> em0      -> v4/DHCP4: 192.168.1.15/24
LAN (lan)      -> em1      -> v4: 172.16.10.1/24
OPT1 (opt1)    -> em2      -> v4: 172.16.11.1/24

0) Logout (SSH only)          9) pfTop
1) Assign Interfaces          10) Filter Logs
2) Set interface(s) IP address 11) Restart webConfigurator
3) Reset webConfigurator password 12) PHP shell + pfSense tools
4) Reset to factory defaults   13) Update from console
5) Reboot system               14) Enable Secure Shell (sshd)
6) Halt system                 15) Restore recent configuration
7) Ping host                   16) Restart PHP-FPM
8) Shell
```

Before moving on to setting up pfSense from the webConfigurator, you must configure the Host-Only network card attached to the hypervisor host. If you are using Windows as your host, and If you haven’t already, you’ll want to visit the “[Unbinding Network Protocols on Windows Virtual Adapters](#)” at a minimum. This will guide you through configuring the VirtualBox Host-Only Ethernet Adapter with an IP address as well as unbinding network protocols on this adapter to increase the security of your Windows hypervisor host. You may also want to consider “[Using Windows Firewall to Limit Exposure of Windows Hypervisor Hosts](#)” to further enhance the security of the Windows host when interacting with your lab VMs.

Linux and OSX users can configure the IP address of their host-only network card by using the command utilities `ifconfig` or `ip addr add`. Try one of the following commands to set the ip address of the `vboxnet0` interface:

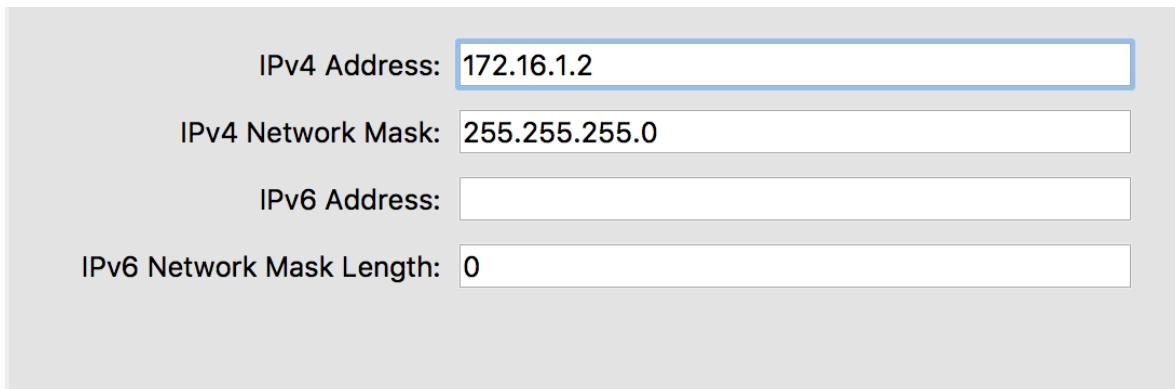
```
ifconfig vboxnet0 172.16.1.2 netmask 255.255.255.0  
ip addr add 172.16.1.2/24 dev vboxnet0
```

Please note that you will need root permissions to run these commands, and that these settings will NOT persist between reboots of your hypervisor host system. Configuring IP address persistence for your host OS is beyond the scope of this guide.

If you don't want to mess around with ifconfig or the ip command you can use the VirtualBox "Preferences" menu to set the ip address of vboxnet0 as well. Navigate to "Network", select the "Host-Only Networks" tab, click vboxnet0 to highlight it, then click the screwdriver icon. A small window will pop up. Select the "Adapter" tab if it isn't already selected, then fill out the following:

IPv4 Address: **172.16.1.2**

IPv4 Network Mask: **255.255.255.0**



Leave the rest of the fields alone, then click OK to exit the configuration menu, then click OK to exit preferences menu.

webConfigurator - Initial Setup

On your VirtualBox host OS, open your favorite web browser (I prefer Firefox - <https://www.mozilla.org/en-US/firefox/new/>) and navigate to <https://172.16.1.1> (or the IP address you assigned to the LAN interface of your pfSense system). The default credentials for access to the web interface are admin/pfsense. If this is your first time logging in, pfSense takes you through a nice little setup wizard. I'll highlight some of the important things to take note of, and/or change as necessary:

Set the primary and secondary DNS servers you plan on using. I typically use 8.8.8.8 (Google public DNS) and 4.2.2.2 (Level 3 public DNS). If you're using this lab at work, your workplace may have restrictions and you may have to use their DNS servers instead.

Primary DNS Server	8.8.8.8
Secondary DNS Server	4.2.2.2

Uncheck the checkbox next to “Block private networks from entering via WAN” rule. Uncheck the checkbox next to “Block Bogon Networks” on this page as well.

RFC1918 Networks

Block RFC1918 Private Networks	<input checked="" type="checkbox"/> Block private networks from entering via WAN
---------------------------------------	--

When set, this option blocks traffic from IP addresses that are reserved for private networks as per RFC 1918 (10/8, 172.16/12, 192.168/16) as well as loopback addresses (127/8). This option should generally be left turned on, unless the WAN network lies in such a private address space, too.

The other default settings on the first time setup wizard should be fine. Please note that you will be asked to change the password for the admin account as a part of the setup wizard. Document the password, and keep it somewhere safe.

Our next order of business is to restrict access to the web interface of the firewall to the machine with the IP address 172.16.1.2. On the menu bar at the top of the page, select Firewall > Rules. Click on LAN to modify the firewall policy for the LAN interface. Click the “Add” button with the arrow facing up. This will add the firewall rule to the top of the firewall policy to where it will evaluated first. The “Edit Firewall Rule” page is fairly straightforward. I’ve highlighted the options to be aware of.

Edit Firewall Rule

Action	Pass			
Choose what to do with packets that match the criteria specified below. Hint: the difference between block and reject is that with reject, a packet (TCP RST or ICMP port unreachable for UDP) is returned to the sender, whereas with block the packet is dropped silently. In either case, the original packet is discarded.				
Disabled	<input type="checkbox"/> Disable this rule Set this option to disable this rule without removing it from the list.			
Interface	LAN			
Choose the interface from which packets must come to match this rule.				
Address Family	IPv4			
Select the Internet Protocol version this rule applies to.				
Protocol	TCP			
Choose which IP protocol this rule should match.				
Source				
Source	<input type="checkbox"/> Invert match.	Single host or alias	172.16.1.2	/
Display Advanced		Display Advanced		
Destination				
Destination	<input type="checkbox"/> Invert match.	Single host or alias	172.16.1.1	/
Destination port range	HTTPS (443)	From	To	Custom
Specify the destination port or port range for this rule. The "To" field may be left empty if only filtering a single port.				
Extra Options				
Log	<input type="checkbox"/> Log packets that are handled by this rule Hint: the firewall has limited local log space. Don't turn on logging for everything. If doing a lot of logging, consider using a remote syslog server (see the Status: System Logs: Settings page).			
Description	pfsense strict anti-lockout			

When you are done, click the Save icon at the bottom of the page. This will take you back to the previous page. A yellow dialogue box with a green button called “Apply Changes” will appear. Click this button to apply this new firewall rule. Next, we want to disable the default anti-lockout rule. Next, let’s navigate to Firewall > Aliases. On the Firewall Aliases page, click on “IP”, then click Add. Create an alias with the following settings:

Firewall / Aliases / Edit

Properties

Name	RFC1918	The name of the alias may only consist of the characters "a-z, A-Z, 0-9 and _".
Description	An alias for all RFC1918 networks	A description may be entered here for administrative reference (not parsed).
Type	Network(s)	

Network(s)

Hint: Networks are specified in CIDR format. Select the CIDR mask that pertains to each entry. /32 specifies a single IPv4 host, /128 specifies a single IPv6 host, /24 specifies 255.255.255.0, /64 specifies a normal IPv6 network, etc. Hostnames (FQDNs) may also be specified, using a /32 mask for IPv4 or /128 for IPv6. An IP range such as 192.168.1.1-192.168.1.254 may also be entered and a list of CIDR networks will be derived to fill the range.			
Network or FQDN	10.0.0.0 / 8	10.x.x.x RFC 1918 networks	
	172.16.0.0 / 12	172.16.x.x RFC 1918 networks	
	192.168.0.0 / 16	192.168.x.x RFC 1918 networks	

Save Add Network

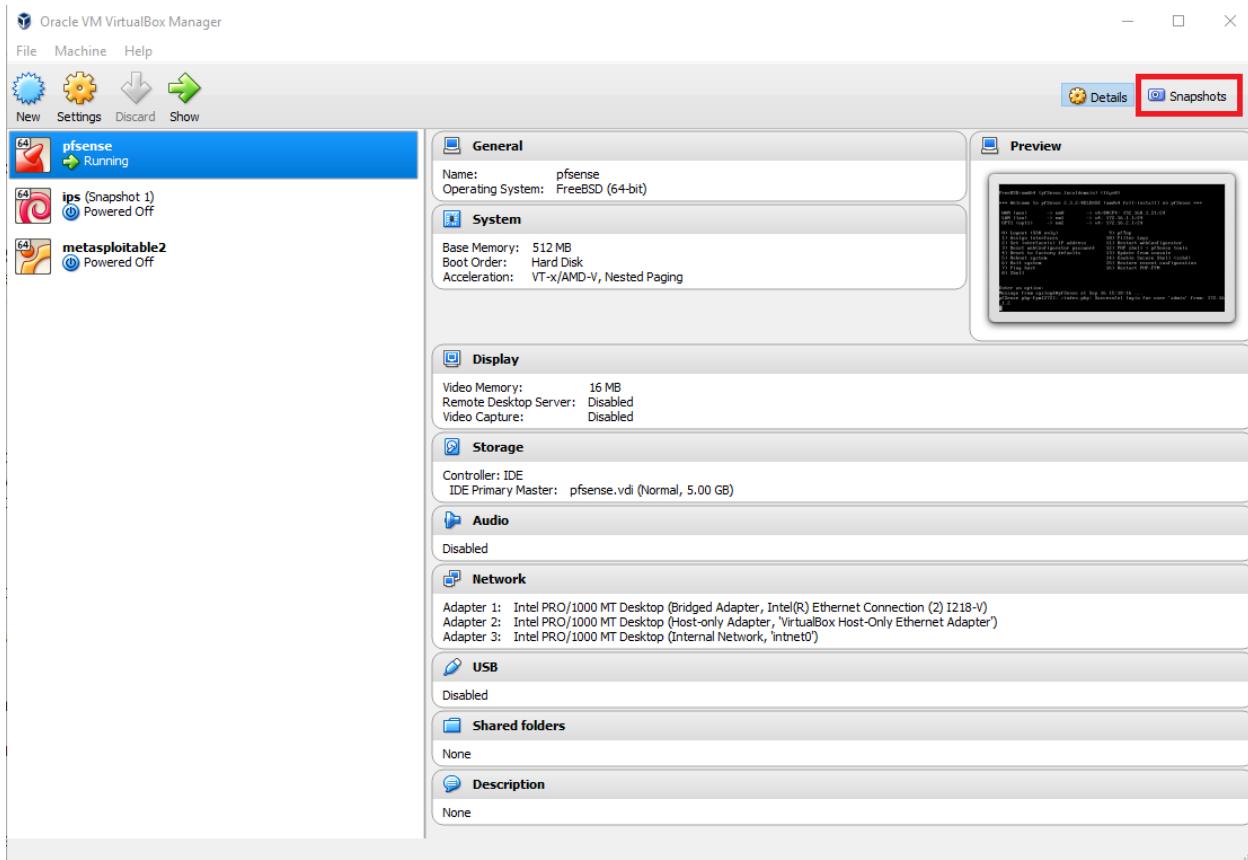
Click Save to be brought back to the previous page, then click “Apply Changes”. We just created an alias for RFC1918 networks (local networks that are not routable through the public internet) This will come in handy later for creating firewall rules around these networks. Next, navigate to System > Advanced. Click to fill in the checkbox next to the option “Disable webConfigurator anti-lockout rule”, and click Save on the bottom of the page.

Anti-lockout Disable webConfigurator anti-lockout rule

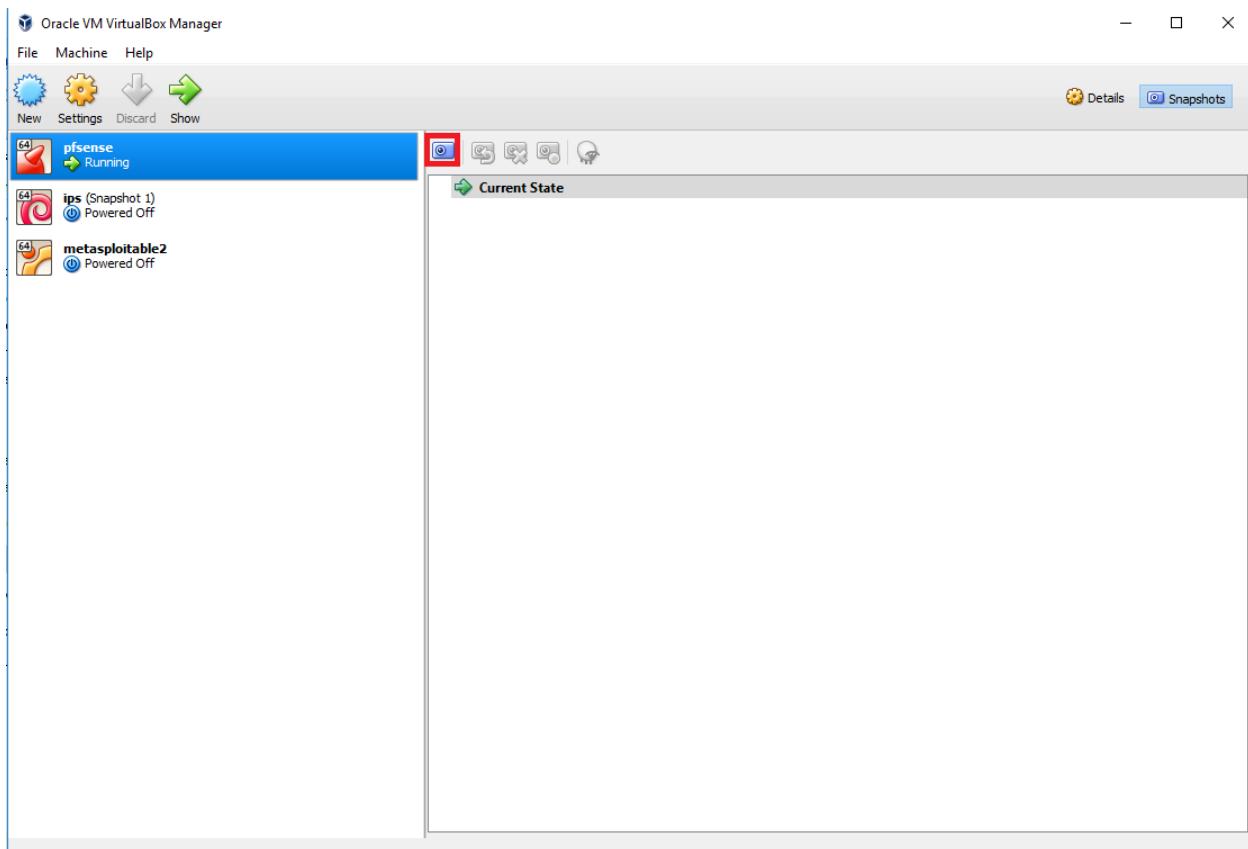
When this is unchecked, access to the webConfigurator on the LAN interface is always permitted, regardless of the user-defined firewall rule set. Check this box to disable this automatically added rule, so access to the webConfigurator is controlled by the user-defined firewall rules (ensure a firewall rule is in place that allows access, to avoid being locked out!) Hint: the “Set interface(s) IP address” option in the console menu resets this setting as well.

Take a Snapshot

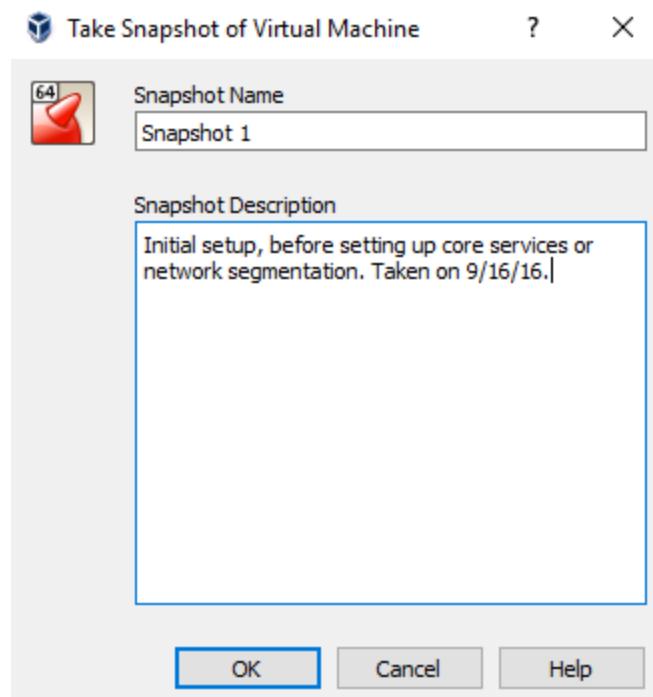
Snapshotting is a VERY important concept with VMs. It lets you restore your virtual machines to its state in the past when the snapshot was taken. This allows you to undo misconfigurations, solve problems or potential issues that might affect the VM, in relatively easy fashion. We’re going to take a snapshot of pfSense in its current state. In the VirtualBox main menu, click on the icon that looks like a little camera in the upper right corner of the menu, labeled “Snapshots”.



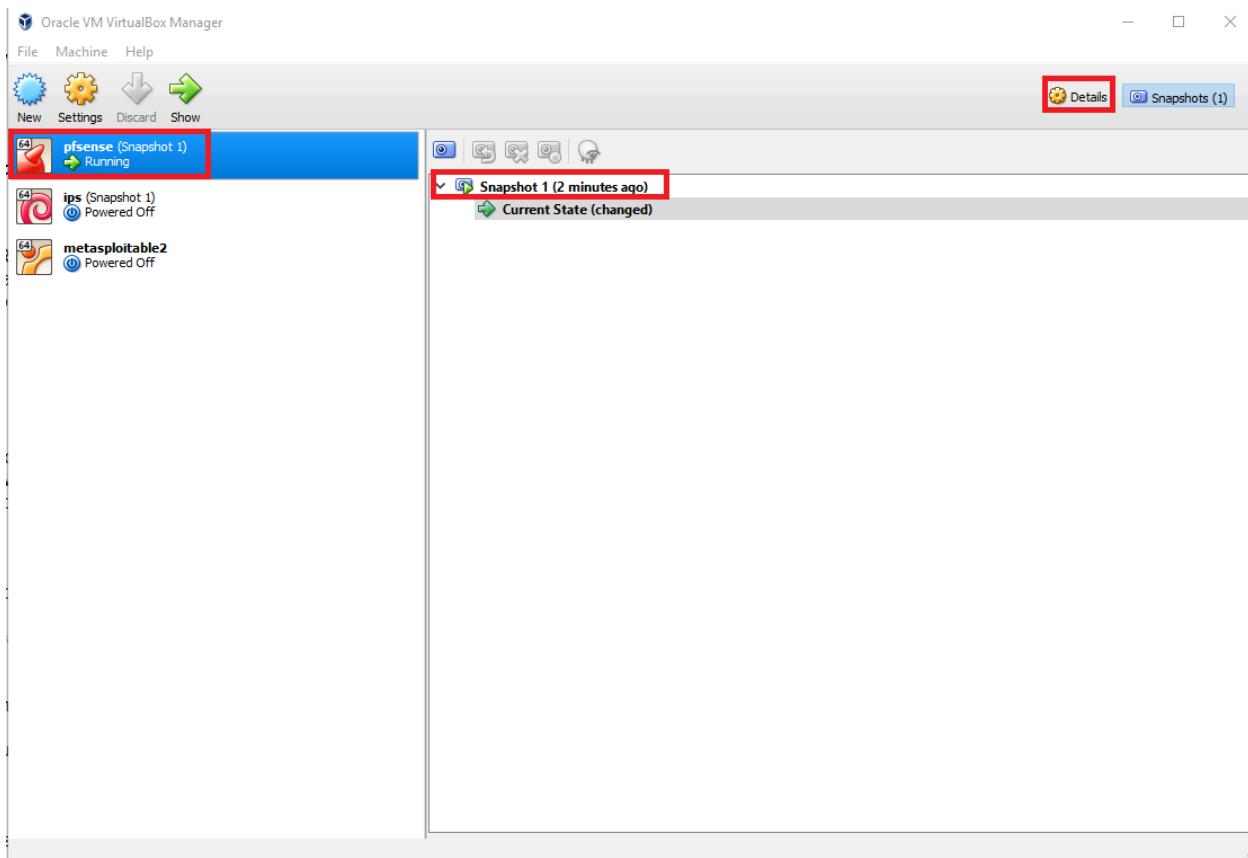
The view changes to display the snapshot manager. You will find another camera icon located in the upper left corner of right pane. Click on this icon to take a snapshot of the VM as it is right now.



Clicking the camera icon brings up a window entitled “Take Snapshot of Virtual Machine”. It asks you to name the snapshot and give it a description. If you keep multiple snapshots make sure you keep good descriptions that include WHEN the snapshot was taken, and WHAT STATE the VM is in when you took the snapshot. Once you’ve named it and added a good, valid description, click OK to generate the snapshot.

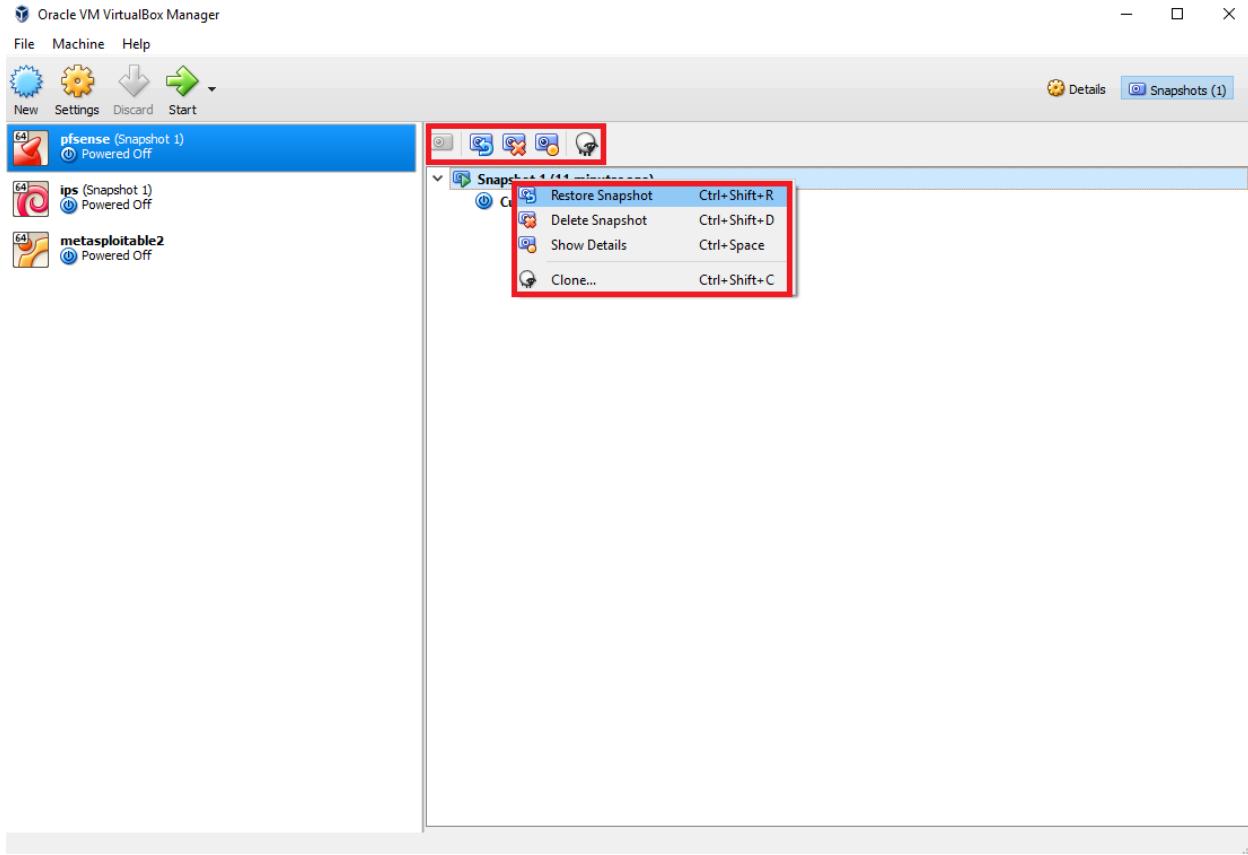


After a few seconds, the snapshot you just created shows up in the snapshot manager for the selected VM, and a little indicator appears next to the name of the VM, displaying the number snapshots available. Remember to delete old snapshots you are NOT planning to use any more, in order to conserve disk space. When you are done, you can click the cog labeled "Details" to return the VirtualBox main menu.



If you ever have to restore a VM from a snapshot, first, shutdown/power off the VM, then navigate to the snapshot manager. Right click on the snapshot you wish to revert to. Choose "Show Details" to view the description you (hopefully) provided for the snapshot, or select "Restore Snapshot" to start the snapshot restore process. This will take a moment or two, and voila, your VM is restored to its previous state.

Note that the icons above the snapshot pane perform the same function as those presented in the menu when right clicking the snapshot: the camera with the U-turn arrow performs a snapshot restore, the camera with the red X deletes the highlighted snapshot, and the camera with the circle shows you details on the highlighted snapshot. The sheep icon allows you to make an exact copy (also known as a clone) of the VM in its current state or at any state it was in, during any of the snapshots.



pfSense Summary

At the end of all this, you should have a pfSense VM with the following settings:

- 512MB of RAM
- 5GB of disk
- 1 Virtual CPU
- DVD drive disabled
- Audio disabled
- USB controller disabled
- 3 Network interfaces:
 - 1 Bridged (Bridge vswitch/WAN interface)
 - 1 Host-Only (Management vswitch/LAN interface)
 - 1 Internal Network (IPS vswitch1/OPT1 interface)

pfSense should be configured as followed:

- The WAN interface should be the VirtualBox network adapter connected to the Bridged Adapter. This interface should automatically get an IP address via DHCP. If this isn't happening, get some help with whoever administers your local network, or start troubleshooting.
- The LAN interface should have an IP address of 172.16.1.1.
- The LAN network should be 172.16.1.0 with a /24 (255.255.255.0) subnet mask

- DHCP should be configured for this network with a scope of 172.16.1.10-172.16.1.254
 - 172.16.1.3-172.16.1.9 are available for setting static DHCP mappings
- The OPT1 interface should have an IP address of 172.16.2.1
- The OPT1 network should be 172.16.2.0 with a /24 (255.255.255.0) subnet mask
- DHCP should be configured for this network with a scope of 172.16.2.10-172.16.2.254
 - 172.16.2.2-172.16.2.9 are available for static DHCP mappings.
- 172.16.1.2 should have a firewall rule configured to allow it access to the firewall over HTTPS; This is the anti-lockout firewall rule.
- You should have an alias for RFC1918 networks configured for use with the firewall policy.
- You should have at least one good snapshot you can revert to if you need to redo a step and somehow got lost/confused.

What's Next?

Now that you have an operational pfSense VM, your next steps are to read "[Network Configuration - Segmentation](#)" guide, followed immediately by "[Network Configuration - Core Network Services](#)". Make sure the firewall policies match those illustrated in the segmentation guide, and make sure that the pfSense VM is hosting all the services mentioned in the core network services guide to make your firewall fully functional and ready to handle your lab network. Once you verify that your firewall policies and services are properly configured, **SNAPSHOT THE VM** *THEN* you can move on, and try your hand at setting up the remaining VMs.

Your turn

The pfSense VM required a lot of custom configuration, both in VirtualBOx, as well as in the VM itself once the OS had been installed. Now that we have done that together, I'm going to have you create the rest of the virtual machines yourself -- with the exception of the Metasploitable 2 VM since it's a special case for reasons you will see momentarily. Below are a set of spec sheets to help you create the remaining virtual machines for the lab on your own. Think of them as checklists you should be able to run through on your own.

All of our Linux-based VMs are built on Debian Linux based distros (Both Ubuntu and Kali Linux are derived from Debian), so setup and configuration of all of our lab VMs should be very similar.

Kali Linux VM

Kali Linux is a popular penetration testing distro. Popular because hackers and script kiddies are lazy, and practically everything you need for performing a penetration test or joining

anonymous is installed by default. Kali is going to be our loud, obnoxious attacker that we're going to use as a noise generator for the express purpose of generating events on our IPS VM. I want you to perform the following tasks:

- Download Kali Linux 64-bit here: <https://www.kali.org/downloads/>
- Create a VM with the following settings:
 - Name the VM "Kali"
 - Choose "Linux" as the type
 - Choose "Debian (64-bit)" as the version
 - 4GB of RAM
 - 80GB fixed size VDI
 - 1-2 virtual CPUs (virtual CPUs can be adjusted under VM Settings > System > Processor tab once the VM has been created)
- Once the VM is created, adjust the following settings:
 - Under the System settings, ensure that the VM is using a PS/2 Mouse
 - Under Storage, point the virtual cd/dvd drive to where you downloaded the Kali Linux ISO, so the VM has a bootable drive, and we have an actual OS to install
 - Disable Audio
 - Disable the USB controller
 - Under Network settings, make sure that Adapter 1 (and only Adapter 1) is enabled and that it is attached to "Internal Network", and that the internal network is named "intnet" (if hosting Virtualbox on Windows) or "intnet0" (if hosting VirtualBox on a non-Windows OS). The internal network name should be the EXACT same as the internal network name for Adapter 3 on the "pfSense" VM
 - Document the MAC address for Adapter 1, under its advanced settings.
- Install Kali Linux
 - **Note:** If you configured the firewall on the pfSense VM, and decided to install and enable squid proxy services, during the apt package retrieval portion of the installer, you will be asked if your network uses a proxy server, and asked to enter the address of the proxy server as well as a username and password (if necessary). You will need to enter `http://172.16.2.1:3128` as the proxy server address. There is no username or password required to use the squid proxy
- After the install is completed, power down the VM and adjust the following settings:
 - Remove the virtual optical drive under the Storage Tree in Storage settings
 - Under system settings, disable the Floppy and Optical drives, and make the Hard Disk the first device the VM boots from
- Log in to the pfSense web UI, navigate to Services > DHCP Server, and add a static mapping to the OPT1 interface DHCP for the MAC address of Adapter 1 for the Kali VM; assign it the IP address 172.16.2.2, making sure to enter a description that tells you which VM this static mapping belongs to
- Power on the virtual machine and do the following:
 - Verify that the static mapping for the VM's IP address worked; open a terminal/shell and run `ifconfig -a | less` and verify that eth0 or whatever the network adapter's name is in the VM was given the IP address you configured its static mapping for in pfSense

- If the VM does not have an ip address, run the dhclient command to have the VM request an IP address from the DHCP server
- Verify the virtual machine can reach the internet
 - In a terminal/shell run the command ping -c 5 www.google.com
 - This will confirm DNS can resolve domain names, and that the VM can reach the internet
- Update/patch the VM
 - In a terminal/shell, run the following, and depending on your network connection, etc., be prepared to wait:
 - `export DEBIAN_FRONTEND=noninteractive; apt-get -q update; apt-get -y -q dist-upgrade`
- Take a snapshot of the VM

Now you should have a functional Kali Linux VM. At this point, the virtual machine should be fully operational, and you should be able to control it from the VM's console. However, if you want to enable and configure SSH access to this host, I have configured a guide for doing so. Jump to "[How to Enable SSH on Kali Linux](#)" to enable the ssh service for this VM.

SIEM VM

SIEM is shorthand for Security Intrusion Events Manager. This is fancy security nomenclature for "log aggregator"; we will be having our IPS VM log its events here. We're going to be running Splunk on this VM. Splunk is a commercial program for managing logs on a pretty large scale. By default, and with no licensing, Splunk only allows you to collect or "index" 500MB worth of logs per day however, there are /ways/ around this that we'll talk about later. Perform the following tasks to set up the "SIEM" VM:

- Download the latest Ubuntu Server 64-bit LTS release. In our case, this is Ubuntu Server 16.04.1, which can be found here: <http://www.ubuntu.com/download/server>
- Create a VM with the following settings:
 - Name the VM "SIEM"
 - Choose "Linux" as the type
 - Choose "Ubuntu (64-bit)" as the version
 - 4GB of RAM
 - 80GB fixed size VDI
 - 1-2 virtual CPUs
- Once the VM is created, adjust the following settings:
 - Under the System settings, ensure the VM is using a PS/2 Mouse
 - Under Storage, point the virtual cd/dvd drive to where you downloaded the Ubuntu Server Linux ISO, so the VM has a bootable drive, and we have an actual OS to install
 - Disable Audio
 - Disable the USB controller

- Under Network settings, make sure that Adapter 1(and only adapter 1) is enabled and that it is attached to “Host-Only Adapter”, and that the name is “VirtualBox Host-Only Ethernet Adapter” -- the same as “Adapter 2” on the pfSense VM
 - Document the MAC address for Adapter 1, under its advanced settings
- Install Ubuntu Server
 - Be sure to install the command line utilities and SSH Server role when asked. You should not need to install any other roles or features for this server
 - **Note:** If you configured the firewall on the pfSense VM, and decided to install and enable squid proxy services, during the apt package retrieval portion of the installer, you will be asked if your network uses a proxy server, and asked to enter the address of the proxy server as well as a username and password (if necessary). You will need to enter <http://172.16.2.1:3128> as the proxy server address. There is no username or password required to use the squid proxy
- After the install is completed, power down the VM and adjust the following settings:
 - Remove the virtual optical drive under the Storage Tree in Storage settings
 - Under system settings, disable the Floppy and Optical drives, and make the Hard Disk the first device the VM boots from
- Log in to the pfSense web UI, navigate to Services > DHCP Server, and add a static mapping to the LAN interface DHCP, for the MAC address of adapter 1 for the SIEM VM; assign it the IP address 172.16.1.3, making sure to enter a description that tells you which VM this static mapping belongs to
- Power on the Virtual Machine and do the following:
 - Verify that the static mapping for the VM’s IP address worked; open a terminal/shell and run `ifconfig -a | less` and verify that eth0 or whatever the network adapter’s name is in the VM was given the IP address you configured its static mapping for in pfSense
 - If the VM does not have an ip address, run the `dhclient` command to have the VM request an IP address from the DHCP server.
 - Verify the virtual machine can reach the internet
 - In a terminal/shell run the command `ping -c 5 www.google.com`
 - This will confirm DNS can resolve domain names, and that the VM can reach the internet
 - Update/patch the VM
 - In a terminal/shell, run the following, and depending on your network connection, etc., be prepared to wait:
 - `sudo export DEBIAN_FRONTEND=noninteractive; sudo apt-get -q update; sudo apt-get -y -q dist-upgrade`
 - Take a snapshot of the VM

IPS VM

This VM is going to be responsible for running the AFPACKET bridge between the IPS 1 and IPS 2 networks. In the case of VirtualBox, these are going to be two Internal Networks. Perform the following actions to set up the IPS VM:

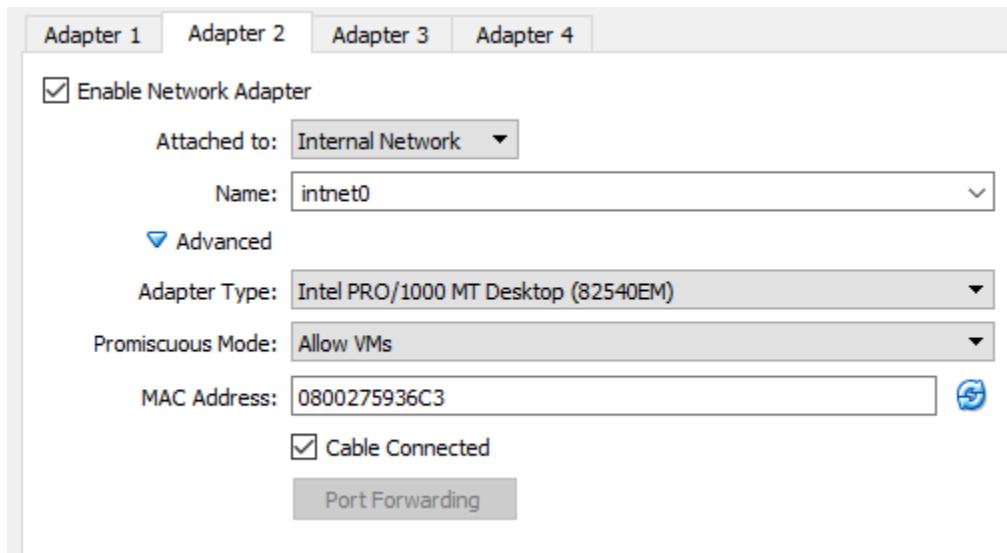
- We will be installing Ubuntu Server 16.04.1 64-bit. If you installed the siem VM already, you shouldn't need to download the ISO again, but just incase, download it here: <http://www.ubuntu.com/download/server>
- Create a VM with the following settings:
 - Name the vm “IPS”
 - Choose “Linux” as the type
 - Choose “Ubuntu (64-bit)” as the version
 - 2GB of RAM
 - 80GB fixed size VDI
 - 1-2 virtual CPUs (virtual CPUs can be adjusted under VM Settings > System > Processor tab once the VM has been created)
- Once the VM is created, adjust the following settings:
 - Under the System settings, ensure that the VM is using a PS/2 Mouse
 - Under Storage, point the virtual cd/dvd drive to where you downloaded the Ubuntu Server Linux ISO, so the VM has a bootable drive, and we have an actual OS to install
 - Disable Audio
 - Disable the USB controller
 - Under Network settings, make sure that Adapter 1, Adapter 2, and Adapter 3 are all enabled, and are using the following settings:
 - Adapter 1 should be attached to “Host-Only Adapater” named “VirtualBox Host-Only Ethernet Adapter”
 - Under Adapter 1, be sure to click “Advanced” and document the MAC address of this interface; This is going to be the primary network interface for this host
 - Adapter 2 should be attached to “Not attached”
 - Click on Advanced settings. Document the MAC address of this interface
 - Adapter 3 should be attached to “Not attached”
 - Click on Advanced settings. Document the MAC address of this interface
- Install Ubuntu Server
 - The installer will reach a section titled “Configure the network” and ask you which network card is the primary network interface for this system
 - Usually, these interfaces are labeled eth0, eth1, and eth2, but may be labeled differently in your case
 - In almost all cases, the first network adapter in Client Hyper-V that is added to the VM will line up with the first network interface listed on the “Configure the network screen (This is almost always eth0). Hit Enter, and wait for it to try and configure the interface via DHCP
 - The installer will try to get an IP address for the network interface via DHCP. If this is not successful, select the <Go Back> option, and select one of the other two network interfaces listed, and try again

- If none of the interfaces work, verify the pfSense VM is running, connected to the “Management Network”, and that the DHCP service is enabled, then try again until DHCP configuration is successful
 - Be sure to install the command line utilities and SSH Server role when asked. You should not need to install any other roles or features for this server
 - **Note:** If you configured the firewall on the pfSense VM, and decided to install and enable squid proxy services, during the apt package retrieval portion of the installer, you will be asked if your network uses a proxy server, and asked to enter the address of the proxy server as well as a username and password (if necessary). You will need to enter `http://172.16.2.1:3128` as the proxy server address. There is no username or password required to use the squid proxy
- After the install is completed, power down the VM and adjust the following settings:
 - Remove the virtual CD/DVD under the Storage Tree in Storage settings
 - Under system settings, disable the Floppy and Optical drives, and make the Hard Disk the first device the VM boots from
 - Adapter 2 should be attached to “Internal Network” named “intnet” or “intnet0” -- the same as the kali VM
 - Under advanced settings, there is a dropdown labeled “Promiscuous Mode”. Set this to “Allow VMs” (***This is VERY important!***)
 - Adapter 3 should be attached to “Internal Network” named “intnet1”. You will have to name this new internal network. This network is the “IPS 2” network
 - Under advanced settings, there is a dropdown labeled “Promiscuous Mode”. Set this to “Allow VMs” (***This is VERY important!***)
- Log in to the pfSense web UI, navigate to Services > DHCP Server, and add a static mapping to the LAN interface DHCP, for the MAC address of adapter 1 for the IPS VM; assign the IP address 172.16.1.4, making sure to enter a description that tells you which VM this static mapping belongs to
- Power on the Virtual Machine and do the following:
 - Determine which interface in the ips VM maps to which interface in the VM’s network settings
 - In the terminal, run `ifconfig -a | egrep HWaddr`
 - This command will show you the name of the interface and its HWaddr (Hardware Address - another name for MAC address). Compare this to the MAC addresses you documented earlier
 - You need to determine the name of the interface that has the MAC address associated with “Adapter1” in the VirtualBox network settings. This is going to be the only network interface this VM can use to talk to other hosts over the network
 - Verify that the static mapping for the VM’s IP address worked; open a terminal/shell and run `ifconfig -a | less` and verify that eth0 or whatever the network adapter’s name is in the VM was given the IP address you configured its static mapping for in pfSense

- If the VM does not have an ip address, run the dhclient command to have the VM request an IP address from the DHCP server
- Verify the virtual machine can reach the internet
 - In a terminal/shell run the command ping -c 5 www.google.com
 - This will confirm DNS can resolve domain names, and that the VM can reach the internet
- Update/patch the VM
 - In a terminal/shell, run the following, and depending on your network connection, etc., be prepared to wait:
 - sudo export DEBIAN_FRONTEND=noninteractive; sudo apt-get -q update; sudo apt-get -y -q dist-upgrade
- Take a snapshot of the VM

Promiscuous Mode

In the configuration checklist above for the “IPS” VM, I very briefly mentioned configuring promiscuous mode for the network adapters on the. This setting MUST be set to “Allow VMs”. Promiscuous mode allows a network card that has this setting enabled to collect or “sniff” traffic observed on a given virtual network from other hosts. This configuration is extremely important and MUST be set on the network adapters connected to “intnet/intnet0” and “intnet1” for the “IPS” VM to perform its function correctly.



Metasploitable 2

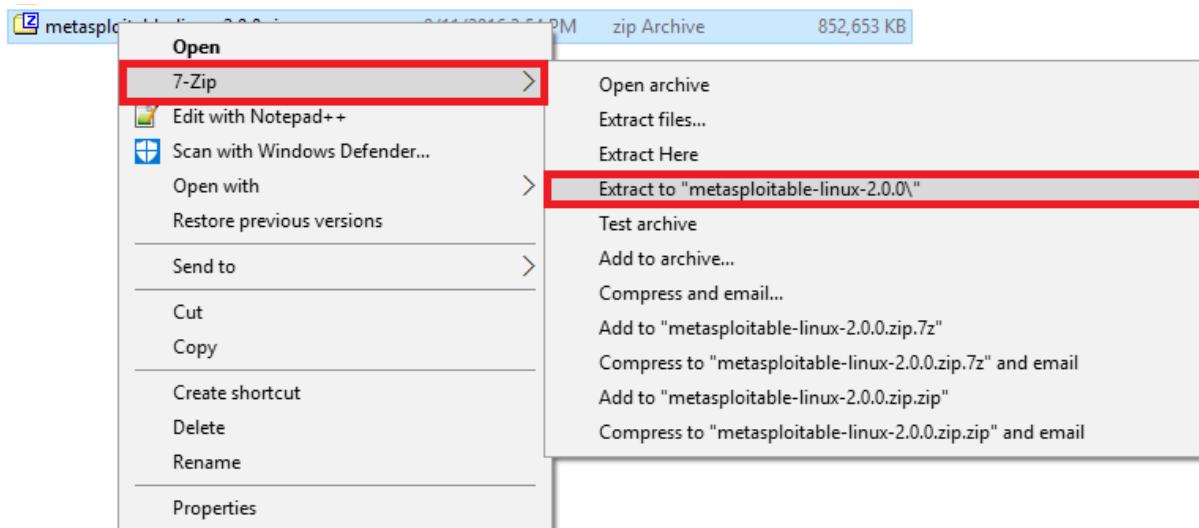
Setting up Metasploitable 2 is a little bit different compared to the other virtual machines, because the VM comes in a prepackaged for use with VMware products. However, VirtualBox is able to utilize these prepackaged files as well. Download Metasploitable 2 from <https://sourceforge.net/projects/metasploitable/files/Metasploitable2/metasploitable-linux-2.0.0.zip/download>. Once you have downloaded the zip file, you will need a compression utility to unzip this file.

Linux/OSX/BSD users usually have the unzip command available. Open a console/shell, navigate to where the file was downloaded (typically ~/Downloads), then run `unzip metasploitable-linux-2.0.0.zip`. You may want to move the `metasploitable-linux-2.0.0` directory to the same directory you are storing the VDI files for your other lab VMs.

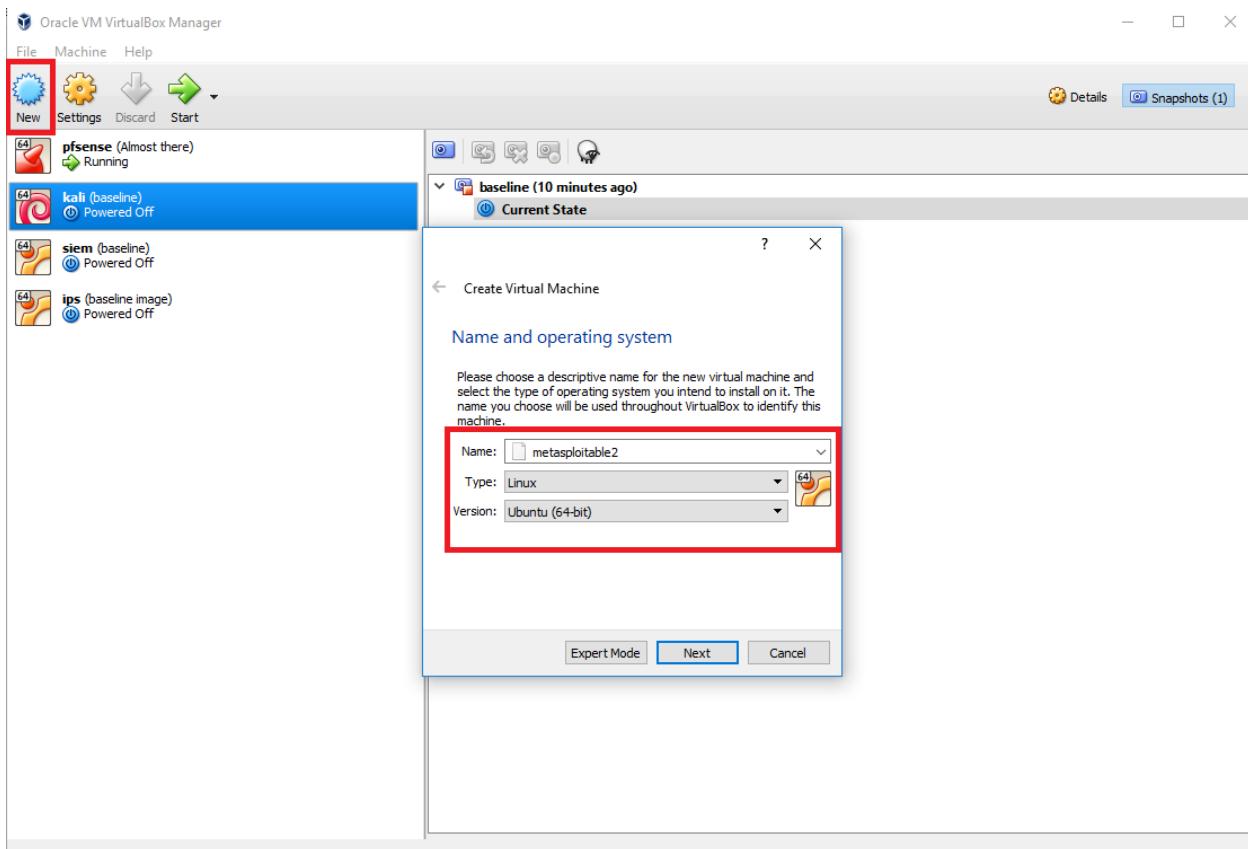
On windows, I prefer 7-Zip (<http://www.7-zip.org/download.html>). Move the .zip file to the directory you are storing your VirtualBox VMs. In my case, I moved the .zip file to D:\vbox

This PC > Data (D:) > vbox			
Name	Date modified	Type	Size
ips	9/19/2016 12:29 PM	File folder	
kali	9/19/2016 12:44 PM	File folder	
pfsense	9/19/2016 12:28 PM	File folder	
siem	9/19/2016 12:29 PM	File folder	
metasploitable-linux-2.0.0.zip	9/11/2016 3:54 PM	zip Archive	852,653 KB

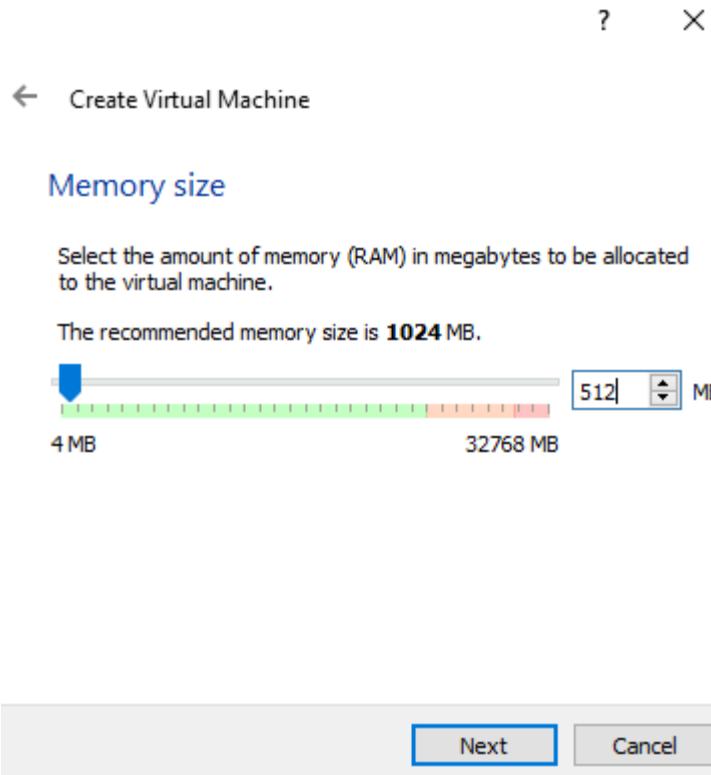
Right click on the zip file, and use 7-Zip to extract the files in the zip to metasploitable-linux-2.0.0



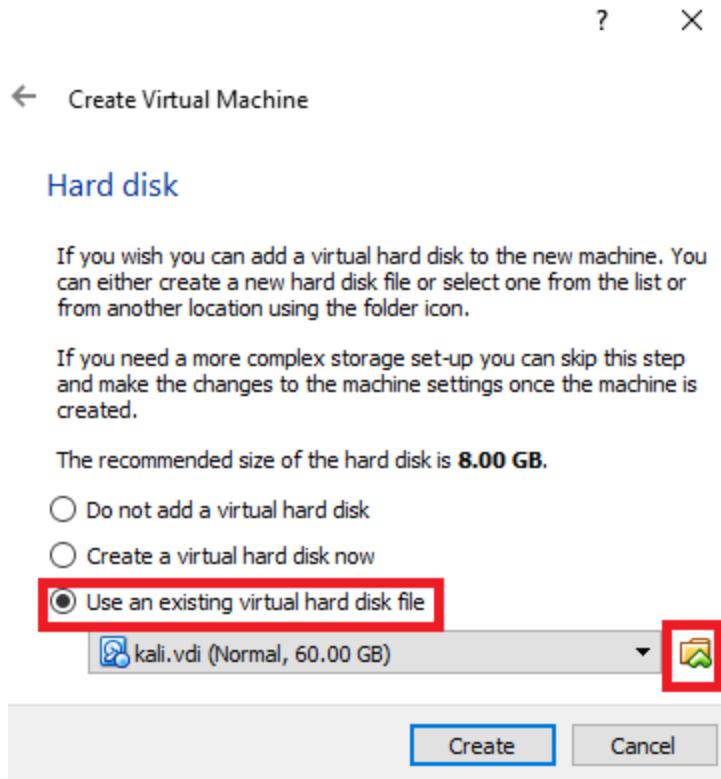
Fire up VirtualBox and select “New” to create a new VM. Name the VM “Metasploitable 2”. For the OS type, select “Linux”. For the version, select “Ubuntu (64-bit)”. Then click next.



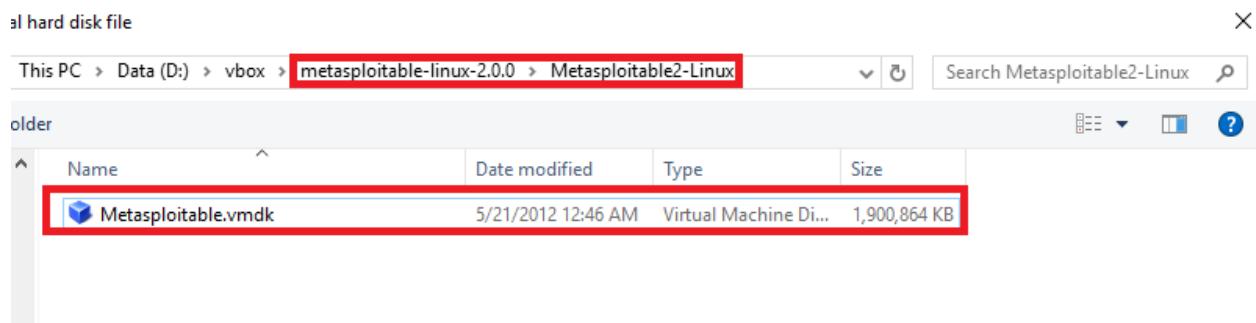
Allocate 512MB of RAM to the VM, and click next.



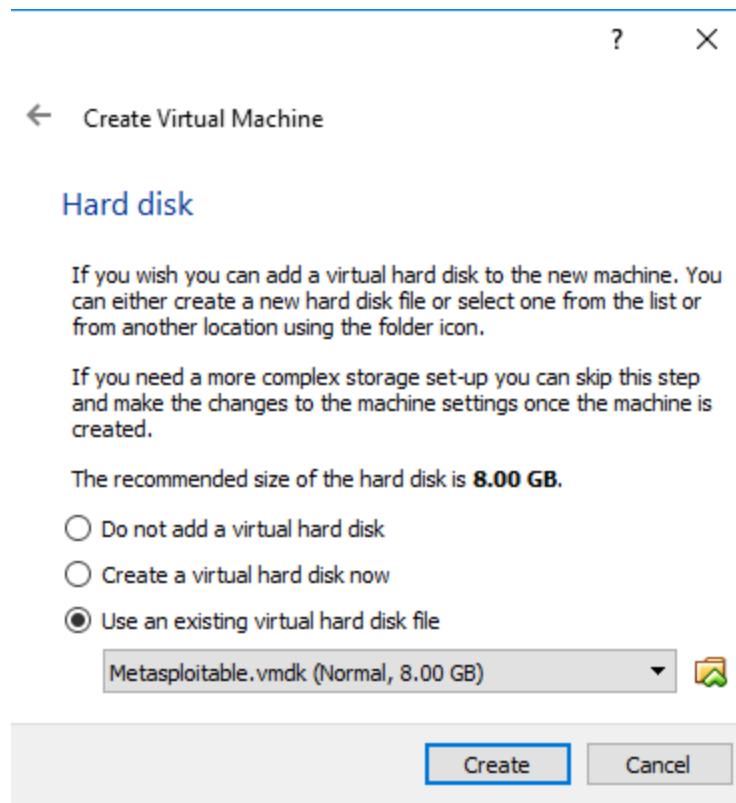
On the next screen, click the “Use an existing virtual hard disk file” radio button, and click the folder with the green icon. This will open an explorer window.



Navigate to the `metasploitable-linux-2.0.0` directory we extracted from the metasploitable 2 zip file moments ago. Inside, is a subdirectory named `Metasploitable2-Linux`. Double click to enter that directory. In my case, the full directory path is `D:\vbox\metasploitable-linux-2.0.0\Metasploitable2-Linux`. Double click on the `Metasploitable.vmdk` file.



Selecting the vmdk file should have brought you back to the “Hard disk” screen. Click the create button. The virtual machine is added to the main VirtualBox menu.



The “Metasploitable 2” VM should be added to our list on the VirtualBox main menu. Next up is something you should be familiar with by now: we are going to adjust the settings of the virtual machine.

- Go to the “System” screen. Under “Boot Order”, uncheck the Floppy and Optical options, and make the Hard Drive the first boot option. Change the “Pointing Device:” to “PS/2 Mouse”
- Remove the virtual CD/DVD under the Storage Tree in Storage settings
- Disable Audio
- Disable the USB controller
- Navigate to “Network” and under the “Adapter 1” tab, ensure the adapter is enabled. Attach the adapter to “Internal Network” named “intnet1” (the EXACT same as Adapter 3 for the “IPS” VM). Click on “Advanced” and document the MAC address of the interface.
- In pfSense, under Services > DHCP Server, add a static DHCP mapping under the OPT1 interface for this virtual machine. Give it the IP address 172.16.2.3 and make sure to enter a description that states which VM the mapping belongs to.

At this point, the VM should be bootable. Feel free to power it on to make sure there aren’t any problems. If you’re greeted with a login prompt, then this indicates everything is working fine. Power down the virtual machine and take a snapshot.

Note: This VM is located on “intnet1”, which we haven’t bridged to “intnet/intnet0” with the “IPS” VM yet. So in spite of having a static DHCP mapping, it has no way of reach out to the “pfSense” virtual machine for an IP address yet. We are going to fix this soon.

Next Steps

All of the VirtualBox setup steps have been completed. However, you're not quite done yet. Here is a checklist of tasks to complete:

- If you are running Virtualbox on a Windows host, and haven't completed the chapter "[Defense in Depth for Windows Hosted Hypervisors](#)", I would very highly suggest doing so in order to harden your hypervisor host.
 - While not strictly necessary, you may also want to complete the "[Remote Lab Management](#)" guide. Specifically, the sections related to [Windows Remote Access](#), [Enabling SSH on Kali Linux](#), and if you're lazy like me, The section on [Securing root SSH access](#). This will allow you to remotely manage all of your lab VMs, as well as finish the IPS and Splunk setup guides much more easily than through the VirtualBox console.
 - You still need to install IDS/IPS software on the IPS VM to get a functioning AFPACKET bridge between the "IPS 1" and "IPS 2" networks. Check out the "[IPS Installation Guide](#)" to learn how to do this with either Snort or Suricata as your IPS software of choice.
 - The SIEM VM needs to have Splunk installed and configured. You'll need to complete the "[Splunk Installation Guide](#)".
 - You may want to consider reading the [Automated Patching for Linux Lab VMs](#) chapter, and implementing the updater.sh script for your Linux VMs.
 - Do you want some ideas on where to take your lab? Check out the chapter "[In Your Own Image](#)", for some tips on how to mold your VM lab to better suit your needs.

Setup - VMware Fusion Pro

Note: This guide is written for VMware Fusion Pro. Specifically, the latest version as of writing, version 8.5. Fortunately for us, OSX is the only OS VMware Fusion was designed for, so you

shouldn't need to worry about OS types. This guide was written using OSX 10.12 "Sierra" the latest version as of writing, but should work just fine with 10.11 "El Capitan".

Be aware that this guide specifically requires VMware Fusion Pro. The pro edition allows us to modify virtual networking and create additional virtual networks, as well as a few other odds and ends. The virtual network editor is what we're concerned with, and regular Fusion does not have it.

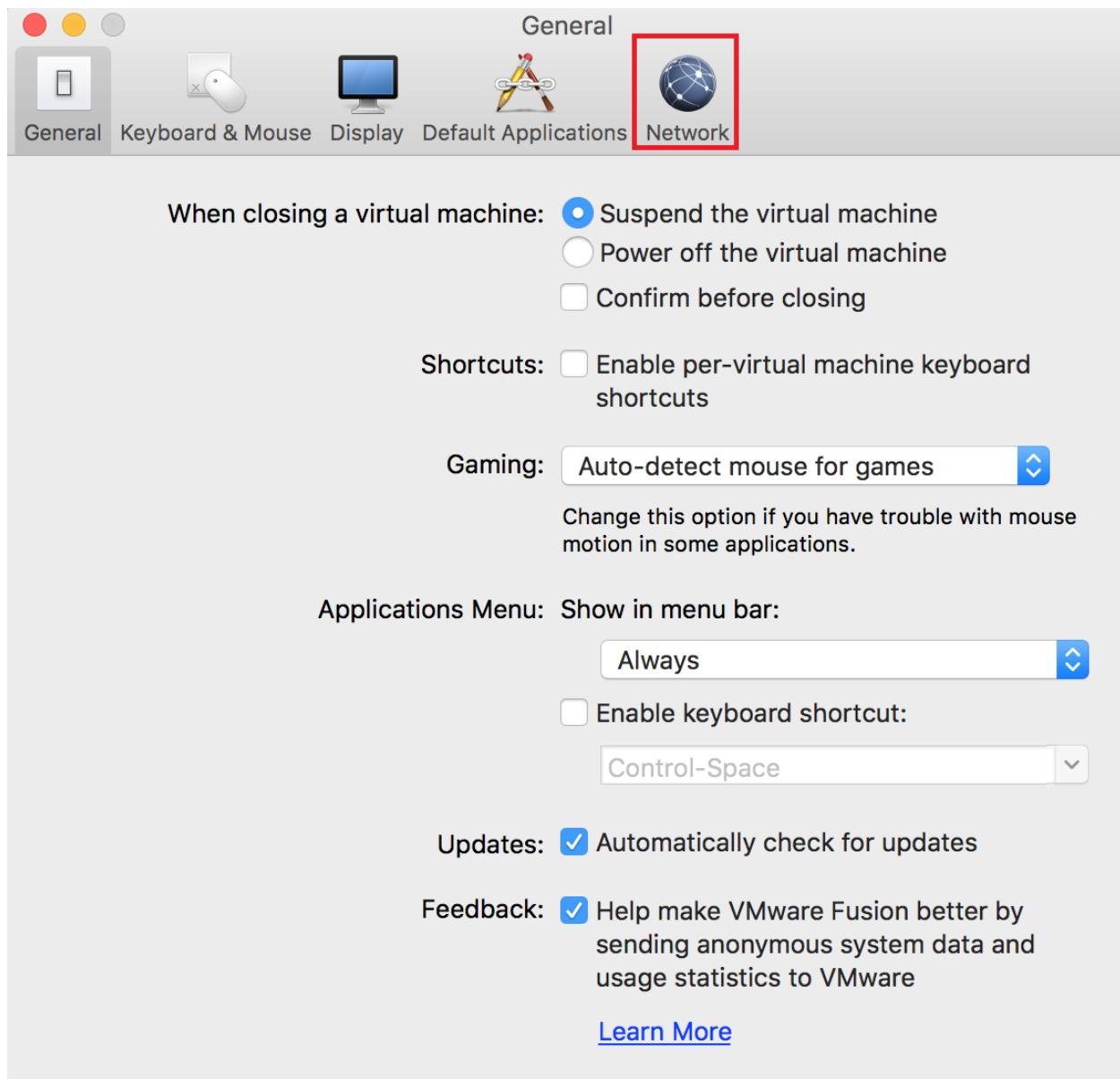
Installation

First, you'll need to download VMware fusion from vmware website. You can a trial copy here: <http://www.vmware.com/products/fusion/fusion-evaluation.html>. Be sure to download and install pro edition. Run through the installer and go through the motions. If you have a license, you'll be prompted to enter it. If you don't continue on and use the trial time you have for now.

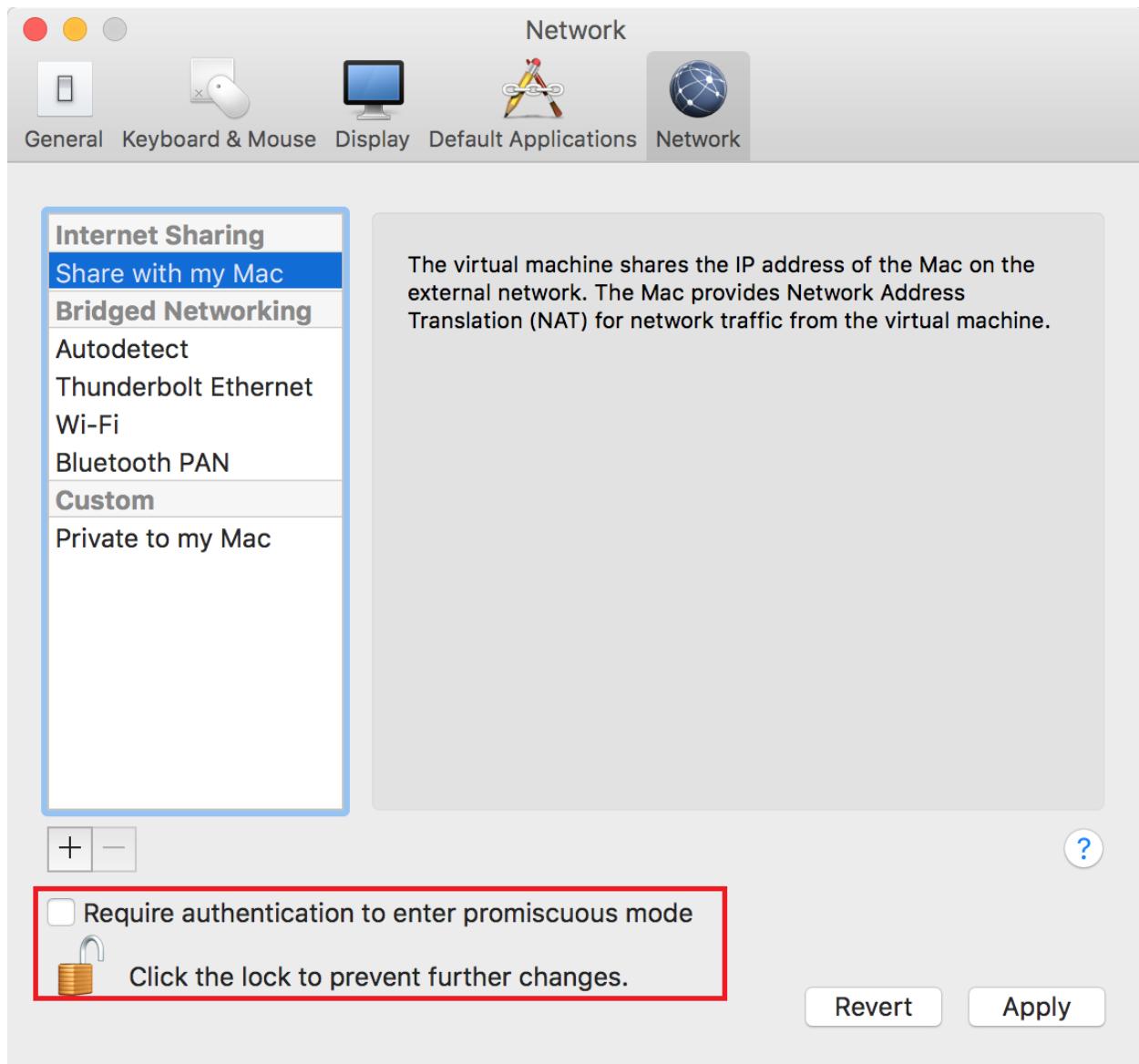
Hypervisor Preferences

Start up VMware fusion and you'll be greeted by a screen that automatically assumes you want to create or add a new VM.

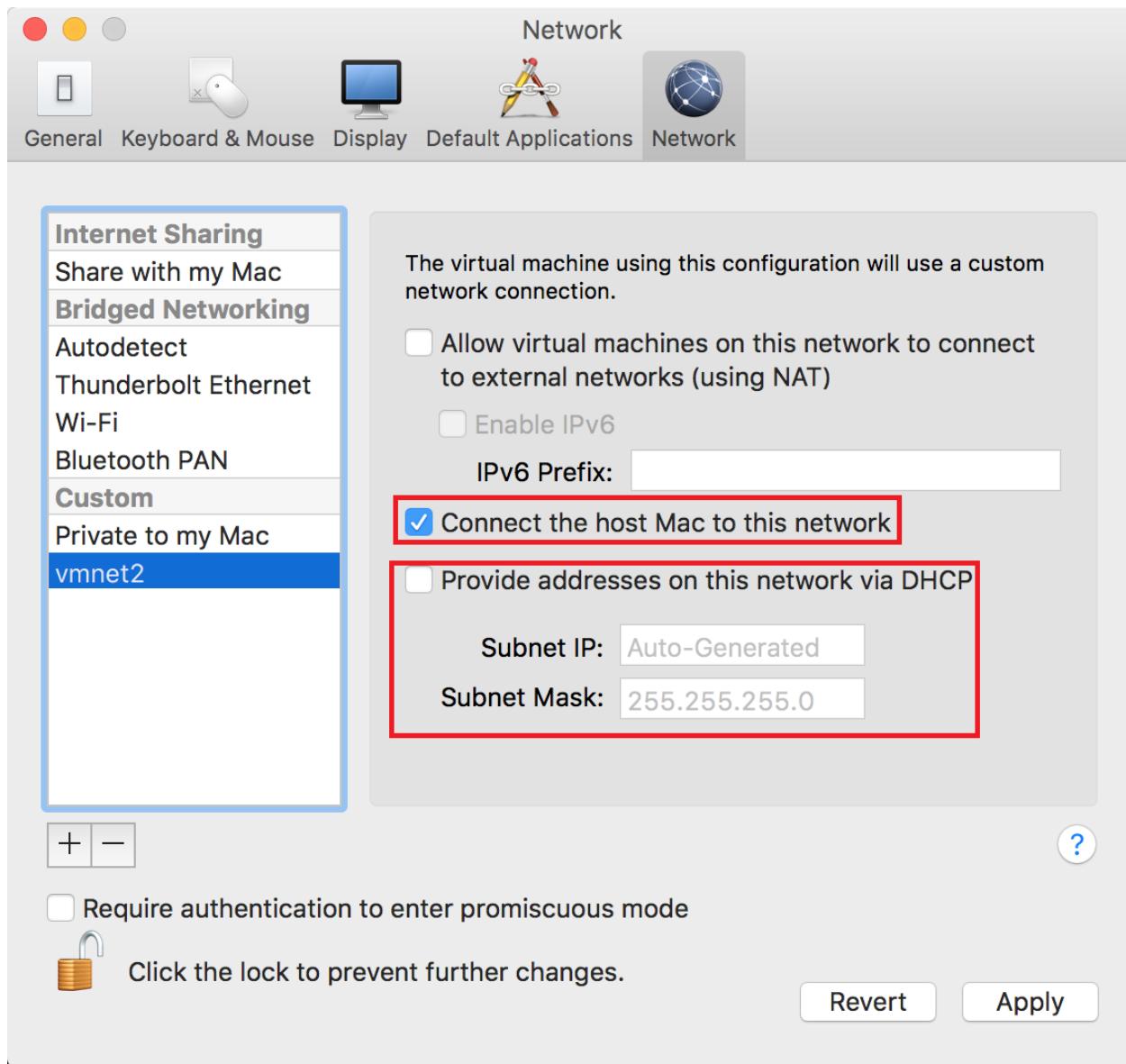
Before we go down that route however, We have to make some changes to Fusion, specifically. In the main menu bar, click on VMware Fusion, then click on "Preferences...". This will open a menu that allows for some modification of vmware fusion. We're interested in the Network options. Click on the "Network" icon.



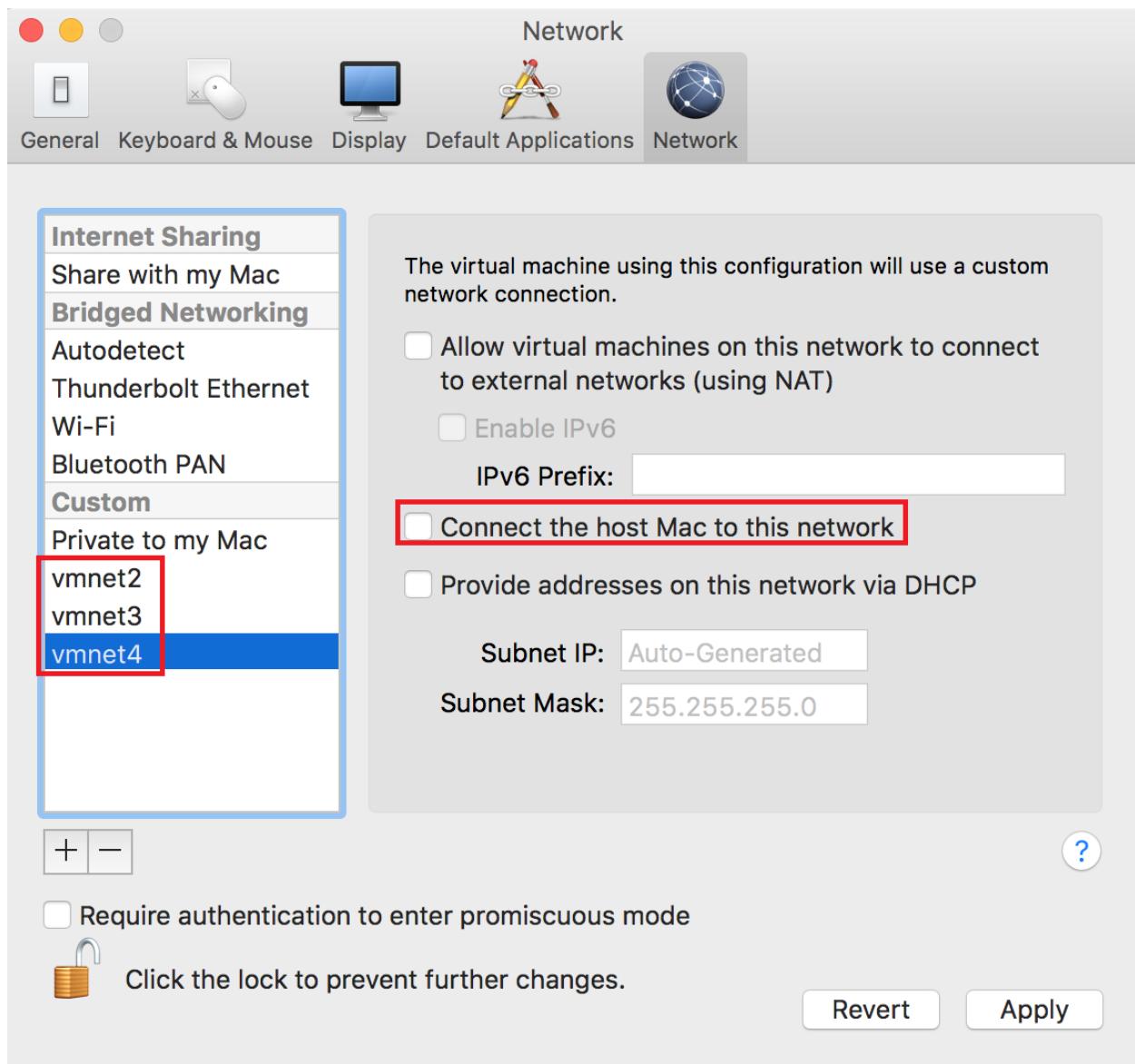
The network settings menu allows you to define network bridging settings, as well as define custom network settings. Before you can do anything, however, you'll need to click the lock on the bottom of the menu, and input your password. After doing so, uncheck the "Require authentication to enter promiscuous mode" button. This makes it so that machines that require promiscuous mode to perform certain tasks (like our IPS VM) will not require you to enter your mac user's password, every time a network interface wants to go in promiscuous mode. If you're paranoid, or want that extra layer of security, leave it enabled, but know that every time you restart the IPS vm, that you'll likely need to enter your password twice (you have authenticate for each interface on each system that wants to go into promiscuous mode).



Next, we have to create three virtual networks. Click the + sign underneath the pane that lists the networks/connections available for VMware Fusion, and a new virtual network should appear under "Custom", labeled "vmnet2". By default, Fusion will highlight this new network and show you its settings. Uncheck the "Provide addresses on this network via DHCP" checkbox. For some reason that is completely beyond me, you have no capability to modify DHCP settings for the host-only network in VMware Fusion (8.5 pro, at least as of this writing), so instead using the preconfigured host-only network that we would normally use as the Management network we have to make our own, since we don't want VMware handling DHCP settings.



Next, we're going to create two more virtual networks. They should be labeled "vmnet3" and "vmnet4" respectively. Just like with vmnet 2, uncheck the "Provide addresses on this network via DHCP" checkbox. Additionally, **uncheck the "Connect the host Mac to this network" checkbox for both vmnet3 and vmnet4. It is very important that you do this.** These networks will serve as the IPS1 and IPS2 networks, and we do not want the macbook to have direct exposure to these networks. When you are finished, you should have three new virtual networks. Vmnet2 will be our Management network, vmnet 3 will be our IPS 1 network, and vmnet4 will be our IPS2 network.



When you're done, click the lock to well.. Lock in your changes. Fusion will prompt you to apply your new settings. Apply the settings, then exit out of the menu by clicking the red circle in the upper left corner of the preferences menu.

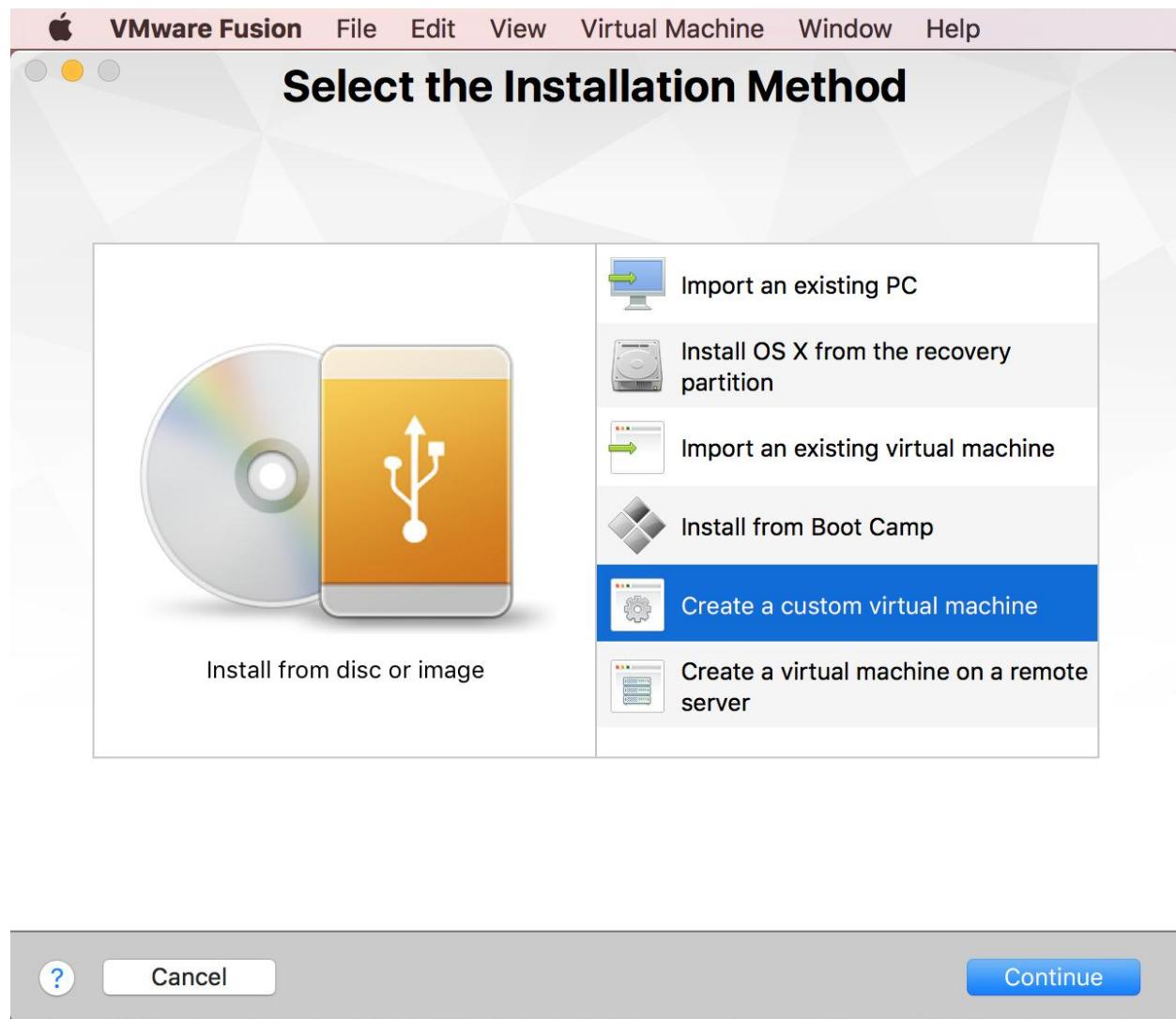
Creating the First VM, pfSense

pfSense is the keystone holding this entire configuration together. Personally, its my favorite firewall distro due to ease of use, the amount of functionality it includes out of the box, combined with a plugin/add-on system for additional functionality ; If you have the CPU, RAM, and Disk, pfSense can easily be converted into a so-called “Next-Generation” firewall. To start, make your way to <https://www.pfsense.org/download/>. Download the latest Installation ISO for the amd64 architecture. After you download the ISO, you will need a decompression tool to uncompress

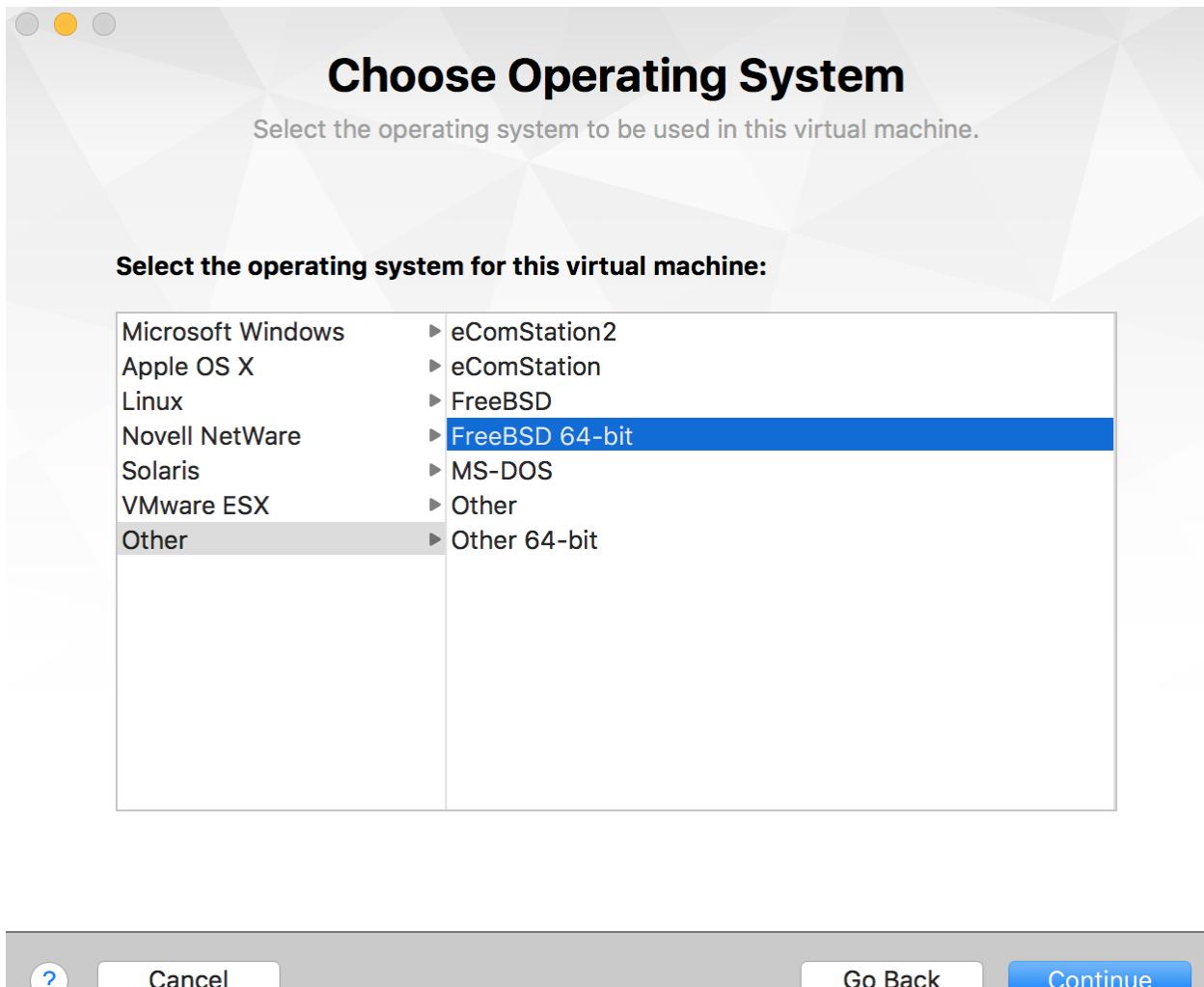
the pfSense ISO. By default, if you double click on a gzipped file in Finder on OSX, it will automatically make a decompressed copy for you. If you prefer to use the command line, the “gunzip” will decompress the file, but will not leave the original, compressed copy.

Adding a New VM

As previously mentioned, when you first started VMware Fusion, its very gung-ho about creating a VM for you, so we're going to start that process now. You should already have a window open, titled “Select the Installation Method”, from starting VMware Fusion already. If you do not (e.g. you hit cancel), On the main menu bar, click File, then click “New...”. To bring up the “Select the Installation Method” window for the new VM wizard. Click on “Create a custom virtual machine”, then click Continue.

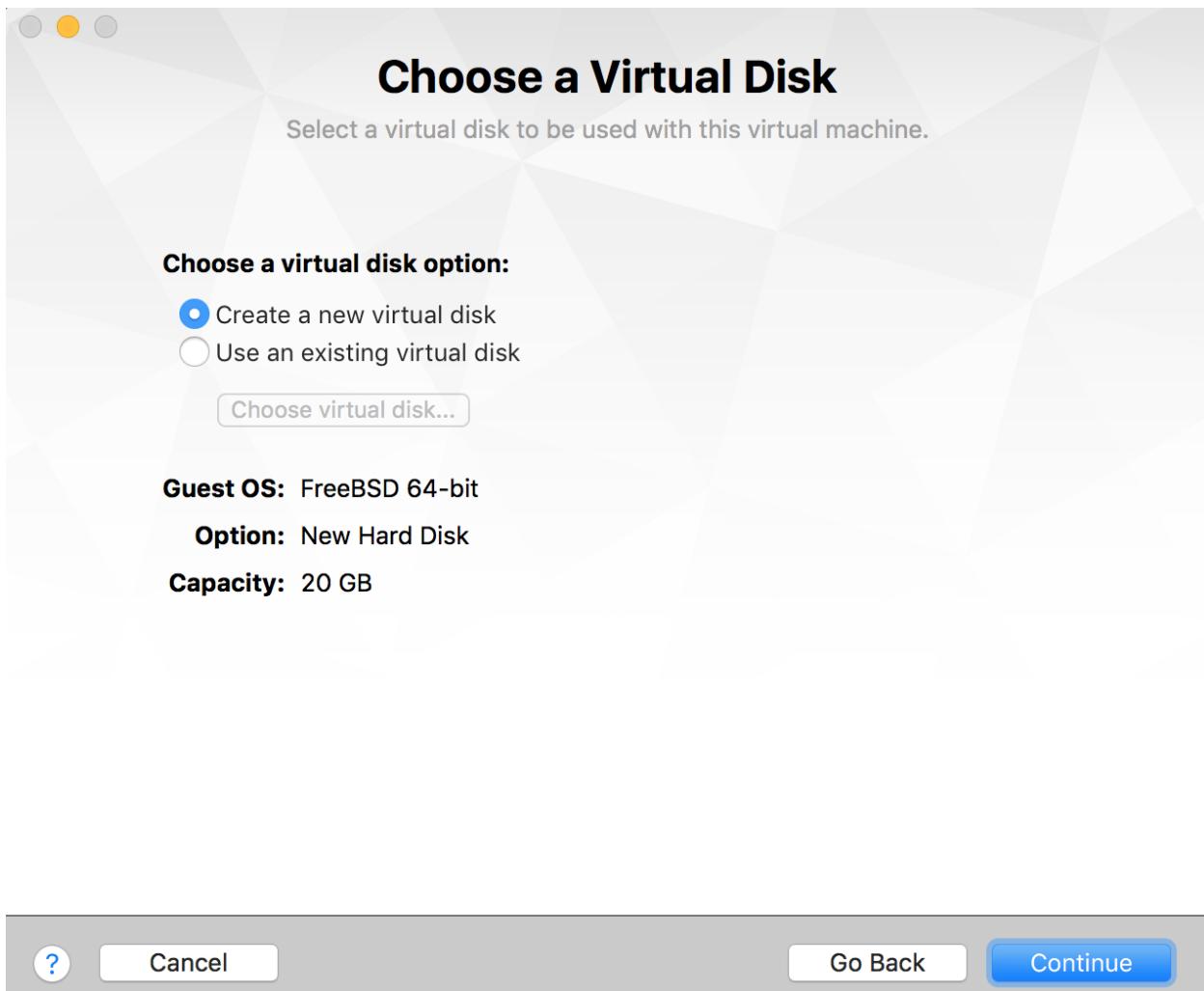


The next screen has you choose the Operating System, and Version. Click on “Other”, then select “FreeBSD 64-bit”. Click Continue.



On the next screen you will create a new virtual disk for the pfSense VM. Ensure that the “Create a new virtual disk” radio button is selected, then click Continue.

The next screen is entitled “Finish”, but we’re not done just yet. Click on the “Customize Settings” button.



A small window asking to save your VM will pop up. Input “pfSense” in the “Save As:” input field. If you have any custom tags you want to associate to this VM, you can enter them in the “Tags:” input field. The “Where:” field allows you to send your VM to a custom location. By default, VMware Fusion saves your VMs to /Users/[username]/Documents/Virtual Machines. If you had another destination you wanted to store your virtual machines to, you can specify that here. For our purposes, the default location suits us just fine. Unfortunately, **there is no universal “Save all of my VMs to this custom folder” option, you have to remember to perform this action on a per-VM basis**. When you have renamed your VM and are satisfied with tags and storage location, click the “Save” button.

Finish

The configuration of the virtual machine is now complete.

Virtual Machine Summary

Guest Operating System FreeBSD 64-bit

New Hard Disk Capacity 20 GB

Memory 256 MB

Networking Share with my Mac (NAT)

Device Summary CD/DVD, USB Controller, Sound Card

To change the default virtual machine settings, click Customize Settings. To run the virtual machine now, click Finish.

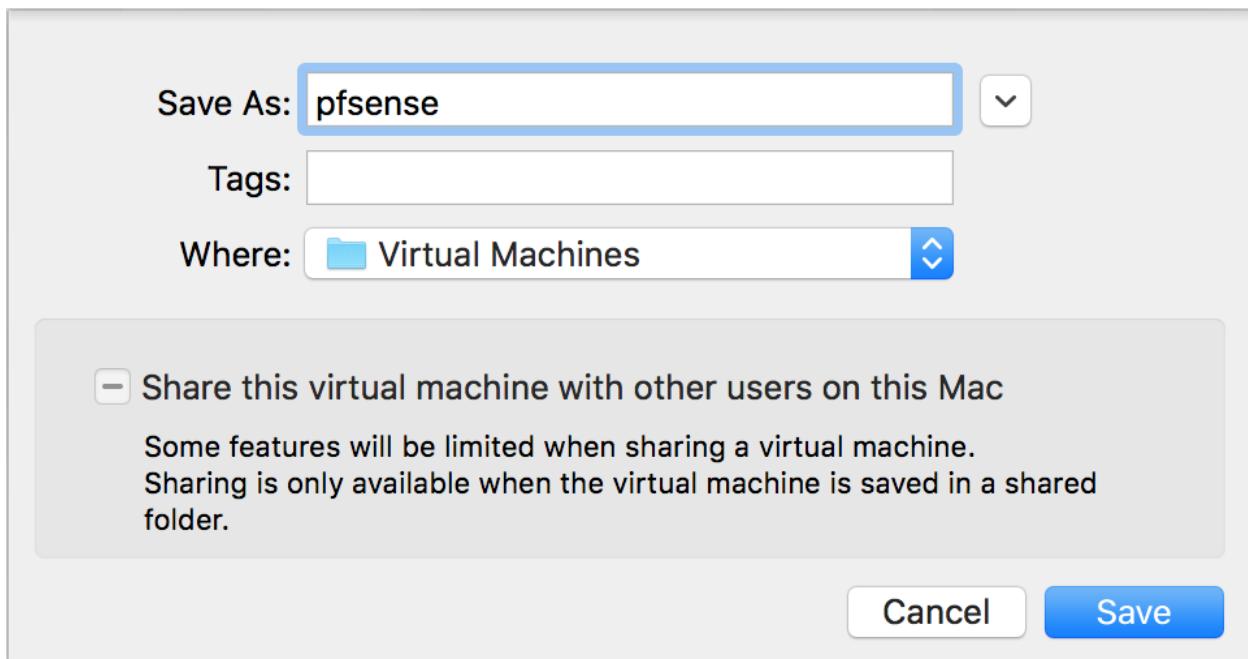
Customize Settings



Cancel

Go Back

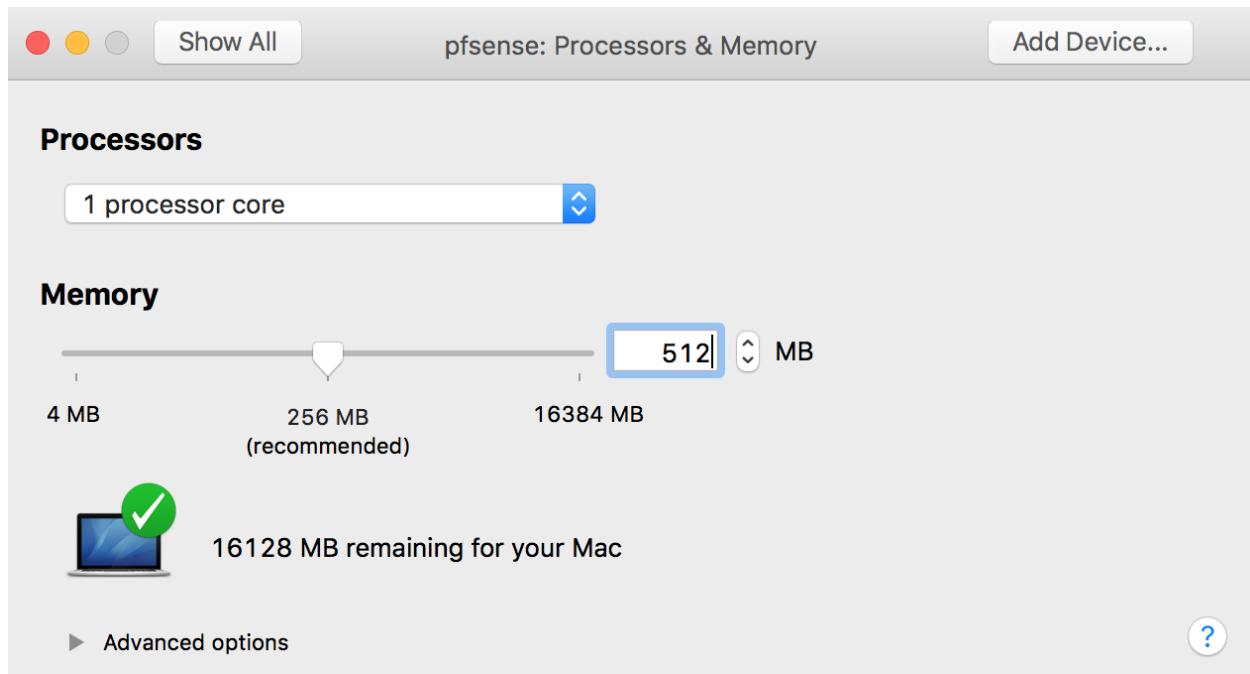
Finish



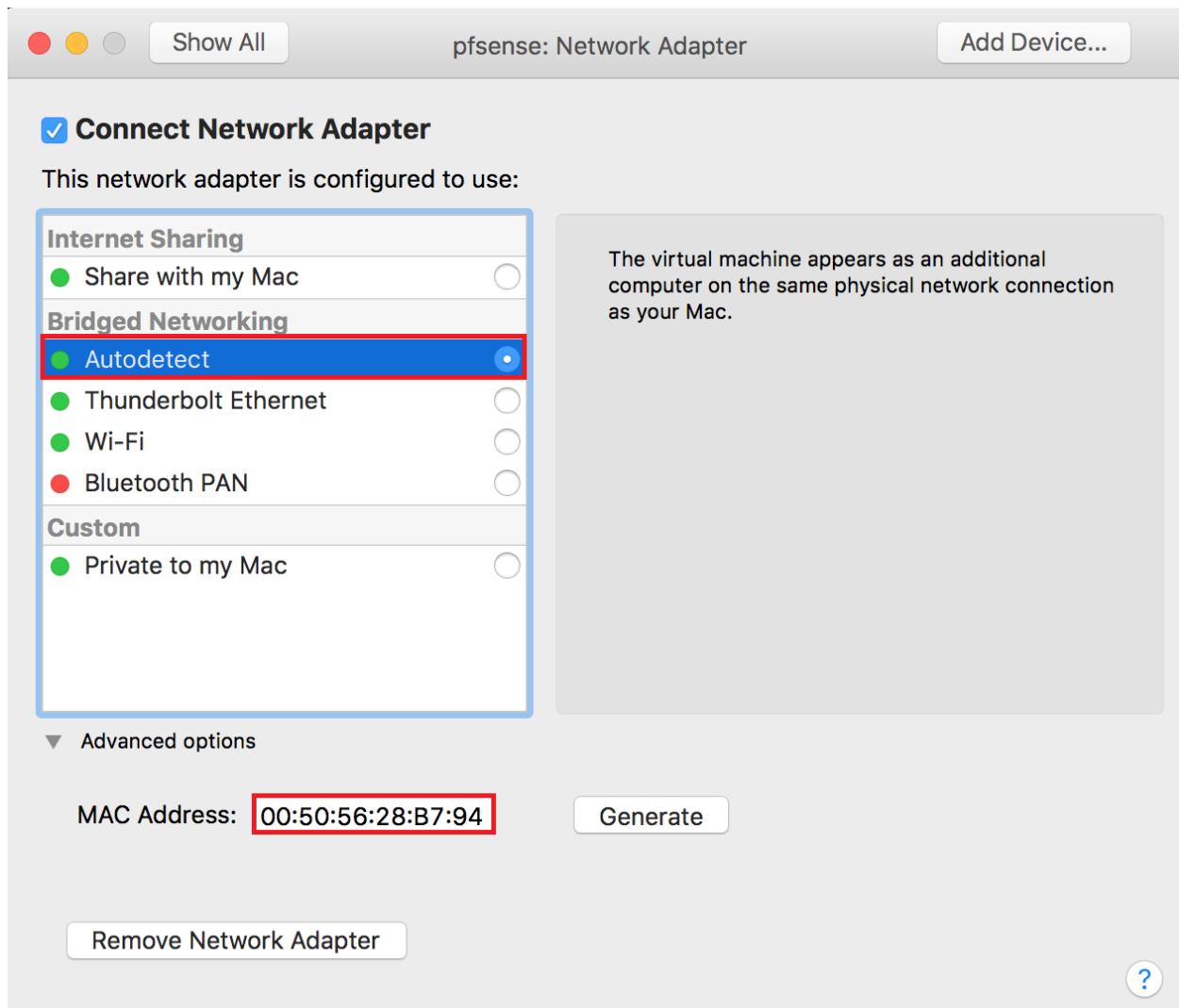
A window for pfSense appears on your desktop, and a smaller window entitled “pfSense: Settings” appears. First, click on “Processors & Memory”.



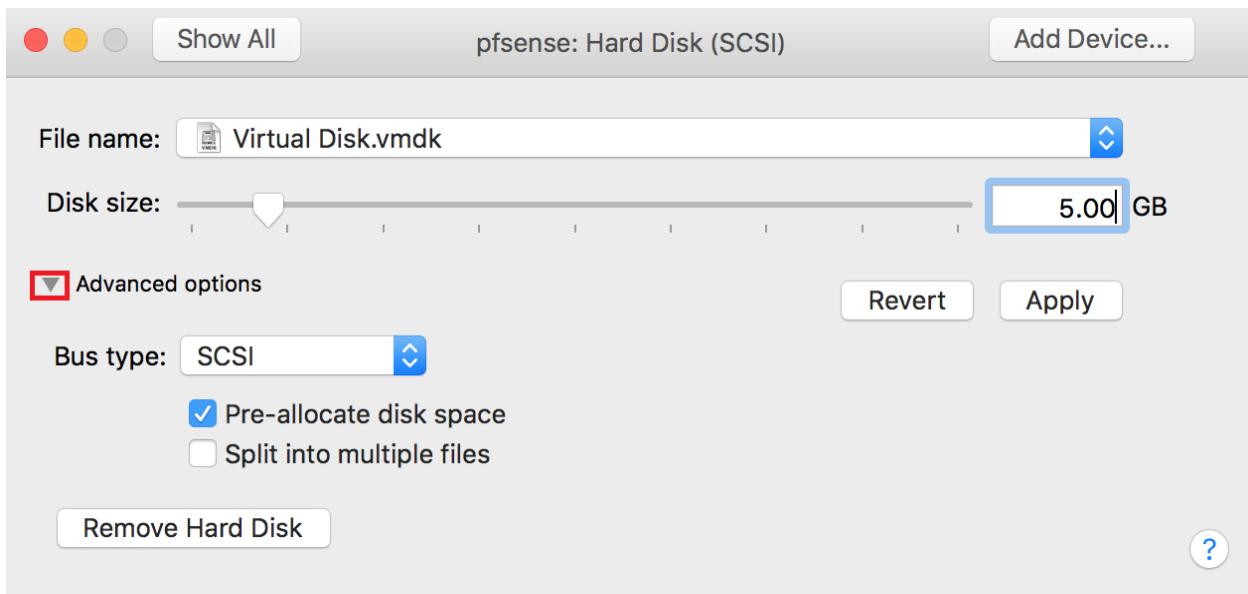
On this screen, you can adjust how many CPU cores and/or how much RAM is available to the VM. Change the input box to 512MB, then click the “Show All” button to proceed back to the main Settings screen.



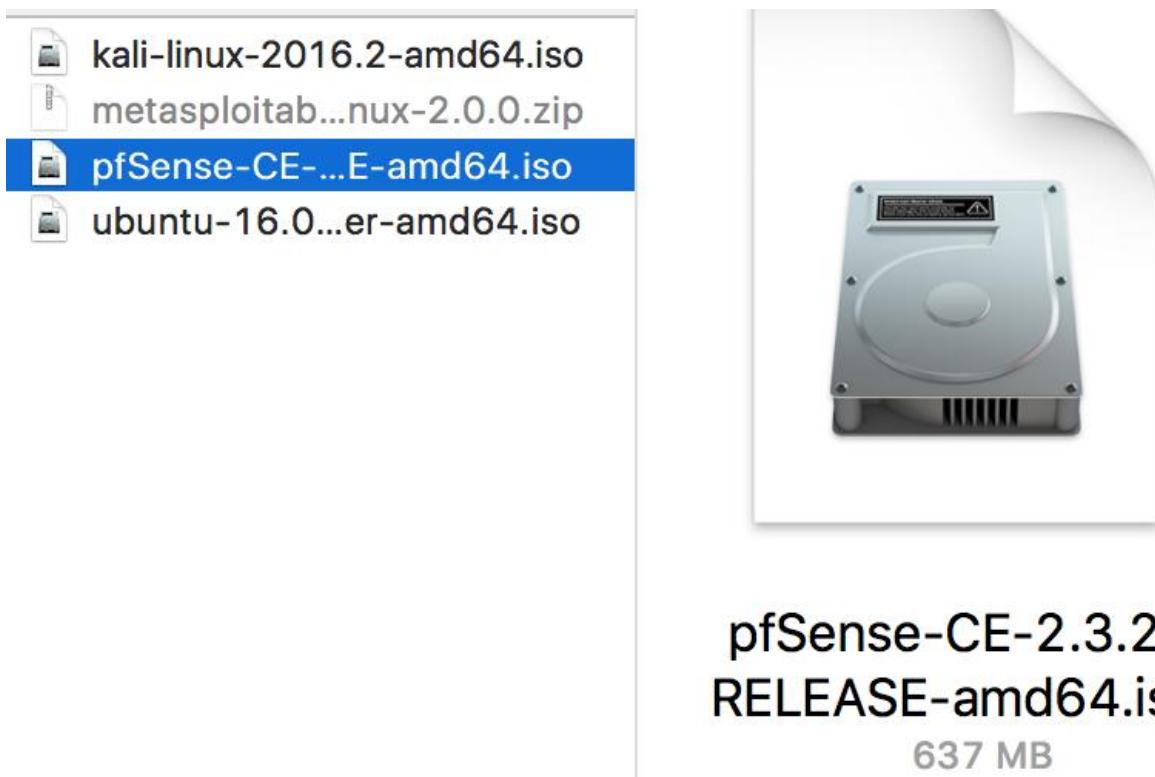
Back on the main screen, click on “Network Adapter” under the “Removable Devices” listing. On this screen, you can adjust what network this network adapter will connect to. Ensure that the “Connect Network Adapter” checkbox is checked, then change the radio button to “Autodetect” under “Bridged Networking”. Click on the triangle next to “Advanced options” under the networking pane. Click on the “Generate” button to the right of the “MAC Address:” field to generate a MAC address for this network adapter. **Record this MAC address, note that it is connected to the bridged network. We'll need this information later.** Click “Show All” to return to the main menu.

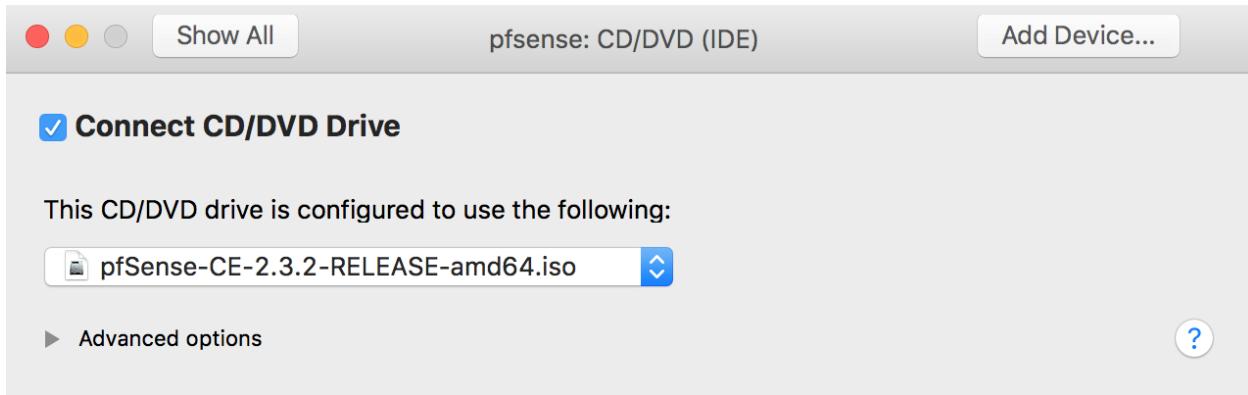


On the main screen under "Removable Devices", select "Hard Disk (SCSI)". This page allows you to modify disk size. In the input box, change the default setting to 5.00GB. Click on the triangle next to "Advanced options". Check the option labeled "Pre-allocate disk space", and uncheck the option "Split into multiple files". When you are finished, click "Show All" to go back to the main menu. When prompted to apply your changes, click Apply. Allocating the disk space may take a moment.

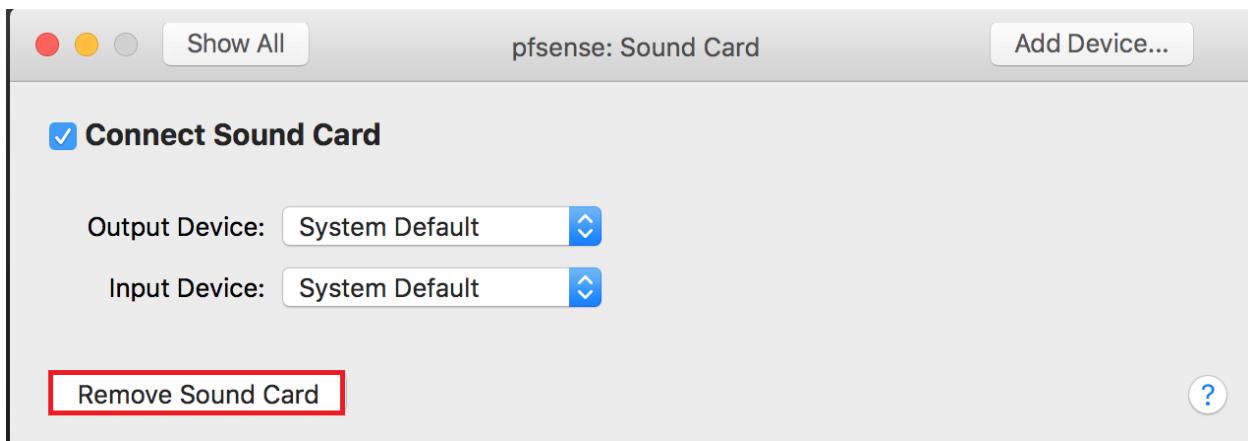


On the main menu, click “CD/DVD (IDE)”. Click the checkbox “Connect CD/DVD Drive”. Next, in the drop-down labeled “This CD/DVD drive is configured to use the following:” select the “Choose a disc or disc image...” option. This will open Finder. Navigate to where you stored the uncompressed/unzipped pfSense ISO. In my case, I placed the ISO in /Users/[username]/Documents/ISOs/. After you have selected the pfSense ISO, click “Show All” to return the main menu.

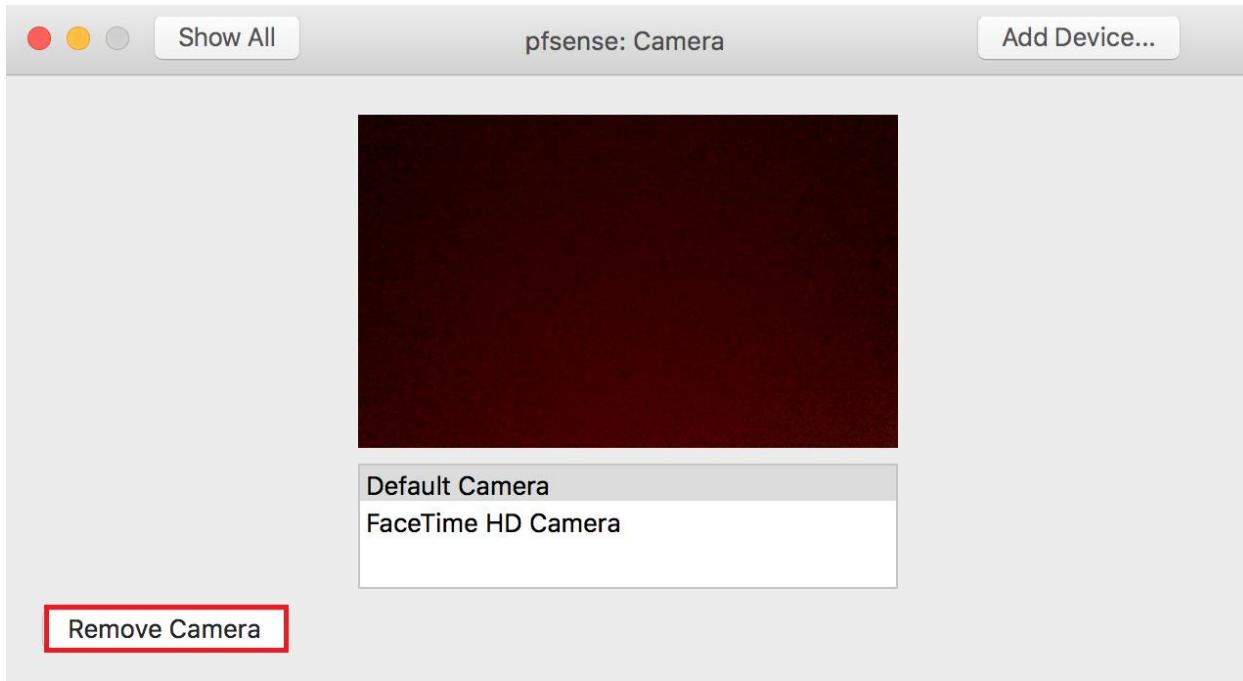




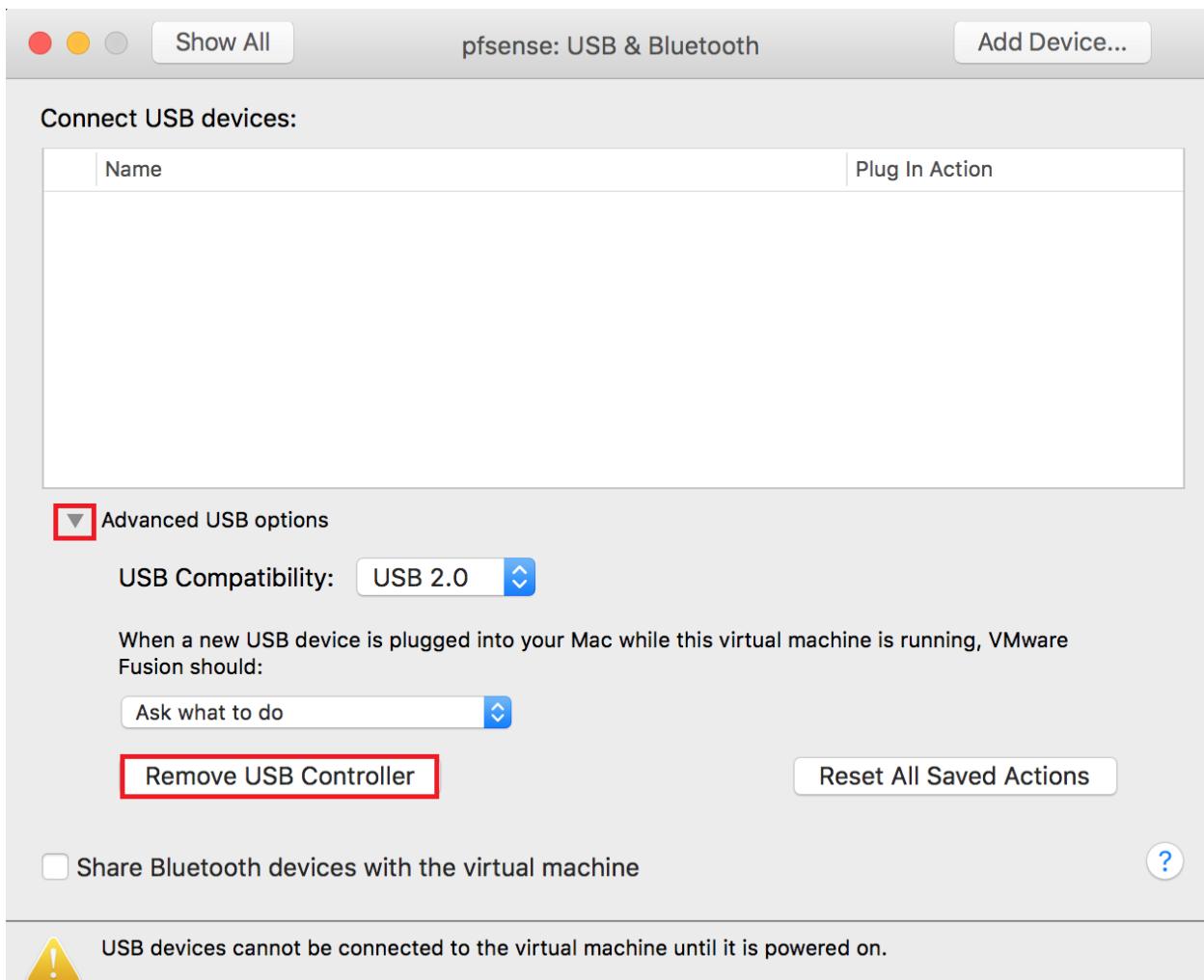
The next few sections are kind of the same, so I'm going to cover them together. I'm going to have you remove the Sound Card, USB & Bluetooth, and Camera options. First, from the main menu, click on "Sound Card". On this screen, simply click the "Remove Sound Card" button. Fusion will ask you to confirm this action; Click the "Remove" button. Doing so should bring you back to the main menu automatically.



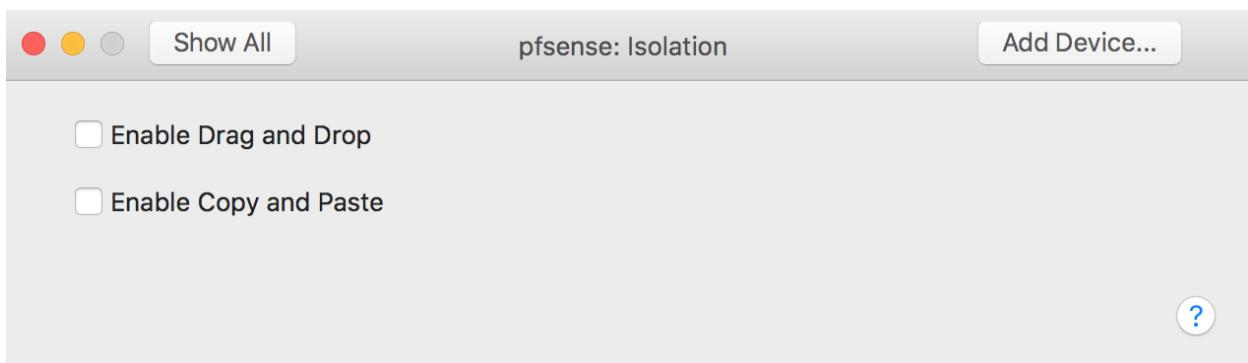
Repeat this process for the "Camera" option. Click on the icon from the main menu, click the "Remove Camera" button, and click "Remove" in the dialogue box. This should bring you back to the main menu again.



Finally, click on “USB & Bluetooth”. Fusion gets tricky and hides the “Remove USB Controller” button under “Advanced USB options”. Click the triangle next to it, to expose the button. Click “Remove USB Controller”, Click “Remove” in the dialogue box that pops up, then you’ll have to click “Show All” to be brought back to the main menu. The “USB & Bluetooth” option will still be there, just with no controller available for your virtual machine.



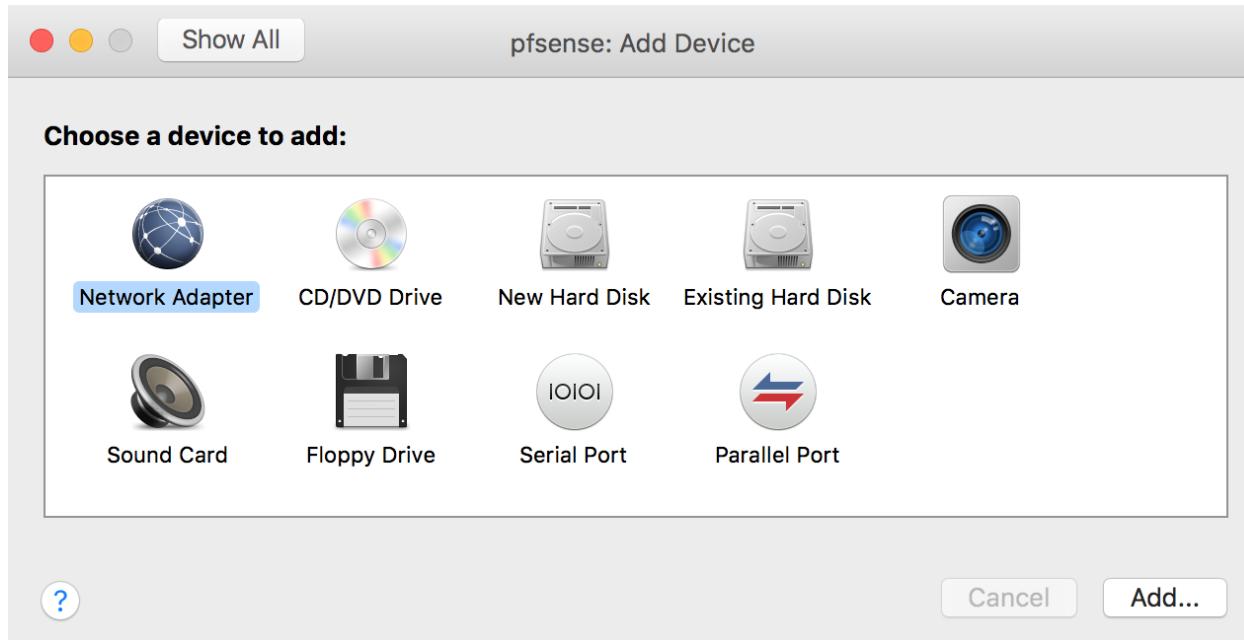
In the main menu, under the “Other” section, click on “Isolation”. These are settings that govern how you are allowed to interact with the VM from OSX, and vice-versa. Since we want to maintain separation as much as we can, Ensure that the “Enable Drag and Drop” and the “Enable Copy and Paste” checkboxes are unchecked. Afterwards, click “Show All” and return to the main menu.



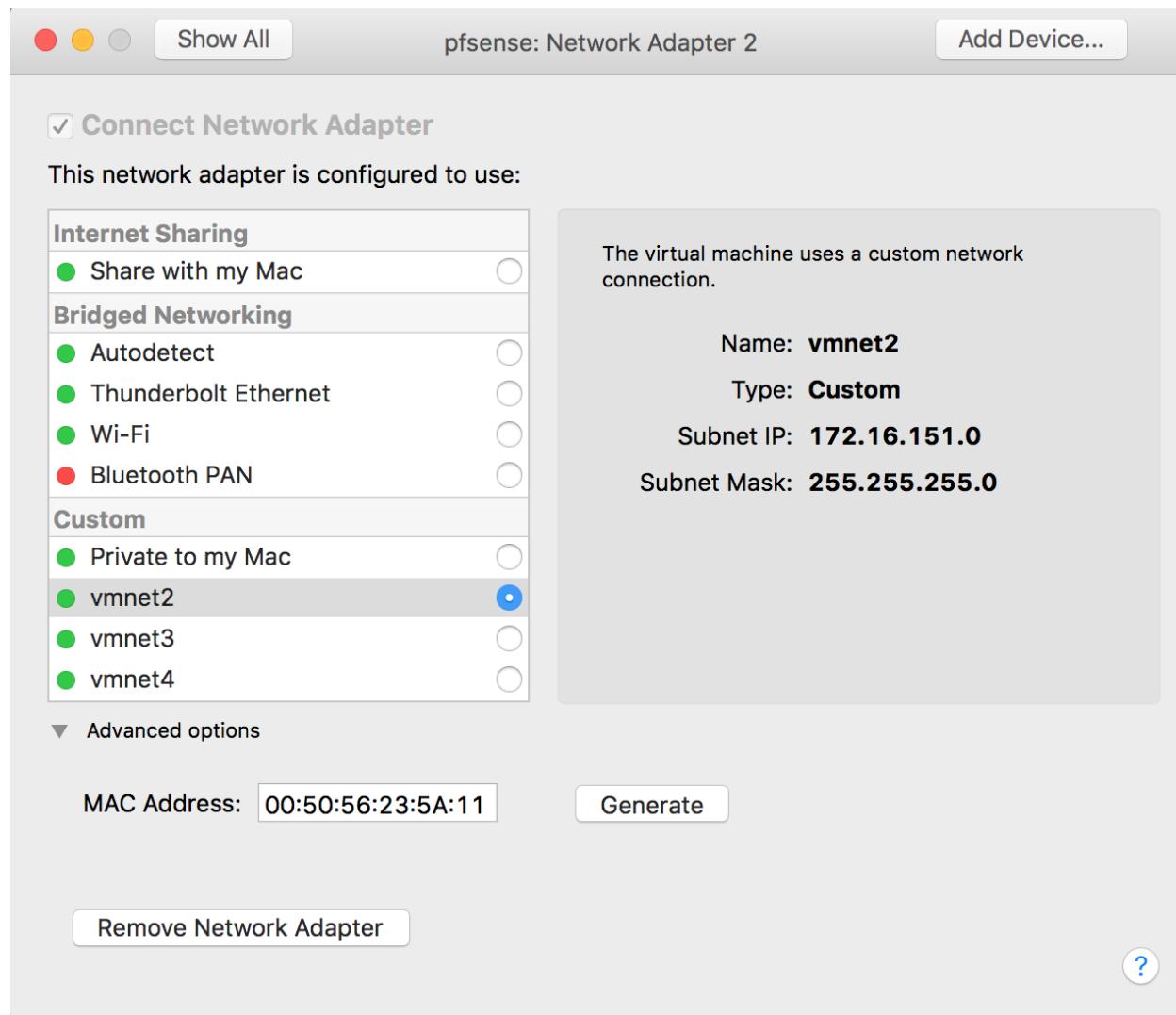
In the main menu, click the “Add Device...” button in the upper right corner.



The next screen asks you choose a device to add. Click on “Network Adapter” to highlight it, then click the “Add...” button. You’ll be brought to the configuration settings for “Network Adapter 2”.

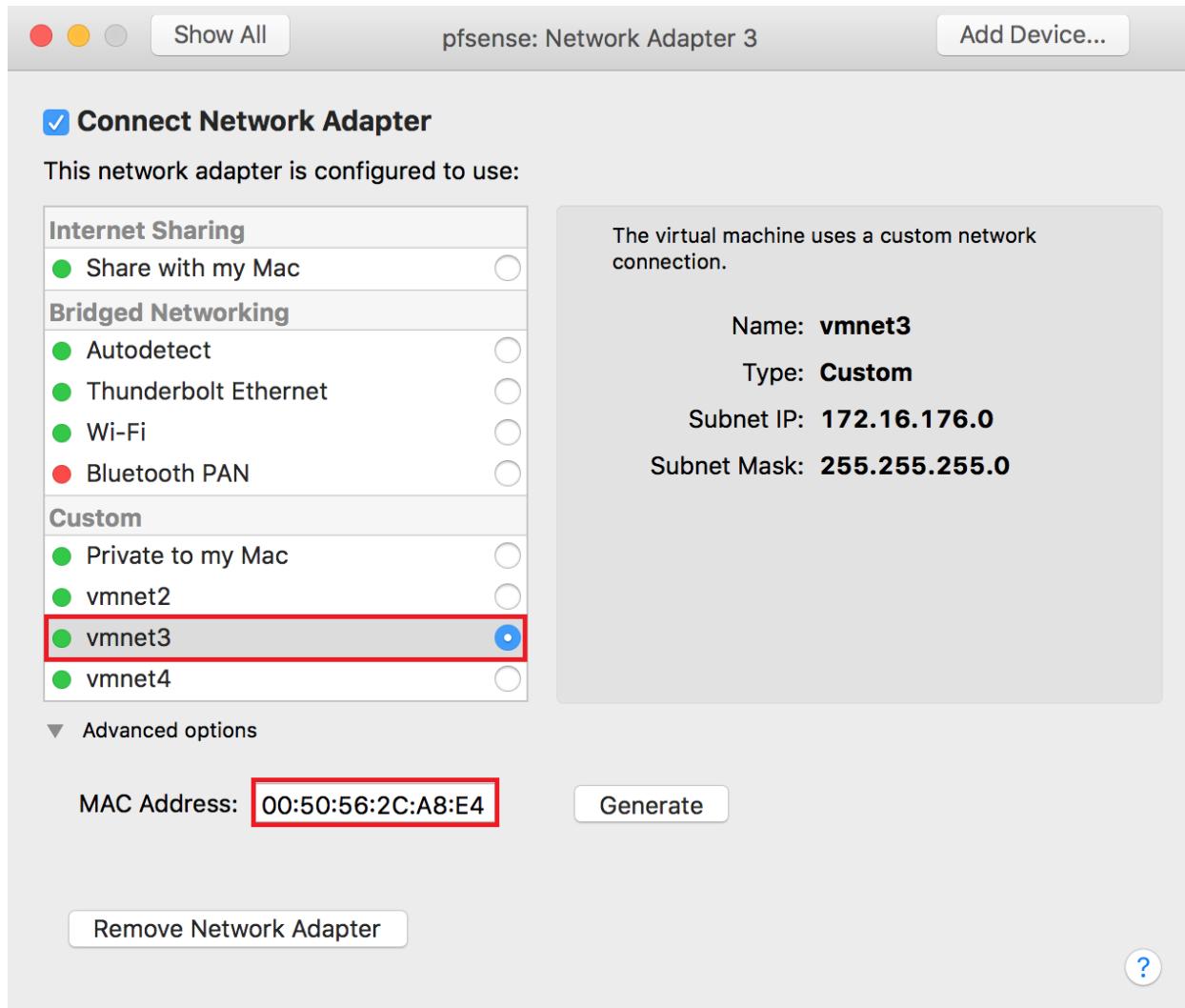


Ensure the “Connect Network Adapter” checkbox is checked. In the networking pane, select the “vmnet2” radio button under “Custom”. Click the triangle next to “Advanced options”, then click the “Generate” button. **Record this MAC address, and document that it is attached to the “Management” network. Additionally, do not worry about the “Subnet IP” and “Subnet Mask” settings that are displayed. These settings DO NOT apply, since we disabled Fusion’s DHCP server for all of our active virtual networks. This applies to the rest of the VMs you create; do not worry about the displayed IP address or subnet mask in the network adapter settings.** After you are done, click on the “Add Device...” button in the upper right corner again.



In the “Add Device” menu, click on “Network Adapter” again to highlight it, then click the “Add...” button. You should be in the settings page for “Network Adapter 3”. Perform the same steps you performed for Network Adapter 2. Connect this network adapter to “vmnet3”, **document the MAC address, and document that it is attached to the “IPS 1” network**. Click the “Show All”

button to return to the main menu one more time, then click the red circle in the upper left corner of the settings menu to exit.

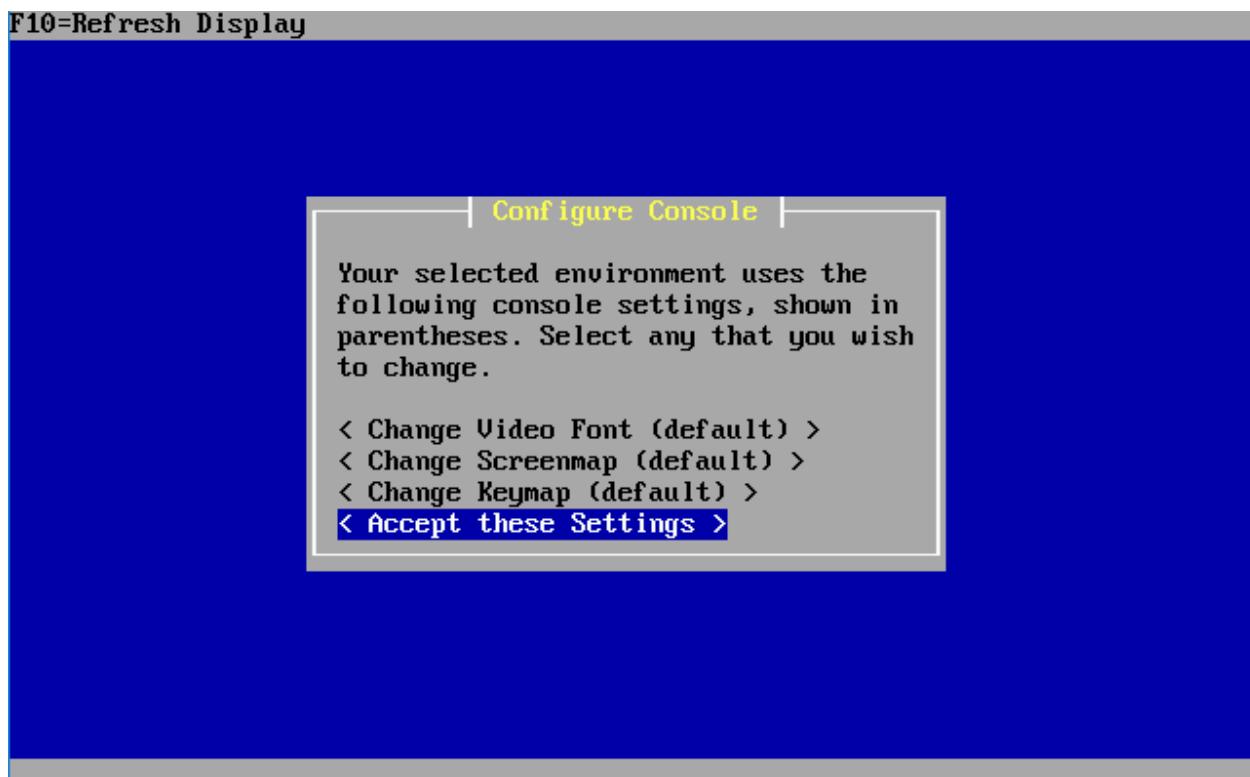


Installing pfSense

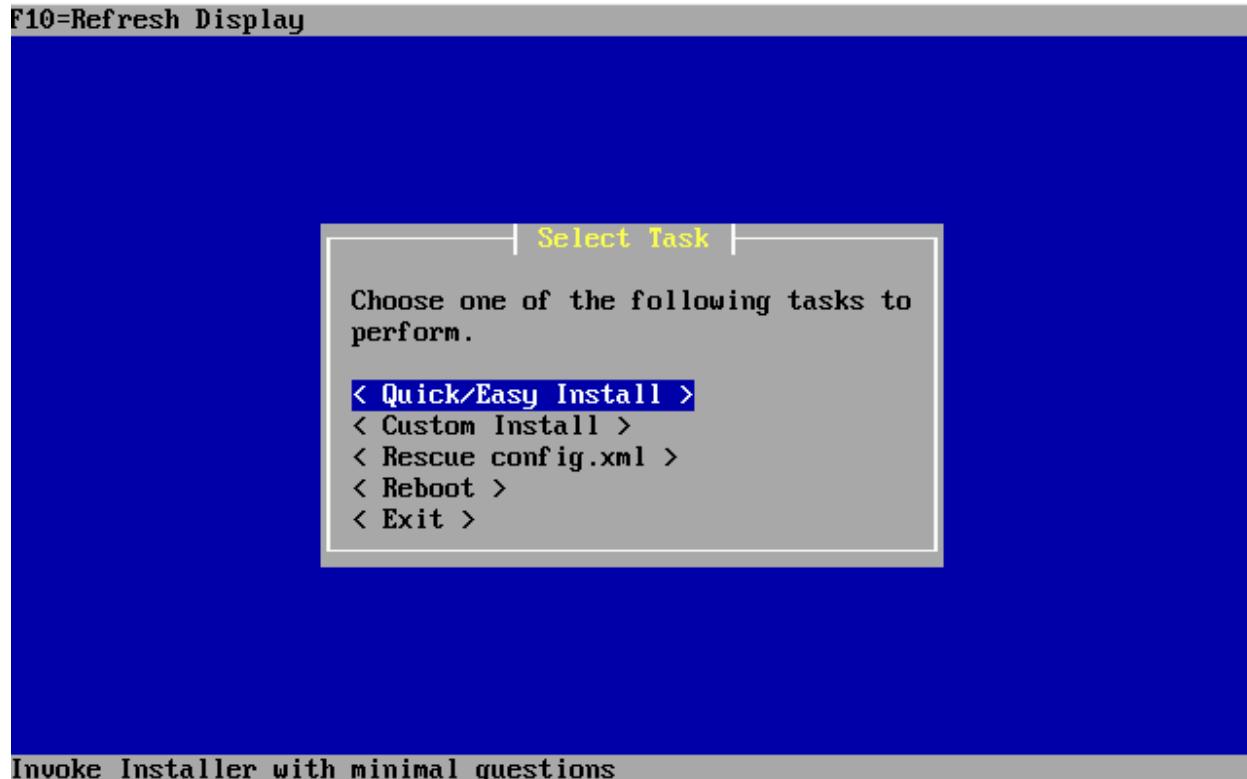
To start the pfSense VM, you can click the big play button on the window entitled “pfSense”, or on the menu bar, click “Virtual Machine”, then click “Start Up”. After doing so, the primary window will switch to view the console of the powered on VM. Clicking on this console will attach your mouse and keyboard to the VM. To detach again, press **ctrl+command**.



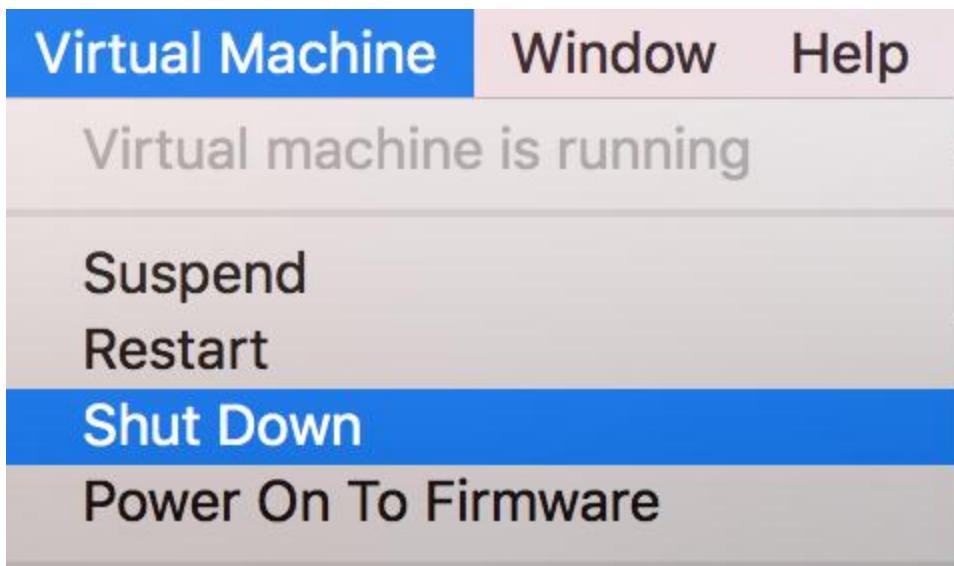
Let pfSense boot up, and the system should automatically run the installer. Adjust your video, screenmap, and keymap settings as necessary, then select “< Accept these Settings >”.



On the next screen, select “< Quick/Easy Install >” and let pfSense do all the heavy lifting. The next screen will inform you that the install will erase the contents of the hard disk. Since our virtual disk is already empty, this doesn’t matter in the least. Select OK. and let the install run.

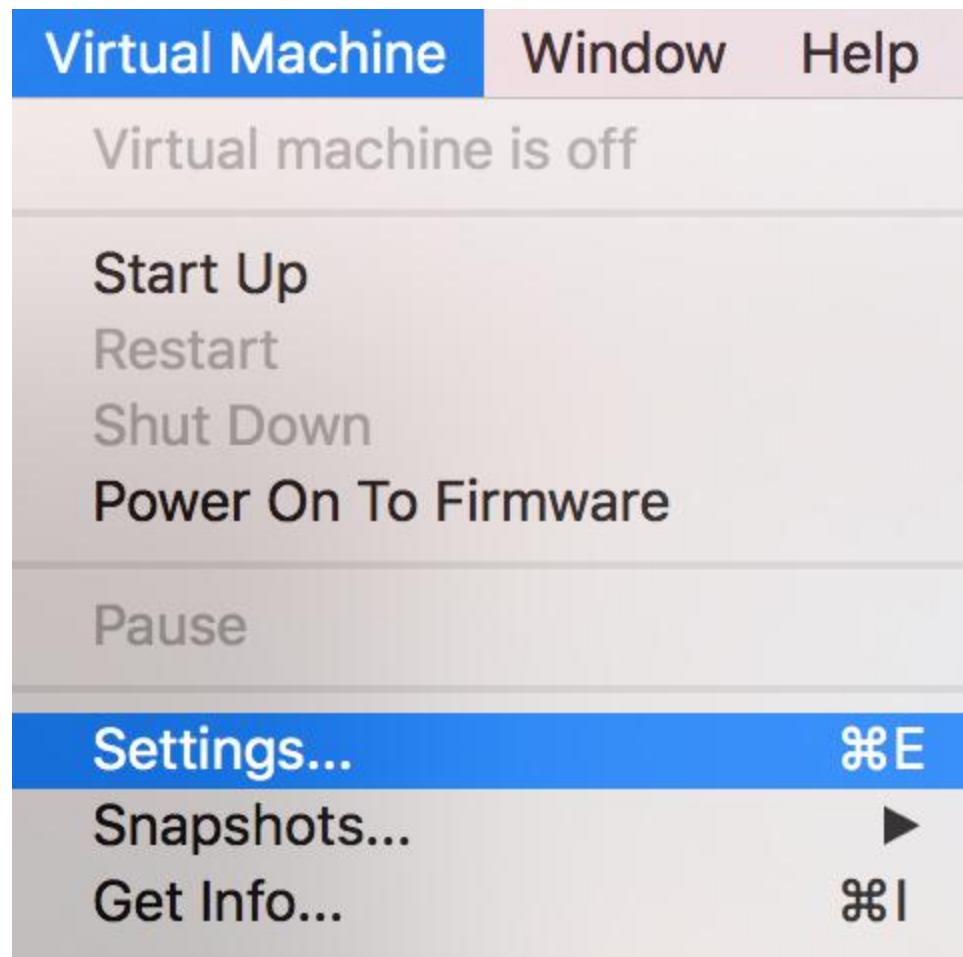


The installer will go through and install pfSense to the virtual hard disk. The installer will ask if you want to install a standard or embedded kernel. Make sure to select “< Standard Kernel >”. Finally, the installer informs you to reboot the machine to boot from the hard drive. The installation is done, however, we’re not going to reboot. Use **ctrl+command** to exit the VM window, then click on “Virtual Machine” in the menu bar, and select “Shut Down”



Final VM Settings

Click on “Virtual Machine” on the menu bar, then select “Settings...” to enter the pfSense: Settings menu.



From the main menu, click “CD/DVD (IDE)” to enter the configuration menu for the CD/DVD drive. Since pfSense is installed, we no longer need the drive anymore. Click the triangle next to “Advanced options”, then click on the “Remove CD/DVD Drive” button. Click on the “Remove” button in the dialogue box to confirm your choice. Afterwards, you should be returned the main menu. Click the red circle in the upper left corner to exit the settings menu.

Network Configuration

Power on the pfSense VM, and allow it to boot up. Eventually you’ll be greeted with the main pfSense menu on the console with 16 options. Select option 1, “Assign Interfaces” to run the interface assignment wizard. This wizard defines which interfaces (e.g. em0, em1, em2, etc.) corresponds to which WAN, LAN and OPT1, etc. networks for pfSense network services and firewall policies.

0) Logout (SSH only)	9) pfTop
1) Assign Interfaces	10) Filter Logs
2) Set interface(s) IP address	11) Restart webConfigurator
3) Reset webConfigurator password	12) PHP shell + pfSense tools
4) Reset to factory defaults	13) Update from console
5) Reboot system	14) Enable Secure Shell (sshd)
6) Halt system	15) Restore recent configuration
7) Ping host	16) Restart PHP-FPM

The WAN interface needs to be assigned to the bridged network adapter (the Bridged network). The LAN interface needs to be assigned to our Host-Only adapter (the Management network), and finally, OPT1 needs to be assigned to our internal network adapter (the IPS 1 network). In my case, em0 became the WAN interface, em1 became the LAN interface, and em2 became the opt2 interface. **Be sure to pay attention to the MAC addresses for em0, em1, and em2, and correlate that to the MAC addresses you documented earlier for network adapter, network adapter 2, and network adapter 3.** After you confirm that the settings are correct, you should automatically be returned to the pfSense main menu.

```
Enter the WAN interface name or 'a' for auto-detection
(em0 em1 em2 or a): em0

Enter the LAN interface name or 'a' for auto-detection
NOTE: this enables full Firewalling/NAT mode.
(em1 em2 a or nothing if finished): em1

Optional interface 1 description found: OPT1
Enter the Optional 1 interface name or 'a' for auto-detection
(em2 a or nothing if finished): em2

Enter the Optional 2 interface name or 'a' for auto-detection
( a or nothing if finished):

The interfaces will be assigned as follows:

WAN -> em0
LAN -> em1
OPT1 -> em2
```

Now that we have assigned the network interfaces, we have to configure IP addresses. In most cases, the “WAN” interface (bridged network) will automatically get an IP address from the device on your physical network that provides DHCP services. As for the LAN and OPT1 networks we will have to manually set the IP address, subnet mask, and DHCP scopes for the networks. Select option 2 in the pfSense main menu to get started.

0) Logout (SSH only)	9) pfTop
1) Assign Interfaces	10) Filter Logs
2) Set interface(s) IP address	11) Restart webConfigurator
3) Reset webConfigurator password	12) PHP shell + pfSense tools
4) Reset to factory defaults	13) Update from console
5) Reboot system	14) Enable Secure Shell (sshd)
6) Halt system	15) Restore recent configuration
7) Ping host	16) Restart PHP-FPM
8) Shell	

The configuration wizard will ask you which interface you want to configure. You'll have to configure the LAN and OPT1 interfaces individually (and if you're providing a STATIC IP address, netmask and gateway for the WAN, you'll have to configure the WAN interface as well). Here are the settings I used for my lab network:

LAN (lan)	-> em1	-> v4: 172.16.1.1/24
OPT1 (opt1)	-> em2	-> v4: 172.16.2.1/24

The wizard will ask you if you want to set up IPv6 for the interfaces and networks. Say no, because that's a can of worms we're not going to deal with here. The wizard will also ask want to use DHCP. For the LAN and OPT1 networks, I said yes. For the LAN network start address, I entered 172.16.1.10. For the end address, I entered 172.16.1.254. For the OPT1 network I chose 172.16.2.10 as the start, and 172.16.2.254 for the end address. If you are using 172.16.1.0/24 or 172.16.2.0/24 in your physical network, choose another network range to assign to the LAN and OPT1 interfaces to avoid network conflicts.

The reason we start at 1.10 and 2.10 is to make room for our other virtual machines that need to have a fixed IP address. The gap allows us to set STATIC DHCP allocations for VMs/systems whose addresses we want to remain fixed. We'll talk about this a little bit more later. For now, we should be done mucking around in the pfSense CLI. From here on out, we get to use the webUI to configure pfSense.

Web Configurator - Initial Setup

Before you are able to log in to the web interface for pfSense, you must configure the Host-Only adapter (aka the management network) on your host. Open up the terminal application for your mac and enter the command:

```
sudo ifconfig vmnet2 172.16.1.2 netmask 255.255.255.0
```

This command will require you to enter your password. This sets the vmnet2 adapter on your macbook to use the IP address 172.16.1.2 with the /24 subnet mask. Run the command "ifconfig vmnet2" to verify that the "inet" field reads 172.16.1.2, and the "broadcast" field reads 172.16.1.255. **Please be aware that using ifconfig to set a static IP address does not persist across system reboots or even VMware Fusion being shut down.** Every time vmware fusion is shut down (e.g. command + q) or the system is shut down/rebooted, the

vmnet interfaces are deleted and recreated with their default values. **If VMware Fusion is restarted, or the system is rebooted/restarted, simply re-run the ifconfig command above to give your hypervisor host direct access to the “Management” virtual network (vmnet 2).**

```
new-host-4:Downloads trobinson$ sudo ifconfig vmnet2 172.16.1.2 netmask 255.255.255.0
Password:
new-host-4:Downloads trobinson$ ifconfig vmnet2
vmnet2: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
    ether 00:50:56:c0:00:02
        inet 172.16.1.2 netmask 0xffffffff broadcast 172.16.1.255
new-host-4:Downloads trobinson$
```

On the host OS open your favorite web browser (I prefer Firefox - <https://www.mozilla.org/en-US/firefox/new/>) and navigate to <https://172.16.1.1> (or the IP address you assigned to the LAN interface of your pfSense system). The default credentials for access to the web interface are admin/pfsense. If this is your first time logging in, pfSense takes you through a nice little setup wizard. I'll highlight some of the important things to take note of, and/or change as necessary:

Set the primary and secondary DNS servers you plan on using. I typically use 8.8.8.8 (google public DNS) and 4.2.2.2 (Level 3 public DNS).

Primary DNS Server 8.8.8.8

Secondary DNS Server 4.2.2.2

Uncheck the “Block private networks from entering via WAN rule. It doesn’t matter. We won’t be letting anything in via the WAN interface.



The other default settings on the first time setup wizard should be fine. Please note that they will make you change the password for the admin account as a part of the setup wizard. Document the password, keep it somewhere safe.

Our next order of business is to restrict access to the web interface to where only 172.16.1.2 can administer the firewall. On the menu bar at the top of the page, select Firewall > Rules. Click on LAN to modify the firewall policy for the LAN interface. Click the “Add” button with the arrow facing up. This adds the firewall rule to the top of the firewall policy to where it is evaluated first. The “Edit Firewall Rule” page is fairly straightforward. I’ve highlighted the options to be aware of.

Edit Firewall Rule

Action	Pass			
Choose what to do with packets that match the criteria specified below. Hint: the difference between block and reject is that with reject, a packet (TCP RST or ICMP port unreachable for UDP) is returned to the sender, whereas with block the packet is dropped silently. In either case, the original packet is discarded.				
Disabled	<input type="checkbox"/> Disable this rule Set this option to disable this rule without removing it from the list.			
Interface	LAN			
Choose the interface from which packets must come to match this rule.				
Address Family	IPv4			
Select the Internet Protocol version this rule applies to.				
Protocol	TCP			
Choose which IP protocol this rule should match.				
Source				
Source	<input type="checkbox"/> Invert match.	Single host or alias	172.16.1.2	/
Display Advanced		Display Advanced		
Destination				
Destination	<input type="checkbox"/> Invert match.	Single host or alias	172.16.1.1	/
Destination port range	HTTPS (443)	From	Custom	To
Specify the destination port or port range for this rule. The "To" field may be left empty if only filtering a single port.				
Extra Options				
Log	<input type="checkbox"/> Log packets that are handled by this rule Hint: the firewall has limited local log space. Don't turn on logging for everything. If doing a lot of logging, consider using a remote syslog server (see Status: System Logs: Settings page).			
Description	pfsense strict anti-lockout			

When you are done, click the Save icon at the bottom of the page. This will take you back to the previous page. A yellow dialogue box with a green button called “Apply Changes” will appear. Click this button to apply this new firewall rule. Next, we want to disable the default anti-lockout rule. Next, lets navigate to Firewall > Aliases. On the firewall Aliases page, click on “IP”, then click Add. Create an alias with the following settings:

Firewall / Aliases / Edit

Properties

Name	RFC1918	The name of the alias may only consist of the characters "a-z, A-Z, 0-9 and _".
Description	An alias for all RFC1918 networks	A description may be entered here for administrative reference (not parsed).
Type	Network(s)	

Network(s)

Hint: Networks are specified in CIDR format. Select the CIDR mask that pertains to each entry. /32 specifies a single IPv4 host, /128 specifies a single IPv6 host, /24 specifies 255.255.255.0, /64 specifies a normal IPv6 network, etc. Hostnames (FQDNs) may also be specified, using a /32 mask for IPv4 or /128 for IPv6. An IP range such as 192.168.1.1-192.168.1.254 may also be entered and a list of CIDR networks will be derived to fill the range.			
Network or FQDN	10.0.0.0 / 8	10.x.x.x RFC 1918 networks	
	172.16.0.0 / 12	172.16.x.x RFC 1918 networks	
	192.168.0.0 / 16	192.168.x.x RFC 1918 networks	

Save Add Network

Click Save to be brought back to the previous page, then click Apply Changes. We just created an alias for RFC1918 networks (local networks that are not routable through the public internet). This will come in handy later for creating firewall rules around these networks. Next, navigate to System > Advanced. Click to fill in the checkbox next to the option “Disable webConfigurator anti-lockout rule”, and click save on the bottom of the page.

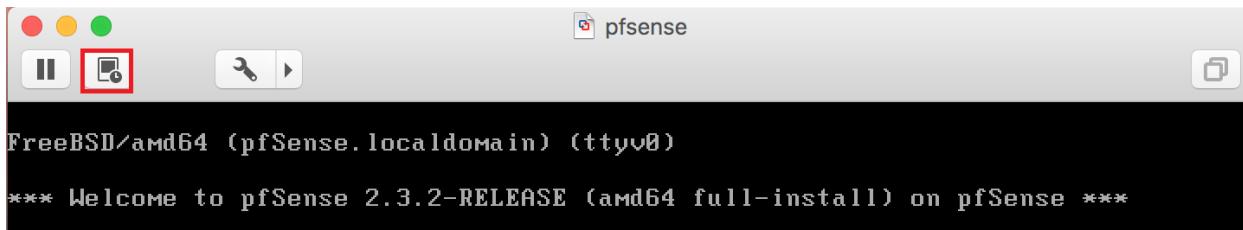
Anti-lockout Disable webConfigurator anti-lockout rule

When this is unchecked, access to the webConfigurator on the LAN interface is always permitted, regardless of the user-defined firewall rule set. Check this box to disable this automatically added rule, so access to the webConfigurator is controlled by the user-defined firewall rules (ensure a firewall rule is in place that allows access, to avoid being locked out!) Hint: the “Set interface(s) IP address” option in the console menu resets this setting as well.

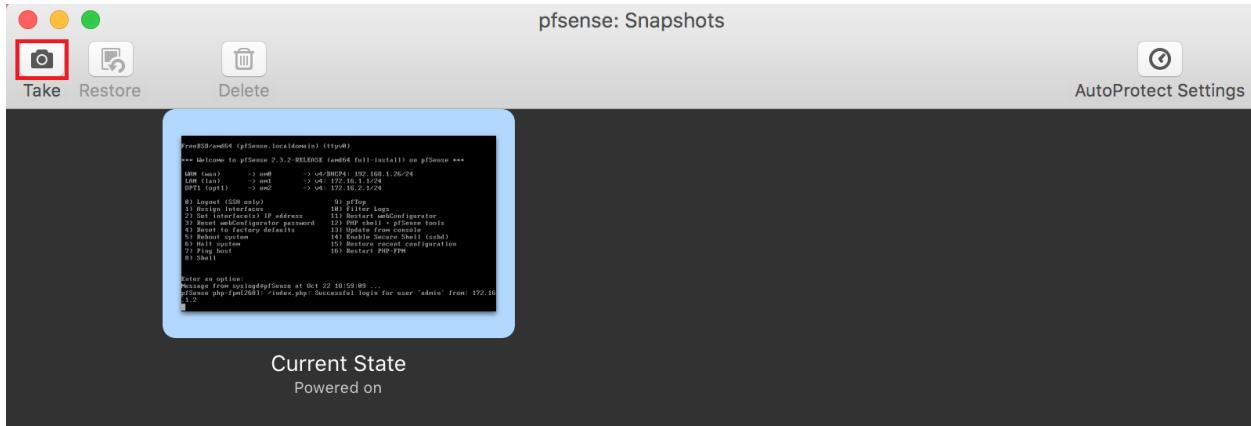
Take a Snapshot

Snapshotting is a VERY important concept with VMs. Snapshots let you restore your virtual machines to a point in time in the past when you took that snapshot. This lets you undo misconfigurations, problems, or potential issues that affect the VM relatively easily. We’re going to take a snapshot of pfSense in its current state.

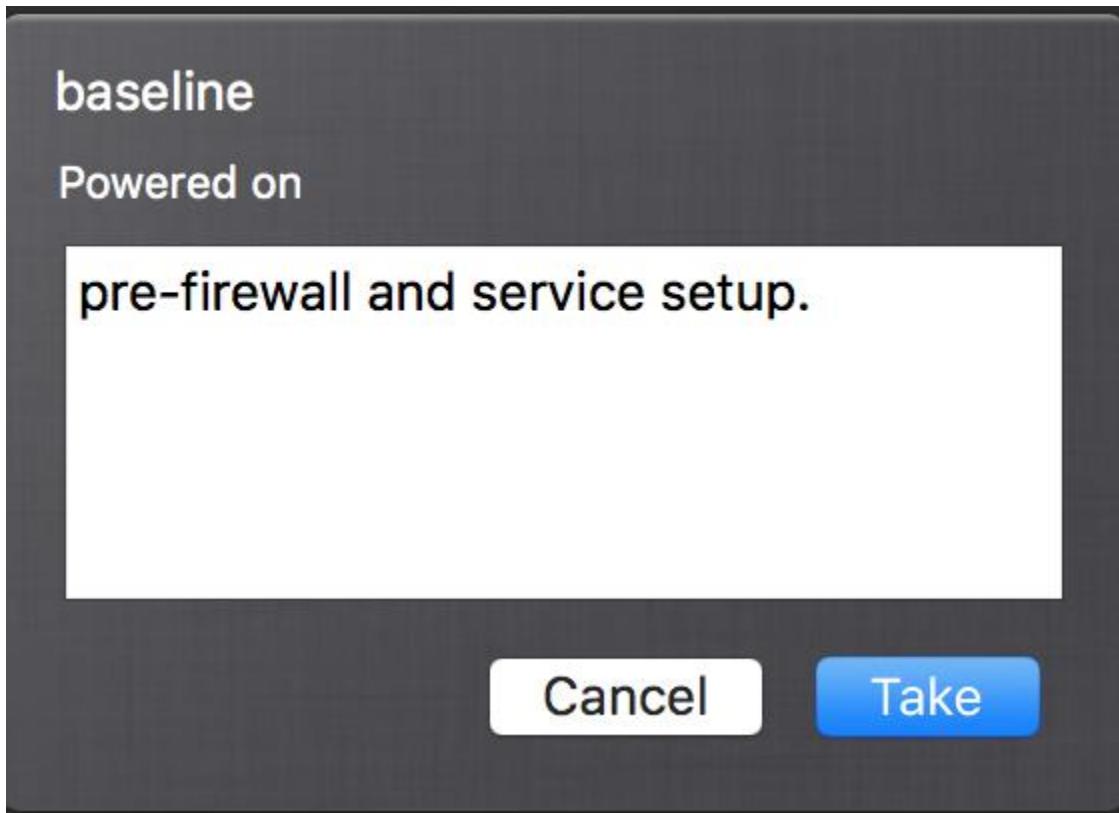
In the pfSense window, click on the icon to the right of the “pause” button. This will open the snapshot manager window.



Once in the snapshot manager, simple click the icon that looks like a camera labeled “Take”.

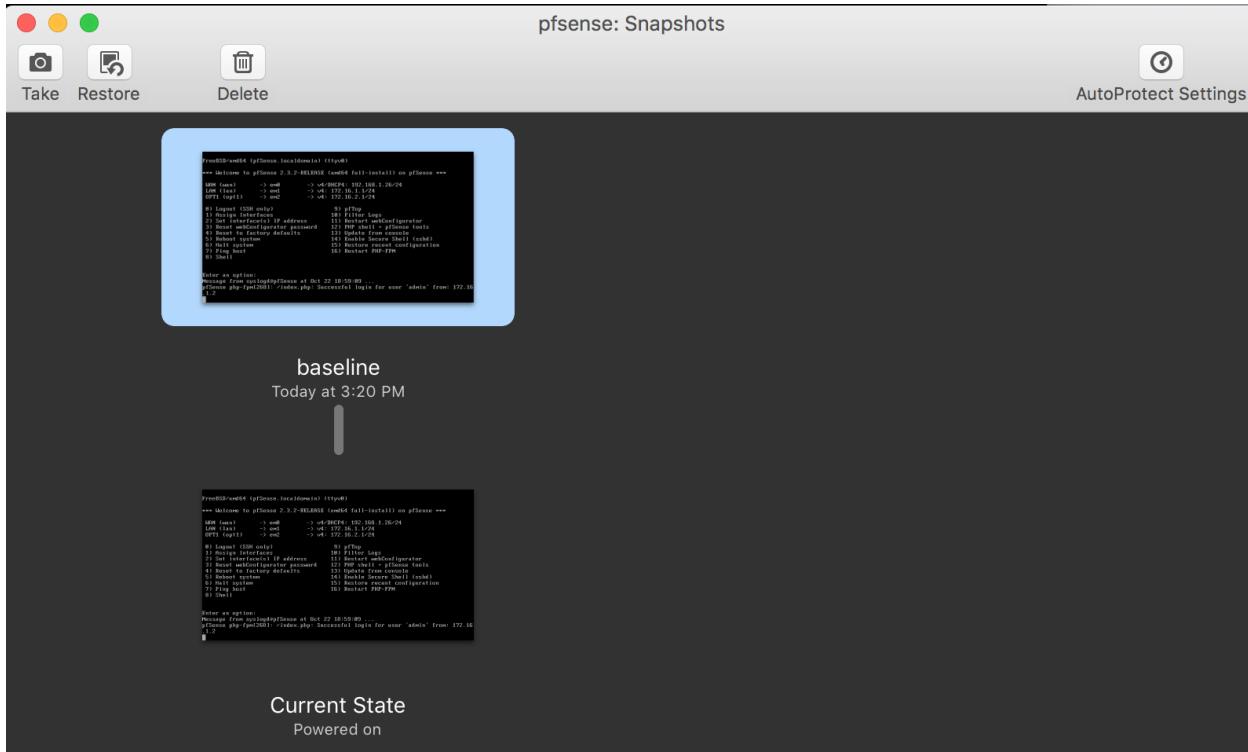


A window pops up asking you to name the snapshot, and add notes. I named mine “baseline” with the notes “pre-firewall and service setup”. After you are done, click the “Take” button for VMware Fusion to generate a snapshot of the VM.



This snapshot will appear in the snapshot manager. You can click on it, and a small “i” will pop up if you hover over it. If you click the “i”, then the name and snapshot notes will appear. If you ever need to restore to a previous snapshot, you can click to highlight the snapshot, then click

“Restore” on the top of the snapshot manager window. When you are finished, click the red circle in the upper left corner of the window.



pfSense Summary

At the end of all this, you should have a pfSense VM with the following settings:

- 512MB of RAM
- 5GB of Disk
- 1 Virtual CPU
- DVD drive disabled
- Audio disabled
- USB controller disabled
- 3 Network interfaces:
 - 1 Bridged (Bridge vswitch/WAN interface)
 - 1 Host-Only (Management vswitch/LAN interface)
 - 1 Internal Network (IPS vswitch1/OPT1 interface)

pfSense should be configured as followed:

- The WAN interface should be the VirtualBox network adapter connected to the Bridged Adapter. This interface should automatically get an IP address via DHCP. If this isn't happening, get some help with whoever administers your local network, or start troubleshooting.
- The LAN interface should have an IP address of 172.16.1.1.

- The LAN network should be 172.16.1.0 with a /24 (255.255.255.0) subnet mask
- DHCP should be configured for this network with a scope of 172.16.1.10-172.16.1.254
 - 172.16.1.3-172.16.1.9 are available for setting static DHCP mappings
- The OPT1 interface should have an IP address of 172.16.2.1
- The OPT1 network should be 172.16.2.0 with a /24 (255.255.255.0) subnet mask
- DHCP should be configured for this network with a scope of 172.16.2.10-172.16.2.254
 - 172.16.2.2-172.16.2.9 are available for static DHCP mappings.
- 172.16.1.2 should have a firewall rule configured to allow it access to the firewall over HTTPS; This is the anti-lockout firewall rule.
- You should have an alias for RFC1918 networks configured for use with the firewall policy.
- You should have at least one good snapshot you can revert to if you need to redo a step and somehow got lost/confused.

What's Next?

Now that you have an operational pfSense VM, your next steps are to read "[Network Configuration - Segmentation](#)" guide, followed immediately by "[Network Configuration - Core Network Services](#)". Make sure the firewall policies match those illustrated in the segmentation guide, and make sure that the pfSense VM is hosting all the services mentioned in the core network services guide to make your firewall fully functional and ready to handle your lab network. Once you verify that your firewall policies and services are properly configured, **SNAPSHOT THE VM THEN** you can move on, and try your hand at setting up the remaining VMs.

Your Turn

The pfSense vm required a lot of custom configuration, both in VirtualBox, as well as in the VM itself once the OS was installed. Now that we did that together, I'm going to have you create the rest of the virtual machines yourself -- with the exception of the metasploitable 2 VM since its a special case for reasons you will see momentarily. Below are a set of spec sheets to help you create the remaining virtual machines for the lab on your own. Think of it as a checklist you should be able to run through on your own.

All of the linux-based VMs in our lab are based on Ubuntu Server (Except for Kali Linux, which is based on the Debian Linux distro, which was the precursor to Ubuntu) so set up and configuration on our lab VMs should be mostly the same.

Kali Linux VM

Kali Linux is a popular penetration testing distro. Popular because hackers and script kiddies are lazy, and practically everything you need for performing a penetration test or joining anonymous is installed by default. Kali is going to be our loud, obnoxious attacker that we're

going to use as a noise generator for the express purpose of generating events on our IPS VM. I want you to perform the following tasks:

- Download Kali Linux 64-bit here: <https://www.kali.org/downloads/>
- Create a new, custom virtual machine with the following settings:
 - Configure the VM to install from the Kali Linux ISO you downloaded
 - Set the Guest OS family to “Linux”
 - Set the Guest OS version to “Debian 8.x 64-bit”
 - Create a new virtual disk
 - Name the VM “kali”
 - Note that after you save the VM and grant it a name, VMware Fusion immediate assumes you want to turn the Virtual Machine off. In the main menu, click “Virtual Machine”, then “Shut Down”; In order to adjust the hardware settings below
- Customize the following hardware settings (Virtual Machine > Settings or you can click the “Customize Settings” button on the finish page of the new VM wizard):
 - Under CD/DVD (IDE), choose the “Choose a disc or disc image” and find the Kali Linux ISO you just downloaded
 - Ensure that the “Connect CD/DVD Drive” option is checked
 - Allocate 4GB (4096MB) of RAM
 - Modify the Hard Disk (SCSI) Settings, and change the disk size to 80.00GB
 - Uncheck the advanced option “Split into multiple files”
 - Check the advanced option “Pre-allocate disk space”
 - Allocate 1-2 processors, where total number of cores is no more than 2
 - Connect the network adapter to VMnet3
 - Ensure that “Connect at power on” is checked
 - Under advanced settings, click the “Generate” button to generate a new MAC address. **Record this MAC address as well as the VM it corresponds to.**
 - Remove the USB Controller
 - Remove the Sound Card
 - Remove the Printer
 - Remove the Camera
 - In the Isolation settings, uncheck the “Enable Drag and Drop” and “Enable Copy and Paste”
- Install Kali Linux
- After the install is completed, power off the VM and adjust the following settings:
 - Remove the virtual CD/DVD drive
- Log in to the pfSense web UI, navigate to Services > DHCP Server, and add a static mapping to the OPT1 interface DHCP, for the MAC address of adapter 1 for the Kali VM; assign it the IP address 172.16.2.2, making sure to enter a description that tells you which VM this static mapping belongs to
- Power on the Virtual Machine and do the following:
 - Verify that the static mapping for the VM’s IP address worked; Open a terminal/shell and run ifconfig -a | less and verify that “eth0” or whatever the

network adapter's name is in the VM was given the IP address you configured its static mapping for in pfSense

- If the VM does not have an ip address, run the “dhclient” command to have the VM request an IP address from the DHCP server.
- Verify the virtual machine can reach the internet.
 - In a terminal/shell run the command ping -c 5 www.google.com
 - This will confirm DNS can resolve domain names, and that the VM can reach the internet
- Update/patch the VM
 - In a terminal/shell, run the following, and depending on your network connection, etc., be prepared to wait:
 - **export DEBIAN_FRONTEND=noninteractive; apt-get -q update; apt-get -y -q dist-upgrade**
- Create a snapshot for the VM

Siem VM

SIEM is shorthand for Security Intrusion Events Manager. This is fancy security nomenclature for “log aggregator”; we will be having our IPS VM log its events here. We’re going to be running Splunk on this VM. Splunk is a commercial program for managing logs on a pretty large scale. By default, and with no licensing, Splunk only allows you to collect or “index” 500MB worth of logs per day however, there are /ways/ around this that we’ll talk about later. I want you perform the following tasks to set up the SIEM VM:

- Download Ubuntu Server 16.04.1 64-bit here: <https://www.ubuntu.com/download/server>
- Create a new, custom virtual machine with the following settings:
 - Configure the VM to install from the Ubuntu Linux ISO you downloaded
 - Set the Guest OS family to “Linux”
 - Set the Guest OS version to “Ubuntu 64-bit”
 - Create a new virtual disk
 - Name the VM “siem”
 - Note that after you save the VM and grant it a name, VMware Fusion immediately assumes you want to turn the Virtual Machine on. In the main menu, click “Virtual Machine”, then “Shut Down”; In order to adjust the hardware settings below.
 - Customize the following hardware settings (Virtual Machine > Settings or you can click the “Customize Settings” button on the finish page of the new VM wizard):
 - Under CD/DVD (IDE), choose the “Choose a disc or disc image” and find the Ubuntu 16.04.1 64-bit Linux ISO you just downloaded
 - Ensure that the “Connect CD/DVD Drive” option is checked
 - Allocate 4GB (4096MB) of RAM
 - Modify the Hard Disk (SCSI) Settings, and change the disk size to 80.00GB
 - Uncheck the advanced option “Split into multiple files”
 - Check the advanced option “Pre-allocate disk space”

- Allocate 1-2 processors, where total number of cores is no more than 2
 - Connect the network adapter to VMnet2
 - Ensure that “Connect at power on” is checked
 - Under advanced settings, click the “Generate” button to generate a new MAC address. **Record this MAC address as well as the VM it corresponds to.**
 - Remove the USB Controller
 - Remove the Sound Card
 - Remove the Printer
 - Remove the Camera
 - In the Isolation settings, uncheck the “Enable Drag and Drop” and “Enable Copy and Paste”
- Install Ubuntu Server
 - Be sure to install the command line utilities and SSH Server role when asked. You should not need to install any other roles or features for this server
- After the install is completed, power off the VM and adjust the following settings:
 - Remove the virtual CD/DVD drive
- Log in to the pfSense web UI, navigate to Services > DHCP Server, and add a static mapping to the LAN interface DHCP, for the MAC address of adapter 1 for the SIEM VM; assign it the IP address 172.16.1.3, making sure to enter a description that tells you which VM this static mapping belongs to!
- Power on the Virtual Machine and do the following:
 - Verify that the static mapping for the VM’s IP address worked; Open a terminal/shell and run ifconfig -a | less and verify that “eth0” or whatever the network adapter’s name is in the VM was given the IP address you configured its static mapping for in pfSense.
 - If the VM does not have an ip address, run the “dhclient” command to have the VM request an IP address from the DHCP server.
 - Verify the virtual machine can reach the internet.
 - In a terminal/shell run the command ping -c 5 www.google.com
 - This will confirm DNS can resolve domain names, and that the VM can reach the internet
 - Update/patch the VM
 - In a terminal/shell, run the following, and depending on your network connection, etc., be prepared to wait:
 - **export DEBIAN_FRONTEND=noninteractive; apt-get -q update; apt-get -y -q dist-upgrade**
 - Create a snapshot for the VM

IPS VM

This VM is going to be responsible for running the AFPPACKET bridge between the IPS 1 (VMnet2) and IPS 2 (VMnet 3) virtual networks. Perform the following tasks to install the IPS VM:

- If you haven't already for the siem VM, Download Ubuntu Server 16.04.1 64-bit here:
<https://www.ubuntu.com/download/server>
- Create a new, custom virtual machine with the following settings:
 - Configure the VM to install from the Kali Linux ISO you downloaded
 - Set the Guest OS family to "Linux"
 - Set the Guest OS version to "Ubuntu 64-bit"
 - Create a new virtual disk
 - Name the VM "ips"
 - Note that after you save the VM and grant it a name, VMware Fusion immediately assumes you want to turn the Virtual Machine on. In the main menu, click "Virtual Machine", then "Shut Down"; In order to adjust the hardware settings below.
 - Customize the following hardware settings (Virtual Machine > Settings or you can click the "Customize Settings" button on the finish page of the new VM wizard):
 - Under CD/DVD (IDE), choose the "Choose a disc or disc image" and find the Ubuntu 16.04.1 64-bit Linux ISO you just downloaded
 - Ensure that the "Connect CD/DVD Drive" option is checked
 - Allocate 2GB (2048MB) of RAM
 - Modify the Hard Disk (SCSI) Settings, and change the disk size to 80.00GB
 - Uncheck the advanced option "Split into multiple files"
 - Check the advanced option "Pre-allocate disk space"
 - Allocate 1-2 processors, where total number of cores is no more than 2
 - Connect the network adapter to VMnet2
 - Ensure that "Connect at power on" is checked
 - Under advanced settings, click the "Generate" button to generate a new MAC address. **Record this MAC address as well as the VM and network it corresponds to.**
 - Remove the USB Controller
 - Remove the Sound Card
 - Remove the Printer
 - Remove the Camera
 - In the Isolation settings, uncheck the "Enable Drag and Drop" and "Enable Copy and Paste"
 - Add two more network adapters
 - Attach network adapter 2 to vmnet 3 (IPS 1)

- Under advanced settings, click the “Generate” button to generate a new MAC address. **Record this MAC address as well as the VM and network it corresponds to.**
- Attach network adapter 3 to vmnet 4 (IPS 2)
 - Under advanced settings, click the “Generate” button to generate a new MAC address. **Record this MAC address as well as the VM and network it corresponds to.**
- Install Ubuntu Server
 - Be sure to install the command line utilities and SSH Server role when asked. You should not need to install any other roles or features for this server
- After the install is completed, power off the VM and adjust the following settings:
 - Remove the virtual CD/DVD drive
- Log in to the pfSense web UI, navigate to Services > DHCP Server, and add a static mapping to the LAN interface DHCP, for the MAC address of the original network adapter for the ips VM (attached to the management network, vmnet2); assign it the IP address 172.16.1.4, making sure to enter a description that tells you which VM this static mapping belongs to!
- Power on the Virtual Machine and do the following:
 - Verify that the static mapping for the VM’s IP address worked; Open a terminal/shell and run ifconfig -a | less and verify that “eth0” or whatever the network adapter’s name is in the VM was given the IP address you configured its static mapping for in pfSense.
 - If the VM does not have an ip address, run the “dhclient” command to have the VM request an IP address from the DHCP server.
 - Verify the virtual machine can reach the internet.
 - In a terminal/shell run the command ping -c 5 www.google.com
 - This will confirm DNS can resolve domain names, and that the VM can reach the internet
 - Update/patch the VM
 - In a terminal/shell, run the following, and depending on your network connection, etc., be prepared to wait:
 - **export DEBIAN_FRONTEND=noninteractive; apt-get -q update; apt-get -y -q dist-upgrade**
 - Create a snapshot for the VM

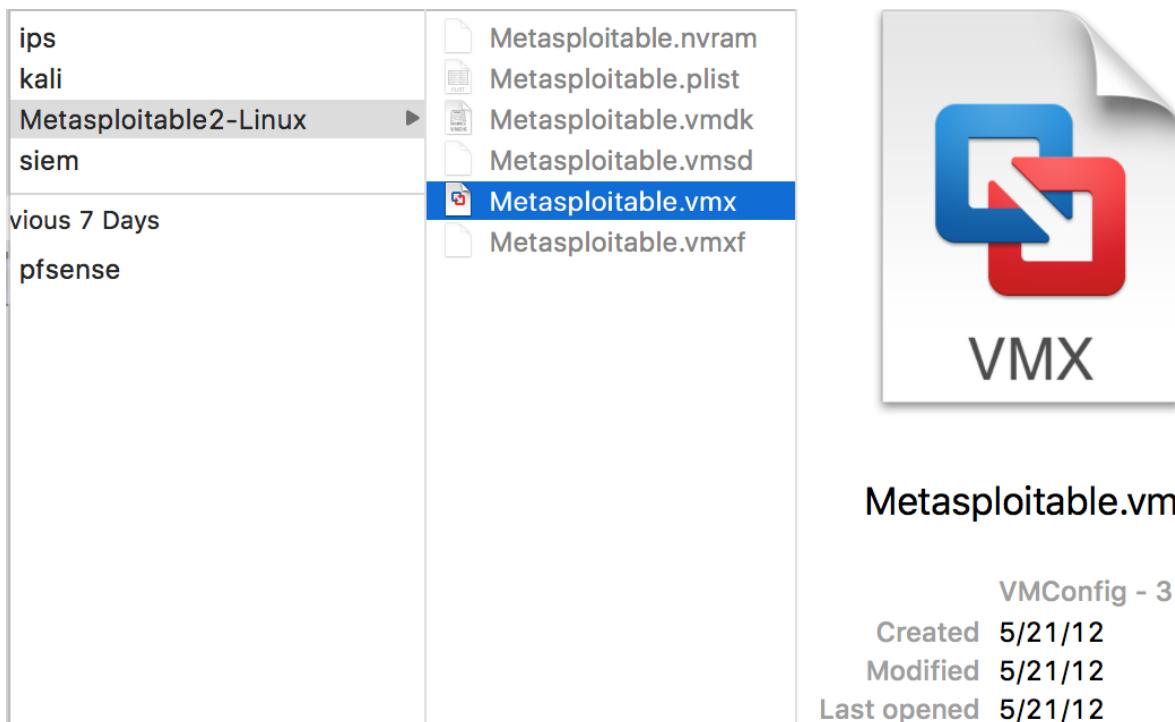
Metasploitable 2

Metasploitable 2 is a little different from the other VMs. It is already pre-created, we just need to import it, and reconfigure some of the virtual hardware. This should be pretty simple to do. First, download a copy of metasploitable 2 from

<https://sourceforge.net/projects/metasploitable/files/Metasploitable2/>. Metasploitable 2 is distributed as a .zip archive. Finder can unzip it by double clicking it, or you can use the command line utility “unzip” to do so.

Download the archive, and decompress it in the download directory. You will be left with a folder labeled “Metasploitable2-Linux”. Move this entire directory to the folder the rest of your VMware Workstation folders are stored; In my case, this is /Users/[username]/Documents/Virtual Machines/.

In VMware Fusion, click “File > Open...” and a browser window will Open. Navigate to /Users/[username]/Documents/Virtual Machines/Metasploitable2-Linux, and double click on the file labeled “Metasploitable.vmx”. Now, VMware Fusion is aware of, and able to run Metasploitable.



Before powering it on, we have to adjust its settings. Click on the wrench in the Metasploitable2-Linux console window, or in “Virtual Machine > Settings...” to bring up the settings menu for Metasploitable 2. Perform the following actions:

- Reconfigure the first Network Adapter and connect it to vmnet 4 (IPS 2)
 - Under advanced options, record the MAC address of this adapter, or generate a new one, via the generate button. **Record this MAC address as well as the VM it corresponds to.**
- Remove Network Adapter 2
- Remove the USB Controller
- Remove the CD/DVD drive
- In the Isolation settings, uncheck the “Enable Drag and Drop” and “Enable Copy and Paste”
- Log in to the pfSense web UI, navigate to Services > DHCP Server, and add a static mapping to the OPT1 interface DHCP, for the MAC address of the original network

adapter for the ips VM (attached to the management network, vmnet2); assign it the IP address 172.16.2.3, making sure to enter a description that tells you which VM this static mapping belongs to!

Power on the metasploitable 2 VM, make sure it is bootable. If so, take a snapshot of the VM, it's ready to go for now. You won't be able to verify the DHCP mapping worked until AFTER the ips VM is configured.

So... What Now?

The VMware Fusion initial setup is all but done at this point. However, you're not quite out of the woods just yet you'll want to do the following:

- While not strictly necessary, you may want to complete "[Remote Lab Management](#)" section, for the Host OS of your choice (e.g. [Linux/Unix Remote Access](#)), [Enabling SSH on Kali Linux](#), and/or if you're lazy like me, The section on [Securing root SSH](#) access. This will allow you to remotely manage all of the linux VMs much more easily than through the VMware Fusion console.
- You have to install IDS/IPS software on the IPS VM. You'll need to complete the "IPS Setup Guide". To learn how to do this with either Snort or Suricata as your IPS software of choice.
- The SIEM VM needs to have splunk installed and configured. You'll need to complete the "Splunk Setup Guide".

Setup - VMware Workstation Pro

Note: This guide is written for VMware Workstation Pro. Specifically, the latest version as of this writing, version 12.5. I'll be writing this guide using Windows as the host Operating System, though the host OS itself should not matter too much; You should be able to easily follow along using Windows or Linux as the host OS.

VMware Workstation is a hosted hypervisor offered by VMware. Our guide specifically needs the Pro edition of workstation due to enhanced virtual networking features available in the pro edition that aren't available in the standard edition of VMware Workstation.

Installation

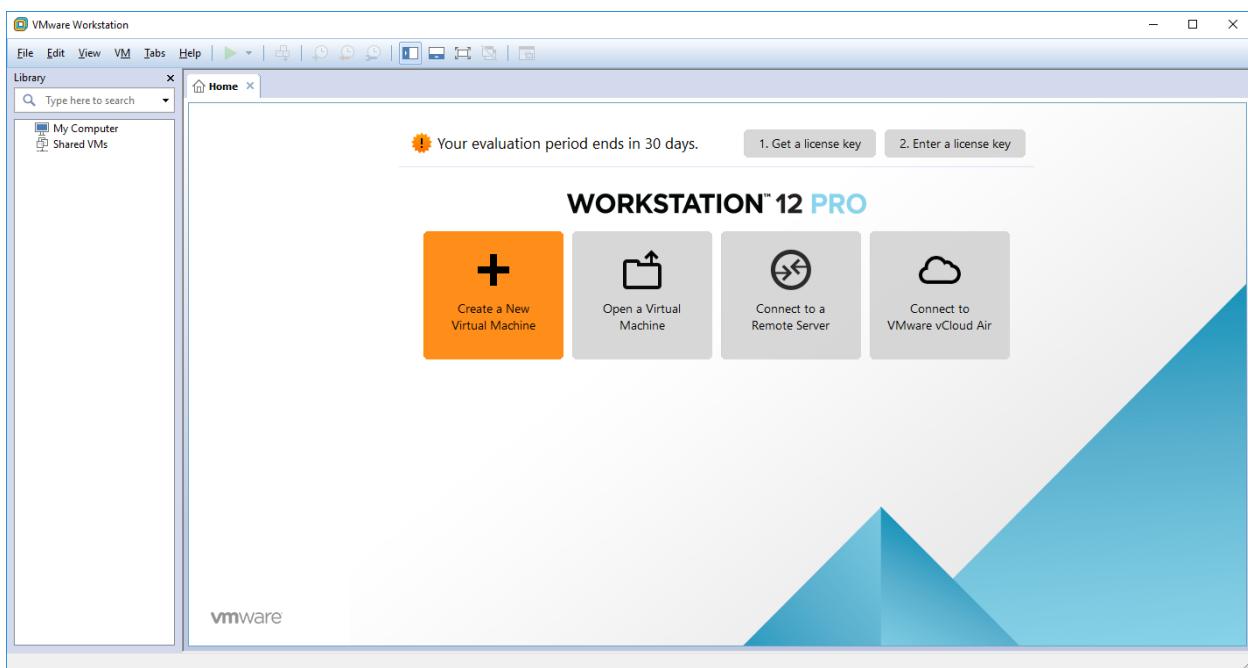
Installing VMware Workstation is pretty straightforward. You can get a trial copy to download and use for free, without even having an account with vmware. As of right now, this is the link to VMware Workstation (Windows and Linux can be downloaded on this page):

<http://www.vmware.com/products/workstation/workstation-evaluation.html>

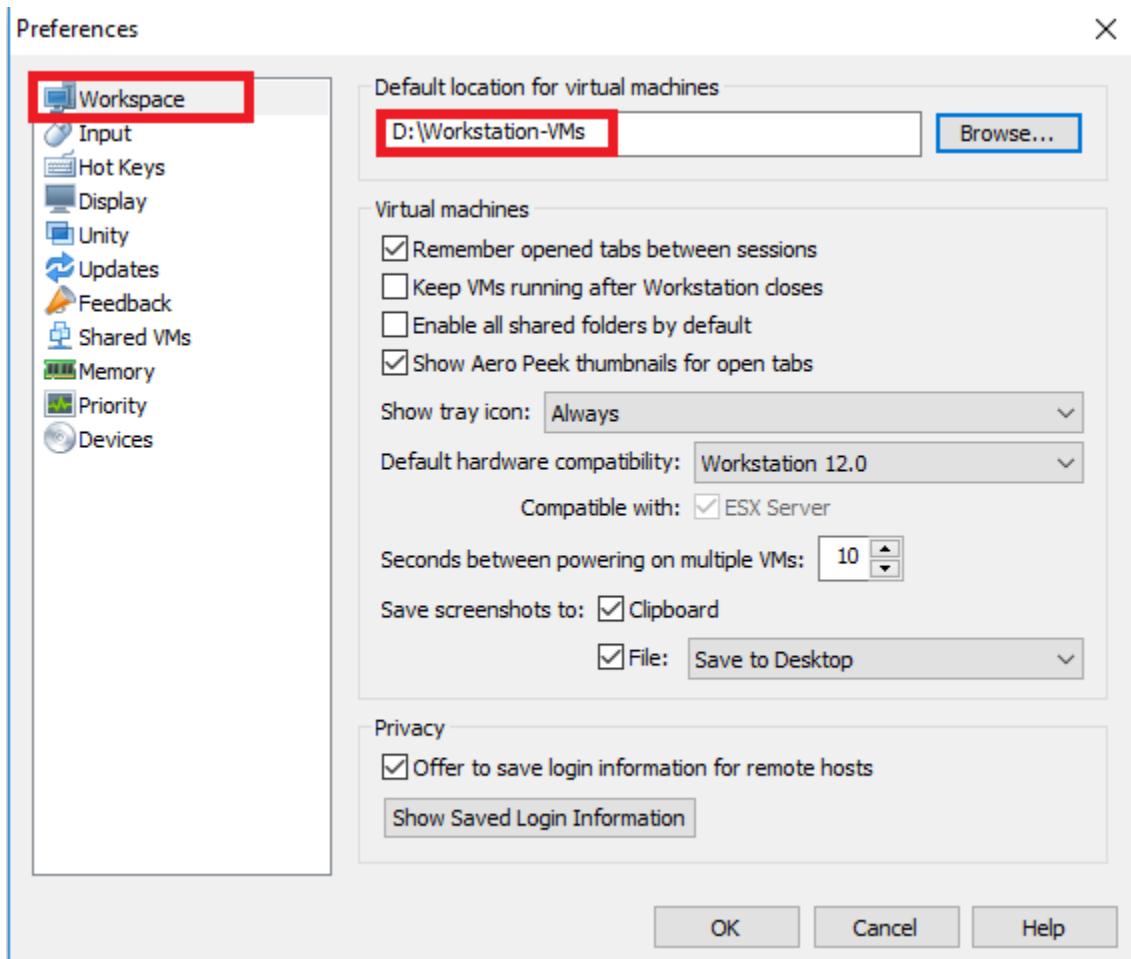
Run the installer and go through the installation steps; the install should finish quickly. Please note that for Linux, you'll need to install kernel modules for vmware as a part of the software installation. To do this, usually you'll need the headers for the kernel you're currently running available. While the name of the package(s) that include the headers varies from distro to distro, and package manager to package manager, usually the headers are included in some sort of a kernel development package.

Hypervisor Preferences

Now that you have workstation installed, start it up. We're going to make some configuration changes here and there, in order to support our lab environment. On first start, the program will ask if you want to input a license, or use the trial period. If you opt to use the trial period, you have to provide a valid e-mail address for VMware to send spam to your inbox. Well, they said a /valid/ e-mail inbox, not /your/ e-mail inbox. You should look into 10-minute mail, Guerilla Mail or other services... I used 10-minute mail for my trial period with no problems to report. After entering your license or e-mail address, you'll be greeted with a main menu that looks like this:

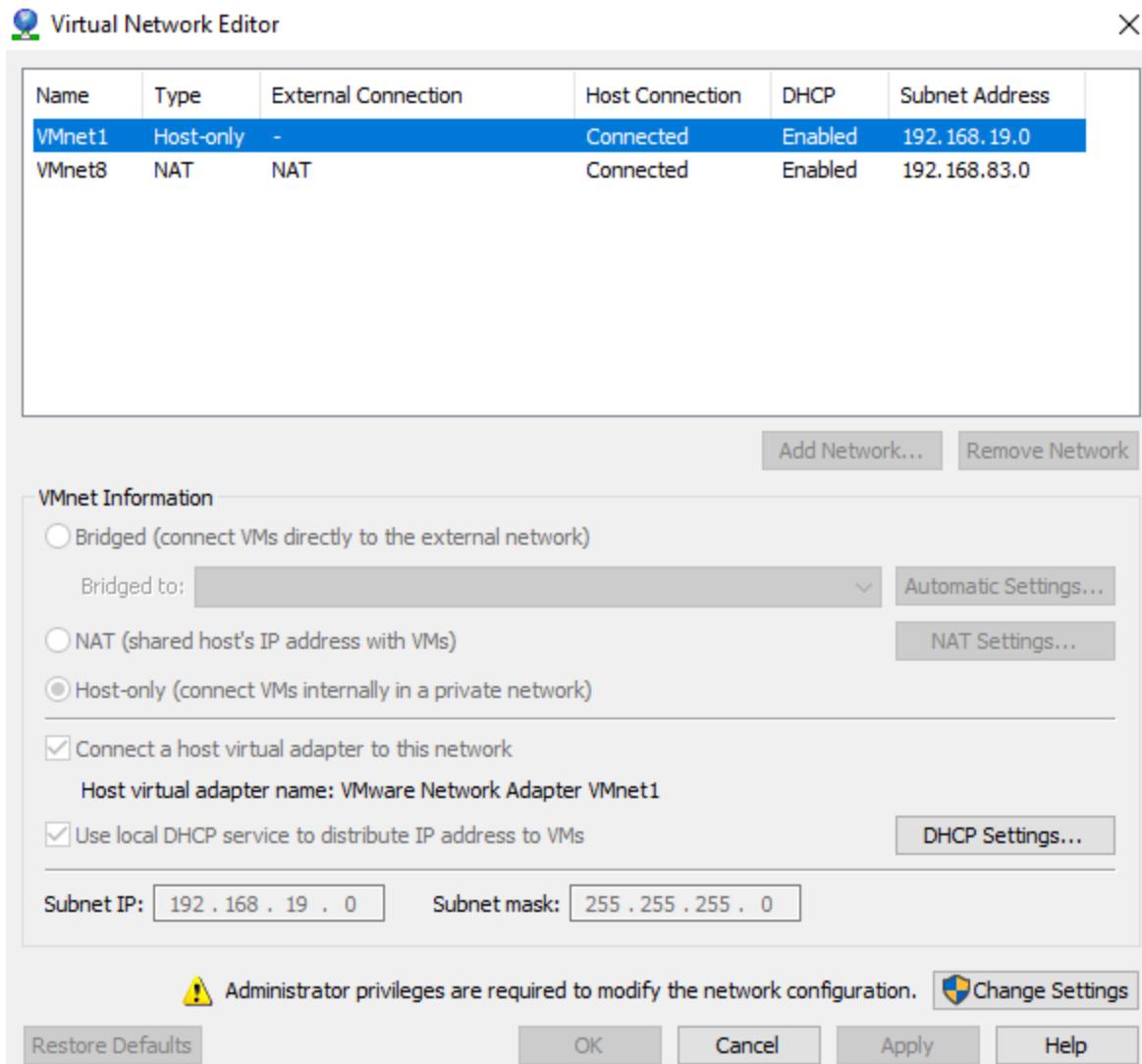


Click Edit on the Menu bar, then click Preferences. The preferences window pops up. Here you can modify a variety of settings with how Workstation operates. You may be interested in checking the other settings and setting things to your liking, but for the time being, all we're concerned with is option titled "Workspace", specifically, the option "Default location for virtual machines". By default, VM files and folders will be placed under your home directory. If you want to change where they are stored, click the Browse button, and choose a new folder to store your VM files in. In my case, I chose to store my virtual machines in D:\Workstation-VMs. When you are done configuring this, or if you are satisfied with the defaults, click OK to close the preferences window.



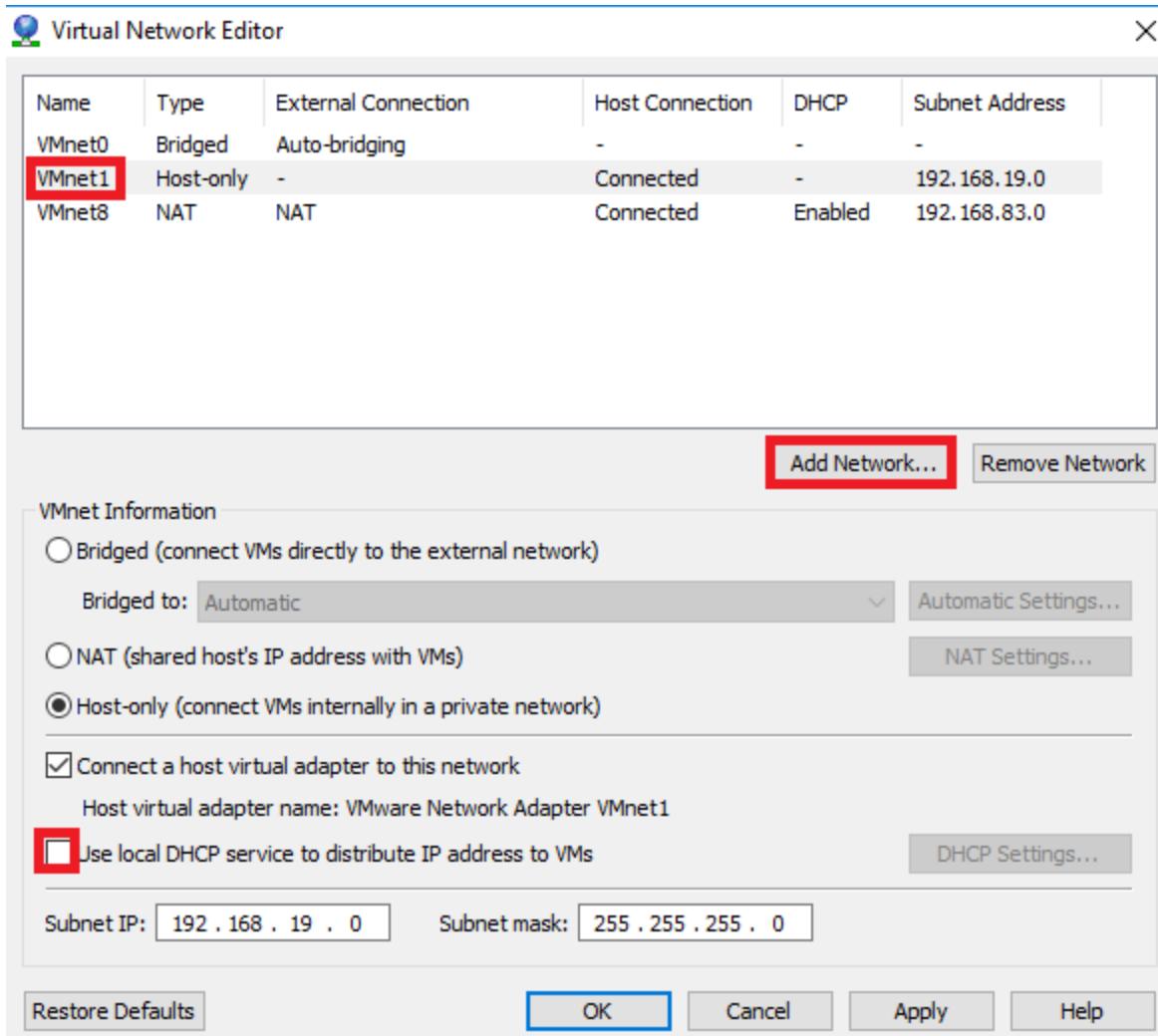
Virtual Networks

Back in the main menu, click on Edit in the menu bar, then click on Virtual Network Editor. The network editor window pops up.

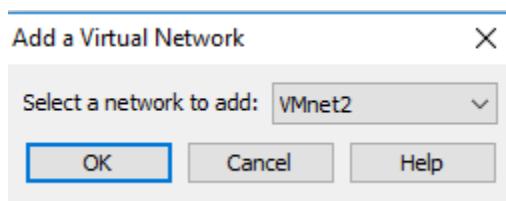


By default, VMware Workstation has two virtual networks defined that you can see right now. VMnet1, a host-only network, and VMnet8, a NAT network. What you don't see currently is that there is also a "VMnet0" network, for bridging to the host system's network connection. For our lab, VMnet0 will serve as our bridged network, VMnet1 will be our management network, and we will have to create two more host-only virtual networks to serve as the IPS 1 and IPS 2 networks.

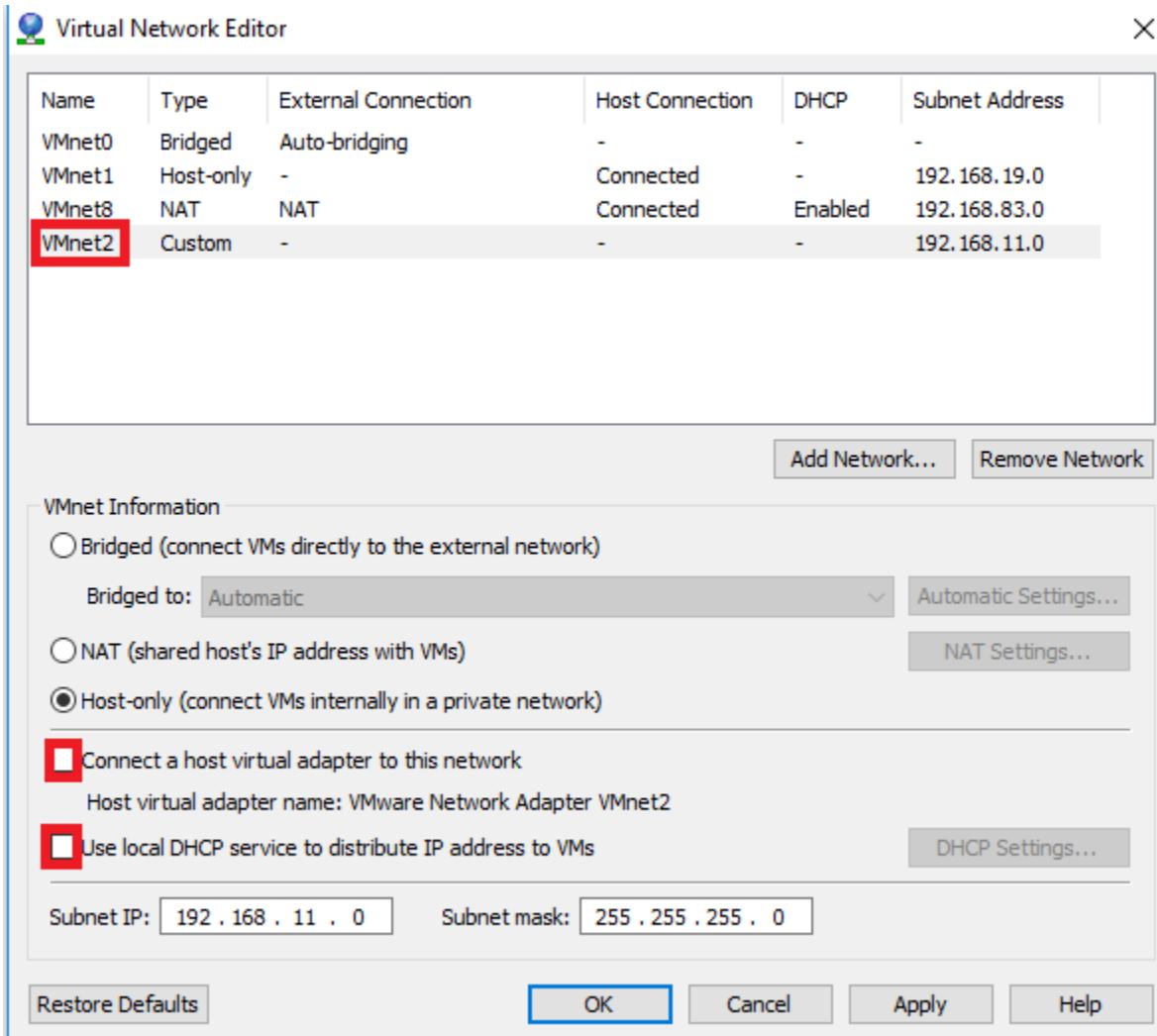
Click on the button in the lower right corner labeled "Change Settings" in order to allow us to reconfigure virtual network settings. Afterwards, the window will pop back up, and many of the VMnet Information options should become available for you to configure, and VMnet0 (Bridged) should be visible. Click on "VMnet1" in the main window pane, then uncheck the checkbox "User local DHCP service to distribute IP address to VMs", then Click on the "Add Network..." button.



A small window pops up with a drop-down labeled “Select a network to add:”. Click okay to choose the default network name (in my case, this was VMnet2). This network is going to serve as the IPS 1 network.



After a moment or two, you should be back at the Virtual Network Editor. The window should have highlighted “Vmnet2” in the primary pane for you already. If it did not, click on “VMnet2” in the primary pane. Verify that the “Host-only (connect VMs internally in a private network)” radio button is selected; this should be the default option. Uncheck the option “Connect a host virtual adapter to this network”, and the option “Use local DHCP service to distribute IP address to VMs”.



You might be wondering why we're disabling DHCP and network adapters for VMnet1 and 2 so far, and why we're disabled the host virtual adapter for VMnet 2. That's because we will be having pfSense handle DHCP services. Leaving DHCP enabled here will cause address conflicts. Additionally the pfSense DHCP server has much more flexibility. Regarding the removal of the host virtual adapter, we're doing this because we do not want the host OS to have direct access to the VMnet2 network, since it will be one of the two IPS networks. This is done to protect the hypervisor host from potential threats.

Repeat this process one more time, because we need to create an “IPS 2” network. Add a new network (in my case the network name was “VMnet3”), verify that it is a host-only network, ensure that the “Connect a host virtual adapter to this network”, and the “Use local DHCP service to distribute IP address to VMs” options are both unchecked. Your virtual network editor primary pane will look like this:

Name	Type	External Connection	Host Connection	DHCP	Subnet Address	
VMnet0	Bridged	Auto-bridging	-	-	-	
VMnet1	Host-only	-	Connected	-	192.168.19.0	
VMnet8	NAT	NAT	Connected	Enabled	192.168.83.0	
VMnet2	Custom	-	-	-	192.168.11.0	
VMnet3	Custom	-	-	-	192.168.131.0	

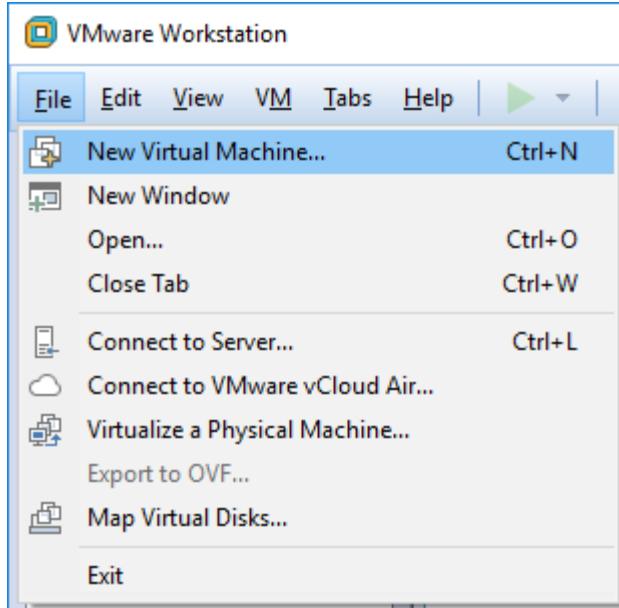
The virtual networks are now configured to support our lab environment. Click the Apply button, then click OK to apply our network settings, and close the Virtual Network Editor window.

Creating the First VM, pfSense

pfSense is the keystone holding this entire configuration together. Personally, its my favorite firewall distro due to ease of use, the amount of functionality it includes out of the box, combined with a plugin/add-on system for additional functionality ; If you have the CPU, RAM, and Disk, pfSense can easily be converted into a so-called “Next-Generation” firewall. To start, make your way to <https://www.pfsense.org/download/>. Download the latest Installation ISO for the amd64 architecture. After download the ISO, you will need a decompression tool to uncompress the pfSense ISO. On linux, the “gunzip” command line utility should work. On Windows, I recommend using 7-zip.

Adding a New VM

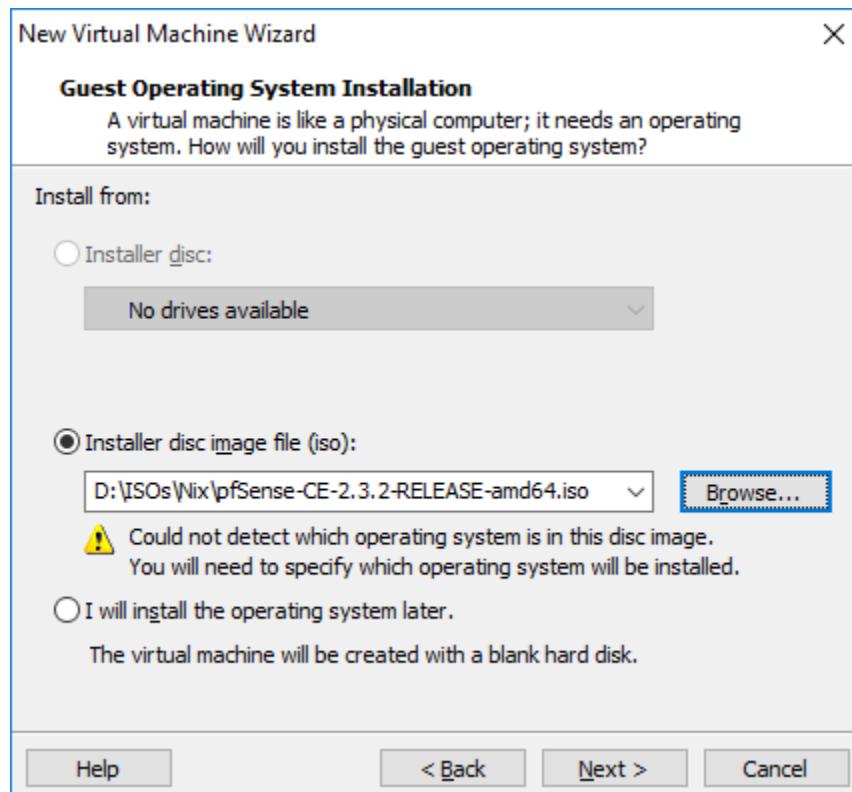
In the VMware Workstation main window, click File on the menu bar, and New Virtual Machine. The “New Virtual Machine Wizard” opens.



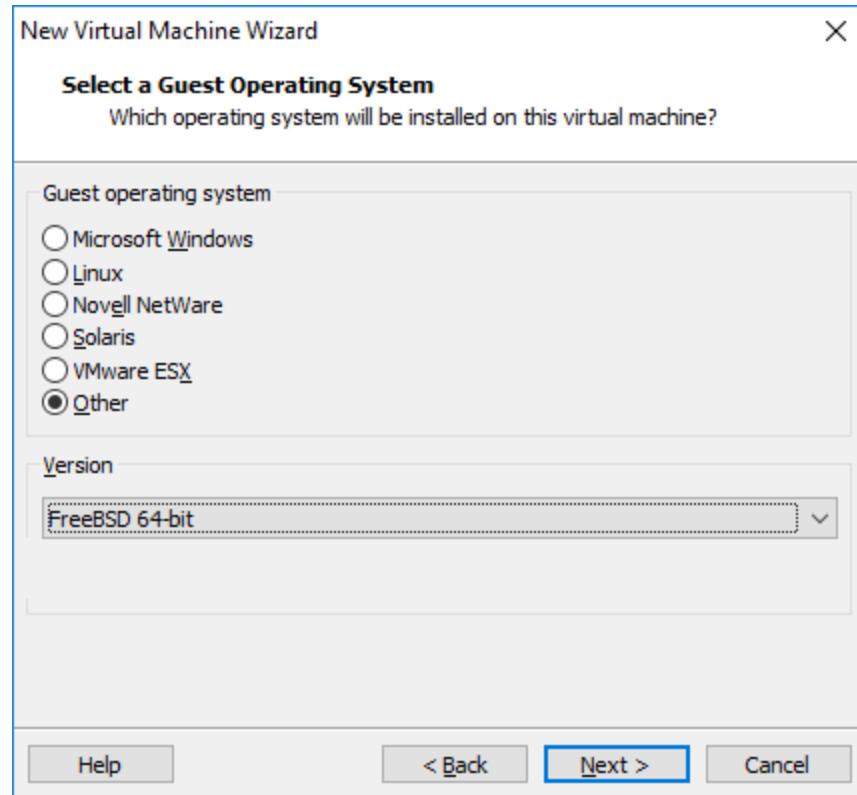
On the first screen of the wizard, select the “Typical (recommended) radio button, then click Next. The Custom radio button allows you to configure a bunch of additional settings on your VM relating to disk types and storage options. For the purposes of our lab, its not necessary to enumerate and customize these options, so for now, creating VMs via the “Typical” option will work.



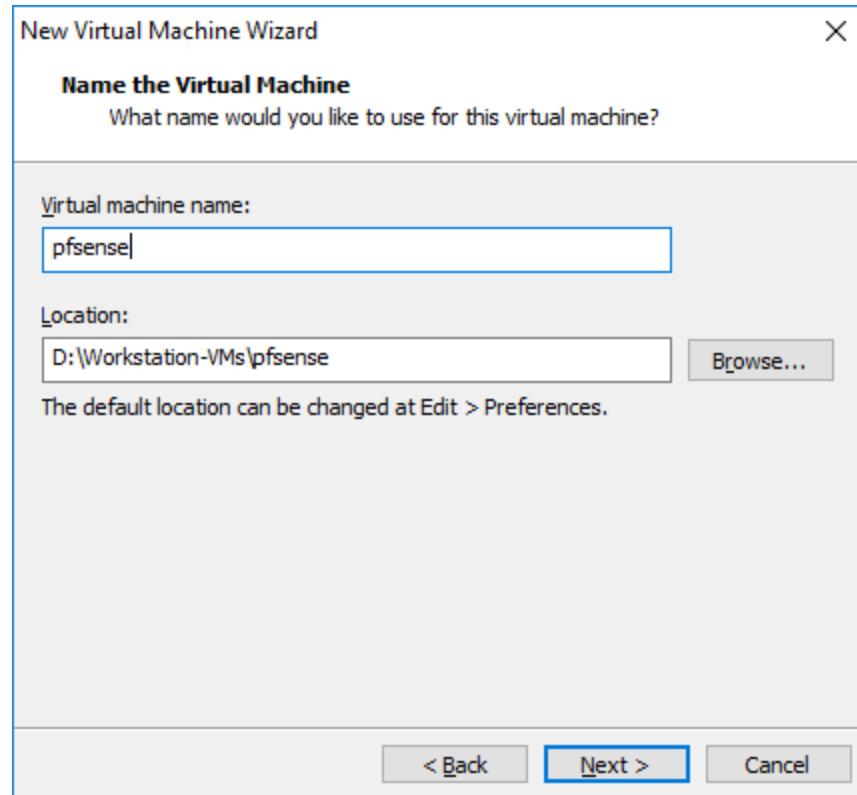
The next screen has us configure where and how we'll be installing our VM's virtual machine. Select the "Installer disc image file (iso):" radio button, then click the browse button. Browse to where you stored the decompressed pfSense ISO file. In my case, I store the ISO in D:\ISOs\Nix. Note that you'll get a warning "Could not detect which operating system is in this disc image." This is fine. After selecting your ISO, click Next.



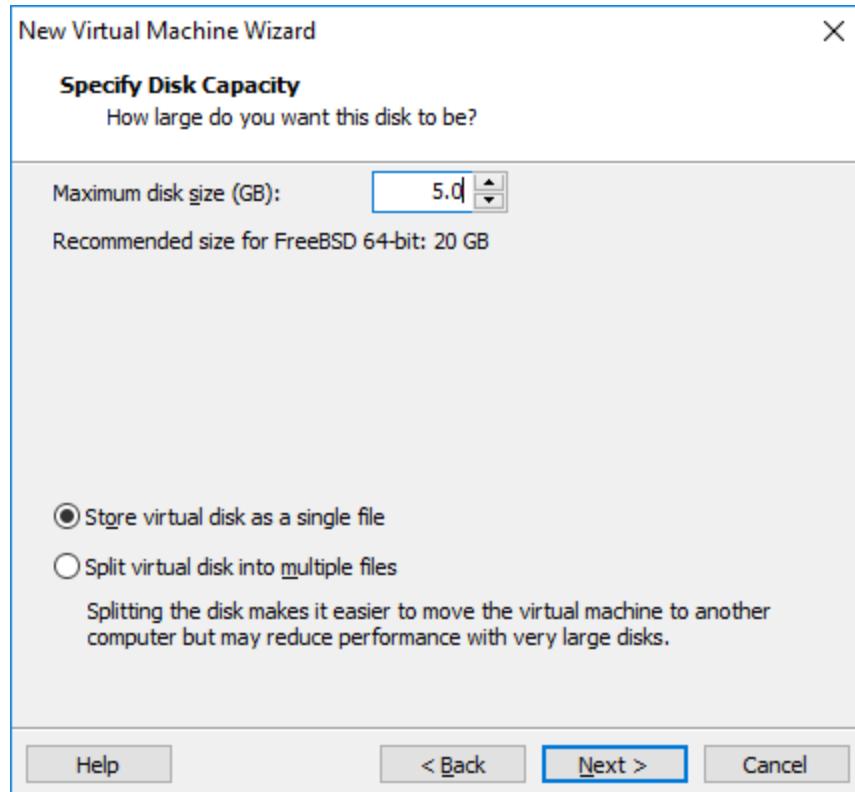
The next screen will ask you to select what guest operating system you will be installing. Click the "Other" radio button under the "Guest operating system" section, then select "FreeBSD 64-bit" from the "Version" drop-down. Click Next.



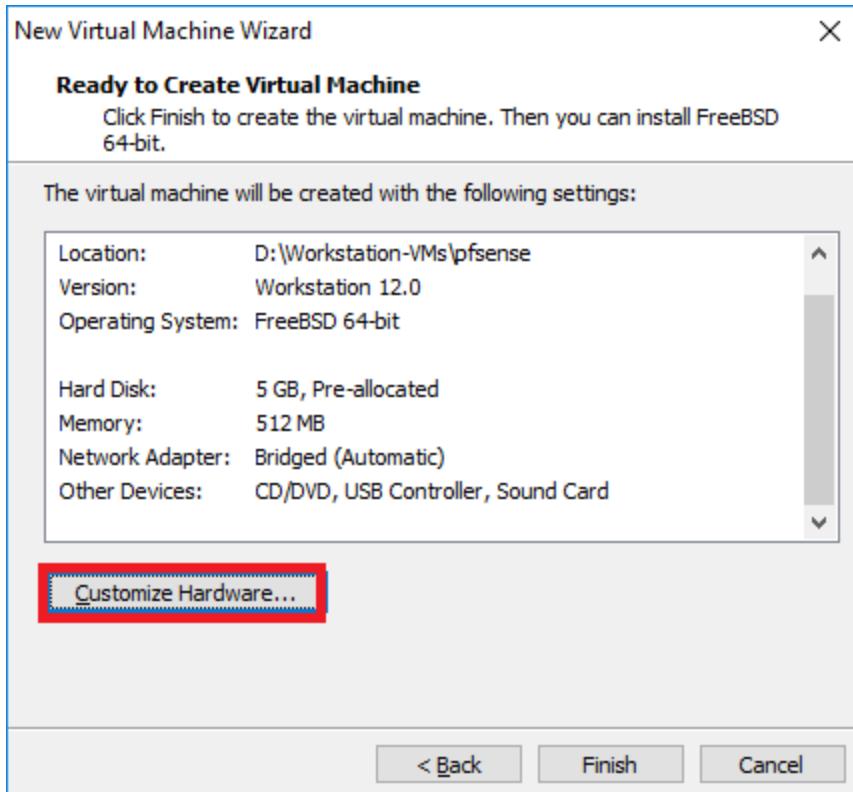
The next screen asks you to name your VM and confirm where you want to store its files. Name the VM “pfSense”, then click Next.



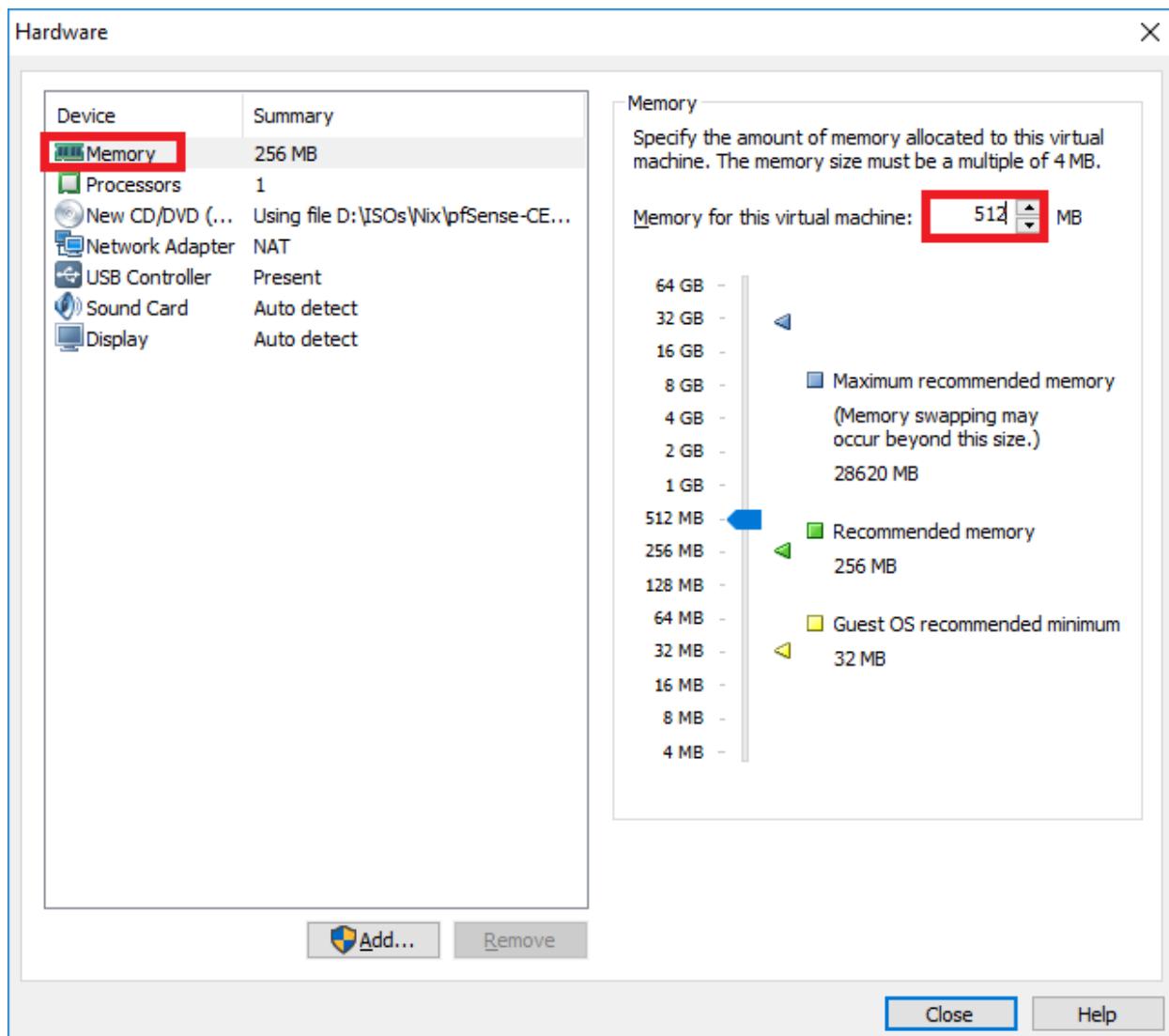
The next screen asks you to specify how much drive space you want to allocate for your VM. Input 5.0 in the “Maximum disk size (GB):” input box, then click the “Store virtual disk as a single file radio button”. Click Next.



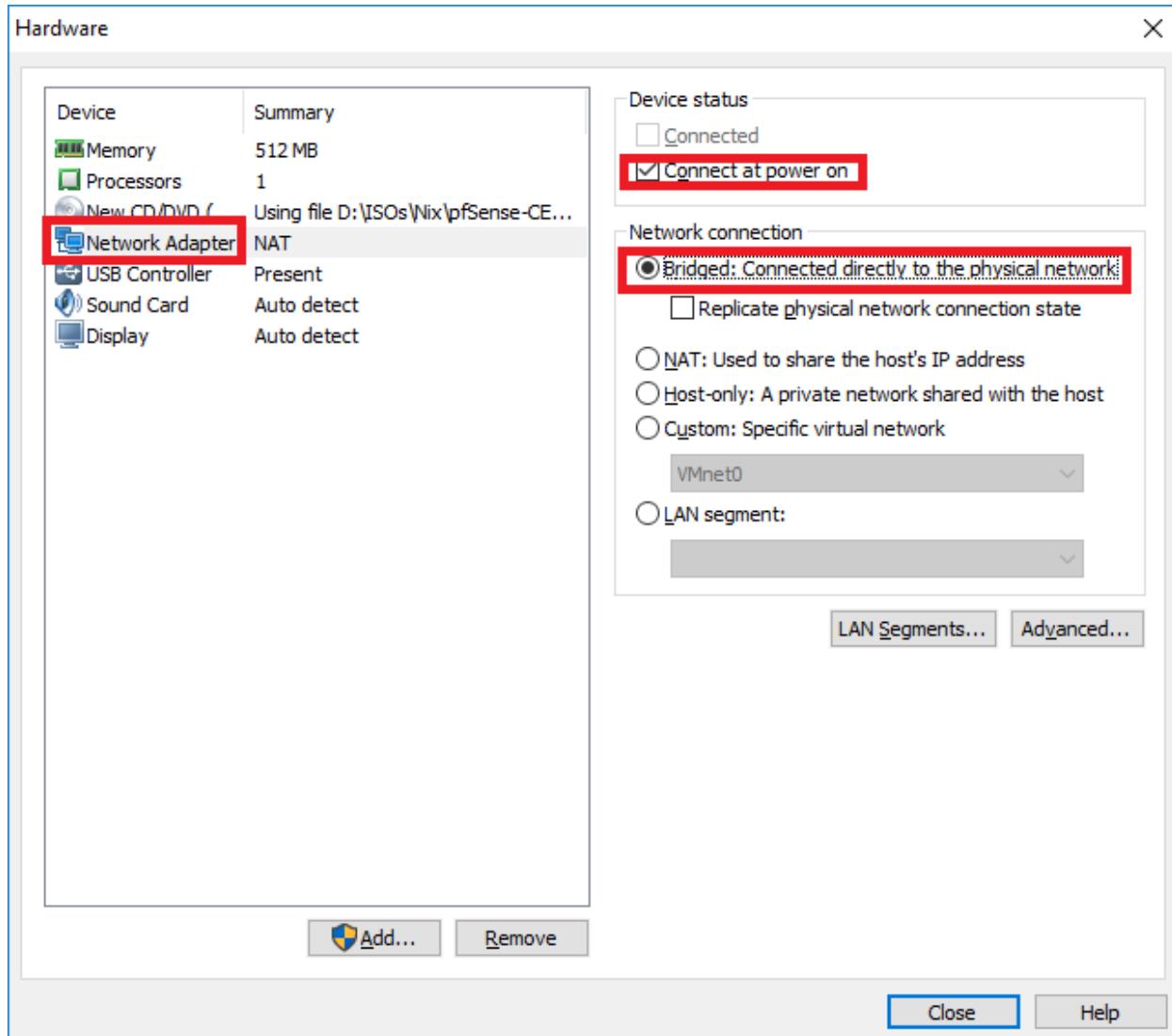
The final screen has you confirm Virtual Machine settings, and gives you the option to customize the hardware present on your virtual machine. Click the “Customize Hardware...” button.



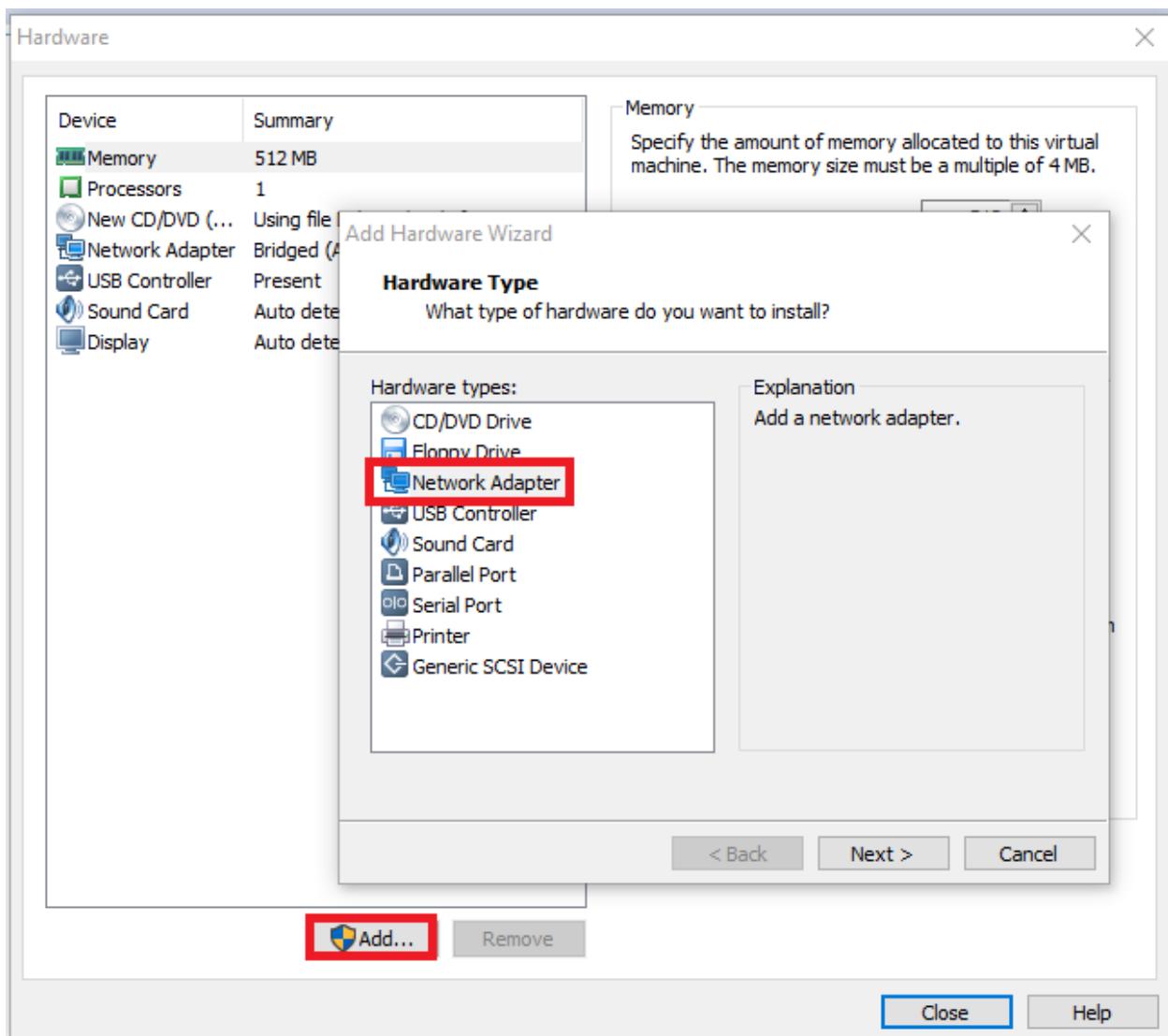
A new window listing the different hardware attached to our new VM will appear. On this new screen, click on “Memory” listed under the “Device” pane to highlight it. This tells us how much memory is currently allocated to our VM, and gives us the ability to change it. We’re going to double the default amount of 256MB to 512MB, by inputting 512 into the “Memory for this virtual machine:” input box.



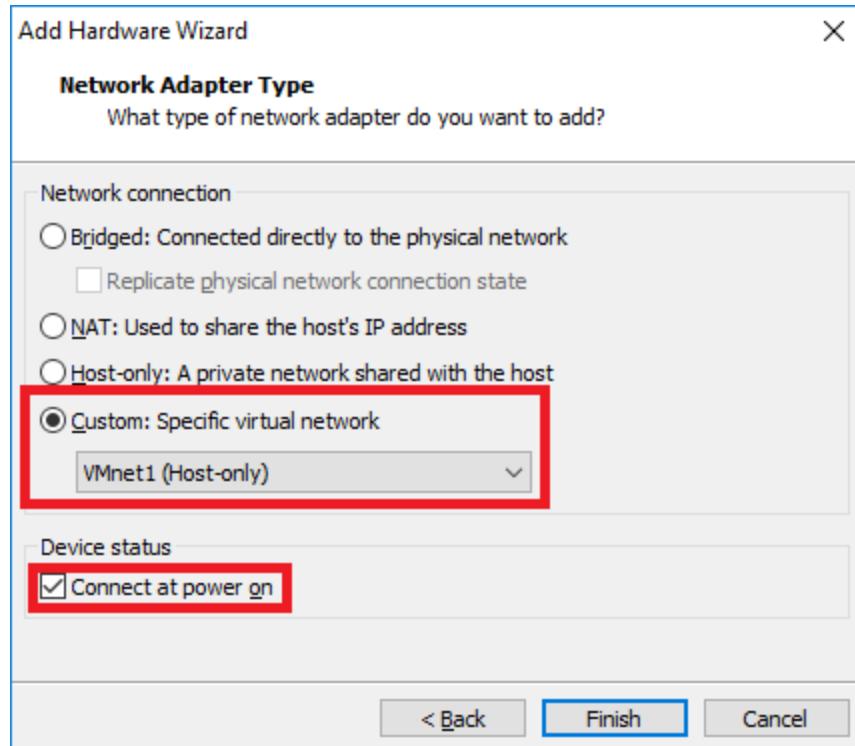
Click on “Network Adapter” in the “Device” pane. This will bring up the settings for this network adapter. Ensure that the option “Connect at power on” is checked, in the “Device Status” section. Change the radio button from the default setting of “NAT: Used to share the host’s IP address” to “Bridged:Connected directly to the physical network.” (note: you do not have to check the “Replication physical network connection state” checkbox)



Click the “Add...” at the bottom of the “Device” pane, and the “Add Hardware Wizard” starts. Click “Network Adapter” under the pane labeled “Hardware types:”, then click Next.

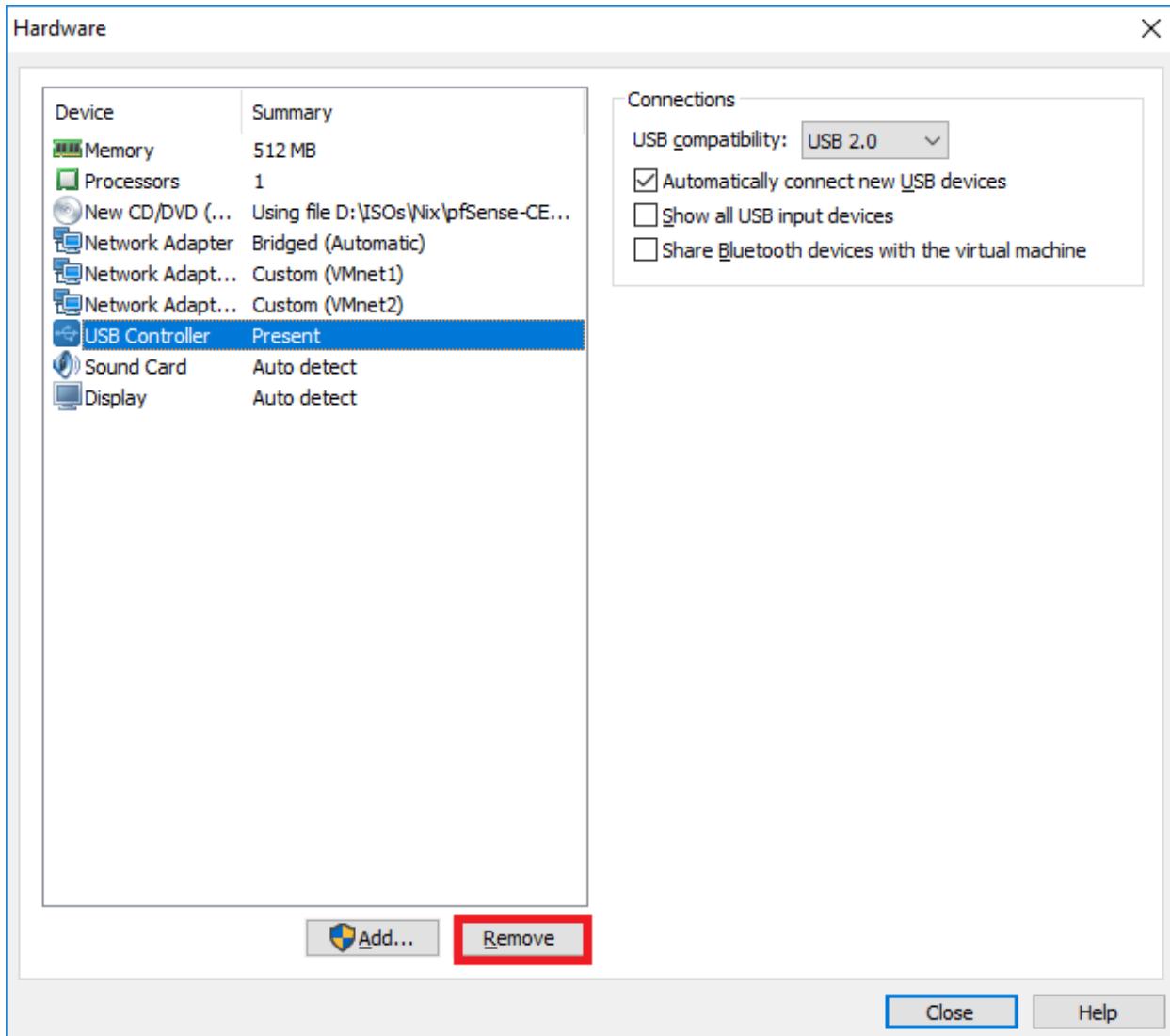


On the next screen, Click the “Custom: Specific virtual network” radio button, then select “VMnet1” in the drop-down menu. Make sure that the “Connect at power on” checkbox is checked, then click Finish.



Repeat this process once more, creating a third network adapter, except this time, connect it to the VMnet2 virtual network.

After adding two additional network cards (for a grand total of 3 network adapters attached to this VM), I'm going to have you remote the USB Controller, and the Sound Card. First click on the USB Controller to highlight it. After highlighting it, select "Remove" under the primary pane.



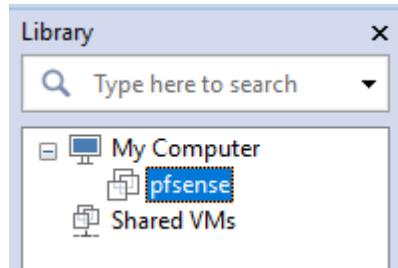
Repeat this process on the Sound Card to remove that as well.

After you are finished, your hardware configuration should look like this:

Device	Summary
Memory	512 MB
Processors	1
New CD/DVD (...)	Using file D:\ISOs\Nix\pfSense-CE...
Network Adapter	Bridged (Automatic)
Network Adapt...	Custom (VMnet2)
Network Adapt...	Custom (VMnet1)
Display	Auto detect

Click the Close button in the hardware window to exit hardware customization. Back in the New Virtual Machine Wizard, click Finish to exit the wizard. The system will take a moment or two to

allocate disk space to the VM, then the main VMware Workstation window should list pfSense under the “Library” pane:



Installing pfSense

At this point, you’re now ready to install pfSense on your VM. To start the VM, in the VMware Workstation main menu, you can click on pfSense in the “Library” pane, then select “Power on this virtual machine” in the main window. After doing so, the primary window will switch to view the console of the powered on VM. Clicking on this console will attach your mouse and keyboard to the VM. To detach again, press ctrl+alt.

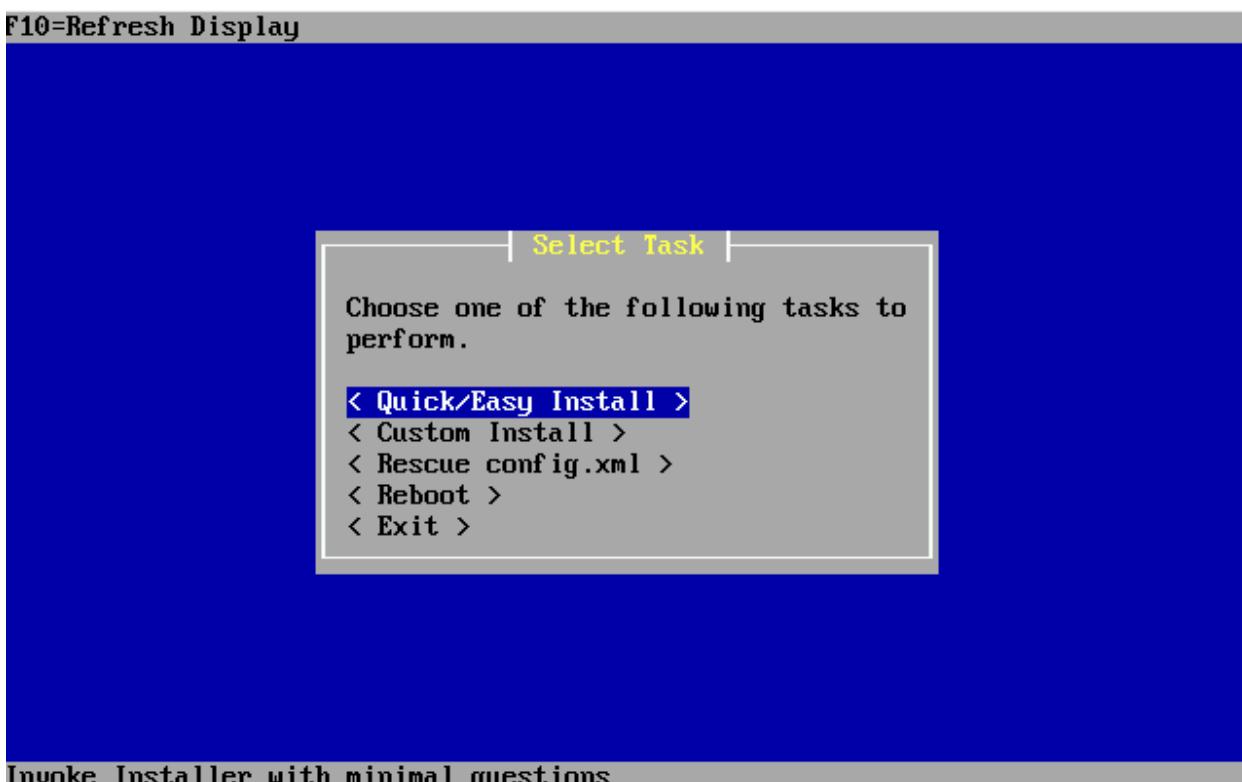


Let pfSense boot up, and the system should automatically run the installer. Adjust your video, screenmap, and keymap settings as necessary, then select “< Accept these Settings >”.



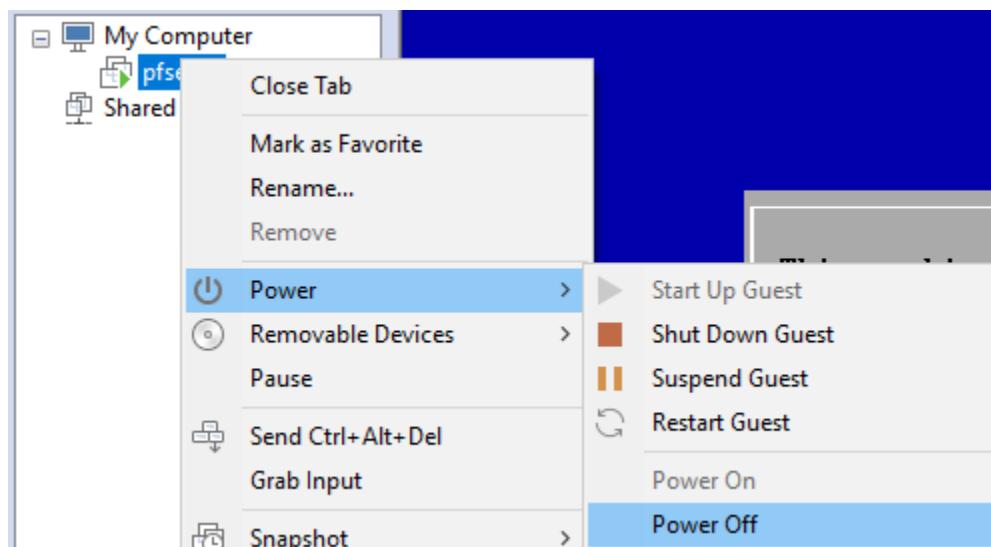
On the next screen, select “< Quick/Easy Install >” and let pfSense do all the heavy lifting. The next screen will inform you that the install will erase the contents of the hard disk. Since our virtual disk is already empty, this doesn’t matter in the least. Select OK. and let the install run.

F10=Refresh Display



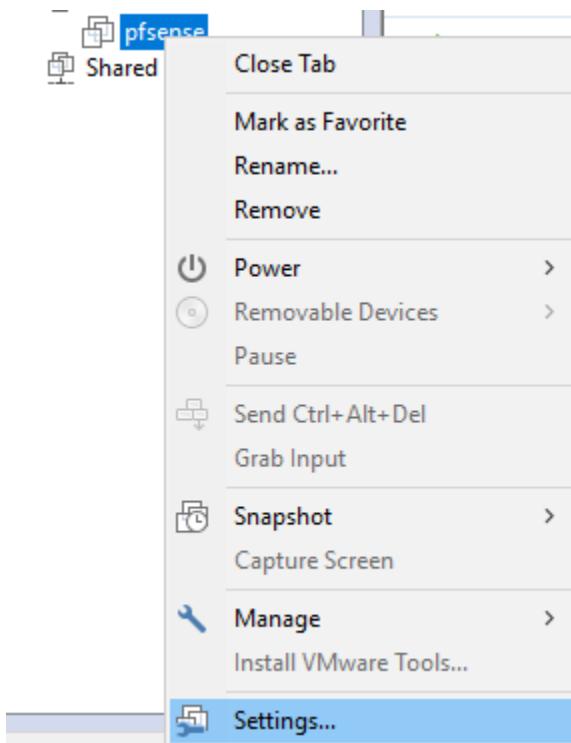
Invoke Installer with minimal questions

The installer will go through and install pfSense to the virtual hard disk. The installer will ask if you want to install a standard or embedded kernel. Make sure to select “< Standard Kernel >”. Finally, the installer informs you to reboot the machine to boot from the hard drive. The installation is done, however, we’re not going to reboot. Right click on pfSense in the “Library” pane, select “Power”, and “Power Off”, and power off the VM.



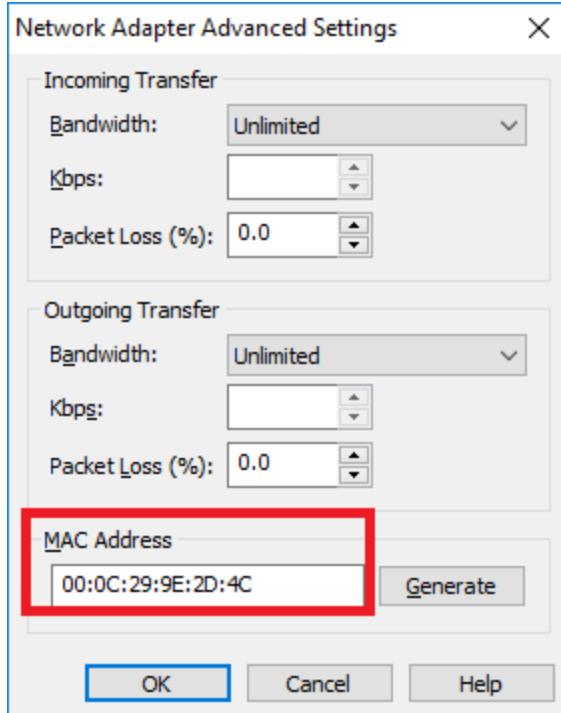
Final VM Settings

Now that pfSense is installed, we no longer need the CD/DVD drive attached to the VM, so we're going to remove it. On the VMware Workstation main screen, right click on pfSense in the "Library" pane, then select "Settings..."



In the "Hardware tab, under the "Devices" pane, click on "CD/DVD (IDE)", then click on the remove button to remove the CD/DVD drive, just like with the sound card and USB controller, earlier.

Click on "Network Adapter" in the "Device" pane, then click on the "Advanced..." button to bring up the advanced network settings menu. Be sure to document the contents of the "MAC Address" section of this screen. Click OK when you are done. Repeat this process for Network Adapter 2 and Network Adapter 3. The goal is for you to know which mac address corresponds to which VMnet (e.g. Bridged, VMnet1 (Management), and VMnet2 (IPS 1)) for configuring Network Settings in pfSense.



Click OK to exit the Virtual Machine Settings menu. Your VM's Device listing should look like this:

Device	Summary
Memory	512 MB
Processors	1
Hard Disk (SCSI)	5 GB
Network Adapter	Bridged (Automatic)
Network Adapter 2	Custom (VMnet1)
Network Adapter 3	Custom (VMnet2)
Display	Auto detect

Network Configuration

Power on the pfSense VM, and allow it to boot up. Eventually you'll be greeted with the main pfSense menu on the console with 16 options. Select option 1, to assign interfaces. This defines which interface (e.g. em0, em1, em2, etc.) corresponds to WAN, LAN, OPT1, etc. networks for pfSense network services and firewall policies.

```

0) Logout (SSH only)
1) Assign Interfaces
2) Set interface(s) IP address
3) Reset webConfigurator password
4) Reset to factory defaults
5) Reboot system
6) Halt system
7) Ping host
8) Help
9) pfTop
10) Filter Logs
11) Restart webConfigurator
12) PHP shell + pfSense tools
13) Update from console
14) Enable Secure Shell (sshd)
15) Restore recent configuration
16) Restart PHP-FPM

```

The WAN interface needs to be assigned to the bridged network adapter. The LAN interface needs to be assigned to our Host-Only adapter (the management network), and finally, OPT1 needs to be assigned to our internal network adapter. In my case, em0 became the WAN interface, em1 became the LAN interface, and em2 became the opt2 interface. **Be sure to pay attention to the MAC addresses for em0, em1, and em2, and correlate that to the MAC addresses you documented earlier for network adapter, network adapter 2, and network adapter 3.** After you confirm that the settings are correct, you should automatically be returned to the pfSense main menu.

```

Enter the WAN interface name or 'a' for auto-detection
(em0 em1 em2 or a): em0

Enter the LAN interface name or 'a' for auto-detection
NOTE: this enables full Firewalling/NAT mode.
(em1 em2 a or nothing if finished): em1

Optional interface 1 description found: OPT1
Enter the Optional 1 interface name or 'a' for auto-detection
(em2 a or nothing if finished): em2

Enter the Optional 2 interface name or 'a' for auto-detection
( a or nothing if finished):

The interfaces will be assigned as follows:

WAN -> em0
LAN -> em1
OPT1 -> em2

```

Now that we have assigned the network interfaces, we have to configure IP addresses. In most cases, the “WAN” interface (bridged network) will automatically get an IP address from the device on your physical network that provides DHCP services. As for the LAN and OPT1 networks we will have to manually set the IP address, subnet mask, and DHCP scopes for the networks. Select option 2 in the pfSense main menu to get started.

- | | |
|---------------------------------------|----------------------------------|
| 0) Logout (SSH only) | 9) pfTop |
| 1) Assign Interfaces | 10) Filter Logs |
| 2) Set interface(s) IP address | 11) Restart webConfigurator |
| 3) Reset webConfigurator password | 12) PHP shell + pfSense tools |
| 4) Reset to factory defaults | 13) Update from console |
| 5) Reboot system | 14) Enable Secure Shell (sshd) |
| 6) Halt system | 15) Restore recent configuration |
| 7) Ping host | 16) Restart PHP-FPM |
| 8) Shell | |

The configuration wizard will ask you which interface you want to configure. You'll have to configure the LAN and OPT1 interfaces individually (and if you're providing a STATIC IP address, netmask and gateway for the WAN, you'll have to configure the WAN interface as well). Here are the settings I used for my lab network:

LAN (lan)	-> em1	-> v4: 172.16.1.1/24
OPT1 (opt1)	-> em2	-> v4: 172.16.2.1/24

The wizard will ask you if you want to set up IPv6 for the interfaces and networks. Say no, because that's a can of worms we're not going to deal with here. The wizard will also ask want to use DHCP. For the LAN and OPT1 networks, I said yes. For the LAN network start address, I entered 172.16.1.10. For the end address, I entered 172.16.1.254. For the OPT1 network I chose 172.16.2.10 as the start, and 172.16.2.254 for the end address. If you are using 172.16.1.0/24 or 172.16.2.0/24 in your physical network, choose another network range to assign to the LAN and OPT1 interfaces to avoid network conflicts.

The reason we start at 1.10 and 2.10 is to make room for our other virtual machines that need to have a fixed IP address. The gap allows us to set STATIC DHCP allocations for VMs/systems whose addresses we want to remain fixed. We'll talk about this a little bit more later. For now, we should be done mucking around in the pfSense CLI. From here on out, we get to use the webUI to configure pfSense.

If so, let's move on to setting up pfSense from the webConfigurator. Before doing so however, you must configure the Host-Only adapter (aka the management network) on your host. If you are using Windows as your host, and If you haven't already, you'll want to visit the "[Unbinding Network Protocols on Windows Virtual Adapters](#)" at a minimum. This document will guide you through configuring the VirtualBox Host-Only Ethernet Adapter with an IP address as well as unbinding network protocols on this adapter to increase the security of your hypervisor host. You may also want to consider "[Using Windows Firewall to Limit Exposure of Windows Hypervisor Hosts](#)" to further enhance the security of the Windows host when interacting with your lab VMs.

Web Configurator - Initial Setup

On the host OS open your a web browser (I prefer Firefox - <https://www.mozilla.org/en-US/firefox/new/>) and navigate to <https://172.16.1.1> (or the IP address you assigned to the LAN interface of your pfSense system). The default credentials for access to the web interface are admin/pfsense. If this is your first time logging in, pfSense takes you through a nice little setup wizard. I'll highlight some of the important things to take note of, and/or change as necessary:

Set the primary and secondary DNS servers you plan on using. I typically use 8.8.8.8 (google public DNS) and 4.2.2.2 (Level 3 public DNS).

The screenshot shows two parts of the pfSense configuration interface. The top part displays the 'Primary DNS Server' as 8.8.8.8 and the 'Secondary DNS Server' as 4.2.2.2. The bottom part shows the 'RFC1918 Networks' section, where the 'Block RFC1918 Private Networks' checkbox is checked, and the 'Block private networks from entering via WAN' checkbox is also checked. A note below explains that this option blocks traffic from reserved private IP ranges (10/8, 172.16/12, 192.168/16) and loopback addresses (127/8), unless the WAN network is in a private address space.

The other default settings on the first time setup wizard should be fine. Please note that they will make you change the password for the admin account as a part of the setup wizard. Document the password, keep it somewhere safe.

Our next order of business is to restrict access to the web interface to where only 172.16.1.2 can administer the firewall. On the menu bar at the top of the page, select Firewall > Rules. Click on LAN to modify the firewall policy for the LAN interface. Click the "Add" button with the arrow facing up. This adds the firewall rule to the top of the firewall policy to where it is evaluated first. The "Edit Firewall Rule" page is fairly straightforward. I've highlighted the options to be aware of.

Edit Firewall Rule

Action	Pass			
Choose what to do with packets that match the criteria specified below. Hint: the difference between block and reject is that with reject, a packet (TCP RST or ICMP port unreachable for UDP) is returned to the sender, whereas with block the packet is dropped silently. In either case, the original packet is discarded.				
Disabled	<input type="checkbox"/> Disable this rule Set this option to disable this rule without removing it from the list.			
Interface	LAN			
Choose the interface from which packets must come to match this rule.				
Address Family	IPv4			
Select the Internet Protocol version this rule applies to.				
Protocol	TCP			
Choose which IP protocol this rule should match.				
Source				
Source	<input type="checkbox"/> Invert match.	Single host or alias	172.16.1.2	/
Display Advanced		Display Advanced		
Destination				
Destination	<input type="checkbox"/> Invert match.	Single host or alias	172.16.1.1	/
Destination port range	HTTPS (443)	From	Custom	To
	HTTPS (443)			Custom
Specify the destination port or port range for this rule. The "To" field may be left empty if only filtering a single port.				
Extra Options				
Log	<input type="checkbox"/> Log packets that are handled by this rule Hint: the firewall has limited local log space. Don't turn on logging for everything. If doing a lot of logging, consider using a remote syslog server (see the Status: System Logs: Settings page).			
Description	pfsense strict anti-lockout			

When you are done, click the Save icon at the bottom of the page. This will take you back to the previous page. A yellow dialogue box with a green button called “Apply Changes” will appear. Click this button to apply this new firewall rule. Next, we want to disable the default anti-lockout rule. Next, lets navigate to Firewall > Aliases. On the firewall Aliases page, click on “IP”, then click Add. Create an alias with the following settings:

Firewall / Aliases / Edit

Properties

Name	RFC1918	The name of the alias may only consist of the characters "a-z, A-Z, 0-9 and _".
Description	An alias for all RFC1918 networks	A description may be entered here for administrative reference (not parsed).
Type	Network(s)	

Network(s)

Hint Networks are specified in CIDR format. Select the CIDR mask that pertains to each entry. /32 specifies a single IPv4 host, /128 specifies a single IPv6 host, /24 specifies 255.255.255.0, /64 specifies a normal IPv6 network, etc. Hostnames (FQDNs) may also be specified, using a /32 mask for IPv4 or /128 for IPv6. An IP range such as 192.168.1.1-192.168.1.254 may also be entered and a list of CIDR networks will be derived to fill the range.			
Network or FQDN	10.0.0.0 / 8	10.x.x.x RFC 1918 networks	
	172.16.0.0 / 12	172.16.x.x RFC 1918 networks	
	192.168.0.0 / 16	192.168.x.x RFC 1918 networks	

Save Add Network

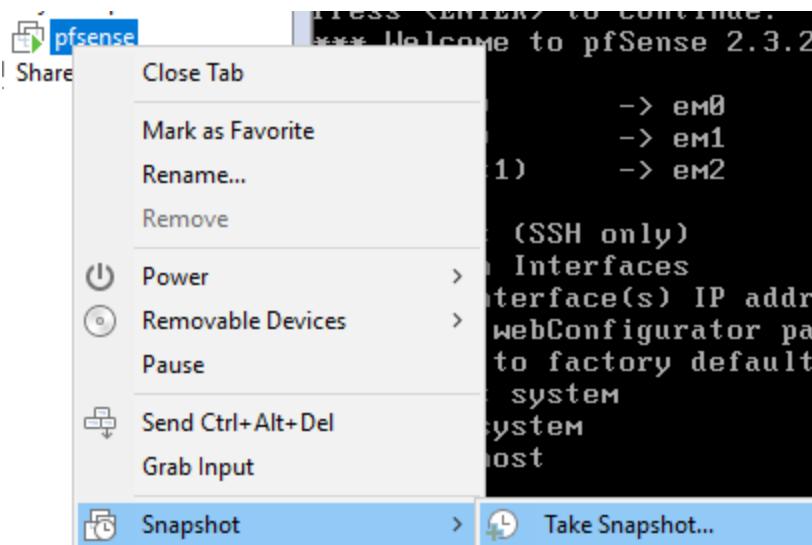
Click Save to be brought back to the previous page, then click Apply Changes. We just created an alias for RFC1918 networks (local networks that are not routable through the public internet). This will come in handy later for creating firewall rules around these networks. Next, navigate to System > Advanced. Click to fill in the checkbox next to the option “Disable webConfigurator anti-lockout rule”, and click save on the bottom of the page.

Anti-lockout Disable webConfigurator anti-lockout rule

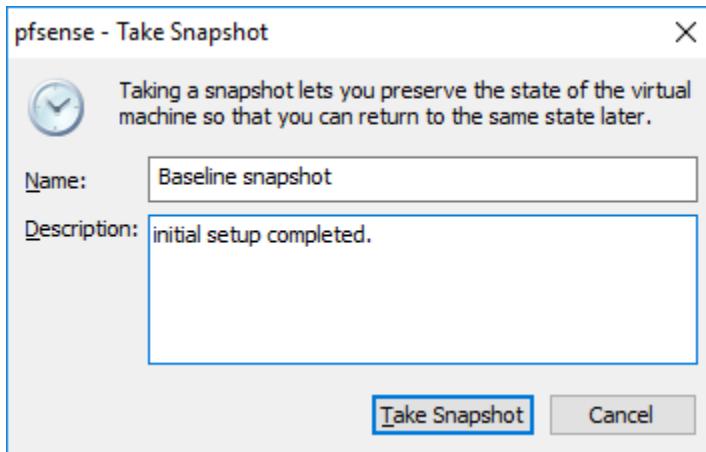
When this is unchecked, access to the webConfigurator on the LAN interface is always permitted, regardless of the user-defined firewall rule set. Check this box to disable this automatically added rule, so access to the webConfigurator is controlled by the user-defined firewall rules (ensure a firewall rule is in place that allows access, to avoid being locked out!) Hint: the “Set interface(s) IP address” option in the console menu resets this setting as well.

Take a Snapshot

Snapshotting is a VERY important concept with VMs. Snapshots let you restore your virtual machines to a point in time in the past when you took that snapshot. This lets you undo misconfigurations, problems, or potential issues that affect the VM relatively easily. We’re going to take a snapshot of pfSense in its current state. In the VMware Workstation main screen, in the “Library” pane, right click on pfSense, click Snapshot > Take Snapshot...



A new window pops up, with an input box for the name of the snapshot, and another for describing the snapshot. I named the snapshot “baseline” and gave a brief description of what the snapshot is for. After you are finished, click the “Take Snapshot” button. VMware will begin generating a snapshot for your VM immediately, and should finish in moments.



If you ever have to restore a VM from a snapshot, you can right click on the VM again in the Library pane, and in the snapshot sub-menu you can revert to the most recent snapshot, or you can open the snapshot manager and choose a specific snapshot to revert to. Snapshot restoration will take a moment or two, and voila, your VM is restored to its previous state.

pfSense Summary

At the end of all this, you should have a pfSense VM with the following settings:

- 512MB of RAM
- 5GB of Disk
- 1 Virtual CPU

- 3 Network interfaces:
 - 1 Interface connected to the VMnet0 (Bridged) network (WAN interface)
 - 1 Interface connected to the VMnet1(Management) network (LAN interface)
 - 1 Interface connected to VMnet2 (IPS 1) network (OPT1 interface)

pfSense should be configured as followed:

- The WAN interface should be the network adapter connected to the Bridged VMnet. This interface should automatically get an IP address via DHCP. If this isn't happening, get some help with whoever administers your local network, or start troubleshooting.

Alternatively, configure a static IP address.

- The LAN interface should have an IP address of 172.16.1.1.
- The LAN network should be 172.16.1.0 with a /24 (255.255.255.0) subnet mask
- DHCP should be configured for this network with a scope of 172.16.1.10-172.16.1.254
 - 172.16.1.3-172.16.1.9 are available for setting static DHCP mappings
- The OPT1 interface should have an IP address of 172.16.2.1
- The OPT1 network should be 172.16.2.0 with a /24 (255.255.255.0) subnet mask
- DHCP should be configured for this network with a scope of 172.16.2.10-172.16.2.254
 - 172.16.2.2-172.16.2.9 are available for static DHCP mappings.
- Your host OS accesses pfSense from the Management interface. You should have one virtual adapter connected to VMnet1 (What we will be using) and VMnet8 (which will NEVER be used. You can delete this interface OR, disable the interface in your host OS.). The VMnet1 adapter should have an IP address of 172.16.1.2. We created a firewall rule that serves as our anti-lock rule for allowing the host OS to access pfSense via the web interface.
- You should have an alias for RFC1918 networks configured for use with the firewall policy.
- You should have at least one good snapshot you can revert to if you need to redo a step and somehow got lost/confused.

What's Next?

Now that you have an operational pfSense VM, your next steps are to read [“Network Configuration - Segmentation”](#) guide, followed immediately by [“Network Configuration - Core Network Services”](#). Make sure the firewall policies match those illustrated in the segmentation guide, and make sure that the pfSense VM is hosting all the services mentioned in the core network services guide to make your firewall fully functional and ready to handle your lab network. Once you verify that your firewall policies and services are properly configured, **It is highly advised you create a new snapshot. *THEN* you can move on, and try your hand at setting up the remaining VMs.**

Your Turn

The pfSense vm required a lot of custom configuration, both in VMware Workstation, as well as in the VM itself once the OS was installed. Now that we did that together, I'm going to have you create the rest of the virtual machines yourself -- with the exception of the metasploitable 2 VM since it's a special case for reasons you will see momentarily. Below are a set of spec sheets to help you create the remaining virtual machines for the lab on your own. Think of it as a checklist you should be able to run through on your own.

All of the linux-based VMs in our lab are based on Ubuntu Server (Except for Kali Linux, which is based on the Debian Linux distro, which was the precursor to Ubuntu) so set up and configuration on our lab VMs should be mostly the same.

Kali Linux VM

Kali Linux is a popular penetration testing distro. Popular because hackers and script kiddies are lazy, and practically everything you need for performing a penetration test or joining anonymous is installed by default. Kali is going to be our loud, obnoxious attacker that we're going to use as a noise generator for the express purpose of generating events on our IPS VM. I want you to perform the following tasks:

- Download Kali Linux 64-bit here: <https://www.kali.org/downloads/>
- Create a VM with the following settings:
 - Configure the VM to install from the Kali Linux ISO you downloaded
 - Set the Guest OS family to "Linux"
 - Set the Guest OS version to "Debian 8.x 64-bit"
 - Name the vm "kali"
 - Allocate 80 GB of space
 - Choose the "store virtual disk as a single file" option
- Customize the following hardware settings
 - Allocate 4GB (4096MB) of RAM
 - Allocate 1-2 processors, where total number of cores is no more than 2
 - Connect the VM's network adapter to VMnet2
 - Ensure that "Connect at power on" is checked
 - Remove the USB Controller
 - Remove the Sound Card
 - Remove the Printer
- Install Kali Linux
- After the install is completed, power off the VM and adjust the following settings:
 - Remove the CD/DVD drive
 - Make sure to document the MAC address under advanced settings for the network adapter.
- Log in to the pfSense web UI, navigate to Services > DHCP Server, and add a static mapping to the OPT1 interface DHCP, for the MAC address of adapter 1 for the Kali VM; assign it the IP address 172.16.2.2, making sure to enter a description that tells you which VM this static mapping belongs to!

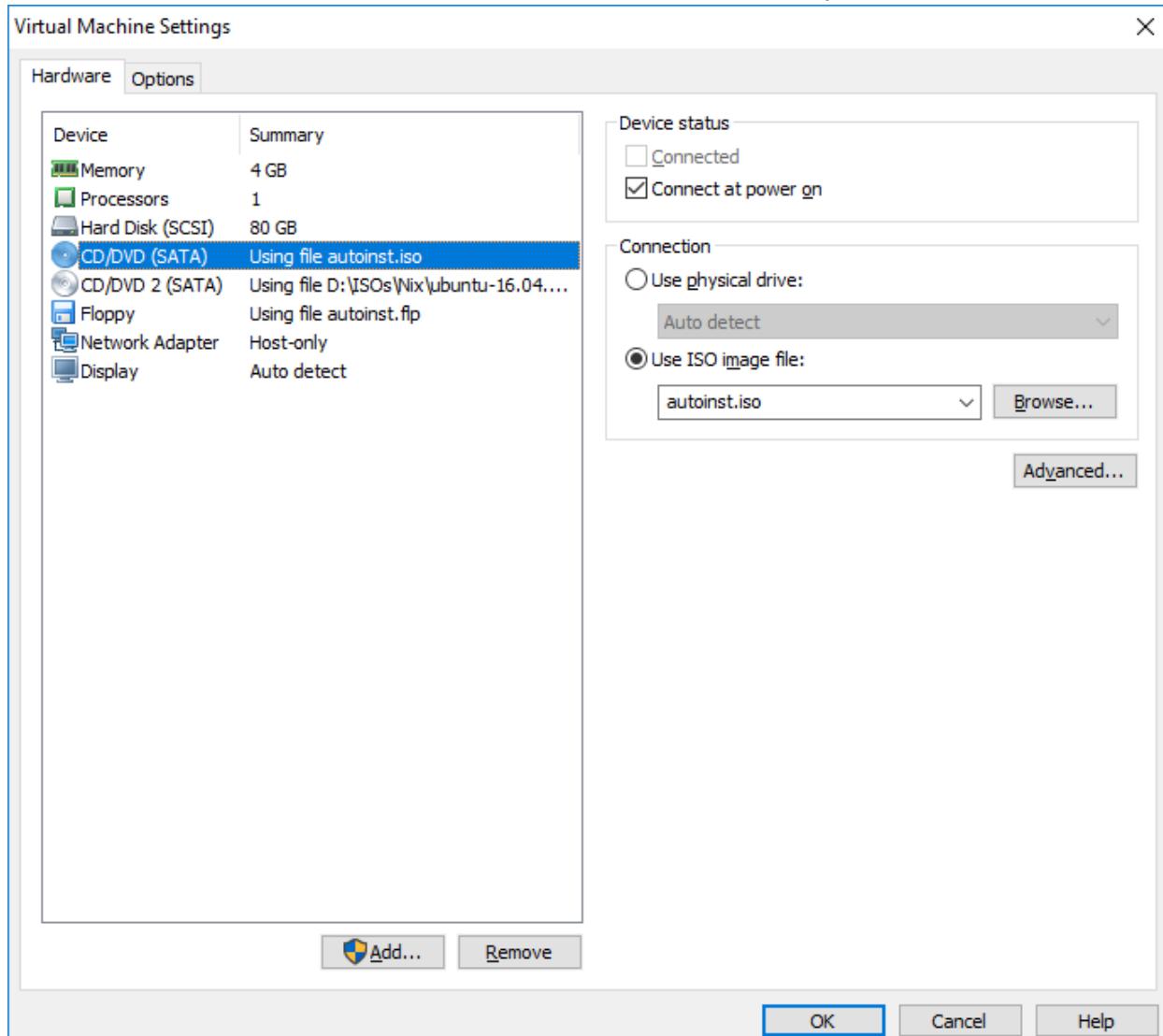
- Power on the Virtual Machine and do the following:
 - Verify that the static mapping for the VM's IP address worked; Open a terminal/shell and run ifconfig -a | less and verify that "eth0" or whatever the network adapter's name is in the VM was given the IP address you configured its static mapping for in pfSense.
 - If the VM does not have an ip address, run the "dhclient" command to have the VM request an IP address from the DHCP server.
 - Verify the virtual machine can reach the internet.
 - In a terminal/shell run the command ping -c 5 www.google.com
 - This will confirm DNS can resolve domain names, and that the VM can reach the internet
 - Update/patch the VM
 - In a terminal/shell, run the following, and depending on your network connection, etc., be prepared to wait:
 - **export DEBIAN_FRONTEND=noninteractive; apt-get -q update; apt-get -y -q dist-upgrade**
 - Create a snapshot for the VM

Siem VM

SIEM is shorthand for Security Intrusion Events Manager. This is fancy security nomenclature for "log aggregator"; we will be having our IPS VM log its events here. We're going to be running Splunk on this VM. Splunk is a commercial program for managing logs on a pretty large scale. By default, and with no licensing, Splunk only allows you to collect or "index" 500MB worth of logs per day however, there are /ways/ around this that we'll talk about later. I want you perform the following tasks to set up the SIEM VM:

- Download Ubuntu Server 16.04.1 64-bit here: <https://www.ubuntu.com/download/server>
- Create a VM with the following settings:
 - Configure the VM to install from the Ubuntu Server ISO you downloaded
 - Ubuntu supports VMware's "Easy Install" option, but we're not going to use it
 - Fill out whatever data you want in the full name, username, password and confirm fields.
 - Name the vm "siem"
 - Allocate 80 GB of space
 - Choose the "store virtual disk as a single file" option
- Customize the following hardware settings
 - Allocate 4GB (4096MB) of RAM
 - Allocate 1-2 processors, where total number of cores is no more than 2
 - Connect the VM's network adapter to "Host-only" (VMnet1)
 - Ensure that "Connect at power on" is checked
 - Remove the USB Controller
 - Remove the Sound Card
 - Remove the Printer
- **Before powering on the VM to install Ubuntu Server..**

- Enter the VM's Hardware settings. Remove the CD/DVD drive that has the file "autoinst.iso" in the virtual drive, as well as the Floppy drive, "autoinst.flp".



- Install Ubuntu Server, making sure to install “standard system utilities” and “OpenSSH server” during the Software selection phase
- After the install is completed, power off the VM and adjust the following settings:
 - Remove the remaining CD/DVD drive
 - Make sure to document the MAC address under advanced settings for the network adapter.
- Log in to the pfSense web UI, navigate to Services > DHCP Server, and add a static mapping to the LAN interface DHCP, for the MAC address of network adapter for the siem VM; assign it the IP address 172.16.1.3, making sure to enter a description that tells you which VM this static mapping belongs to!
- Power on the Virtual Machine and do the following:
 - Verify that the static mapping for the VM's IP address worked; Open a terminal/shell and run ifconfig -a | less and verify that “eth0” or whatever the

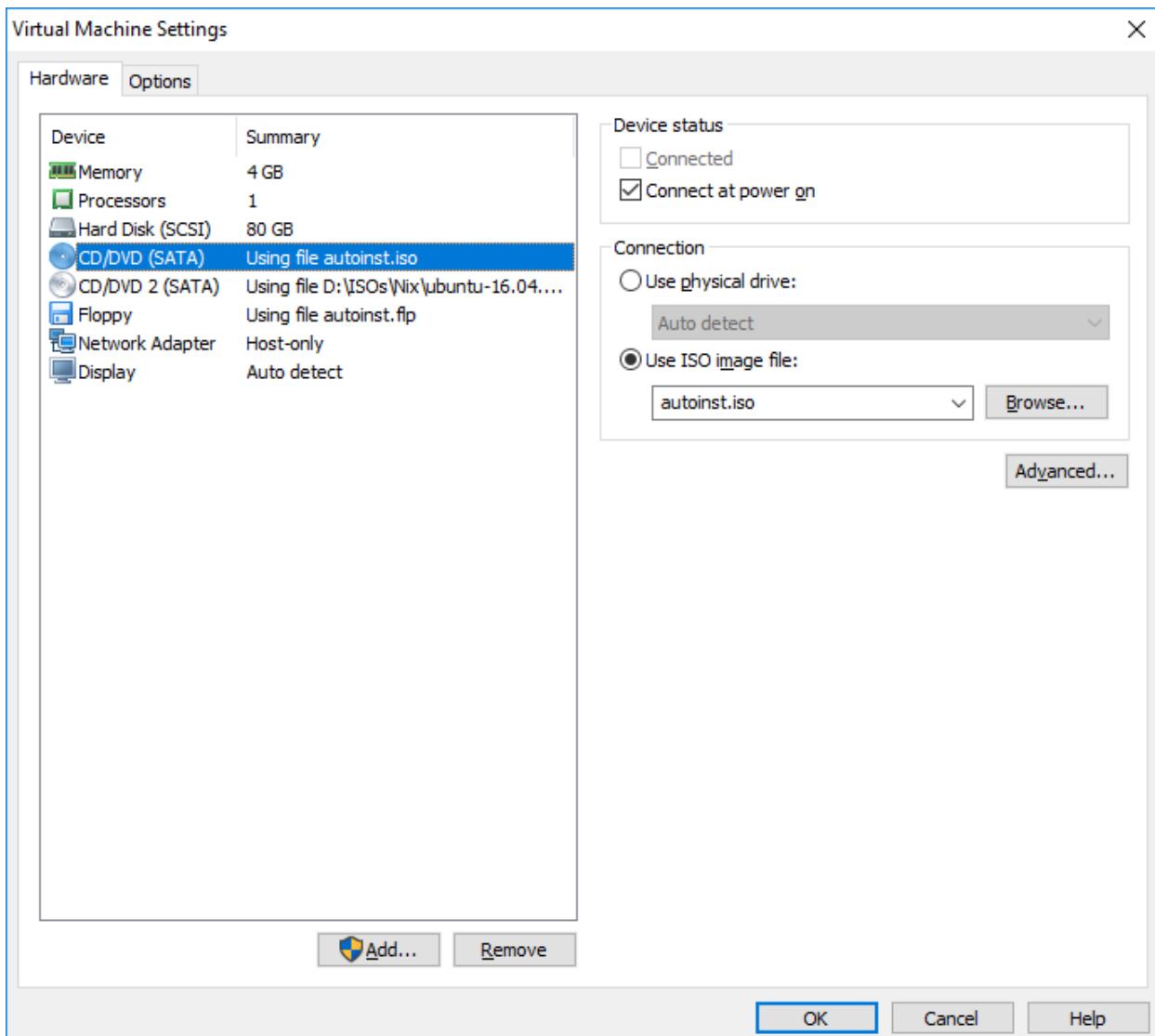
network adapter's name is in the VM was given the IP address you configured its static mapping for in pfSense.

- If the VM does not have an IP address, run the “dhclient” command to have the VM request an IP address from the DHCP server.
- Verify the virtual machine can reach the internet.
 - In a terminal/shell run the command ping -c 5 www.google.com
 - This will confirm DNS can resolve domain names, and that the VM can reach the internet
- Update/patch the VM
 - In a terminal/shell, run the following, and depending on your network connection, etc., be prepared to wait:
 - **export DEBIAN_FRONTEND=noninteractive; apt-get -q update; apt-get -y -q dist-upgrade**
- Create a snapshot for the VM

IPS VM

This VM is going to be responsible for running the AFPACKET bridge between the IPS 1 (VMnet2) and IPS 2 (VMnet 3) virtual networks. Perform the following tasks to install the IPS VM:

- If you haven't already for the siem VM, Download Ubuntu Server 16.04.1 64-bit here: <https://www.ubuntu.com/download/server>
- Create a VM with the following settings:
 - Configure the VM to install from the Ubuntu Server ISO you downloaded
 - Ubuntu supports VMware's “Easy Install” option, but we're not going to use it
 - Fill out whatever data you want in the full name, username, password and confirm fields.
 - Name the vm “siem”
 - Allocate 80 GB of space
 - Choose the “store virtual disk as a single file” option
- Customize the following hardware settings
 - Allocate 2GB (2048MB) of RAM
 - Allocate 1-2 processors, where total number of cores is no more than 2
 - Connect the VM's first network adapter to “Host-only” (VMnet1)
 - Ensure that “Connect at power on” is checked
 - Create two additional network adapters.
 - Attach Network Adapter 2 to VMnet2 (IPS 1)
 - Attach Network Adapter 3 to VMnet3 (IPS 2)
 - Remove the USB Controller
 - Remove the Sound Card
 - Remove the Printer
- **Before powering on the VM to install Ubuntu Server..**
 - Enter the VM's Hardware settings. Remove the CD/DVD drive that has the file “autoinst.iso” in the virtual drive, as well as the Floppy drive, “autoinst.flp”.



- Install Ubuntu Server, making sure to install “standard system utilities” and “OpenSSH server” during the Software selection phase
- After the install is completed, power off the VM and adjust the following settings:
 - Remove the remaining CD/DVD drive
 - Make sure to document the MAC address under advanced settings for all three network adapters, making sure to note which MAC address pairs up to which network adapter, connected to which VMnet.
- Log in to the pfSense web UI, navigate to Services > DHCP Server, and add a static mapping to the LAN interface DHCP, for the MAC address of network adapter connected to VMnet1 (Management network, Host-only VMnet); assign it the IP address 172.16.1.4, making sure to enter a description that tells you which VM this static mapping belongs to!
- Power on the Virtual Machine and do the following:
 - Verify that the static mapping for the VM’s IP address worked; Open a terminal/shell and run ifconfig -a | less and verify that “eth0” or whatever the

network adapter's name is in the VM was given the IP address you configured its static mapping for in pfSense.

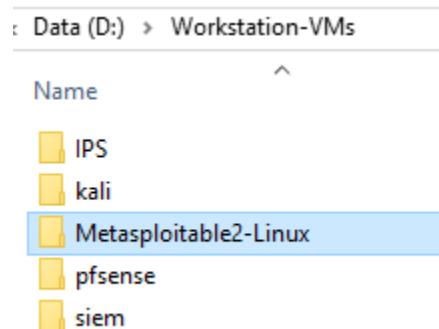
- If the VM does not have an IP address, run the “dhclient” command to have the VM request an IP address from the DHCP server.
- Verify the virtual machine can reach the internet.
 - In a terminal/shell run the command ping -c 5 www.google.com
 - This will confirm DNS can resolve domain names, and that the VM can reach the internet
- Update/patch the VM
 - In a terminal/shell, run the following, and depending on your network connection, etc., be prepared to wait:
 - **export DEBIAN_FRONTEND=noninteractive; apt-get -q update; apt-get -y -q dist-upgrade**
- Create a snapshot for the VM

Metasploitable 2

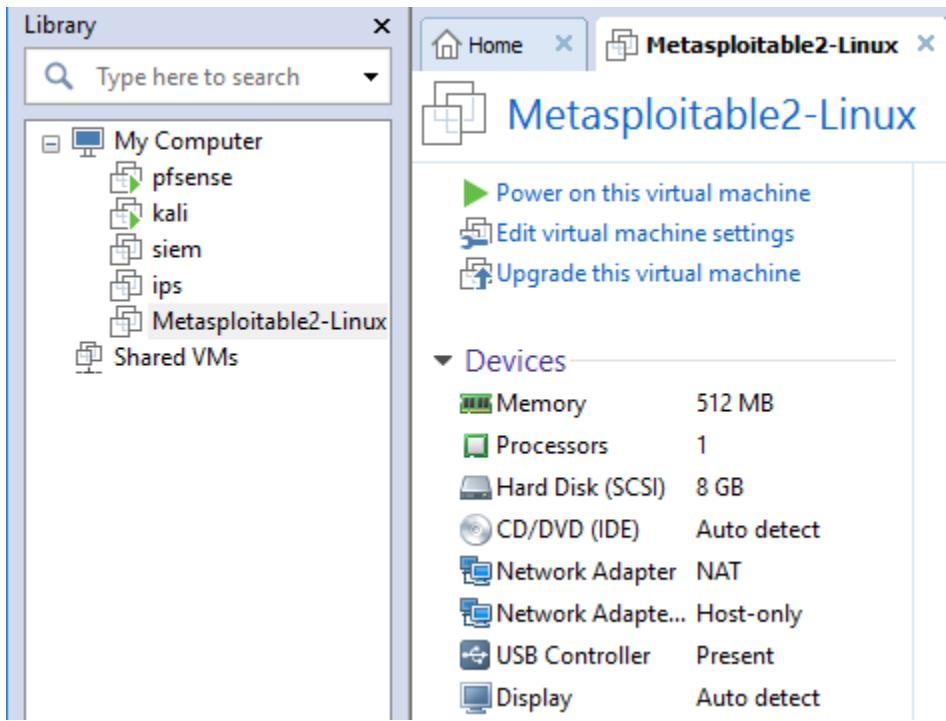
Metasploitable 2 is a little different from the other VMs. It is already pre-created, we just need to import it, and reconfigure some of the virtual hardware. This should be pretty simple to do. First, download a copy of metasploitable 2 from

<https://sourceforge.net/projects/metasploitable/files/Metasploitable2/>. Metasploitable 2 is distributed as a .zip archive so you'll need a compression tool to extract it. On Linux/Unix systems, you can use the command line utility unzip. On windows, I recommend 7-zip, which can be found at <http://www.7-zip.org/download.html>

Download the archive, and decompress it in the download directory. You will be left with a folder labeled “metasploitable-linux-2.0.0.0”. Inside of that folder is another folder labeled “Metasploitable2-Linux”. Move this entire directory to the folder the rest of your VMware Workstation folders are stored. In my case, this is D:\Workstation-VMs



In the VMware Workstation main window, on the menu bar, click File > Open... and Navigate to the Folder you just moved the Metasploitable2-Linux directory to. In my case. I navigated to D:\Workstation-Vms\Metasploitable2-Linux. Select the “Metasploitable.vmx” file in this directory. A new VM should appear in the “Library” pane in VMware Workstation.



There are a few more settings we need to adjust before starting the VM. Using what you know from setting up the previous VMs, perform the following actions in the settings menu, before attempting to boot the VM:

- Remove the USB Controller
- Remove Network Adapter 2
- Remove the CD/DVD drive
- Reconfigure The Network Adapter from “NAT” to “VMnet3” (IPS 2 Network)
 - Document the MAC address of this network interface
- Log in to the pfSense web UI, navigate to Services > DHCP Server, and add a static mapping to the LAN interface DHCP, for the MAC address of network adapter connected to VMnet1 (Management network, Host-only VMnet); assign it the IP address 172.16.2.3, making sure to enter a description that tells you which VM this static mapping belongs to!

Power on the metasploitable 2 VM, make sure it is bootable. If so, take a snapshot of the VM, it's ready to go for now. You won't be able to verify the DHCP mapping worked until AFTER the ips VM is configured.

So... What Now?

The VMware Workstation initial setup is all but done at this point. However, you're not quite out of the woods just yet you'll want to do the following:

- If you haven't completed the section "[Defense in Depth for Windows Hosts](#)", and you are using Windows as a hypervisor host, I would very highly suggest doing so.
- While not strictly necessary, you may want to complete "[Remote Lab Management](#)" section, for the Host OS of your choice (e.g. [Windows Remote Access](#), [Linux/Unix Remote Access](#)), [Enabling SSH on Kali Linux](#), and/or if you're lazy like me, The section on [Securing root SSH](#) access. This will allow you to remotely manage all of the linux VMs much more easily than through the VMware Workstation console.
- You have to install IDS/IPS software on the IPS VM. You'll need to complete the "IPS Setup Guide". To learn how to do this with either Snort or Suricata as your IPS software of choice.
- The SIEM VM needs to have splunk installed and configured. You'll need to complete the "Splunk Setup Guide".

Setup - VMware VSphere Hypervisor(ESXi)

Note: This guide was made using VMware vSphere Hypervisor 6.0, Update 2 with free licensing, and assumes that users of this guide are this software version or higher.

VMware VSphere Hypervisor (which from now on I am going to refer to it as "ESXi", its traditional name) is a baremetal hypervisor, as opposed to a hosted hypervisor, like the rest of the virtualization suites we've covered so far. This means that you need to have dedicated hardware to install ESXi, and a second workstation or laptop capable of managing the ESXi server. Configuration, installation, and setup are also a little different compared to hosted hypervisors. Read on, pay attention, and you'll come out of this just fine.

Installation

As mentioned in "[Hypervisor and Hardware Considerations](#)", ESXi is pretty picky about what hardware it will recognize and install on. And then, even if it supports your hardware, there may be unusual cases where all the features of the motherboard/chipset aren't supported fully. This isn't so much a problem if you are using commercial server solutions from a known hardware vendor (e.g. Dell or HP servers), but if you're trying to build your own server (also known as a "white box" server) you may have some problems. For instance, I built a server that had a supermicro motherboard that was supported by ESXi, but the intel RAID controller wasn't. The server would recognize hard drives, but not RAID arrays. Others have told me about experiences where their motherboard was supported, but the integrated ethernet cards were not, so they were required to buy PCI/PCI-e network cards in order to run ESXi.

Some people have managed to install ESXi on some pretty exotic hardware - shuttle PCs, intel NUCs, among other things. A lot of these builds require rebuilding the installation ISO using a technique called "slipstreaming" to ensure that drivers for specific chipsets are included in the ISO. Unfortunately, this is beyond the scope of this guide. I can't guarantee your success if you

try to install ESXi on exotic hardware, but if you want to build your own server (instead of buying a new or used commercial server), here are some things to consider:

- ESXi works really well with Intel chipsets and network cards. There's a good chance that if you have an Intel chipset on your motherboard and/or intel network cards (integrated or PCI cards), that you will do just fine.
- While most prefer Intel chipsets and network cards where possible, the next best solution is to utilize broadcom network cards. A lot of sysadmins I know dislike broadcom due to loose interpretations of various standards wherever they are involved, but support for broadcom NICs in ESXi is pretty decent, and you don't have the same requirements they do; this is a lab, not a production environment.
- VMware has a hardware compatibility list you can search here: <https://www.VMware.com/resources/compatibility/search.php>. But generally, the compatibility list is very cumbersome to use. A better solution would be to google search for "part/model number" "esxi" to see if there are any hits for compatibility problems for the server model you want to use or the server component you want to use.

So now that we got hardware compatibility out of the way, you need to actually download ESXi and install it. In order to do so, you need to create an account on VMware's website. After you have signed up, and logged in, Navigate to Downloads and find "VMware vSphere Hypervisor(ESXi)". VMware changes links around all the time, but for the time being, the page for ESXi can be found here: <https://my.VMware.com/en/web/VMware/evalcenter?p=free-esxi6>. On the download page, you will be presented with a free product key for the free version of ESXi. While there are some significant differences between ESXi free and enterprise versions, for our purposes, the free version and licensing will work just fine.

Make sure to download the latest version of ESXi (which is currently 6.0, Update 2), the latest version of VMware vsphere client (If you are using windows), and to document the product key for your ESXi installation.

License Information

COMPONENT	LICENSE KEYS
VMware vSphere Hypervisor 6 License	[REDACTED]

Download Packages

Your downloads are available below

VMware vSphere Hypervisor 6.0 Update 2 - Binaries

ESXi ISO image (Includes VMware Tools)
2016-03-15 | 6.0U2 | 357.95 MB | iso

Boot your server with this image in order to install or upgrade to ESXi (ESXi requires 64-bit capable servers). This ESXi image includes VMware Tools.

MD5SUM(): 7b85a48eb67e277186d2422ebd42f6b6
SHA1SUM(): 5a93f457980d18f7061c8b550c509682070cadc7
SHA256SUM(): b8eb47ef17bd5e7eee92bee6d0bbb95ab18d0ab48c3cc6322b67815da1c9fc44

Manually Download

VMware vSphere Client 6.0 Update 2
2016-03-15 | 6.0U2 | 348.81 MB | exe

Separate installer for the VMware vSphere Client. Note: vSphere Web Client can be installed using the vCenter Server installer

MD5SUM(): 8f491cb58d5d0e78e953f68978f2524a
SHA1SUM(): 06e30707c03fe617604756df0876fdbcb51a708c
SHA256SUM(): 710c05704d3593485fc328e19fb9445c7d5946ea1755b55a330259b12e5f8f5

Manually Download

So now that you have an ISO, you need to figure out how you're going to install ESXi on your server. Traditionally, you would use a CD Burning utility to burn the ISO to a CD or DVD, and simply boot from the CD. However, many newer PCs and/or servers are opting for no optical drives, but instead allow you to boot from a USB drive. You'll have to get a little bit creative and use a utility to make a bootable USB drive. I found a great guide on how to create a USB installer for ESXi here: <http://www.virten.net/2014/12/howto-create-a-bootable-esxi-installer-usb-flash-drive/>

So now that you have a bootable USB drive, you have some choices as to where you want to install ESXi. IT professionals have been found to install ESXi on external media like USB drives or even SD cards. The reason for this is that ESXi itself only requires around 4 GB of space to install properly, with most of the components for ESXi residing in memory, so a flash drive or SD card more than serves the purpose. Installing on external media saves space on the drives you'll be using to store virtual machines, and also ensures that the Virtual Machines you run won't be contending for disk I/O against ESXi itself.. If you don't have a spare SD card or USB drive laying around, don't worry about this too much, just bear in mind that this is an installation option available, and that a lot of sysadmins do this. Also bear in mind that **If you do decide to install to an SD card, or USB drive, system logging will be disabled by default; ESXi will NOT write logs to a USB drive or SD card by default.** To enable logging, you can configure ESXi to use a datastore disk (where VMs are stored) to write logs (in the advanced host settings), or you can specify the IP address of a syslog server to send its logs to.

The installation itself is pretty straightforward. During the install you will be asked to set the root password for the device, create datastores on attached hard drives for actually storing your VMs and their files, as well as configure the management interface (e.g. the network interface and IP address you will be using to connect to this device). If at all possible, you should consider setting a static IP address for your ESXi host, or a static DHCP mapping to ensure that your ESXi server always has the same IP address in the event the system reboots. Spending a portion of your day to figure out that DHCP granted your ESXi server a new IP address is pretty annoying.

Accessing ESXi

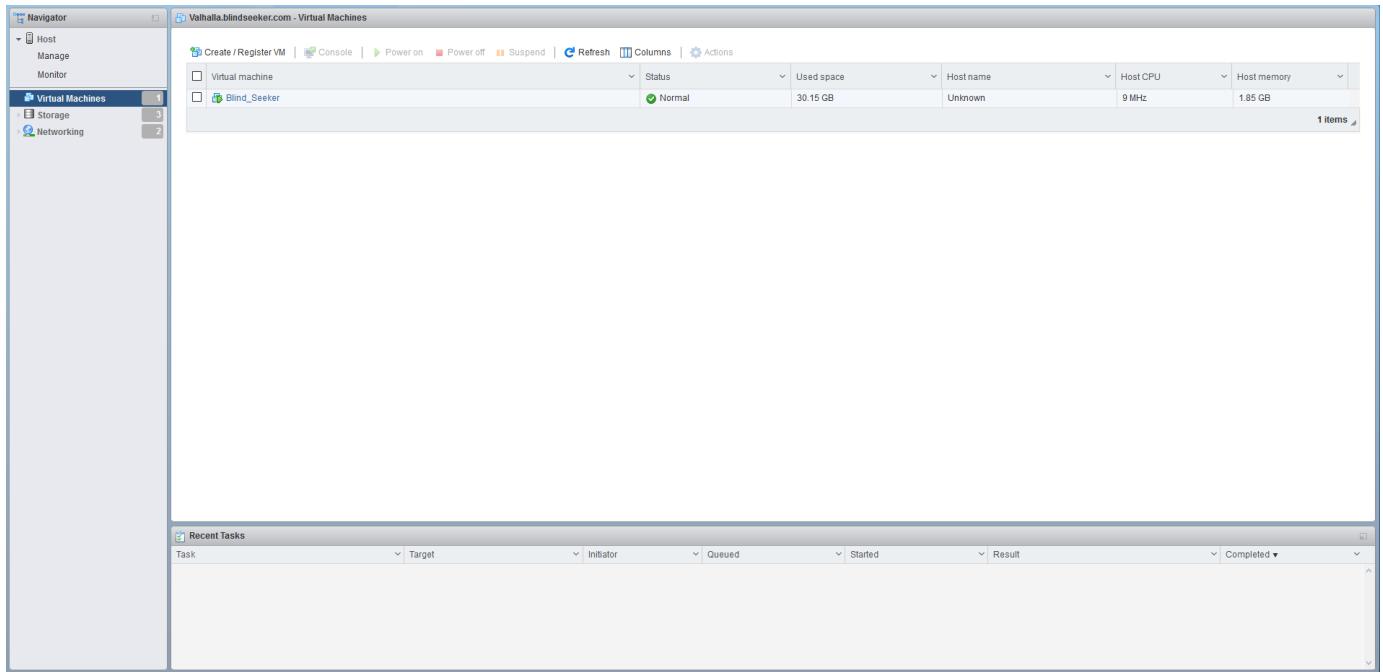
Once you have ESXi installed, and can confirm that it is reachable over the network (e.g. ping), you should be able to perform administrative tasks over the network. If you're using a windows host, you have the option of utilizing the VMware Infrastructure Client to access your ESXi server. Simply plug in the IP address of your ESXi host, enter a username and password (in this case, you want to log in as the root user you created during installation)



If you are not using windows, or you would rather not use the vSphere client, you can point your web browser to https://[server ip]/ui. And you can log in to the HTML5-based web interface for VMware ESXi. The web interface can be used on Windows, OS X, or Linux, provided you have something resembling a modern web browser able to understand HTML5 content. For the sake of simplicity, **we will be using the HTML5 web interface to interact with the ESXi server and perform all of our configuration, except where noted and/or if workarounds are required.** Please note that the HTML5 web interface is still relatively new, and prone to bugs here and there.



When you successfully login, you should be greeted with a dashboard that shows you a nice list of VMs installed on your ESXi instance. In your case, you shouldn't have any installed yet. In my case, since I'm literally running a web server out of my basement, my first and only VM on the server, BlindSeeker is listed.



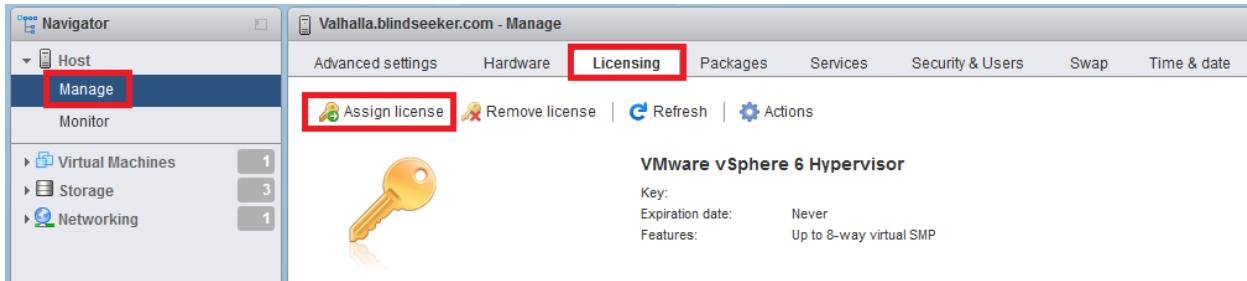
Get use to this interface. Know it and love it, since we'll be spending a lot of time here. The window pane on the left, entitled "Navigator" is going to be your best friend. As the name implies, you use it to navigate the various settings available to you for administering the ESXi server. From checking resource utilization, to creating VMs, to enabling/disabling features on the ESXi host, the navigator pane is the key to getting around.

Hypervisor Setup

ESXi is a little different from our hosted hypervisors. We need to perform some initial configuration and post-installation tasks in order to make it ready for us to get started.

Licensing

On the Navigator pane, under "Host", there is an option called "Manage". When you click on it, the middle pane will have a series of tabs you can use to Manage your ESXi host. Click on the "License" Tab, then click "Assign License". Input the key we got from the VMware website earlier. By default, ESXi will give you the full featureset of ESXi for 60 days, but then you need a license. Its best to get that taken care of up front, rather than letting it become a problem later. If your key is valid and has been entered successfully, your Licensing page should look like this. Please note that I intentionally blanked out my license key on this illustration.

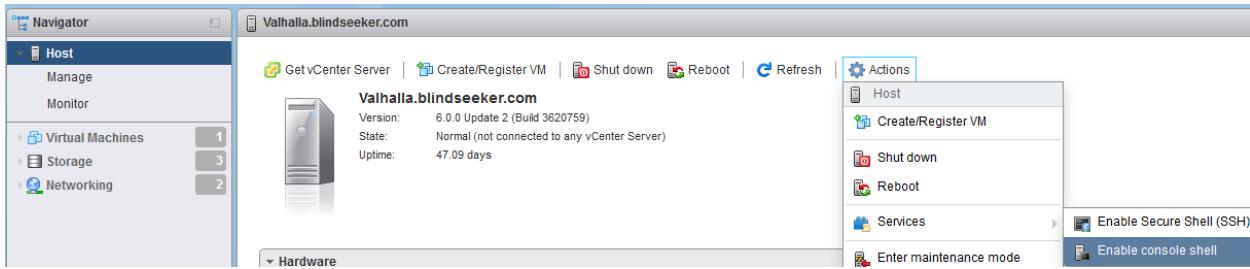


Resolving Some Interface Bugs

I noticed while I was making this guide, that there are some bugs and issues with the HTML5-based ESXi web interface. Specifically for our purposes, there are some significant bugs with configuring port groups which will become very important to us momentarily. Please note that as of this writing, these interface bugs appear to affect the entire ESXi 6.X line (up to and including ESXi 6.0 Update 2, the current ESXi release that you should be using). Fortunately, there are workarounds available, especially if you have a Windows installation available with the vSphere client installed. For the sake of keeping your options open, I'll document workarounds to web interface issues I had to use initially, but I'm also going to show you how to update the ESXi embedded web interface using VMware labs "flings" to fix the problem. Please be aware that software versions, website designs and download links may be subject to change. Always keep an eye out. Special thanks to @JohnPittman for pointing me towards this solution.

Flings are experimental offerings that VMware makes available for system administrators and virtualization enthusiasts to try out before they get accepted as official offerings or mainline additions to VMware products. If you try to download anything on flings, they make you click a checkbox that says you agree to not install these things on production systems or hold VMware accountable if something terrible happens. In a nutshell, flings are the equivalent of doing public beta testing of new features and applications. I'm going to walk you through installing the latest "ESXi Embedded Host Client" fling.

First and foremost, you'll need to enable access to the esxi shell. By default, VMware disables access to the command line shell because more often than not, your average user doesn't really have a compelling reason to be mucking around in there, unless they're working with VMware support, or doing strange things to the OS (kinda like what we're about to do) so from a security standpoint, it makes sense. No need to worry however, enabling this access is trivial. From the web interface click on "Host" under Navigator. In the middle pane, click on "Actions" and a menu appears. Click on "Services", then click on "Enable console shell".



This enables access to the shell interface. If you planned on connecting a mouse and keyboard or were going to use a lights-out management system or IPMI viewer application of some sort, this is the only service you need to enable. Otherwise, you may have noticed that the other option under services is “Enable Secure Shell (SSH)”. In addition to enabling the console, you will need to click this to start the SSH service on your ESXi host on port 22. From here, you can use your favorite ssh client combined with the root username and password to connect to your ESXi server over SSH. If your SSH login is successful, and console access is enabled, you should be greeted with this prompt:

```
Using username "root".
The time and date of this login have been sent to the system logs.

VMware offers supported, powerful system administration tools. Please
see www.vmware.com/go/sysadmintools for details.

The ESXi Shell can be disabled by an administrative user. See the
vSphere Security documentation for more information.
[root@Valhalla:~] 
```

Otherwise, if you prefer to use keyboard and mouse, LOM, or IPMI, when you get access to the ESXi console, hit alt+f1 to gain access to the shell login prompt. If you need to regain access to the ESXi console main menu, hit alt+f2. If console access is enabled, you’ll be greeted with a login prompt:



So now that we have CLI access enabled, you’ll want to go to <https://labs.VMware.com/flings/esxi-embedded-host-client> and verify the current version and filename we will be installing. In my case, the filename is “esxui-signed-4393350.vib and version 1.9.1.



September 16, 2016

v1.9.1

- I have read and agree to the
[Technical Preview License](#)
I also understand that Flings
are experimental and should
not be run on production
systems.

esxui-signed-4393350.vib

Now that you confirmed the latest version of the esxui package, be sure to copy the download here link listed on the page. In my case, the download link was <http://download3.VMware.com/software/vmw-tools/esxui/esxui-signed-4393350.vib>. Copy it to your clipboard, or to a text editor of your choice, and go back to your ESXi console or SSH session.

Download the VIB [here](#)

- [Open Link in New Tab](#)
- [Open Link in New Window](#)
- [Open Link in New Private Window](#)
- [Bookmark This Link](#)
- [Save Link As...](#)
- [Save Link to Pocket](#)
- [Copy Link Location](#)

Now, we need to run the following command:

```
esxcli software vib install -v [url here]
```

In my case, the full command is:

```
esxcli software vib install -v http://download3.VMware.com/software/vmw-tools/esxui/esxui-signed-4393350.vib
```

```
[root@Valhalla:] esxcli software vib update -v http://download3.vmware.com/software/vmw-tools/esxui/esxui-signed-4393350.vib
```

If the command completed successfully, you'll get a pair of single quotes (") and the prompt will return to you. If you get any errors following some basic troubleshooting steps -- make sure that the ESXi server has a DNS server and default gateway configured, its allowed to use HTTP outbound, if there is a proxy installed that it is configured to use the proxy, etc. You may need to use the offline installation package and/or use some method of copying the file to the server. If you enabled SSH, then SCP is going to be the easiest way to get the installation package to the server. If you need further guidance on installing the esxui package, go here: <https://labs.VMware.com/flings/esxi-embedded-host-client#instructions>.

If you get the following error:

```
[root@Valhalla:] esxcli software vib update -v http://download3.vmware.com/software/vmw-tools/esxui/esxui-signed-4393350.vib
Installation Result
  Message: Host is not changed.
  Reboot Required: false
  VIBs Installed:
  VIBs Renoved:
  VIBs Skipped: VMware_bootbank_esx-ui_1.9.1-4393350
```

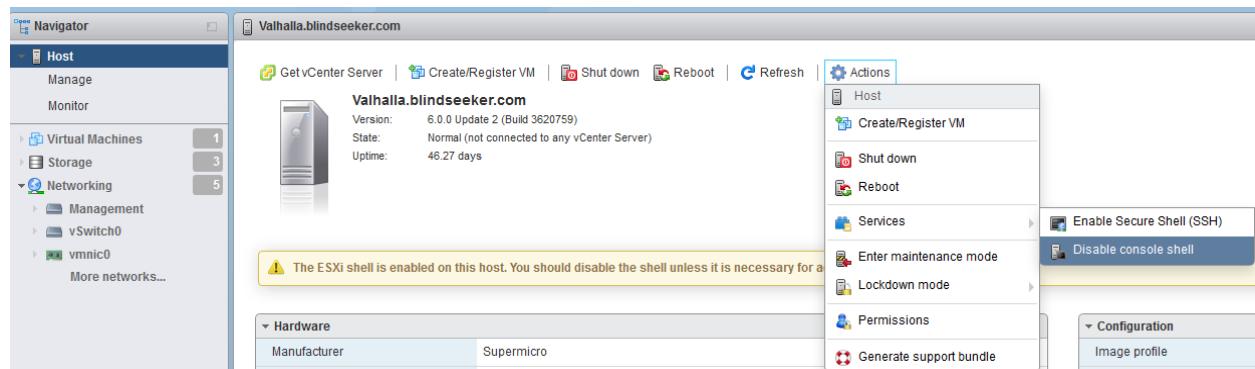
This means that the package has already been installed. You can confirm that the package has been successfully installed by running the following command:

```
esxcli software vib list | less
```

Look for the package esx-ui, and verify it matches the version from the VMware labs site we just downloaded.

Name	Version	Vendor	Acceptance Level	Install Date
nt_ip32xx-native	3.8.5-1vmw.600.0.0.2494585	VMWARE	VMwareCertified	2016-08-02
ata-pata-and	0.3.10-3vmw.600.0.0.2494585	VMware	VMwareCertified	2016-08-02
ata-pata-atixp	0.4.6-4vmw.600.0.0.2494585	VMware	VMwareCertified	2016-08-02
ata-pata-cmd64x	0.2.5-3vmw.600.0.0.2494585	VMware	VMwareCertified	2016-08-02
ata-pata-hpt32n	0.3.4-3vmw.600.0.0.2494585	VMware	VMwareCertified	2016-08-02
ata-pata-pdc2027x	1.0-3vmw.600.0.0.2494585	VMware	VMwareCertified	2016-08-02
ata-pata-serverworks	0.4.3-3vmw.600.0.0.2494585	VMware	VMwareCertified	2016-08-02
ata-pata-si1680	0.4.8-3vmw.600.0.0.2494585	VMware	VMwareCertified	2016-08-02
ata-pata-via	0.3.3-2vmw.600.0.0.2494585	VMware	VMwareCertified	2016-08-02
block-cciss	3.6.14-10vmw.600.0.0.2494585	VMware	VMwareCertified	2016-08-02
cpu-microcode	6.0.0-0.0.2494585	VMware	VMwareCertified	2016-08-02
ehci-ehci-hcd	1.0-3vmw.600.2.34.3620759	VMware	VMwareCertified	2016-08-02
elxnet	10.2.309.6v-1vmw.600.0.0.2494585	VMware	VMwareCertified	2016-08-02
emulex-esx-elxnetcli	10.2.309.6v-0.0.2494585	VMware	VMwareCertified	2016-08-02
esx-base	6.0.0-2.34.3620759	VMware	VMwareCertified	2016-08-02
esx-dvfilter-generic-fastpath	6.0.0-0.0.2494585	VMware	VMwareCertified	2016-08-02
esx-thrott	6.0.0-2.34.3620759	VMware	VMwareCertified	2016-08-02
esx-ui	1.9.1-4393350	VMware	VMwareCertified	2016-10-04
esx-xserver	6.0.0-0.0.2494585	VMware	VMwareCertified	2016-08-02

Once you have verified the package has been installed, exit your console session. You'll also have to logout and log back into your web UI session as well. After logging back in, navigate back to the Host > Actions > Services and disable the console and/or SSH services as necessary. **The console and/or SSH services should not remain enabled.**



Networking and Virtual Switches

ESXi's Virtual Networking is based off of creating virtual switches, and then assigning port groups that the VMs can use on those virtual switches.. There are no NAT networks, host-only networks, or internal networks here; just virtual switches and port groups. To provide access to a physical network, the virtual switch can be uplinked to a physical network port on the ESXi server to serve the same purpose as a bridged network, or can be stand-alone to serve the same purpose as a private or host-only network. For us to do network for our ESXi server the *right way*, you need to have at least two network cards on your ESXi server that are connected to your physical network. One of the network cards/ports is reserved by the ESXi host's "vmkernel" for managing the ESXi server itself, while the other network card will be dedicated to network traffic to and from our VMs.

To verify that your network cards/ports have been recognized by ESXi properly, under Navigator, click “Networking”, then click the “Physical NICs” tab. ESXi will display the status of each connected network card including whether or not the cable is plugged in and/or what link speed the network card has negotiated to. If your network card(s) are NOT showing up here, ESXi does NOT have drivers for them or they are not supported. In my case, I have four network ports (vmnic0-vmnic3), two of which are using the e1000e driver (Intel), and two of which are using the bnx (broadcom) driver. Additionally you can see that two of the network ports are physically connected and have a link speed of 1gbps.

Name	Driver	MAC address	Auto-negotiate	Link speed
vmnic0	e1000e	00:25:90:79:43:86	Enabled	100Mbps, full duplex Link down
vmnic1	bnx2	00:10:18:69:cff8	Enabled	Link down
vmnic2	bnx2	00:10:18:69:cffa	Enabled	Link down
vmnic3	e1000e	00:25:90:79:43:87	Enabled	100Mbps, full duplex

Don't have two network cards? Don't fret. there are ways to force the vmkernel and virtual machines to use the same network card/port. Though I must state, that this goes against best practice, is not technically supported, and may not be an option in the future. I'll show you how to do it anyhow, because this is a lab environment and you shouldn't be hosting anything of great importance here anyhow, right? Right. Now, let's create some virtual switches.

Creating Virtual Switches

Under the Navigator pane, click on “Networking”, then click on the “Virtual Switches” tab. You'll notice vswitch0 has been created. This is the vswitch that ESXi creates to attach the vmkernel to the management interface, and we won't be using it. If you were to click “Edit Settings” for this vswitch, you would see that this vswitch is uplinked to the network card you choose as your management interface when you configured networking during the initial ESXi installation. Click the “Add standard virtual switch” option.

Name	Port groups
vSwitch0	1

Name the first Virtual Switch “Management”. Click the “X” in the “Uplink 1” field to remove the drop-down. We do NOT want this virtual switch to have an uplink. Click the Add button to create this vswitch.

Add standard virtual switch - Management

Add uplink

vSwitch Name	Management
MTU	1500
► Security	Click to expand

Add Cancel

Next, click the “Add standard virtual switch” option again. Name this virtual switch “IPS 1”. Same as with the management virtual switch, remove the “Uplink 1” field by clicking the “X” next to the drop-down. Click the triangle next to “Security” to expand the security settings for the virtual switch. Click the “Accept” radio button for all three settings - Promiscuous mode, MAC address changes, and Forged transmits.

Add standard virtual switch - IPS 1

Add uplink

vSwitch Name	IPS 1
MTU	1500
▼ Security	
Promiscuous mode	<input checked="" type="radio"/> Accept <input type="radio"/> Reject
MAC address changes	<input checked="" type="radio"/> Accept <input type="radio"/> Reject
Forged transmits	<input checked="" type="radio"/> Accept <input type="radio"/> Reject

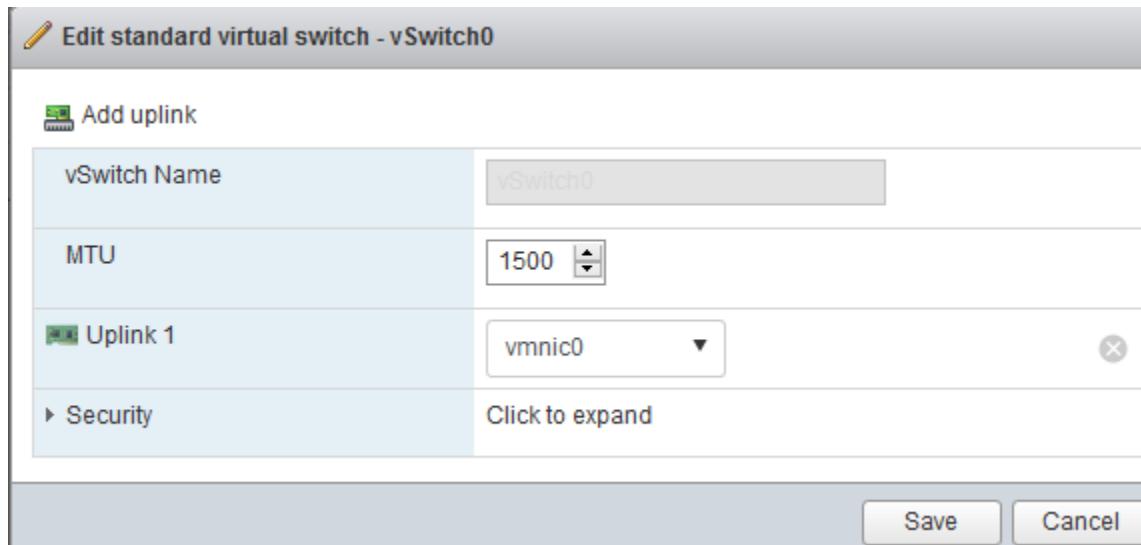
Add Cancel

Add another virtual switch with the exact same settings (all of the Security radio buttons set to “Accept”), and name it IPS 2.

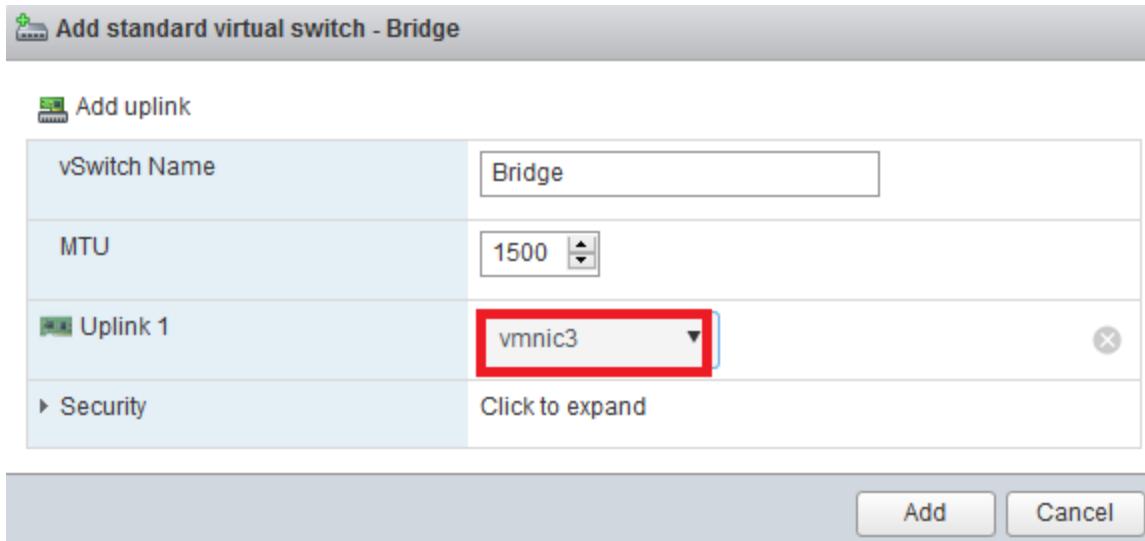
We need to create one more virtual switch, This one is going to bridged to one of the host’s network cards/ports. Since one network port is already uplinked to vSwitch0, you need to

ensure that you have at least one additional network connection available, and physically connected. If you do not have a second network card or port available, please visit the section “Adding a Virtual Machine Port Group to vSwitch0 - No Secondary NIC Available” for further instructions.

First, we need to confirm which network card vSwitch0 is uplinked to. In the Virtual Switches tab, click on “vSwitch0”. This brings you to the settings for the first vswitch. Click on “Edit Settings”. Take notice of the network interface listed in the “Uplink 1” field. Click cancel to exit.



Remember how I had you review which physical NICs were connected under the Physical NICs tab? Your physical NICs listing may be different, but in my case, vmnic0 and vmnic3 were connected to my physical network. Thanks to the illustration above, we know that vmnic0 is being used by vswitch0. Navigate back to Network > Virtual switches and click “Add standard virtual switch”. Name this virtual switch “Bridge”. This time, we’re going to set “Uplink 1” to another vmnic interface that is physically connected to your physical network. In my case, I am using vmnic3. Click the Add button.



Back in the Virtual switches tab, you should have a list of vswitches that looks something like this:

Name	Port groups	Uplinks	Type
vSwitch0	1	1	Standard vSwitch
Management	0	0	Standard vSwitch
IPS 1	0	0	Standard vSwitch
IPS 2	0	0	Standard vSwitch
Bridge	0	1	Standard vSwitch

If you do not have a secondary NIC available, please check out the section “Using the Windows vSphere Client to work around ESXi Web Interface Bugs” to learn how you can work around this.

Port Groups

Port groups are required to allow a VM’s network interfaces to be attached to a virtual switch. Port group also allow you to logically separate groups of virtual machines connected to the same virtual switch using VLANs. For example, you could have a port group assigned to an “engineering” VLAN, and another port group assigned to a “Human Resources VLAN”. They reside on the same ESXi server, use the same vswitch, but are on different VLANs and are thus segmented. There’s more to segmentation than that, but... that is outside the scope of this document.

Depending on whether or not you installed the esxui fling from “Resolving Some Interface Bugs” earlier, you may need a Windows system with vSphere client installed to work around some web interface issues. There is currently a bug where the “Port groups” tab under Networking will not display properly during initial setup, not without at least one virtual machine port group defined. If you did not download vSphere client from the VMware website, you can usually download this from the ESXi server by opening up your web browser and pointing it to the

management IP address of your ESXi host. For example, if your ESXi server's IP address is 192.168.1.20, point your web browser to <https://192.168.1.20>, then click on the "Download vSphere client for Windows" link to download the client we will be using.

VMware ESXi
Welcome

Getting Started

If you need to access this host remotely, use the following program to install vSphere Client software. After running the installer, start the client and log in to this host.

- [Download vSphere Client for Windows](#)
- [Open the VMware Host Client](#)

Adding Port Groups via the ESX Web Interface

Provided you installed the esxui VIB package earlier, creating port groups is easy. Under the Navigator pane on the web interface, Click on "Networking", then click on the "Port groups" tab, Then click "Add port group"

Name	Active ports	VLAN ID	Type	vSwitch
Management Network	1	0	Standard port group	vSwitch0

On the window that pops up, name the port group "Bridged" and in the "Virtual Switch" drop-down select "Bridge". **If you only have a single network port available to ESXi, and you installed the esxui fling, you can use vSwitch0 as your switch for the "Bridged" port group.** When you are done, click the Add button.

Add port group - Bridged

Name	Bridged
VLAN ID	0
Virtual switch	Bridge
▶ Security	Click to expand

Add **Cancel**

You should be back at the “Port groups” tab. Repeat this process three more times, creating a port group for the Management, IPS 1 and IPS 2 vswitches. Your “Port groups” list should look like this when completed:

Name	Active ports	VLAN ID	Type	vSwitch
Management Network	1	0	Standard port group	vSwitch0
Bridged	0	0	Standard port group	Bridge
Management	0	0	Standard port group	Management
IPS 1	0	0	Standard port group	IPS 1
IPS 2	0	0	Standard port group	IPS 2

Using the Windows vSphere Client to work around ESXi Web Interface Bugs

If you’re not comfortable installing vib packages or mucking around the command line in ESXi, using a Windows system, with the vSphere client, its possible to create an initial virtual machine port group, then create additional port groups in the ESXi web interface. Open the Windows vSphere client, and input the IP address of your ESXi server, username (root) and password, then click login. Note that the client will likely give you an SSL cert warning if you are using the default SSL cert for ESXi. Accept the warning and allow the connection.



Once you have successfully logged in, you should be dumped into an interface with a series of tabs for controlling the ESXi host. Click on the “Configuration” tab.

File Edit View Inventory Administration Plug-ins Help

Home > Inventory > Inventory

192.168.1.7

Valhalla.blindseeker.com VMware ESXi, 6.0.0, 3620759

Getting Started Summary Virtual Machines Resource Allocation Performance Configuration Users Events Permissions close tab X

What is a Host?

A host is a computer that uses virtualization software, such as ESX or ESXi, to run virtual machines. Hosts provide the CPU and memory resources that virtual machines use and give virtual machines access to storage and network connectivity.

You can add a virtual machine to a host by creating a new one or by deploying a virtual appliance.

The easiest way to add a virtual machine is to deploy a virtual appliance. A virtual appliance is a pre-built virtual machine with an operating system and software already installed. A new virtual machine will need an operating system installed on it, such as Windows or Linux.

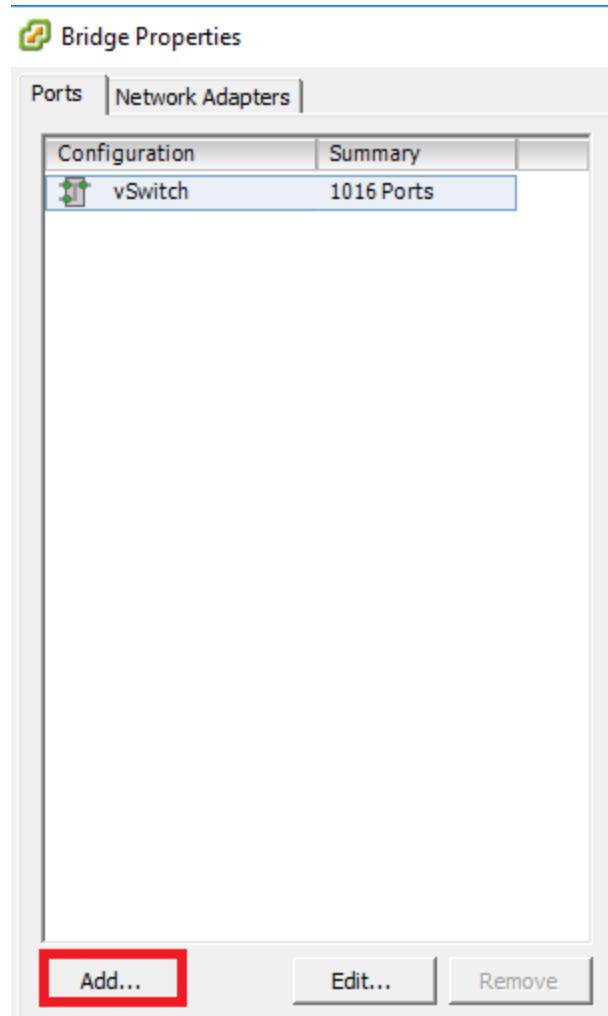
Virtual Machines
Host
vSphere Client

On the Configuration tab, you'll see a pane on the left labeled "Hardware". Click on Networking. This has a list of all the virtual switches and their logical network configuration. We will be creating our initial port group on the "Bridge" virtual switch. Click on "Properties..." to the right of "Standard Switch: Bridge". **If you don't have a Bridge vswitch due to not having a second network card available to dedicate to VM traffic, use vSwitch0 to create your initial port group instead.**

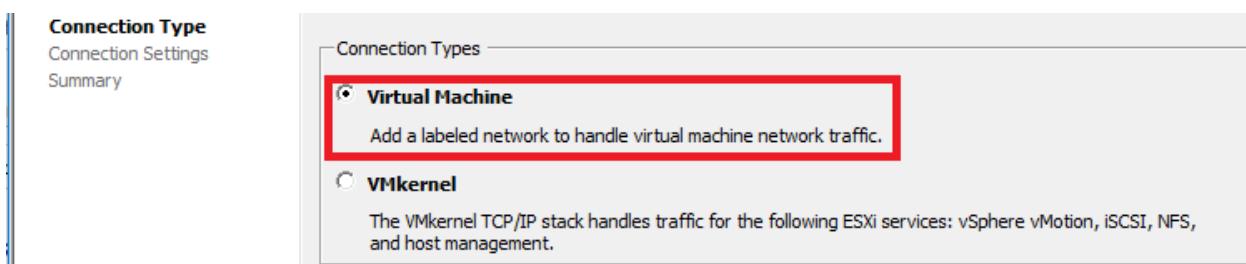
The screenshot shows the vSphere Configuration interface with the following details:

- Hardware Tab:** Selected. Submenu: Health Status, Processors, Memory, Storage, Networking (selected), Storage Adapters, Network Adapters, Advanced Settings, Power Management.
- Software Tab:** Selected. Submenu: Licensed Features, Time Configuration, DNS and Routing, Authentication Services, Virtual Machine Startup/Shutdown, Virtual Machine Swapfile Location, Security Profile, Host Cache Configuration, System Resource Reservation, Agent VM Settings, Advanced Settings.
- View:** vSphere Standard Switch
- Networking:** Standard Switch: vSwitch0
 - VMkernel Port: Management Network (IP: vmk0 : 192.168.1.7, MAC: fe80::225:90ff:fe79:4386)
 - Physical Adapters: vmnic0 (1000 Full)
- Standard Switch: Bridge** (Properties... button highlighted with a red box)
 - No associated port groups
 - Physical Adapters: vmnic3 (1000 Full)
- Standard Switch: Management**
 - No associated port groups
 - Physical Adapters: No adapters
- Standard Switch: IPS 1**
 - No associated port groups
 - Physical Adapters: No adapters
- Standard Switch: IPS 2**
 - No associated port groups
 - Physical Adapters: No adapters

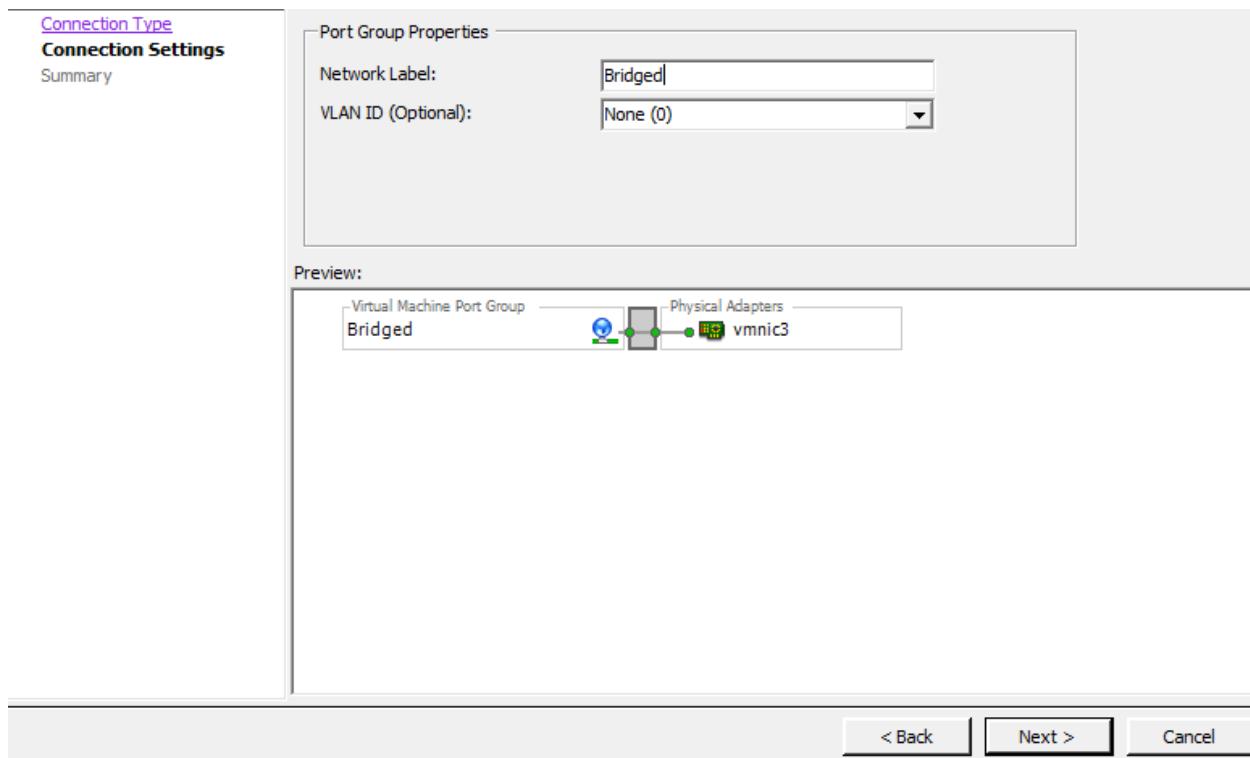
A new window show you the properties for the Bridge vswitch will pop up. Click on the "Add..." button on the bottom left corner to start the add network wizard.



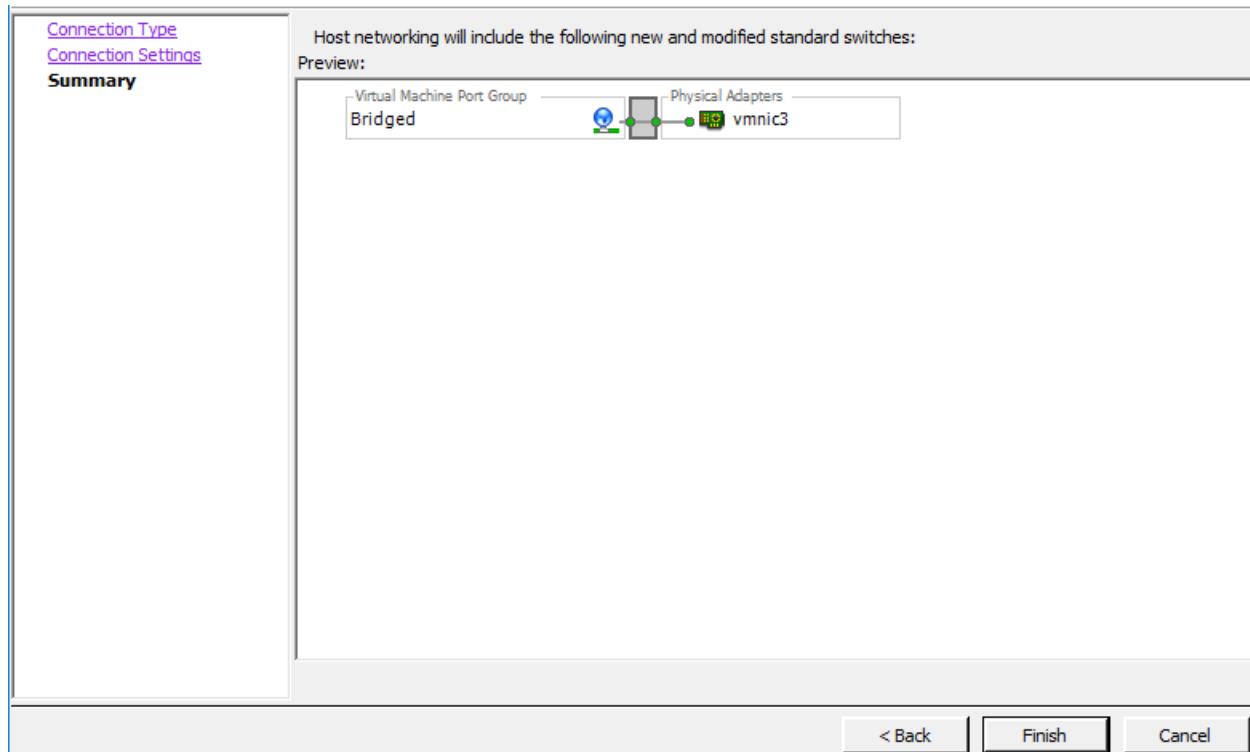
On the first screen, under “Connection Type”, click the “Virtual Machine” radio button, then click Next.



Under the “Port Group Properties” section, Change the “Network Label:” input box to “Bridged”, then click Next again.



The final page will confirm your settings. Click Finish.



Close the Bridge virtual switch properties. Back under Configuration > Networking, you should see a new port group on the Bridge virtual switch. If you'd like, you could repeat the process and create port groups for the other virtual switches, but so long as you have created a single port group, you can create the other port groups for the remaining virtual switches on the ESXi web interface.

Final Flight Check

At this point, you should have ESXi installed. You may or may not have installed the latest esxui fling to fix some of the web interface issues, or barring that, you have a windows system with vSphere client installed to work around some web UI issues. You should have have 5 virtual switches (four if you only have a single physical network port available), and at least 5 port groups (including the default "Management Network" port group used for managing ESXi).

You'll want to make sure that ESXi is recognizing all the RAM you have installed, and that all of the disks (or RAID arrays, if you're fancy) you have attached to your ESXi server are recognized. Quickest way to do this would be to fire up the ESXi web interface, log in, and under the Navigator pane, click on "Host" option. The central pane has a nice status report in a section called oddly enough, "Hardware":

Hardware																	
Manufacturer	Supermicro																
Model	X9SCL/X9SCM																
► CPU	4 CPUs x Intel(R) Xeon(R) CPU E31220 @ 3.10GHz																
Memory	31.97 GB																
► Virtual flash	0 B used, 0 B capacity																
▼ Networking																	
Hostname	Valhalla.blindseeker.com																
IP addresses	1. vmk0: 192.168.1.7 2. vmk0: fe80::225:90ff:fe79:4386																
DNS servers	1. 192.168.1.1																
Default gateway	192.168.1.1																
IPv6 enabled	Yes																
Host adapters	4																
Networks	<table border="1"> <thead> <tr> <th>Name</th><th>VMs</th></tr> </thead> <tbody> <tr> <td>Bridged</td><td>0</td></tr> <tr> <td>IPS 1</td><td>0</td></tr> <tr> <td>IPS 2</td><td>0</td></tr> <tr> <td>Management</td><td>0</td></tr> </tbody> </table>	Name	VMs	Bridged	0	IPS 1	0	IPS 2	0	Management	0						
Name	VMs																
Bridged	0																
IPS 1	0																
IPS 2	0																
Management	0																
▼ Storage																	
Physical adapters	8																
Datastores	<table border="1"> <thead> <tr> <th>Name</th><th>Type</th><th>Capacity</th><th>Free</th></tr> </thead> <tbody> <tr> <td>datastore1</td><td>VMFS5</td><td>78.75 GB</td><td>820 MB</td></tr> <tr> <td>Rogue 1</td><td>VMFS5</td><td>931.25 GB</td><td>930.3 GB</td></tr> <tr> <td>Rogue 2</td><td>VMFS5</td><td>931.25 GB</td><td>897.28 GB</td></tr> </tbody> </table>	Name	Type	Capacity	Free	datastore1	VMFS5	78.75 GB	820 MB	Rogue 1	VMFS5	931.25 GB	930.3 GB	Rogue 2	VMFS5	931.25 GB	897.28 GB
Name	Type	Capacity	Free														
datastore1	VMFS5	78.75 GB	820 MB														
Rogue 1	VMFS5	931.25 GB	930.3 GB														
Rogue 2	VMFS5	931.25 GB	897.28 GB														

The hardware window tells me how much RAM, how many network cards/ports, and how many CPU cores were detected. Additionally it tells me what the network settings of the actual ESXi host are (ip address, gateway, DNS, etc. finally, It lists all the configured port groups, and all of the connected data stores (disk drives we can use for storing VMs and other files). In the upper right corner, there are a set of bars that will give you a quick at a glance view of resource utilization on the server. For instance, you can see that I have approximately 1.9TB of space available, with a little over 100GB of space used, I have 32GB of ram in total, with 32GB in use, and CPU usage is minimal at 111mhz out of an available 12.4GHz (3.10 Ghz x 4 cores). This system is as ready as it is going to get for hosting VMs.



If for some reason you think all of your storage space isn't be detected or utilized, consider visiting the Storage interface under the Navigator pane. Try seeing if perhaps your array or disks simply weren't formatted or converted to datastores by click the "Devices" tab and seeing if there are some uninitialized disks or disks with free space listed then format and initialize the disks as needed.

Name	Status	Type	Capacity	Queue Depth	Vendor
Local ATA Disk (t10 ATA_____Corsair_Force_3_SSD_____1219821800FF16400004)	Normal	Disk (SSD)	83.85 GB	31	ATA
Local ATA Disk (t10 ATA_____ST1000DM0032D9YN162_____S1D3H7LC)	Normal	Disk	931.51 GB	31	ATA
Local ATA Disk (t10 ATA_____ST1000DM0032D9YN162_____S1D3JKJN)	Normal	Disk	931.51 GB	31	ATA

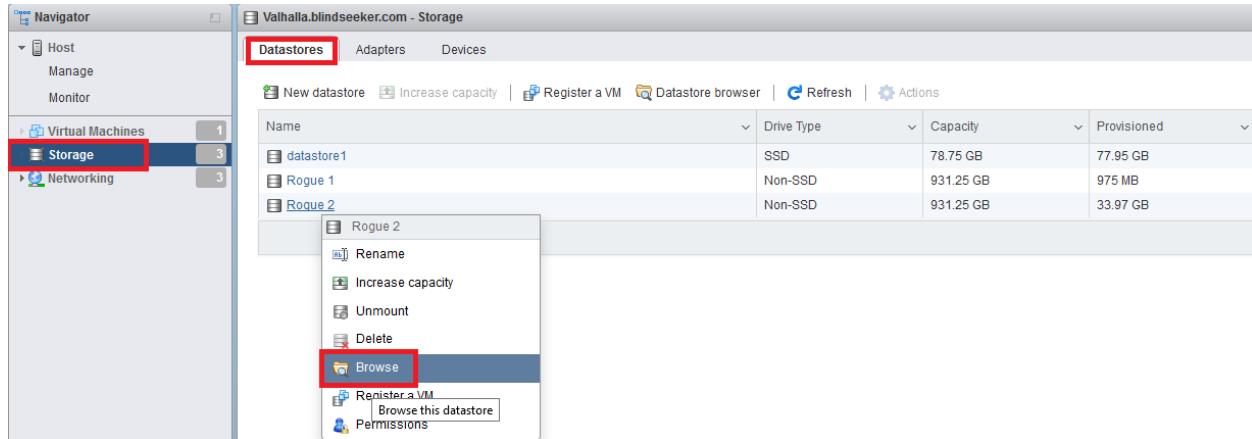
In regards to other hardware-related issues, make sure all your hardware is seated and installed properly. If your network cards aren't being recognized, you may have to acquire new network cards that are recognized by ESXi. If RAM or CPU is not being recognized by ESXi, verify that the hardware is being detected by the system BIOS; as the hardware may simply be bad. If everything in your server shows up properly and you have all the resources you need, let's move on.

Creating the First VM, pfSense

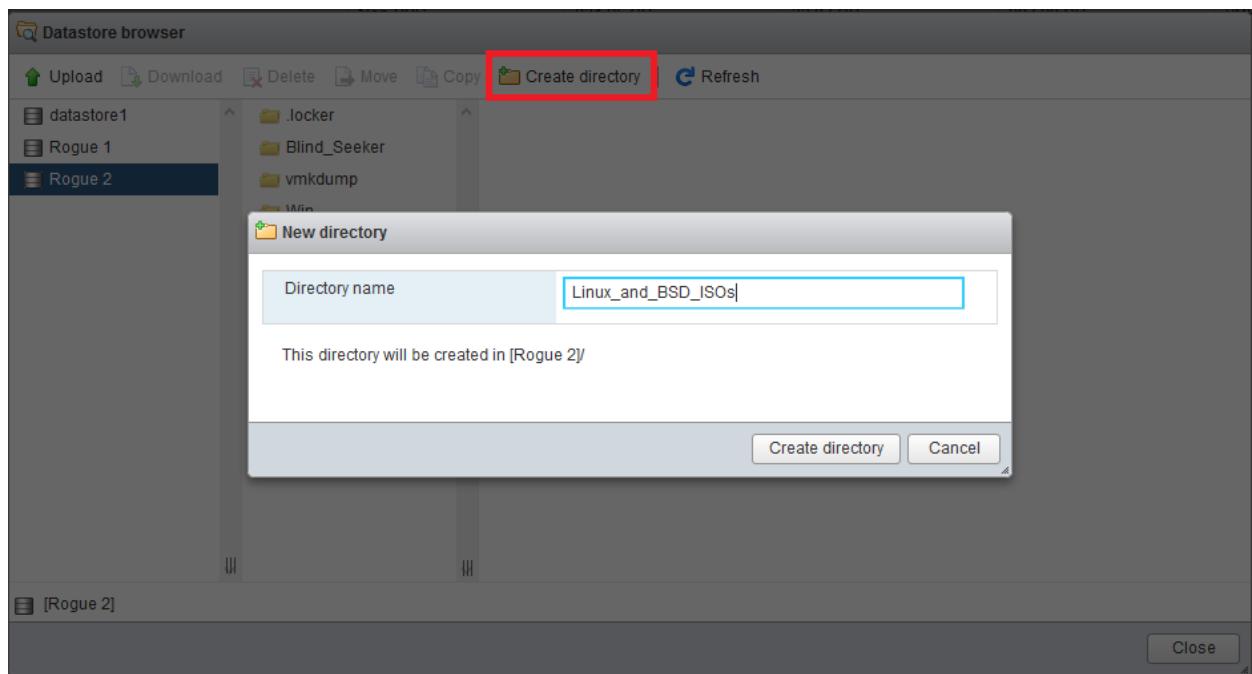
pfSense is the keystone holding this entire configuration together. Personally, it's my favorite firewall distro due to ease of use, the amount of functionality it includes out of the box, combined with a plugin/add-on system for additional functionality ; If you have the CPU, RAM, and Disk, pfSense can easily be converted into a so-called "Next-Generation" firewall. To start, make your way to <https://www.pfsense.org/download/>. Download the latest Installation ISO for the amd64 architecture (as of right now this should be "pfSense-CE-2.3.2-RELEASE-amd64.iso.gz")

Once you have downloaded the ISO, depending on what operating system you are using to manage your ESXi server, you may need to acquire a compression utility; pfSense ISO files are compressed with gzip (.iso.gz), so you'll need to decompress it using gunzip (mac/linux) or 7-zip (Windows). After you have decompressed it, you'll want to upload it to ESXi and put it on a datastore. On the web interface, under the Navigator pane, click on "Storage", then ensure the

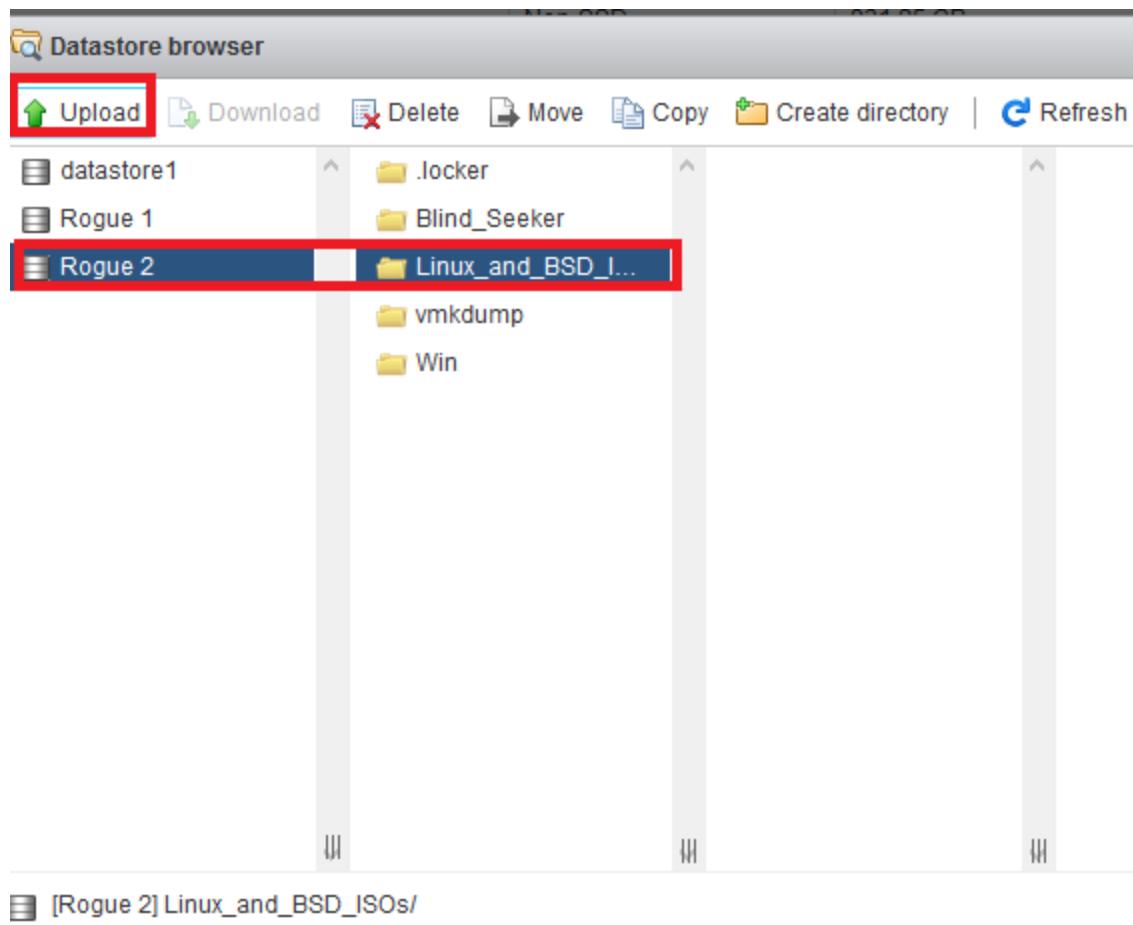
“Datastores” tab is selected. Right click on the datastore you wish to store the ISO on, then click “Browse”. In my case, I have Three datastores. I’m going to use the “Rogue 2” datastore.



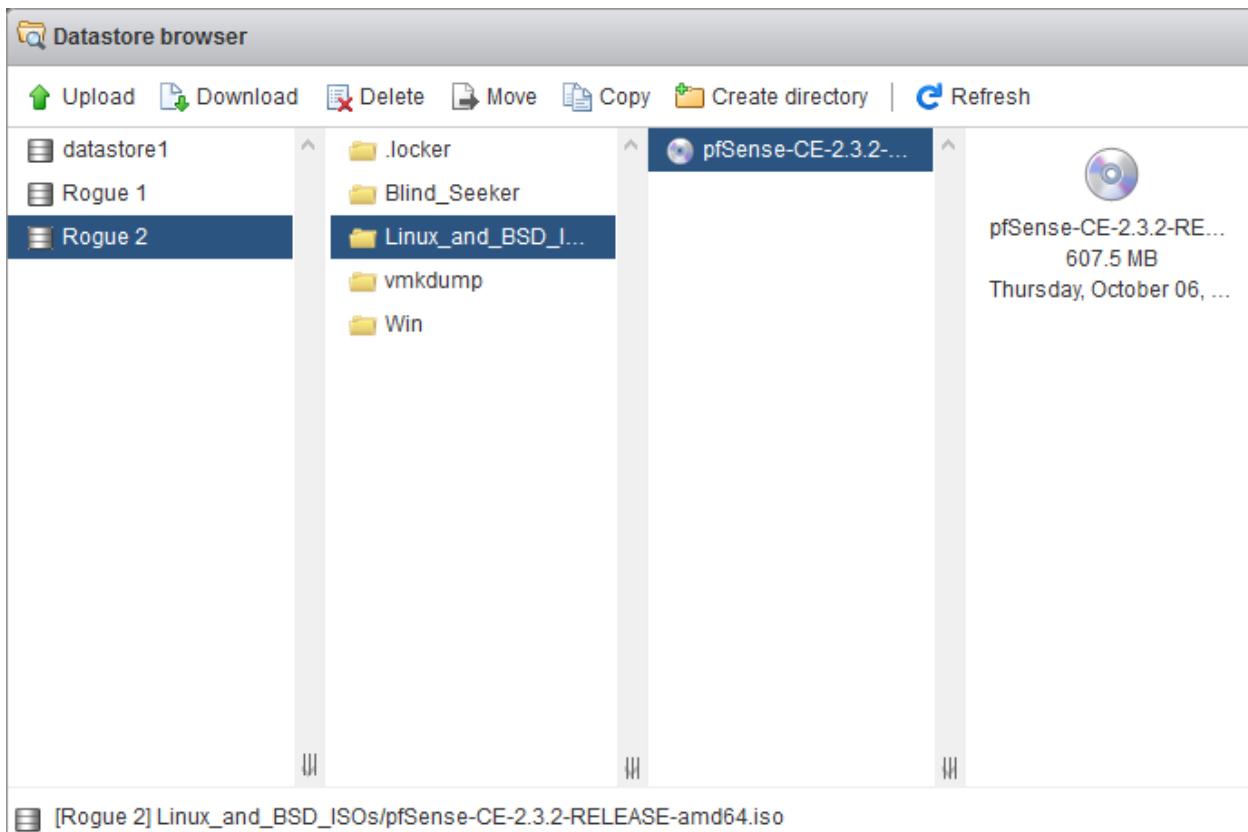
This will bring up the datastore browser. Click the “Create directory” option; we’re going to make a directory we use to store all of our Linux and BSD ISOs, so lets name it “Linux_and_BSD_ISOs”. Click the “Create directory” button.



Now, click on the new directory we created, then click “Upload” in the upper right portion of the window.



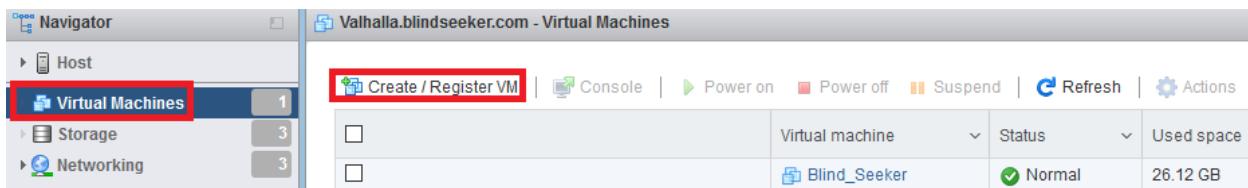
This will open a file browser on your local machine. You need to navigate to where you stored the unzipped pfSense ISO. In my case, I'm using Windows, and I stored my ISO in D:\ISOs\Nix. When you select the ISO file, it will start the process of uploading it to the ESXi server's datastore. When the upload process has completed, you should be able to click on the ISO file to get file statistics.



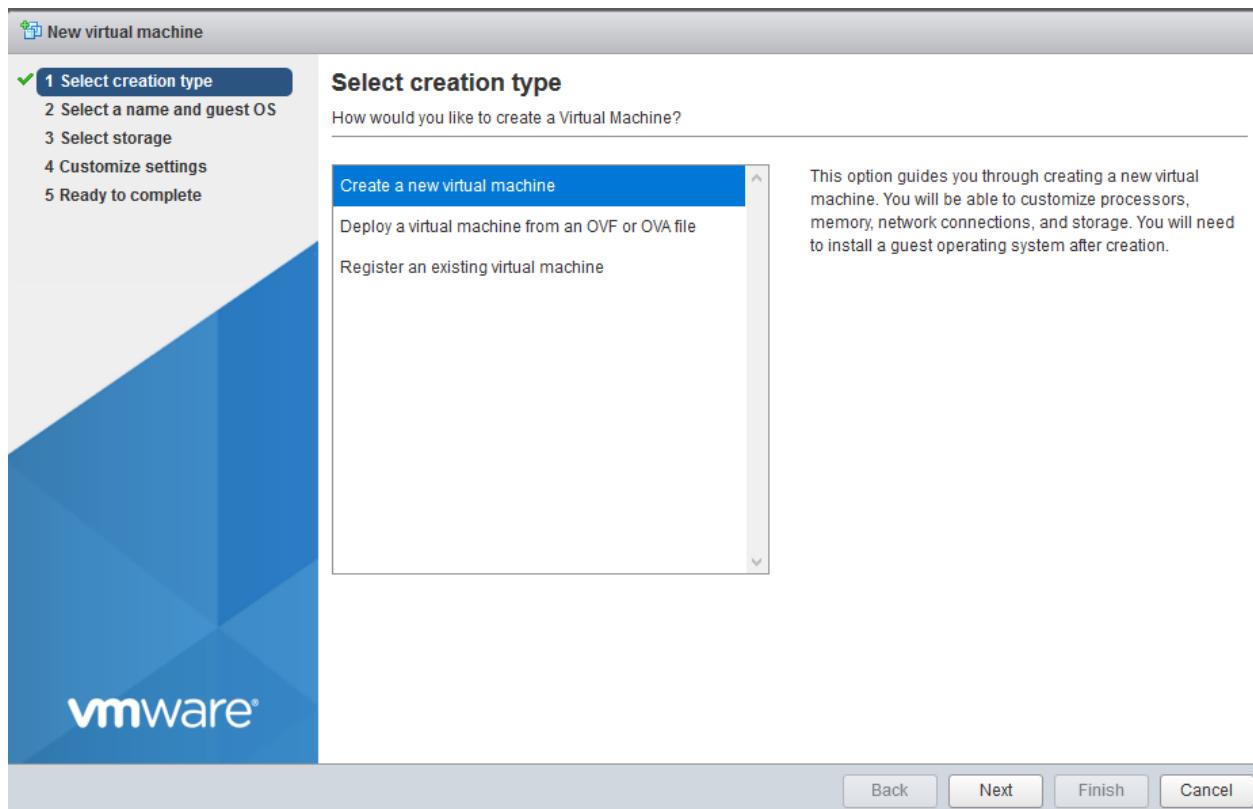
After you have uploaded the ISO file to the desired datastore, close the Datastore browser.

Adding a New VM

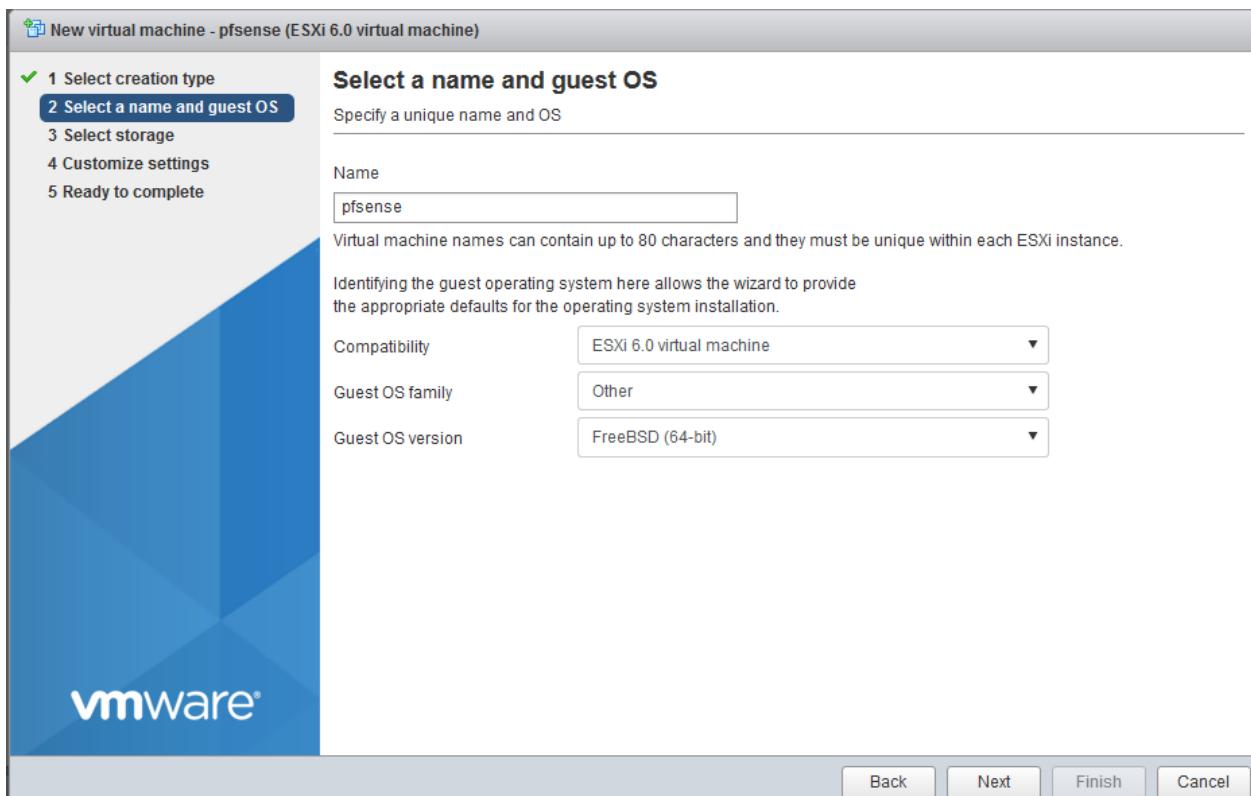
Now that we have an ISO image to boot from, let's create our first VM. On the ESXi web interface, click on "Virtual Machines" under the Navigator pane. This brings you to a list of currently registered VMs on the server. This list should be empty on your server, but in my case, since my web server is here, I have one VM named "BlindSeeker." Click on "Create/Register VM" to start the "New virtual machine" wizard.



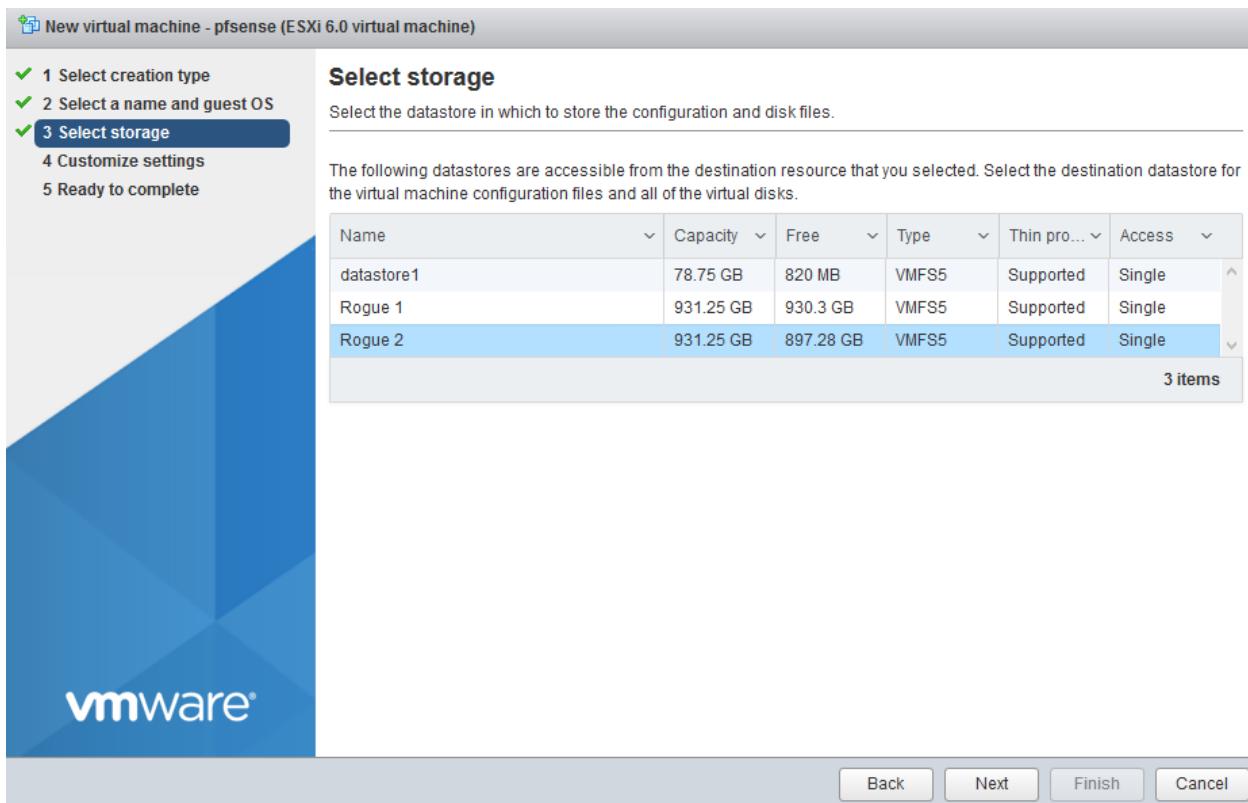
On the first screen, select the option "Create a new virtual machine" then click the Next button.



The next screen has you give your VM a name, and choose the OS family and version. Enter “pfSense” in the “Name” field. For the OS family drop-down, choose other. For the OS version drop-down, choose FreeBSD (64-bit), then click Next.



The next page will have you choose what datastore you wish to place your VM on. In my case, I have three choices. I'm going to choose “Rogue 2” as my datastore by clicking on it. Once you have made your decision (if you have more than one datastore), click next.



The next page allows us to make various customizations to our Virtual Machine's hardware. The first and most obvious settings you'll see are CPU, Memory and Hard disk space. For the pfSense VM, we'll be using 1 CPU core, 512mb of RAM, and 5GB of disk space, but there are some configuration changes we want to make to how the disk space is provided to the VM. Click the triangle next to "Hard disk 1" to bring up the hard disk settings. In the "Disk Provisioning" section, click the "Thick provisioned, lazily zeroed" radio button, and in the "Virtual Device Node" section, select "SATA controller 0" in the first drop-down. The second drop-down should read "Sata (0:0)"

Customize settings

Configure the virtual machine hardware and virtual machine additional options

Add hard disk Add network adapter Add other device

▶ CPU	1	i	
▶ Memory	512	MB	
▼ Hard disk 1	5	GB	X
Maximum Size	897.28 GB		
Location	[Rogue 2] pfsense		Browse...
Disk Provisioning	<input type="radio"/> Thin provisioned <input checked="" type="radio"/> Thick provisioned, lazily zeroed <input type="radio"/> Thick provisioned, eagerly zeroed		
Shares	Normal	1000	▼
Limit - IOPs	Unlimited		
Virtual Device Node	SATA controller 0	SATA (0:0)	▼

Next, in the “SCSI Controller 0” field, click the X on the right to remove the SCSI controller entirely; we won’t be needed it.



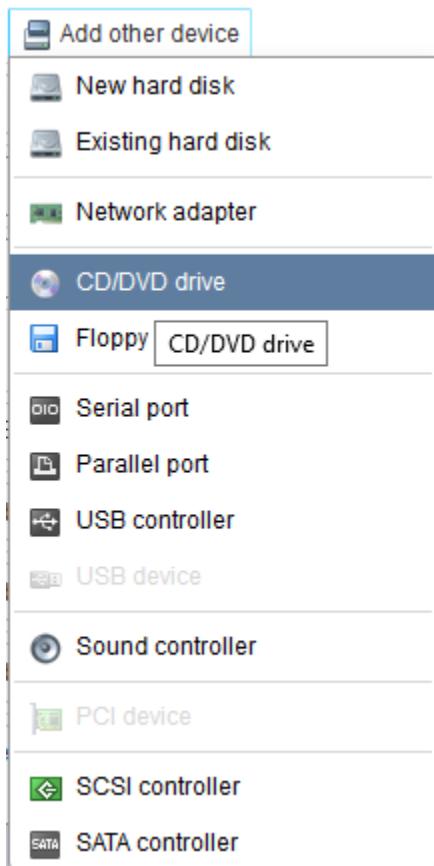
Click the “Add network adapter” button twice. Two more network cards appear named “New Network Adapter”, immediately underneath “Network Adapter 1”. Configure the first network adapter to connect to the “Bridged” port group, the second adapter to connect to the “Management” port group, and the third adapter to connect to the “IPS 1” port group. Ensure that the checkbox labeled “Connect” is checked for all three adapters.

▶ Network Adapter 1 Bridged Connect X

▶ New Network Adapter Management Connect X

▶ New Network Adapter IPS 1 Connect X

Finally, click the “Add other device” option, and a drop-down featuring a series of choices appears. Select “CD/DVD” drive.



“New CD/DVD drive” should appear immediately underneath the network adapters. Click on the triangle next to this text to bring up settings for the virtual CD/DVD drive. Ensure that the checkbox “Connected at power on” is checked, then in the drop-down field set to “Host device”, change this setting to “Datastore ISO file”. When you do so, the datastore browser window should open up. Browse to the location you uploaded your pfSense ISO to.

A screenshot of a configuration dialog for a new CD/DVD drive. The dialog has several fields:

- "New CD/DVD Drive": A dropdown menu currently set to "Datastore ISO file".
- "Status": A checkbox labeled "Connect at power on" which is checked.
- "CD/DVD Media": A dropdown menu containing the path "[Rogue 2] Linux_and_BSD_ISOs/pfSense-CE-2.3.2-F" with a "Browse..." button to its right.
- "Virtual Device Node": Two dropdown menus: the first is set to "SATA controller 0" and the second is set to "SATA (0:1)".

Afterwards, click the Next button to move on to the final page of the wizard. This page is a confirmation page to review all the settings we have selected for this VM. Your settings should look similar to this:

Name	pfsense
Datastore	Rogue 2
Guest OS name	FreeBSD (64-bit)
Compatibility	ESXi 6.0 virtual machine
vCPUs	1
Memory	512 MB
Network adapters	3
Network adapter 1 network	Bridged
Network adapter 1 type	E1000
Network adapter 2 network	Management
Network adapter 2 type	E1000
Network adapter 3 network	IPS 1
Network adapter 3 type	E1000
IDE controller 0	IDE 0
IDE controller 1	IDE 1
SATA controller 0	New SATA controller
Hard disk 1	
Capacity	5GB
Datastore	[Rogue 2] pfsense
Mode	Dependent
Provisioning	Thick provisioned, lazily zeroed
Controller	SATA controller 0 : 0
CD/DVD drive 1	
Backing	[Rogue 2] Linux_and_BSD_ISOs/pfSense-CE-2.3.2-RELEASE-amd64.iso
Connected	Yes

Click finish and allow some time for ESXi to create your new VM. After completion you should be dropped back into the virtual machines listing, and you should be able to see the new VM.



Note: For whatever reason, ESXi appears to have reversed the order of network adapters from what I expected. If you right click on the VM you just created and select “Edit Settings”, you may see that “Network Adapter 1” is set to “IPS 1” as its port group, “Network Adapter 2” is set to “Management” and “Network Adapter 3” is set to “Bridged”.

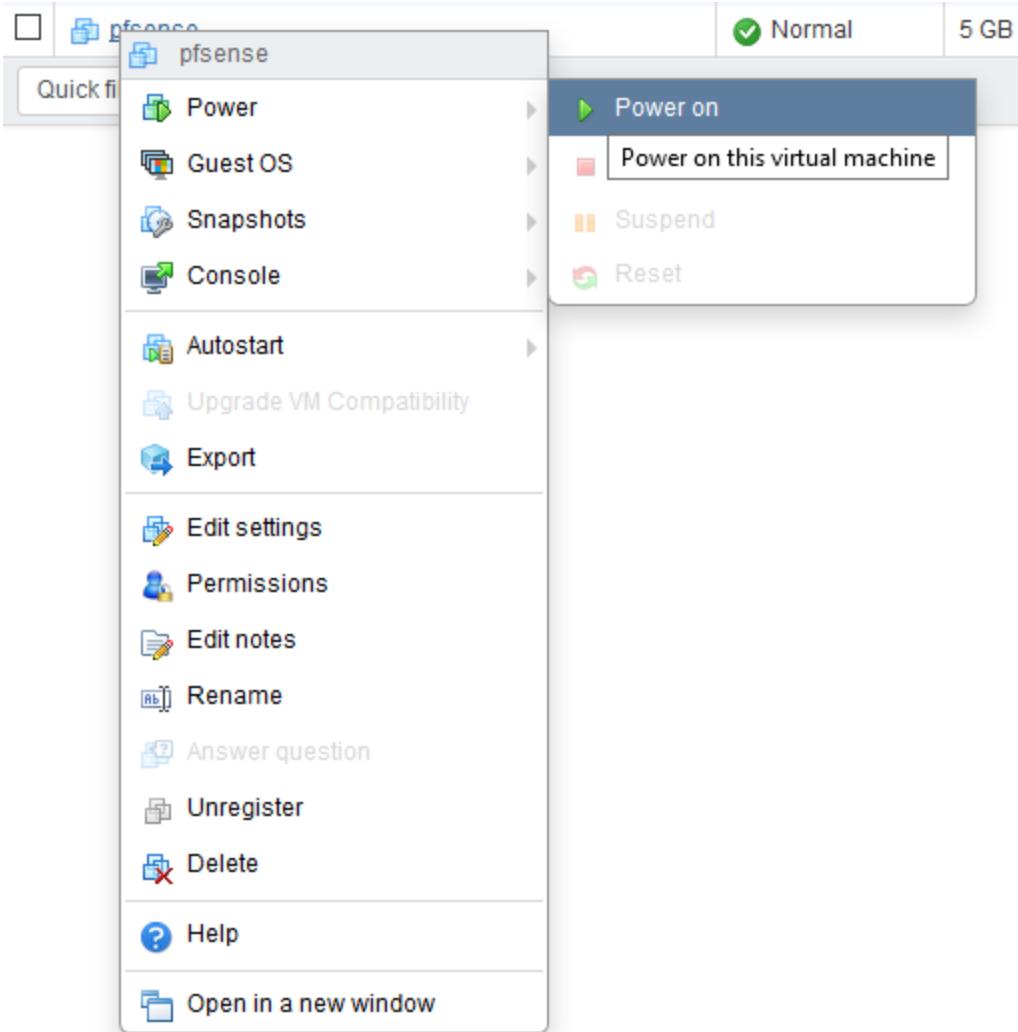
▶ Network Adapter 1	IPS 1	<input checked="" type="checkbox"/> Connect	
▶ Network Adapter 2	Management	<input checked="" type="checkbox"/> Connect	
▶ Network Adapter 3	Bridged	<input checked="" type="checkbox"/> Connect	

Apparently, ESXi considers the newest created network adapter to be “Network Adapter 1”, then 2, and so on. Currently, this ordering doesn’t really matter, however if you’re OCD like I am, Simply swap the port group settings for Adapter 1 and Adapter 3, then click the Save button. The only other VM we will be making with multiple network adapters is the IPS VM, so don’t worry too much, just keep this in mind.

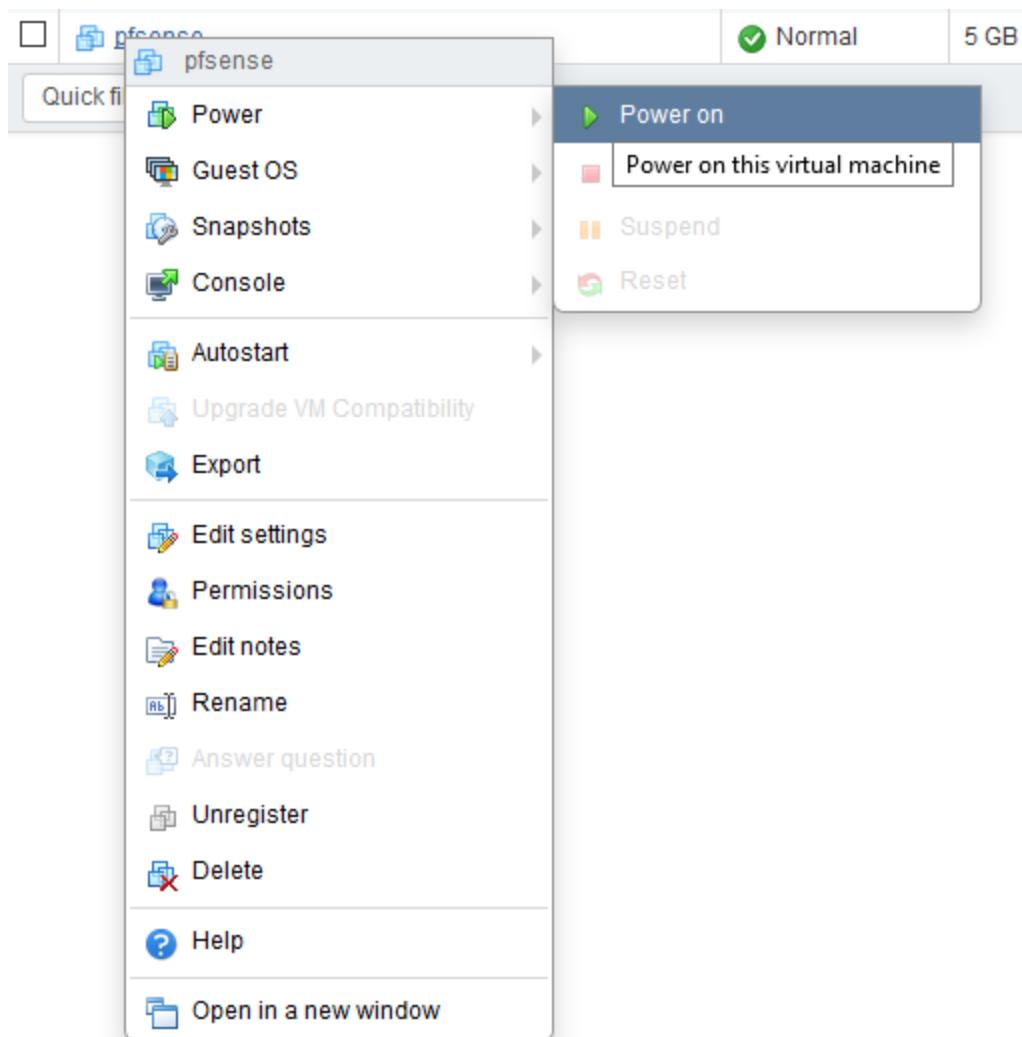
▶ Network Adapter 1	Bridged	<input checked="" type="checkbox"/> Connect	
▶ Network Adapter 2	Management	<input checked="" type="checkbox"/> Connect	
▶ Network Adapter 3	IPS 1	<input checked="" type="checkbox"/> Connect	

Installing pfSense

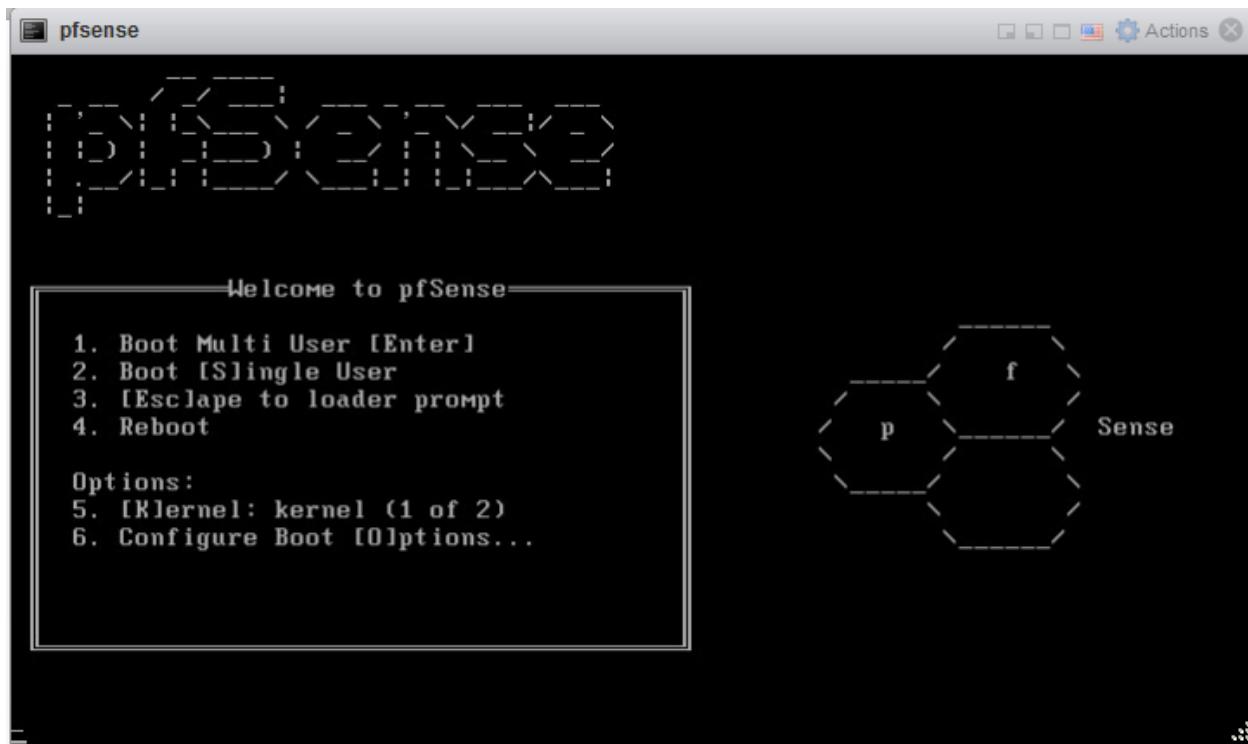
At this point, the VM is ready to be powered on to have the operating system installed. pfSense is really easy to install if you’ve installed any operating system’s before. You can start the VM by navigating to “Virtual Machines” in the navigator pane, then either right click pfSense and selecting Power > Power on, or click to highlight pfSense, and on the option bar above the VM list, click “Power on”.



In either case, the VM should start up. You'll want to open a console window to interact with the VM. Right click on pfSense in the virtual machines list again. This time, select "Console > Browser Console" to open a pop-up window that will act as a console to interact with the virtual machine.

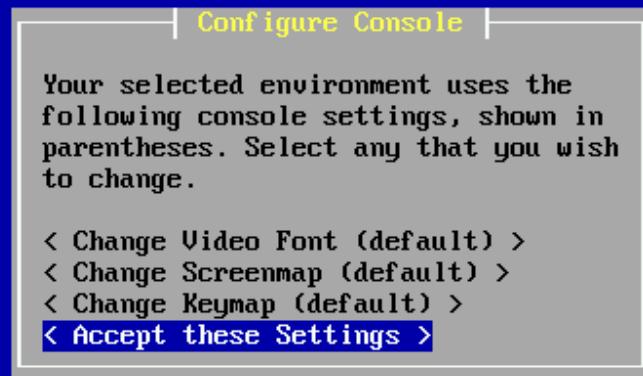


The console window should pop up, and you should be able to interact with the host via the new window.



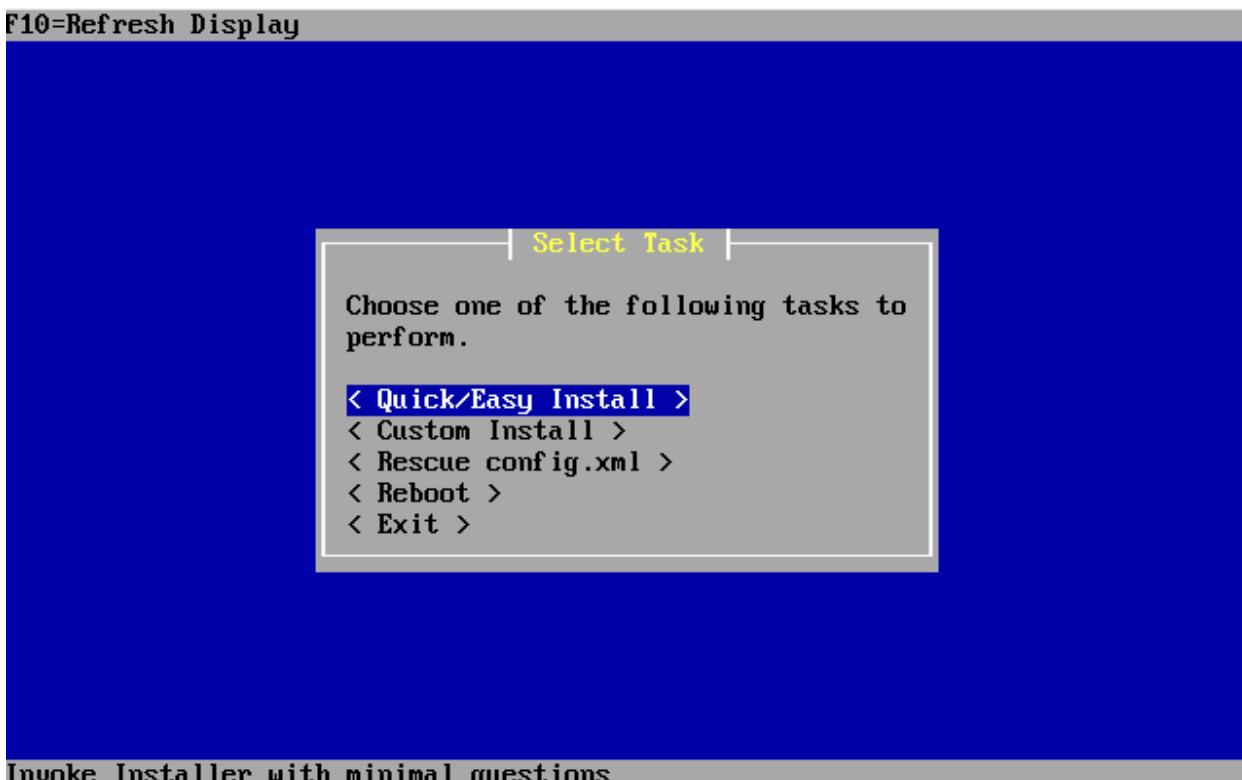
Installing pfSense is very straightforward if you've installed any OS from optical media or USB before, it should be fairly easy to figure out. You'll want to select the option 1 "Boot Multi User [Enter]" By hitting the enter key. Let pfSense boot up, and the system should automatically run the installer. Adjust your video, screenmap, and keymap settings as necessary, then select "< Accept these Settings >".

F10=Refresh Display



On the next screen, select “< Quick/Easy Install >” and let pfSense do all the heavy lifting. The next screen will inform you that the install will erase the contents of the hard disk. Since our virtual disk is already empty, this doesn’t matter in the least. Select OK. and let the install run.

F10=Refresh Display

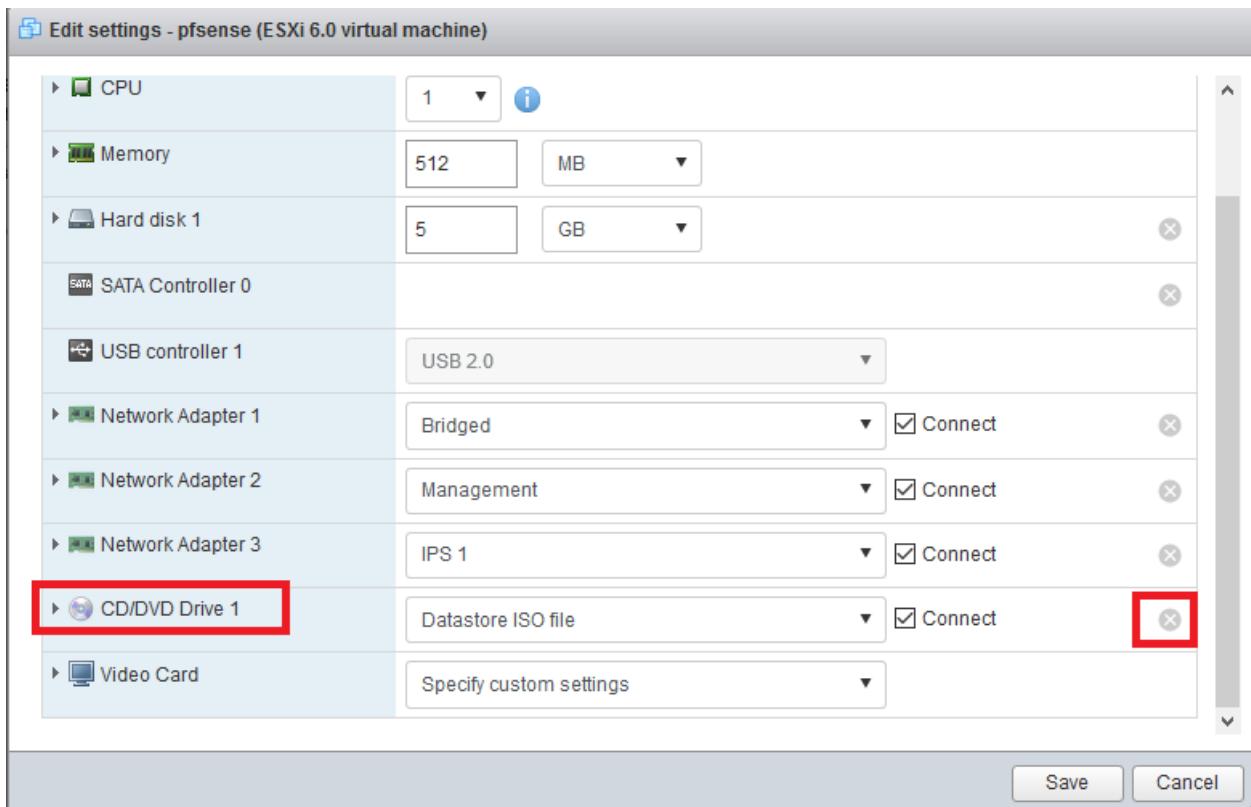


Invoke Installer with minimal questions

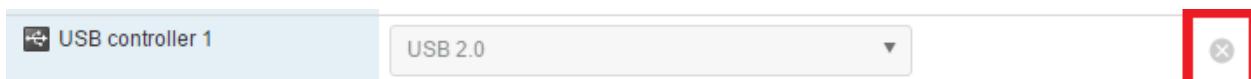
The installer will go through and install pfSense to the virtual hard disk. The installer will ask if you want to install a standard or embedded kernel. Make sure to select "< Standard Kernel >". Finally, the installer informs you to reboot the machine to boot from the hard drive. The installation is done, however, we're not going to reboot. In the ESXi web interface, close the console, right click on pfSense, and select "Power > Power off" to turn off the VM.

Final VM Settings

In the "Virtual Machines" pane under Navigator, right click on pfSense and select "Edit settings". In the section labeled "CD/DVD drive", we're going to click the X to the right to remove the CD/DVD drive, since we won't need it anymore.



Follow the same instructions for the CD/DVD drive above and remove “USB controller 1” as well:

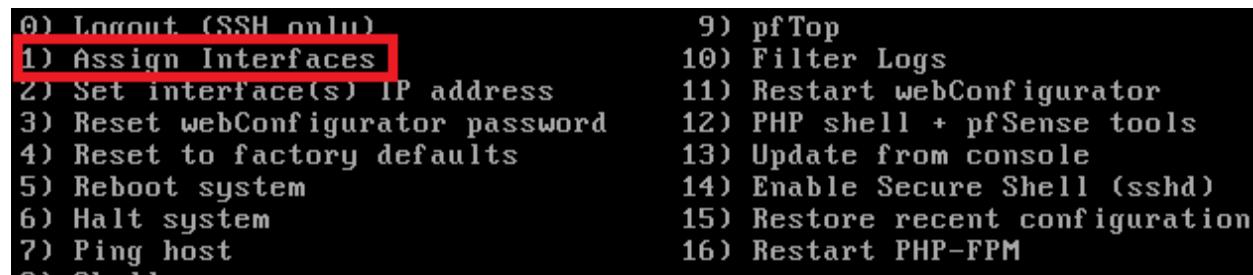


Afterwards, click the triangle next to “Network Adapter 1”, “Network Adapter 2”, and “Network Adapter 3”. Be sure to document somewhere the contents of the MAC address field, and which MAC address is connected to which port group. We’ll be needing this information in the next section for performing network configuration. After you are finished, click the Save button.

Network Adapter 1	Bridged
Status	<input checked="" type="checkbox"/> Connect at power on
Adapter Type	E1000
MAC Address	Automatic <input type="button" value="▼"/> 00:0c:29:49:38:3e
Network Adapter 2	Management
Status	<input checked="" type="checkbox"/> Connect at power on
Adapter Type	E1000
MAC Address	Automatic <input type="button" value="▼"/> 00:0c:29:49:38:48
Network Adapter 3	IPS 1
Status	<input checked="" type="checkbox"/> Connect at power on
Adapter Type	E1000
MAC Address	Automatic <input type="button" value="▼"/> 00:0c:29:49:38:52

Network Configuration

Power on your pfSense VM, and connect to the console. Once the system is fully booted, You should be dropped into a menu system with 16 options. Select option 1, to assign interfaces. This defines which interface (e.g. em0, em1, em2, etc.) corresponds to WAN, LAN, OPT1, etc. networks for pfSense network services and firewall policies.



In “Final VM Settings”, I had you document the MAC address of each network adapter. We need to know which interface in pfSense corresponds to which network adapter so we can designate

the WAN (bridged network), LAN (Host-Only Network) and OPT1 (Internal Network) properly in pfSense. In my case, the MAC address for em0 is 00:0c:29:49:38:3e. This corresponds to the MAC address assigned to network adapter 1 in the ESXi virtual machine settings for pfSense. em1 corresponds to adapter 2, and em2 corresponds to adapter 3. Once you have figured out which interface corresponds to which VirtualBox adapter we need to assign em0, em1 and em2 to be the WAN, LAN, and OPT1 interfaces respectively. The WAN interface needs to be assigned to the network adapter connected to the bridged port group. The LAN interface needs to be assigned to the network adapter connected to the management port group, and finally, OPT1 needs to be assigned to the network adapter connected to the IPS 1 port group. In my case, em0 became the WAN interface, em1 became the LAN interface, and em2 became the opt2 interface. After you confirm that the settings are correct, you should automatically be returned to the pfSense main menu.

```
Enter the WAN interface name or 'a' for auto-detection  
(em0 em1 em2 or a): em0  
  
Enter the LAN interface name or 'a' for auto-detection  
NOTE: this enables full Firewalling/NAT mode.  
(em1 em2 a or nothing if finished): em1  
  
Optional interface 1 description found: OPT1  
Enter the Optional 1 interface name or 'a' for auto-detection  
(em2 a or nothing if finished): em2  
  
Enter the Optional 2 interface name or 'a' for auto-detection  
( a or nothing if finished):  
  
The interfaces will be assigned as follows:  
  
WAN -> em0  
LAN -> em1  
OPT1 -> em2
```

Now that we have assigned the network interfaces, we have to configure IP addresses. In most cases, the “WAN” interface (bridged network) will automatically get an IP address from the device on your physical network that provides DHCP services. Baremetal hypervisors are different from hosted hypervisors. **You'll want to ensure that this IP address is either has a static DHCP allocation, or that you statically assign the network information for the WAN interface as necessary for your network.** This is primarily due to our management system for both the hypervisor and the VMs needing to know the IP address for the WAN interface if we're going to use remote access to manage the VMs. **We need to be able to set up a firewall rules and a persistent route to access the lab network externally, and this requires having the WAN IP address be consistent.** In my case, my WAN interface was assigned an IP address of 192.168.1.22, that I made a static DHCP mapping for. As for the LAN and OPT1 networks, we will have to manually set the IP address, subnet mask, and DHCP scopes for the networks. Select option 2 in the pfSense main menu to get started.

```

0) Logout (SSH only)
1) Assign Interfaces
2) Set interface(s) IP address
3) Reset webConfigurator password
4) Reset to factory defaults
5) Reboot system
6) Halt system
7) Ping host
8) Shell
9) pfTop
10) Filter Logs
11) Restart webConfigurator
12) PHP shell + pfSense tools
13) Update from console
14) Enable Secure Shell (sshd)
15) Restore recent configuration
16) Restart PHP-FPM

```

The configuration wizard will ask you which interface you want to configure you'll have to configure the LAN and OPT1 interfaces individually. Here are the settings I used for my lab network:

LAN (lan)	->	em1	->	v4: 172.16.1.1/24
OPT1 (opt1)	->	em2	->	v4: 172.16.2.1/24

The wizard will ask you if you want to set up IPv6 for the interfaces and networks. Say no, because that's a can of worms we're not going to deal with here. The wizard will also ask if you want to configure DHCP services for the LAN and OPT1 interfaces; select yes. For the LAN network start address, I entered 172.16.1.10. For the end address, I entered 172.16.1.254. For the OPT1 network I chose 172.16.2.10 as the start, and 172.16.2.254 for the end address. If you are using 172.16.1.0/24 or 172.16.2.0/24 in your physical network, choose another network range to assign to the LAN and OPT1 interfaces to avoid network conflicts.

The reason we start at 1.10 and 2.10 is to make room for our other virtual machines that need to have a fixed IP address. The gap allows us to set STATIC DHCP allocations for VMs/systems whose addresses we want to remain fixed. We'll talk about this a little bit more later. Before we're done with the command line, there is one more thing we'll need to do in order to access the pfSense web interface. Select option 8 from the pfSense menu to open up a shell. Once you get a shell prompt, run the following commands:

```
pfctl -d
easymode pass wan tcp 192.168.1.17 192.168.1.22 443
```

```
[2.3.2-RELEASE][root@pfSense.locaLdomaIn]# pfctl -d
pf disabled
```

```
[2.3.2-RELEASE][root@pfSense.locaLdomaIn]# easymode pass wan tcp 192.168.1.17 192.168.1.22 443
Successfully added pass rule!
```

The first command, `pfctl -d`, temporarily disables the firewall on pfSense. We have to disable the firewall because the WAN interface by default drops ALL connections from RFC1918 addresses -- These are IP addresses that are NOT routable over the public internet and technically should NOT be encountered on a WAN interface. The WAN interface of our VM sits

on an internal private network and our management system, which sits on this same internal network has to use the WAN interface to access the firewall, so this presents a problem.

The first command disables the firewall entirely. This is a temporary solution for us to do initial setup of the firewall and fix this problem permanently. The second command, `easymode pass wan tcp 192.168.1.17 192.168.1.22 443`, adds a firewall rule to the WAN interface of pfSense that tells the firewall to allow 192.168.1.17 (the address of my Windows workstation) to access the WAN interface of the pfSense VM (192.168.1.22) on port 443 (https). Adjust the IP addresses for the source and destination of this rule as needed for your network at home or work. **You need to run both of these commands before moving on to the next section.**

Web Configurator - Initial Setup

On your workstation, open your a web browser (I prefer Firefox - <https://www.mozilla.org/en-US/firefox/new/>) and navigate to the WAN address of the pfSense firewall (in our example, mine is <https://192.168.1.22>). The default credentials for access to the web interface are admin/pfsense. If this is your first time logging in, pfSense takes you through a nice little setup wizard. I'll highlight some of the important things to take note of, and/or change as necessary:

Set the primary and secondary DNS servers you plan on using. I typically use 8.8.8.8 (google public DNS) and 4.2.2.2 (Level 3 public DNS).

Primary DNS Server	8.8.8.8
Secondary DNS Server	4.2.2.2

Be sure to uncheck the “Block private networks from entering via WAN rule, **This setting must be unchecked in order for your management system (or jumpbox) to be able to access the pfSense web interface.**

RFC1918 Networks	
Block RFC1918 Private Networks	<input checked="" type="checkbox"/> Block private networks from entering via WAN <small>When set, this option blocks traffic from IP addresses that are reserved for private networks as per RFC 1918 (10/8, 172.16/12, 192.168/16) as well as loopback addresses (127/8). This option should generally be left turned on, unless the WAN network lies in such a private address space, too.</small>

The other default settings on the first time setup wizard should be fine. Please note that they will make you change the password for the admin account as a part of the setup wizard. Document the password, keep it somewhere safe, use a password manager if necessary.

Next, let's navigate to Firewall > Aliases. On the firewall Aliases page, click on “IP”, then click Add. Create an alias with the following settings:

Firewall / Aliases / Edit

Properties

Name	RFC1918	The name of the alias may only consist of the characters "a-z, A-Z, 0-9 and _".
Description	An alias for all RFC1918 networks	A description may be entered here for administrative reference (not parsed).
Type	Network(s)	

Network(s)

Hint: Networks are specified in CIDR format. Select the CIDR mask that pertains to each entry. /32 specifies a single IPv4 host, /128 specifies a single IPv6 host, /24 specifies 255.255.255.0, /64 specifies a normal IPv6 network, etc. Hostnames (FQDNs) may also be specified, using a /32 mask for IPv4 or /128 for IPv6. An IP range such as 192.168.1.1-192.168.1.254 may also be entered and a list of CIDR networks will be derived to fill the range.			
Network or FQDN	10.0.0.0 / 8	10.x.x.x RFC 1918 networks	
	172.16.0.0 / 12	172.16.x.x RFC 1918 networks	
	192.168.0.0 / 16	192.168.x.x RFC 1918 networks	

Save + Add Network

Click Save to be brought back to the previous page, then click Apply Changes. We just created an alias for RFC1918 networks (local networks that are not routable through the public internet). This will come in handy later for creating firewall rules around these networks.

Next, navigate to Firewall > Rules, and click on WAN, if it isn't already selected. Your firewall rules for the WAN interface should look like this:

Floating	WAN	LAN	OPT1							
Rules (Drag to Change Order)										
States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
0/6 KiB	*	Reserved Not assigned by IANA	*	*	*	*	*	*	Block bogon networks	
<input type="checkbox"/> 2/126 KiB	IPv4 TCP	192.168.1.17	*	192.168.1.22	443 (HTTPS)	*	none		Easy Rule: Passed from Firewall Log View	

Add Add Delete Save + Separator

That second firewall rule? Never remove it. That was the rule we entered using the `easyrule` command. This rule is absolutely mandatory. never place any firewall rules on the WAN interface above this rule (besides the “Block bogon networks” rule). This is the only thing keeping you from being locked out of the firewall web interface.

Next, navigate to System > Advanced. Click to fill in the checkbox next to the option “Disable webConfigurator anti-lockout rule”, and click save on the bottom of the page. We’re disabling the anti-lockout rule since without the firewall rule on the WAN interface we can’t access pfSense at all. The web configurator anti-lockout lets any machine connected on the LAN

network (Management network) try to connect to the pfSense web interface. This is unnecessary exposure, so disabling the rule makes sense.

Anti-lockout Disable webConfigurator anti-lockout rule

When this is unchecked, access to the webConfigurator on the LAN interface is always permitted, regardless of the user-defined firewall rule set. Check this box to disable this automatically added rule, so access to the webConfigurator is controlled by the user-defined firewall rules (ensure a firewall rule is in place that allows access, to avoid being locked out!) Hint: the "Set interface(s) IP address" option in the console menu resets this setting as well.

Before moving on to the next section, open a console session to the pfSense VM, and selection option 8 to open a shell session on the firewall (if the one we opened earlier is no longer available). Enter the following command:

```
pfctl -e
```

As a part of the initial firewall setup, pfSense reloads the firewall config and SHOULD re-enable the firewall, but just to make sure the firewall is enabled and operating normally again, we'll check anyhow. This is what you'll get if the firewall is still disabled:

```
[2.3.2-RELEASE][root@pfSense.localdomain]# pfctl -e  
pf disabled
```

And this is the output you'll get if the firewall was re-enabled already:

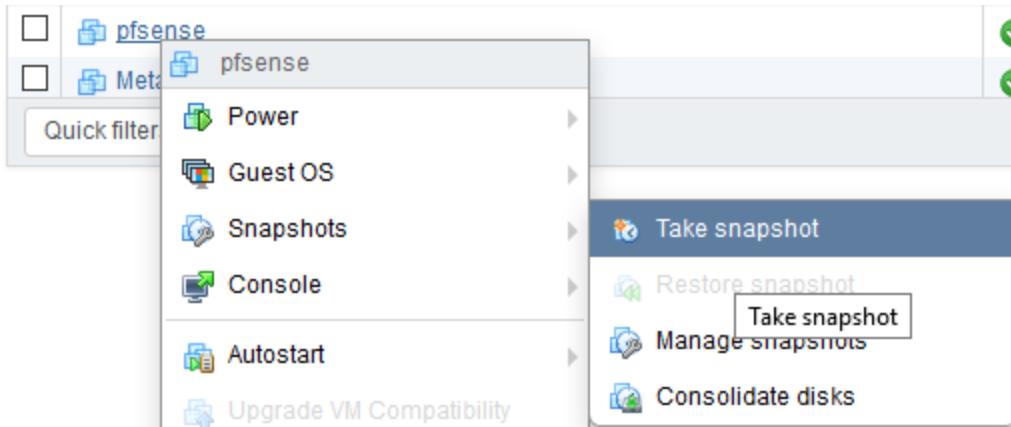
```
[2.3.2-RELEASE][root@pfSense.localdomain]# pfctl -e  
pf enabled
```

Make sure to verify that the firewall is re-enabled, and that you are able to access the firewall's web UI before moving on to the next section (e.g. before you take a snapshot)

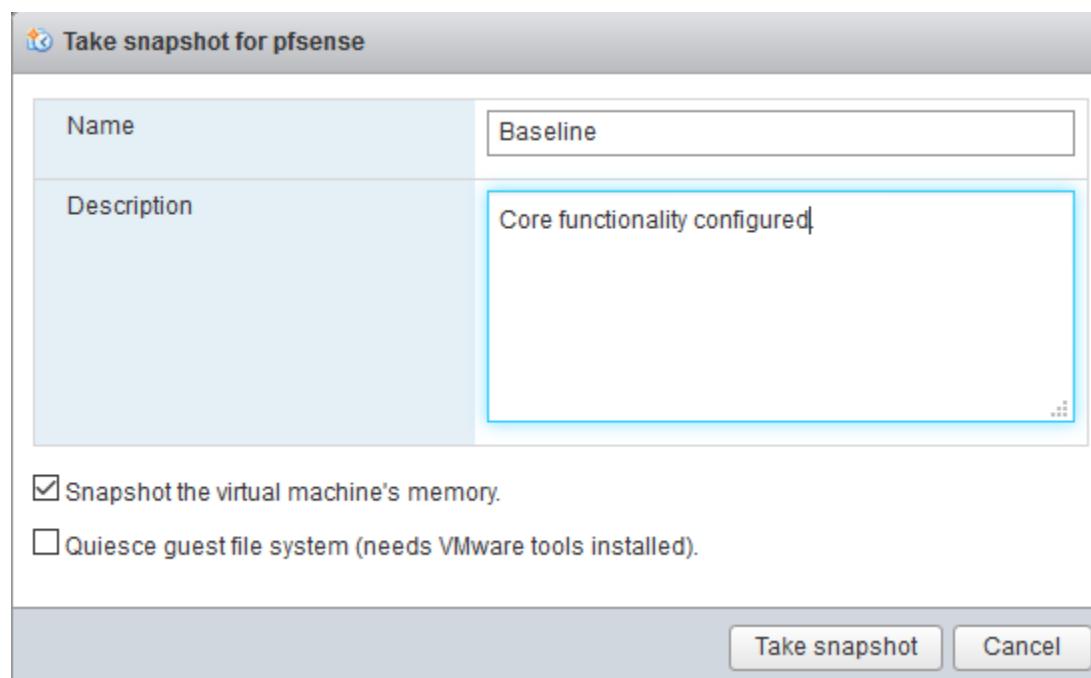
Take a Snapshot

Snapshotting is a VERY important concept with VMs. Snapshots let you restore your virtual machines to a point in time in the past when you took that snapshot. This lets you undo misconfigurations, problems, or potential issues that affect the VM relatively easily. We're going to take a snapshot of pfSense in its current state.

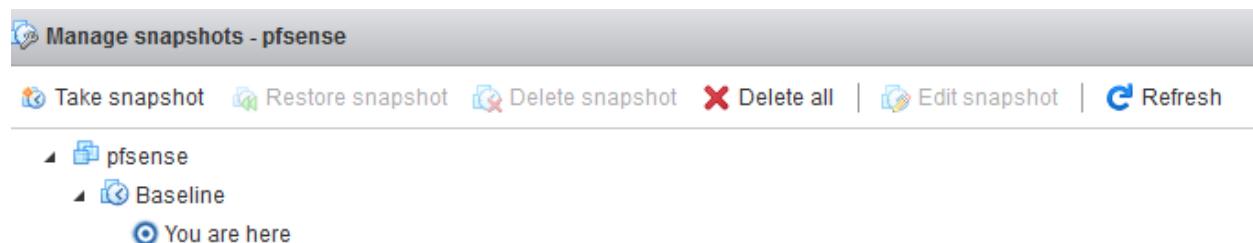
On the ESXi web interface, click on “Virtual Machines” under the navigator pane. In the Virtual machine list, right click on pfSense, Select “Snapshots > Take Snapshot”.



On the window that pops up, enter a name for the snapshot, and a description, then click the Take Snapshot button.



To access previous snapshots, right click on the VM, and choose "Schemas > Manage snapshots" to get a tree view of snapshots and various options for restoring from previous snapshots, deleting snapshots, etc.



pfSense Summary

At the end of all this, you should have a pfSense VM with the following settings:

- 512MB of RAM
- 5GB of Disk
- 1 Virtual CPU
- 3 Network interfaces:
 - 1 Interface connected to the Bridged port group (WAN interface)
 - 1 Interface connected to the Management port group (LAN interface)
 - 1 Interface connected to the OPT1 port group (OPT1 interface)

pfSense should be configured as followed:

- The WAN interface should be the ESXi network adapter connected to the Bridged port group, meaning it should have an IP address on your physical network. **Configure a static IP address or ensure this interface has a static DHCP mapping so that the address does not change.**
- The LAN interface should have an IP address of 172.16.1.1.
- The LAN network should be 172.16.1.0 with a /24 (255.255.255.0) subnet mask
- DHCP should be configured for this network with a scope of 172.16.1.10-172.16.1.254
 - 172.16.1.3-172.16.1.9 are available for setting static DHCP mappings
- The OPT1 interface should have an IP address of 172.16.2.1
- The OPT1 network should be 172.16.2.0 with a /24 (255.255.255.0) subnet mask
- DHCP should be configured for this network with a scope of 172.16.2.10-172.16.2.254
 - 172.16.2.2-172.16.2.9 are available for static DHCP mappings.
- Your management workstation accesses pfSense from the WAN interface. We created a firewall rule that serves as our anti-lock rule for the management interface.
 - Be aware that if the IP address of your workstation or WAN interface change, you'll need to reconfigure this rule from the ESXi console for pfSense.
 - Alternatively, you may wish to consider "jumpboxes".
- You should have an alias for RFC1918 networks configured for use with the firewall policy.
- You should have at least one good checkpoint you can revert to if you need to redo a step and somehow got lost/confused.

What's Next?

Since this pfSense VM is slightly different, you'll need to consult the section "Firewall Policies for Baremetal Hypervisors". The firewall rules are going to be a little bit different, depending on how you configured access to your ESXi lab network. In addition to the firewall rules, make sure that the pfSense VM is hosting all the services mentioned in the core network services guide to make your firewall fully functional and ready to handle your lab network. Once you verify that your firewall policies and services are properly configured, **It is highly advised you create a new checkpoint. THEN you can move on, and try your hand at setting up the remaining VMs.**

Your Turn

The pfSense vm required a lot of custom configuration, both in the ESXi web interface, as well as in the VM itself once the OS was installed. Now that we did that together, I'm going to have you create the rest of the virtual machines yourself -- with the exception of the metasploitable 2 VM since its a special case for reasons you will see momentarily. Below are a set of spec sheets to help you create the other virtual machines for the lab on your own. Think of it as a checklist you should be able to run through on your own.

All of the linux-based VMs in our lab are based on Ubuntu Server (Except for Kali Linux, which is based on the Debian Linux distro, which was the precursor to Ubuntu) so set up and configuration on our lab VMs should be mostly the same.

Kali Linux VM

Kali Linux is a popular penetration testing distro. Popular because hackers and script kiddies are lazy, and practically everything you need for performing a penetration test or joining anonymous is installed by default. Kali is going to be our loud, obnoxious attacker that we're going to use as a noise generator for the express purpose of generating events on our IPS VM. I want you to perform the following tasks:

- Download Kali Linux 64-bit here: <https://www.kali.org/downloads/>
- Upload your Kali Linux ISO to your ESXi server's datastore using the datastore browser
- Create a VM with the following settings:
 - Name the vm "kali"
 - Set the Guest OS family to "Linux"
 - Set the Guest OS version to "Debian GNU/Linux 8 (64-bit)"
 - Allocate 1 or 2 CPUs.
 - Allocate 4GB of RAM
 - Connect the VM's network adapter to the "IPS 1" port group
 - Allocate 80 GB of space for hard disk 1
 - Configure the VM for Thick Provisioning, lazily zeroed
 - Configure the Virtual Device Node to Sata Controller 0 (SATA (0:0))
 - Remove the SCSI controller
 - Add a CD/DVD drive configured to use the Kali Linux iso file you uploaded to the datastore.
 - Ensure that the "Connect at power on" is checked
 - Adjust the Video Card setting "Total Video Memory" to at least 32MB
 - This is required in order for the VM to render the GUI.
- Install Kali Linux
- After the install is completed, power off the VM and adjust the following settings:
 - Remove the CD/DVD drive
 - Remove the USB controller
 - Under Network Adapter 1, make sure to document the MAC address of this VM

- Log in to the pfSense web UI, navigate to Services > DHCP Server, and add a static mapping to the OPT1 interface DHCP, for the MAC address of adapter 1 for the Kali Linux VM; assign it the IP address 172.16.2.2, making sure to enter a description that tells you which VM this static mapping belongs to!
- Power on the Virtual Machine and do the following:
 - Verify that the static mapping for the VM's IP address worked; Open a terminal/shell and run ifconfig -a | less and verify that "eth0" or whatever the network adapter's name is in the VM was given the IP address you configured its static mapping for in pfSense.
 - If the VM does not have an ip address, run the "dhclient" command to have the VM request an IP address from the DHCP server.
 - Verify the virtual machine can reach the internet.
 - In a terminal/shell run the command ping -c 5 www.google.com
 - This will confirm DNS can resolve domain names, and that the VM can reach the internet
 - Update/patch the VM
 - In a terminal/shell, run the following, and depending on your network connection, etc., be prepared to wait:
 - export DEBIAN_FRONTEND=noninteractive; apt-get -q update; apt-get -y -q dist-upgrade
 - Create a snapshot for the VM

Siem VM

SIEM is shorthand for Security Intrusion Events Manager. This is fancy security nomenclature for "log aggregator"; we will be having our IPS VM log its events here. We're going to be running Splunk on this VM. Splunk is a commercial program for managing logs on a pretty large scale. By default, and with no licensing, Splunk only allows you to collect or "index" 500MB worth of logs per day however, there are /ways/ around this that we'll talk about later. I want you perform the following tasks to set up the SIEM VM:

- Download the latest Ubuntu Server 64-bit LTS release. In our case, this is Ubuntu Server 16.04.1, which can be found here: <http://www.ubuntu.com/download/server>
- Upload the Ubuntu Server ISO to your ESXi server's datastore
- Create a VM with the following settings:
 - Name the vm "SIEM"
 - Set the Guest OS family to "Linux"
 - Set the Guest OS version to "Ubuntu Linux (64-bit)"
 - Allocate 1 or 2 CPUs.
 - Allocate 4GB of RAM
 - Connect the VM's network adapter to the "Management" port group
 - Allocate 80 GB of space for hard disk 1
 - Configure the VM for Thick Provisioning, lazily zeroed
 - Configure the Virtual Device Node to Sata Controller 0 (SATA (0:0))

- Remove the SCSI controller
 - Add a CD/DVD drive configured to use the Ubuntu Linux iso file you uploaded to the datastore.
 - Ensure that the “Connect at power on” is checked
- Install Ubuntu Server
 - Be sure to install the command line utilities and SSH Server role when asked. You should not need to install any other roles or features for this server
- After the install is completed, power off the VM and adjust the following settings:
 - Remove the CD/DVD drive
 - Remove the USB controller
 - Under Network Adapter 1, make sure to document the MAC address of this VM
- Log in to the pfSense web UI, navigate to Services > DHCP Server, and add a static mapping to the LAN interface DHCP, for the MAC address of adapter 1 for the SIEM VM; assign it the IP address 172.16.1.3, making sure to enter a description that tells you which VM this static mapping belongs to!
- Power on the Virtual Machine and do the following:
 - Verify that the static mapping for the VM’s IP address worked; Open a terminal/shell and run ifconfig -a | less and verify that “eth0” or whatever the network adapter’s name is in the VM was given the IP address you configured its static mapping for in pfSense.
 - If the VM does not have an ip address, run the “dhclient” command to have the VM request an IP address from the DHCP server.
 - Verify the virtual machine can reach the internet.
 - In a terminal/shell run the command ping -c 5 www.google.com
 - This will confirm DNS can resolve domain names, and that the VM can reach the internet
 - Update/patch the VM
 - In a terminal/shell, run the following, and depending on your network connection, etc., be prepared to wait:
 - export DEBIAN_FRONTEND=noninteractive; apt-get -q update; apt-get -y -q dist-upgrade
 - These commands require root privileges. Run sudo su - and the password for the user account you initially created to become the root user.
 - Create a snapshot for the VM

IPS VM

This VM is going to be responsible for running the AFPACKET bridge between the IPS 1 and IPS 2 virtual switches. Perform the following tasks to install the IPS VM:

- If you installed the SIEM VM already, you should already have Ubuntu Server downloaded. If not, download the latest Ubuntu Server 64-bit LTS release. In our case, this is Ubuntu Server 16.04.1, which can be found here:
<http://www.ubuntu.com/download/server>

- Upload the Ubuntu Server ISO to your ESXi server's datastore (if it isn't already there)
- Create a VM with the following settings:
 - Name the VM "ips"
 - Set the Guest OS family to "Linux"
 - Set the Guest OS version to "Ubuntu Linux (64-bit)"
 - Allocate 1 CPU.
 - Allocate 2GB of RAM
 - Create two additional network adapters (in addition to the one ESXi provides you)
 - Connect network adapter 1 to the "Management" port group
 - Connect network adapter 2 to the "IPS 1" port group
 - Connect network adapter 3 to the "IPS 2" port group
 - Verify that that the promiscuous mode and forged transmit options are allowed on the IPS 1 and IPS 2 virtual switches. This is very important
 - Allocate 80 GB of space for hard disk 1
 - Configure the VM for Thick Provisioning, lazily zeroed
 - Configure the Virtual Device Node to SATA Controller 0 (SATA (0:0))
 - Remove the SCSI controller
 - Add a CD/DVD drive configured to use the Ubuntu Linux iso file you uploaded to the datastore.
 - Ensure that the "Connect at power on" is checked
- Install Ubuntu Server
 - When prompted, select the option to install security updates automatically
 - Be sure to install the standard system utilities and OpenSSH Server software packages when asked. You should not need to install any other roles or features for this server.
- After the install is completed, power off the VM and adjust the following settings:
 - Remove the CD/DVD drive
 - Remove the USB controller
 - Under Network Adapter 1, 2, and 3, make sure to document the MAC addresses and network(s) they are connected to.
- Log in to the pfSense web UI, navigate to Services > DHCP Server, and add a static mapping to the LAN interface DHCP, for the MAC address of the network adapter connected to the "Management" network for the ips VM; assign it the IP address 172.16.1.4, making sure to enter a description that tells you which VM this static mapping belongs to!
- Power on the Virtual Machine and do the following:
 - Verify that the static mapping for the VM's IP address worked; Open a terminal/shell and run ifconfig -a | less and verify that "eth0" or whatever the network adapter's name is in the VM was given the IP address you configured its static mapping for in pfSense.
 - If the VM does not have an ip address, run the dhclient command to have the VM request an IP address from the DHCP server.
 - Verify the virtual machine can reach the internet.

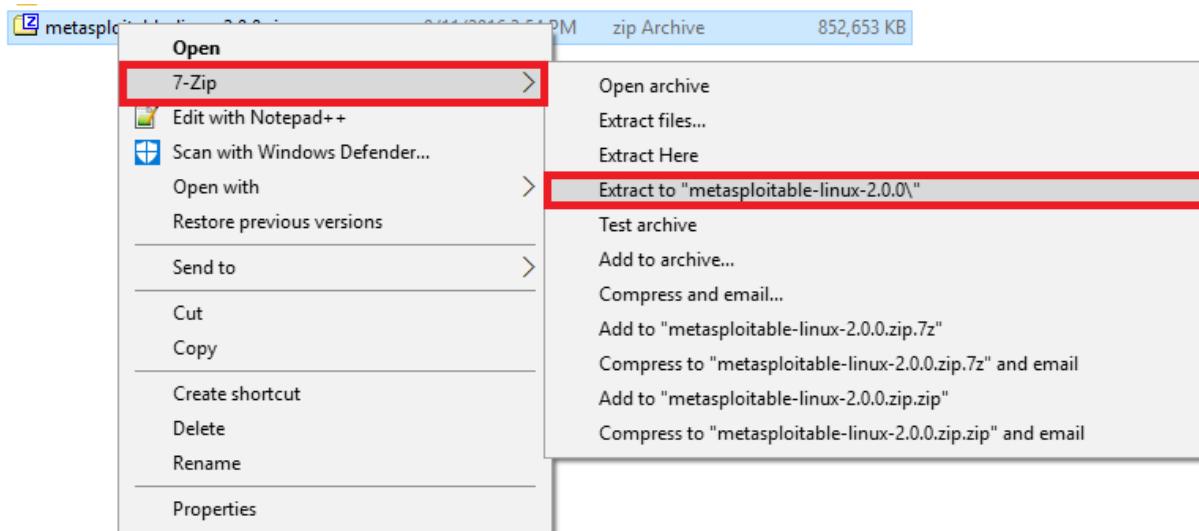
- In a terminal/shell run the command `ping -c 5 www.google.com`
 - This will confirm DNS can resolve domain names, and that the VM can reach the internet
- Update/patch the VM
 - In a terminal/shell, run the following, and depending on your network connection, etc., be prepared to wait:
 - `export DEBIAN_FRONTEND=noninteractive; apt-get -q update; apt-get -y -q dist-upgrade`
 - These commands require root privileges. Run `sudo su -` and the password for the user account you initially created to become the root user.
 - Create a snapshot of the VM

Metasploitable 2

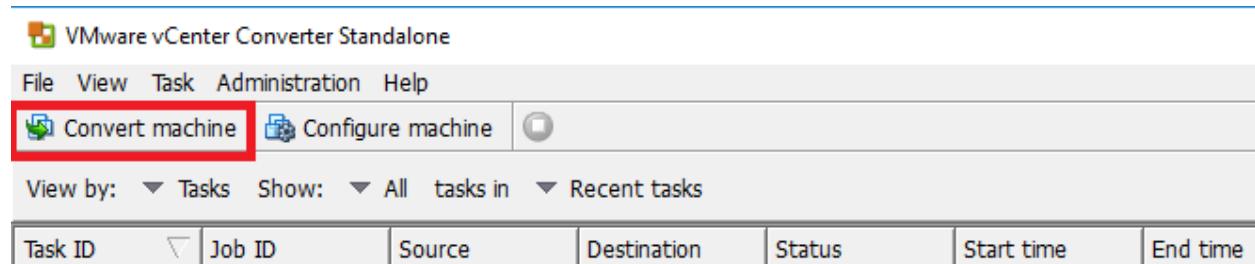
Setting up Metasploitable 2 on ESXi is slightly more complex than the just creating a virtual machine. While there are ways to get the VM on to ESXi, The easiest and fastest way to do so will be through VMware's vCenter converter standalone. To download and use this tool, you'll need a Windows system, and an account on VMware's website, which you should already have from downloading ESXi earlier. Go download the installer here:

<https://my.VMware.com/group/VMware/evalcenter?p=converter>

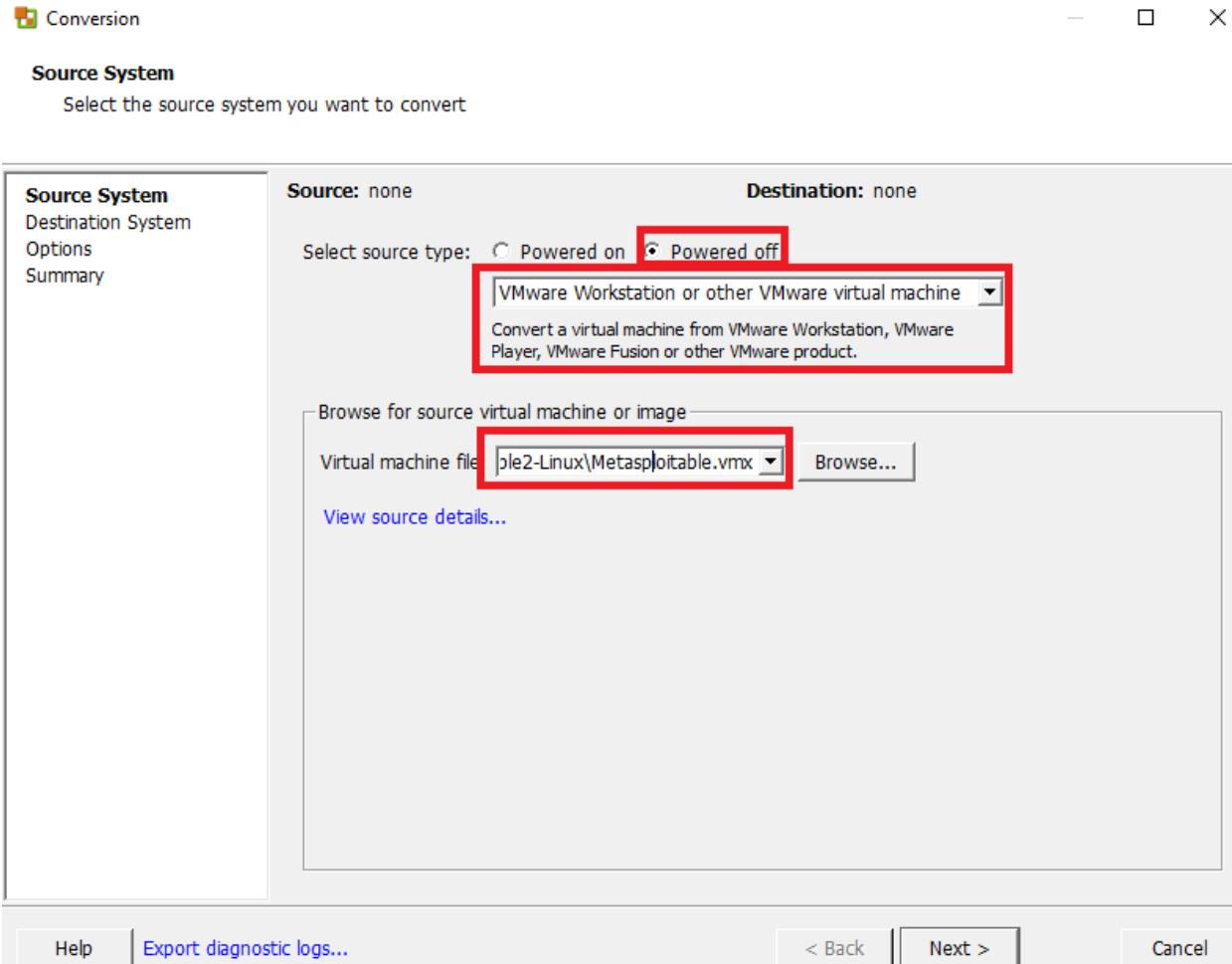
After downloading and installing VMware converter, go and download metasploitable 2 from <https://sourceforge.net/projects/metasploitable/files/Metasploitable2/metasploitable-linux-2.0.0.zip/download>. Once the zip file has been downloaded, you'll need a compression utility to unzip the zip file. I recommend using 7-zip (<http://www.7-zip.org/download.html>). Once downloaded, I moved the zip file to D:\VMs, where the rest of my VM files are stored. We'll need to extract the zip file for use with the converter tool. Right click on the zip file and select 7-zip > Extract to "metasploitable-linux-2.0.0\"



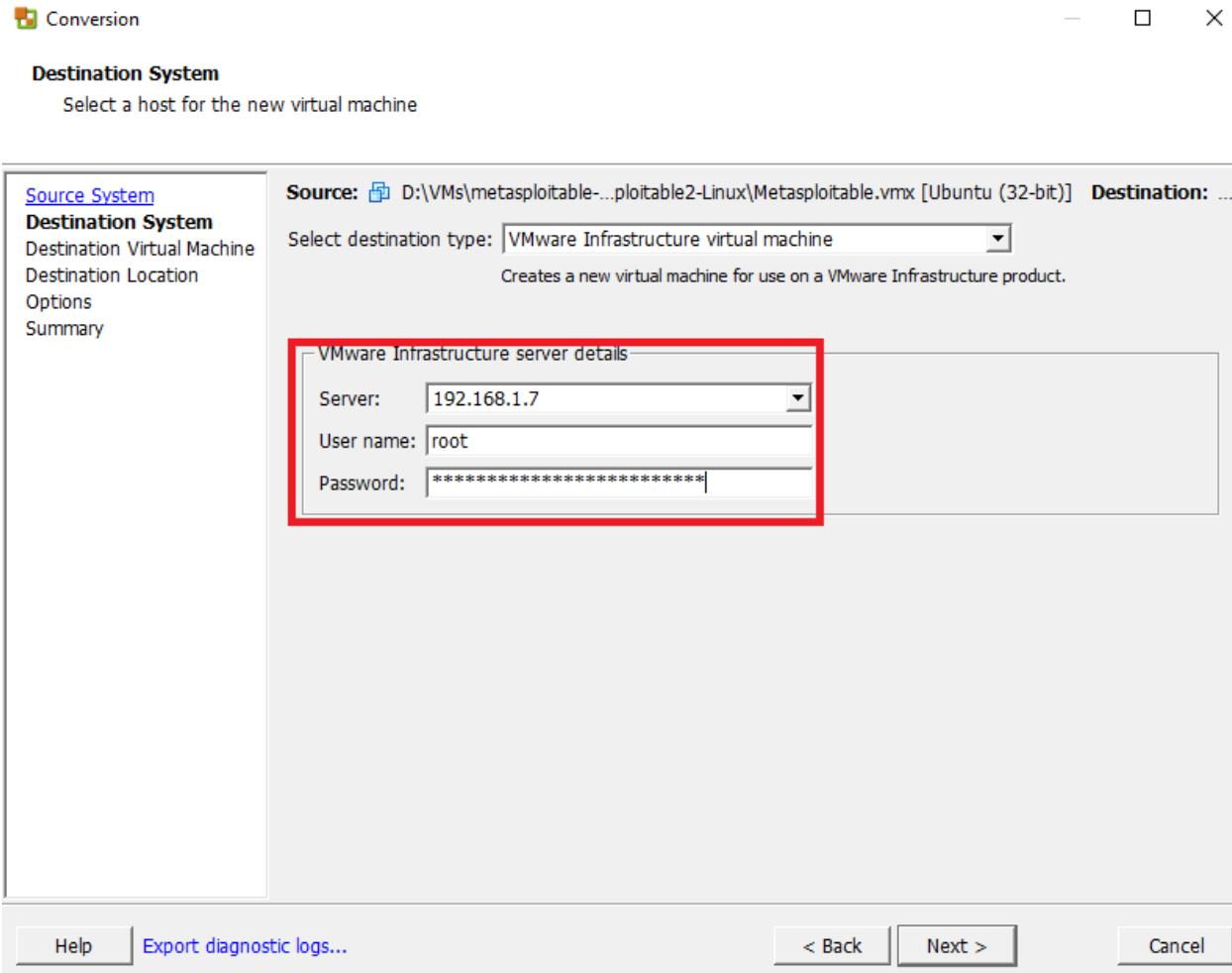
Next, open VMware vCenter converter standalone and click “Convert machine”.



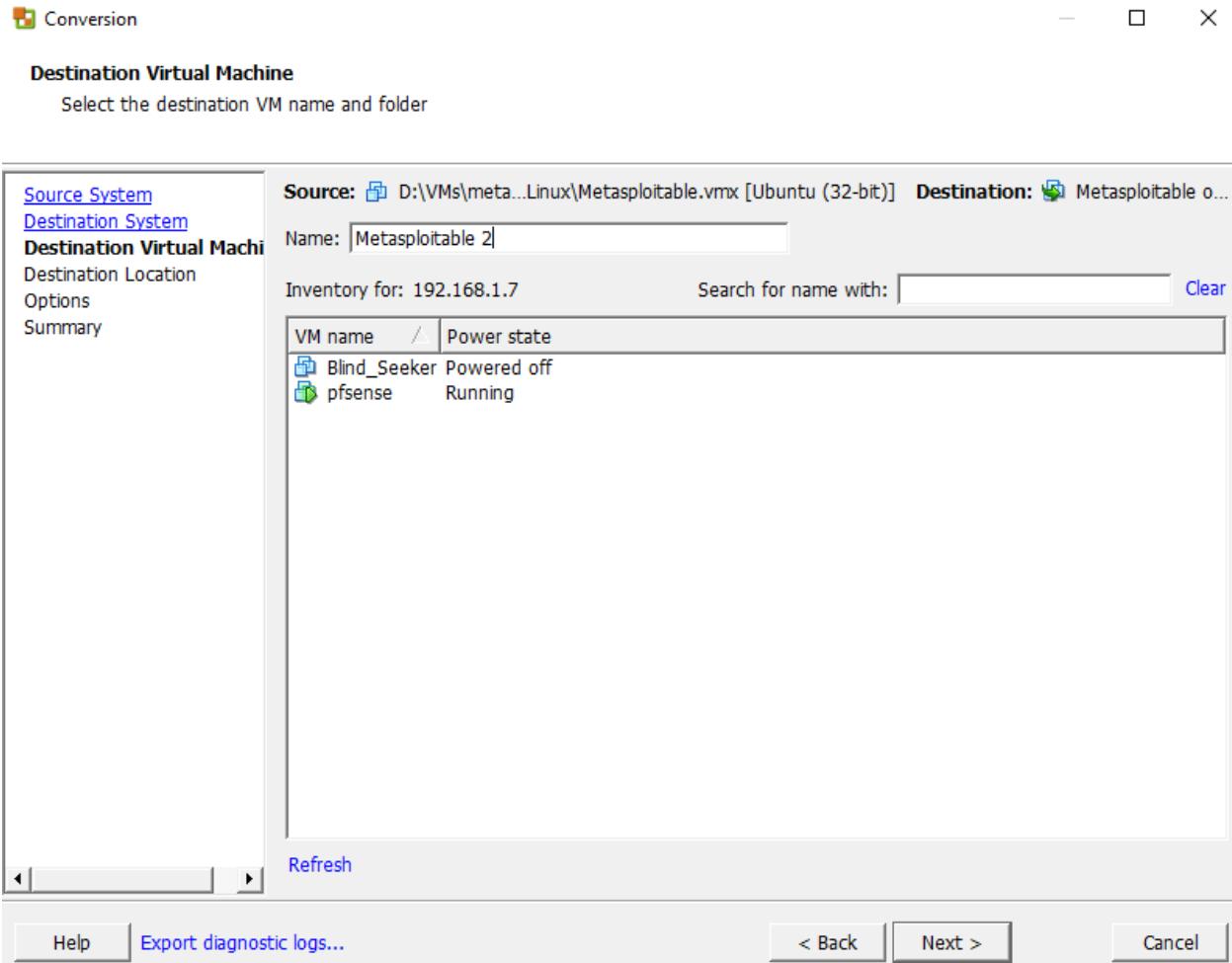
You're guided through a wizard with a variety of steps. The first step is choosing the source system to convert. Click the “Powered off” radio button, then in the drop-down select “VMware Workstation or other VMware virtual machine”. In the section below entitled “Browse for source virtual machine or image”, click the browse button. Browse to the directory you extracted Metasploit 2 in. You want to find the “Metasploitable.vmx” file, select it. In my case, this was located in D:\VMs\metasploitable-linux-2.0.0\Metasploitable2-Linux\Metasploitable.vmx. After selecting the vmx file, click Next.



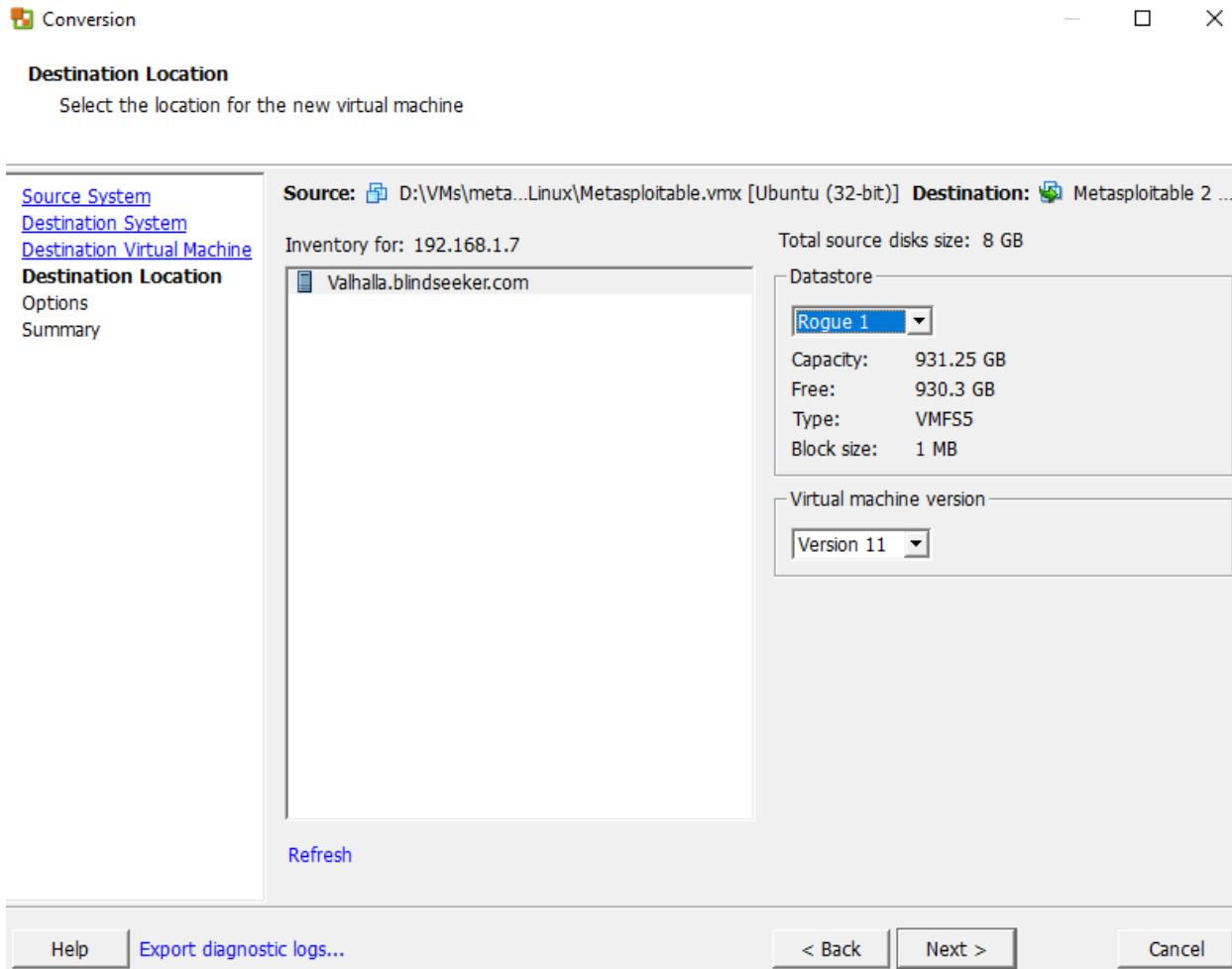
On the next screen, we need to put in details and credentials for our ESXi server; the converter is going to do an on the fly conversion and upload. Enter the IP address or host, and username/password in the “VMware Infrastructure server details” section, then click Next.



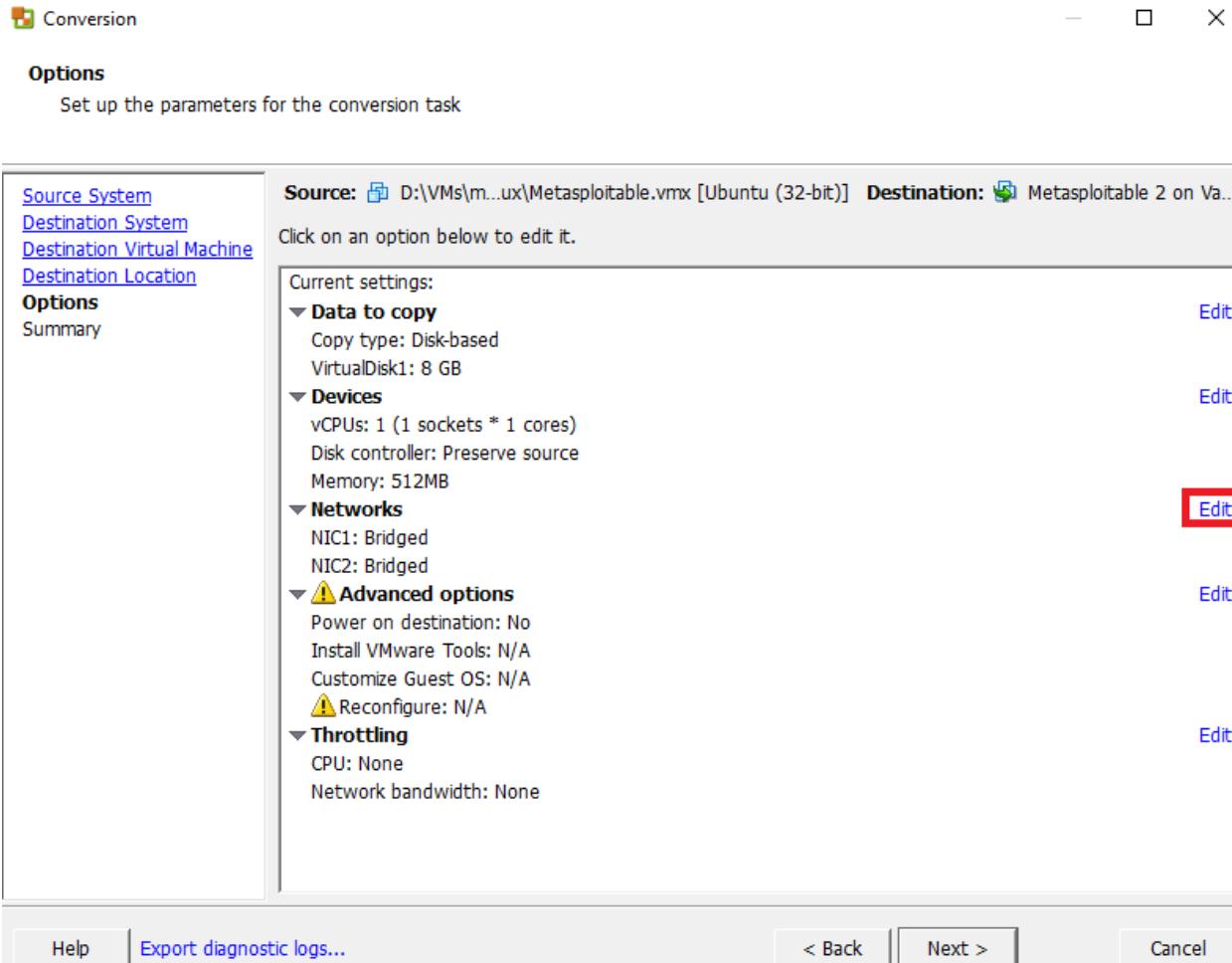
If it was able to successfully connect and authenticate, you'll get a listing of VMs on the next screen. In the name field, Change the name to "Metasploitable 2", then click Next.



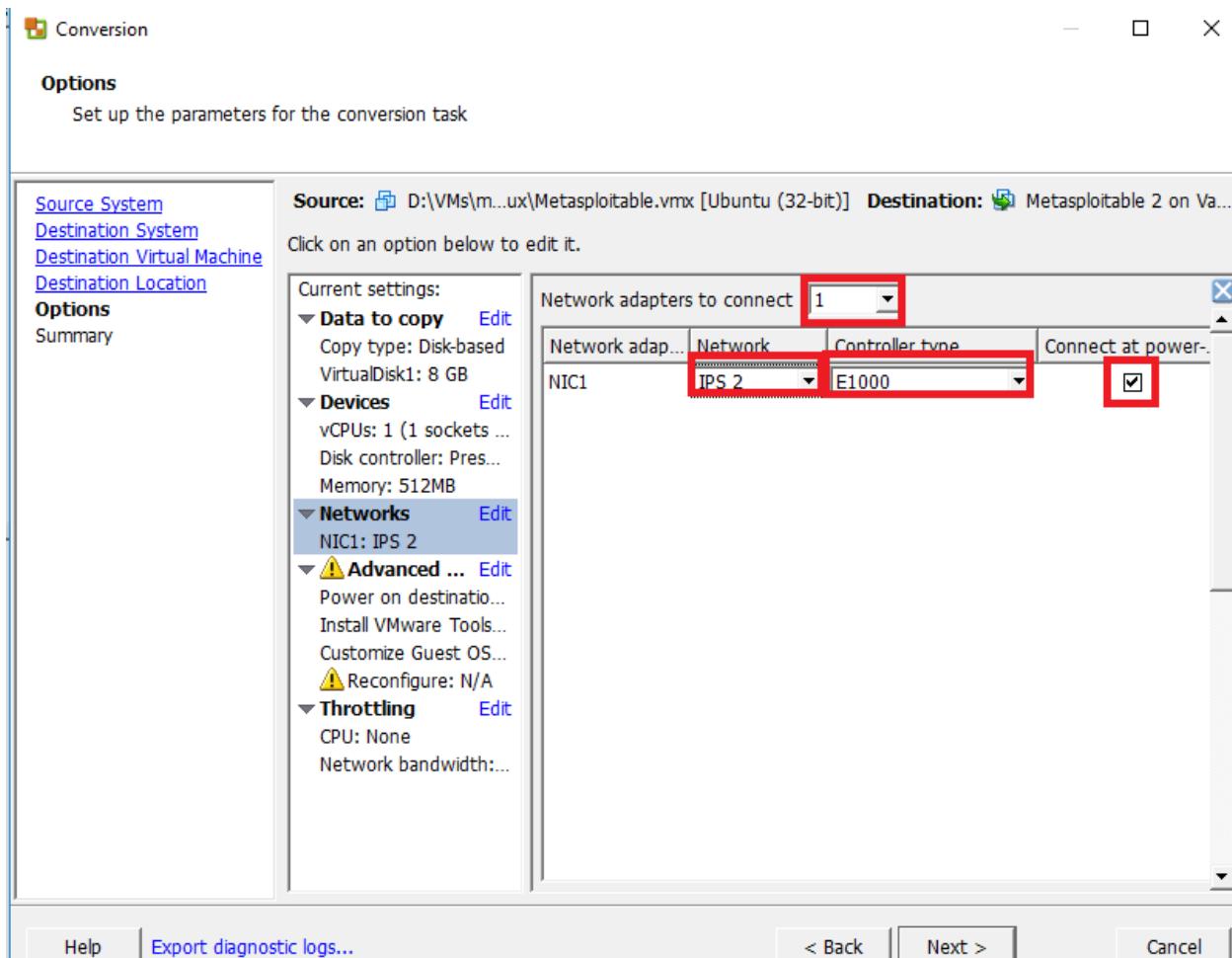
The next screen will have you choose what datastore you want to store the VM in and/or the virtual machine hardware version. In this case, I'm going to store my VM on the "Rogue 1" datastore. Click Next when you have selected your datastore.



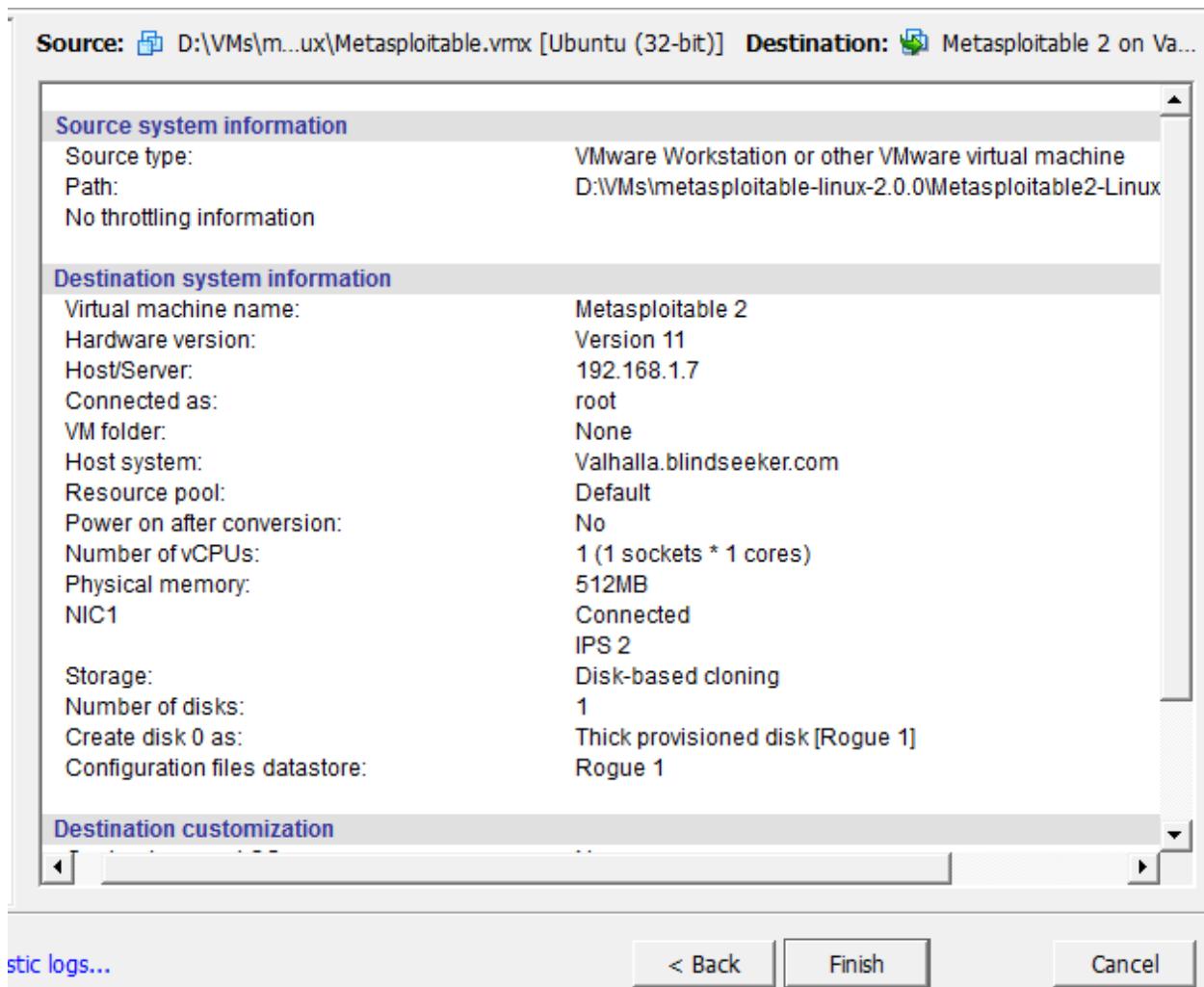
The next screen has you confirm and set various options for the VM. We need to edit the “Networks” option, remove one of the network cards and change the port group the VM will be attached to “IPS 2”. Click the blue edit text to the right of “Networks”.



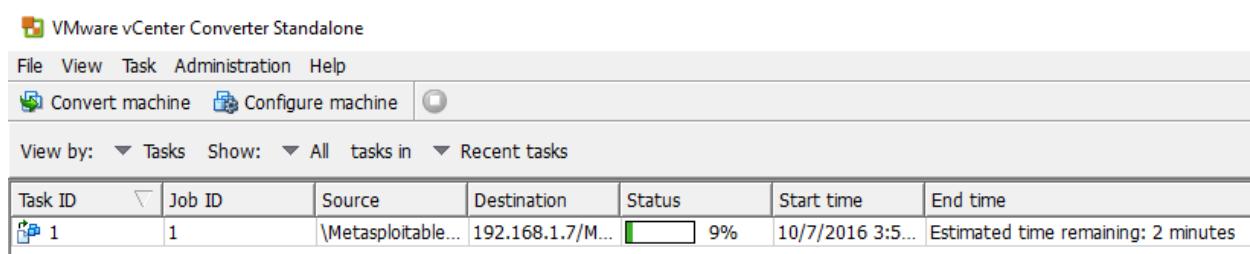
In the “Networks to connect” drop-down, select 1. For NIC1, set the “Network” drop-down to “IPS 2”. For the “Controller Type” select “E1000”. Ensure that the “Connect at power on” checkbox is checked. Afterwards, click Next.



The final screen is a summary of all the configuration settings we will be using to convert the VM. The source, the destination, hardware settings, OS settings, etc. Click Finish to begin the conversion process.



The main application window for converter will update, letting you know the progress of the VM conversion. Wait for it to get done. Depending on hardware, this could take a while.



After the conversion is complete, the new VM should appear in the Virtual Machines list on your ESXi web interface.



So... What Now?

Baremetal hypervisors are quite a bit different than hosted hypervisors, however some of the same concepts apply. You may still consider reading “Defense in Depth for Windows Hosted Hypervisors” for some ideas on how to harden your Management workstation against possible attack by your ESXi lab VMs, but some of the configurations may or may not apply. For instance, in a home environment with other windows systems, or a work environment, unbinding network protocols may or may not make sense depending on your situation. Setting Windows firewall rules may or may not make sense as well. Read the sections to see if the security recommendations apply to you.

Additionally, consider reading the section on “Remote Lab Management”, for the management operating system of choice you are using. You may also want to consider reading up on jump boxes if you need to ensure consistent access to your lab environment, but the ip address and/or location of your management system constantly changes.

Once you’ve hardened your management system sufficiently, and configured remote access to your liking, move on to configuring the IPS and SIEM vms.

pfSense Firewall Rules and Network Services Guide

In all of the hypervisor setup guides, I guided you through how to perform initial setup of the pfSense VM. At this point, performing the finishing touches is more or less the same across all the hypervisors. This chapter will guide you through how to finish setting up pfSense with firewall rules to enforce segmentation of the lab networks from each other, as well as physical networks you have your lab attached to. The end goal here is to have a self-contained lab that only your hypervisor host (for hosted hypervisor users), or management workstation/jump box (for baremetal hypervisor users) will be able to access. After we finish setting up firewall rules, we will discuss setting up core network services for the various network segments, focusing on NTP (Network Time Protocol), DHCP (Dynamic Host Configuration Protocol -- though really, the setup guides have already covered this), DNS (Domain Name Service), and setting up your firewall to act as a squid proxy server (this is optional).

Network Configuration - Segmentation and Firewall Config

Firewall Rules for the WAN Network

States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
■ ✘ 0/55.47 MiB	IPv4+6 *	*	*	*	*	*	none		explicit deny any	

The illustration above is a screen capture from the pfSense web interface. This is the firewall policy for the “WAN” network. This interface of the pfSense firewall is connected to the bridge vswitch for internet access. pfSense has an implicit deny any/any firewall rule, but I made an explicit deny any/any rule. The purpose of this firewall rule is to prevent hosts outside of the virtual network from initiating any connections to any VMs inside of our lab network. This firewall rule does NOT prevent the VMs in our network from interacting with hosts on the internet however; firewall rules for that are handled on the other interfaces.

Firewall Rules for the Management Network

States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
✘ 0/22 KiB	*	Reserved Not assigned by IANA	*	*	*	*	*		Block bogon networks	
■ ✓ 0/28 KiB	IPv4 TCP	172.16.1.0/24	*	172.16.1.1	3128	*	none		Proxy Access	
■ ✓ 0/56 KiB	IPv4 UDP	172.16.1.0/24	*	172.16.1.1	123 (NTP)	*	none		allow ntp in	
■ ✓ 3/176 KiB	IPv4 UDP	172.16.1.0/24	*	172.16.1.1	53 (DNS)	*	none		allow dns access	
■ ✓ 0/47 KiB	IPv4 TCP	172.16.1.2	*	172.16.2.2	47129	*	none		Bahamut to KalAYY SSH	
■ ✓ 1/1.50 MiB	IPv4 TCP	172.16.1.2	*	172.16.1.1	443 (HTTPS)	*	none		anti-lockout(HTTPS)	
■ ✘ 0/90 KiB	IPv4 *	*	*	RFC 1918	*	*	none		deny access to LAN	
■ ✓ 12/15.52 MiB	IPv4 TCP	172.16.1.0/24	*	*	443 (HTTPS)	*	none		allow HTTPS out (patching)	
■ ✘ 0/9 KiB	IPv4+6 *	*	*	*	*	*	none		Deny All	

This is the firewall policy for the pfSense interface connected to the management switch. First and foremost, **Firewall Rule Order is important**. If there is a rule that would deny your network traffic placed ABOVE a rule that allows certain network traffic, that deny rule will be checked

first, and the traffic WILL be dropped. **pfSense (and most other firewalls) evaluates firewall rules in a policy from top to bottom.** Be mindful of this!

The rules that allow access from the 172.16.1.0/24 net to 172.16.1.1 are to allow the VMs on the management network to use the pfSense gateway for network services; specifically NTP (port 123/udp), DNS (port 53/udp), and HTTP proxy (port 3128/tcp). By default, pfSense has an “anti-lockout” rule that allows any system on the LAN network to access the firewall over HTTPS. I didn’t like how open this rule was, so I manually made an anti-lockout rule that only allows access to the web interface from a single IP address on this network 172.16.1.2 -- the IP address I statically set to the virtual interface allocated to the host operating system attached to this network.

When the management vswitch was created, it was created in such a way that the host OS got a virtual network card directly connected to this network. So we can use this virtual interface to directly interact with the SIEM and the IPS’ management interface on this same network segment without having to worry about pfSense firewall rules to allow our Host OS to manage those assets. However, you probably did notice the random rule that allows access from 172.16.1.2 to 172.16.2.2 over port 47129/tcp. This rule is to allow SSH connectivity to the Kali Linux box. I changed the default listening port for SSH on Kali to a random port out of habit (in this case... port 47129). I’m just weird like that. If you’re not weird like me, allowing port 22/tcp (the default for SSH) from 172.16.1.2 to 172.16.2.1 should allow you to SSH to your Kali VM with ease.

The last few rules exist to explicitly deny access to any local networks (RFC 1918 addresses - 172.16.0.0/12, 192.168.0.0/16, and 10.0.0.0/8). This ensures VMs in this network cannot talk to the physical network or the other virtual networks without having allow rules to specifically allow this defined above. The rule to allow HTTPS traffic outbound is to allow machines in this network access to the internet over HTTPS to allow for application patches and updates. Most update systems use HTTP, FTP or HTTPS for acquiring updates. The pfSense HTTP proxy handles both HTTP and FTP, so I only needed to define a rule for HTTPS traffic explicitly. This rule is placed BELOW the rule to block RFC 1918 addresses in order to enforce the concept of not allowing any of these VMs to talk to the other networks, only allowing for HTTPS connectivity to the internet. Finally, I made an explicit deny any/any catch-all rule to block anything else not defined in the firewall policy.

Firewall Rules for the IPS Network

Firewall / Rules / OPT1										
Floating	WAN	LAN	OPT1	Rules (Drag to Change Order)						
States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
✗ 0/22 KiB	*	Reserved Not assigned by IANA	*	*	*	*	*		Block bogon networks	
■ ✗ 0/5 KiB	IPv4 TCP	*	*	*	80 (HTTP)	*	none		deny http out	
■ ✗ 0/0 B	IPv4 TCP	172.16.2.0/24	*	*	21 (FTP)	*	none		Deny FTP out	
■ ✓ 0/116.10 MiB	IPv4 TCP	172.16.2.0/24	*	172.16.2.1	3128	*	none		Squid Proxy	
■ ✓ 0/0 B	IPv4 TCP	172.16.2.0/24	*	172.16.2.1	123 (NTP)	*	none		ntpd service access	
■ ✓ 0/13 KiB	IPv4 UDP	172.16.2.0/24	*	172.16.2.1	53 (DNS)	*	none		allow dns in	
■ ✗ 0/167 KiB	IPv4 *	*	*	RFC 1918	*	*	none		disallow access to LAN	
■ ✓ 0/14 KiB	IPv4 *	*	*	*	*	*	none		allow OPT1 to any	

Add Add Delete Save Separator

This is the IPS network firewall policy. It's mostly the same as the LAN network policy, but with a few more accommodations.. First and foremost, at some point I'm going to be turning this into a malware analysis network, so instead of only allowing HTTP, HTTPS, and FTP outbound, I have a catch-all rule at the end of this policy to ALLOW anything not explicitly defined in this policy. Why? Because malware is weird and most of the time, will not use common service ports to connect out to the internet, so that last rule is there so I have the option of observing malware that uses unusual ports for C2 activity.

Because the last rule allows any traffic outbound, I have to explicitly deny HTTP and FTP outbound to force systems on this network to use the proxy on port 3128, so all HTTP and FTP activity can be logged/controlled. Just like on the management network, there are allow rules to allow clients on this network to access the proxy for DNS, NTP, and HTTP Proxy services, and there is also the rule that denies systems on this network from talking to other RFC1918 hosts to enforce segmentation/boundaries as well; this means machines infected with malware shouldn't be able to interact with hosts on other network segments.

Network Configuration - Core Network Services

As hinted above, I have pfSense with a few core services enabled.

NTP

NTP is shorthand for Network Time Protocol. It is a service that allows systems to synchronize their internal clocks to a known good time source. It prevents system clocks from drifting, which has very important connotations for incident response, since you need to ensure that the timestamps on all the data collected are accurate.

On the pfSense webUI, navigate to Services > NTP. make sure that the LAN and OPT1 interfaces are highlighted to ensure NTP services are available for those two networks (in Windows, hit the ctrl key, then click LAN and then click OPT1 to highlight them both). By default, pfSense runs their own NTP pool, and in most cases, this should work well enough. However, it never hurts to know alternatives. Visit <http://www.pool.ntp.org/en/> and find the region of the world you live in. In my case, I select “North America” and on the next page, I’m presented with the following NTP server information:

- 0.north-america.pool.ntp.org
- 1.north-america.pool.ntp.org
- 2.north-america.pool.ntp.org
- 3.north-america.pool.ntp.org

Using this information, you can click the green “Add” button to add additional NTP servers. I added all four servers, and clicked “Delete” next to the default NTP server pool that pfSense provides. After making changes/enabling the service for the LAN and OPT1 interfaces, click Save to enable the service.

NTP Server Configuration

Interface: **WAN** **LAN** **OPT1**

Interfaces without an IP address will not be shown.
Selecting no interfaces will listen on all interfaces with a wildcard.
Selecting all interfaces will explicitly listen on only the interfaces/IPs specified.

0.north-america.pool.ntp.org	<input type="checkbox"/> Prefer	<input type="checkbox"/> No Select	
1.north-america.pool.ntp.org	<input type="checkbox"/> Prefer	<input type="checkbox"/> No Select	
2.north-america.pool.ntp.org	<input type="checkbox"/> Prefer	<input type="checkbox"/> No Select	
3.north-america.pool.ntp.org	<input type="checkbox"/> Prefer	<input type="checkbox"/> No Select	

Add

For best results three to five servers should be configured here.
The prefer option indicates that NTP should favor the use of this server more than all others.
The no select option indicates that NTP should not use this server for time, but stats for this server will be collected and displayed.

Orphan Mode

Orphan mode allows the system clock to be used when no other clocks are available. The number here specifies the stratum reported during orphan mode and should normally be set to a number high enough to insure that any other servers available to clients are preferred over this server (default: 12).

NTP Graphs Enable RRD graphs of NTP statistics (default: disabled).

Logging Log peer messages (default: disabled).

DHCP

If you followed the instructions on how to configure the network interfaces on the pfSense VM, then DHCP should be enabled. The LAN network should be serving addresses from 172.16.1.10 to 172.16.1.254, and OPT1 should be serving addresses from 172.16.2.10 to 172.16.2.254. While the CLI lets you set the IP address scopes easily enough, the web interface allows a lot more granular control. DHCP configuration can be controlled via the Services > DHCP Server page on the web UI.

The screenshot shows the 'General Options' section of the LAN interface configuration. The 'Enable' checkbox is checked, and the 'Enable DHCP server on LAN interface' option is selected. The 'Range' field is set to 172.16.1.10 - 172.16.1.254. The 'From' value is 172.16.1.10 and the 'To' value is 172.16.1.254. A red box highlights the 'Range' input field.

As you can see, there are sub-pages for the LAN and OPT1 interfaces. Ensure the “Enable DHCP Server on the LAN and OPT1 tabs are checked, and confirm the IP address range for DHCP is correct. By default, DHCP assumes that the IP address of the pfSense firewall on the LAN and OPT1 network(s) is the default gateway, and is also the DNS server for that network segment as well. For the purposes of our lab, this is perfect since pfSense handles routing the network traffic, and also has the DNS Resolver service enabled by default and will use the DNS servers we configure in the initial pfSense setup wizard for forward DNS requests.

pfSense also allows us to configure static DHCP allocations. You can add static mappings for any network pfSense is running DHCP for. The static mappings have to be for IP addresses that are still within the network that interface is in, but NOT in the DHCP network range. In the illustration above, the DHCP range is from 172.16.1.10-172.16.1.254. 172.16.1.1 and 172.16.1.2 are already used by pfSense and our host system’s network interfaces. That means we can only use 172.16.1.3-172.16.1.9 for static DHCP allocations, unless the standard DHCP range is adjusted.

Static mappings are configured by MAC address. This ensures that virtual machines with a specific MAC address connected to a specific network ALWAYS get the same IP address from the DHCP server.

DHCP Static Mappings for this Interface				
Static ARP	MAC address	IP address	Hostname	Description
	00:15:5d:01:11:12	172.16.1.3	squadsight	IPS sensor management interface
	00:15:5d:01:11:19	172.16.1.4	Avenger	The logger.

As an example, the following illustration is the static DHCP mapping I use for “squadsight”, the IPS vm’s management interface, connected to the “LAN/management” network. All you have to do is identify the mac address for the network interface you want to statically map, then choose what IP address you want that interface to always get, then save the settings. Note that the DNS server configuration is redundant and you could get away with leaving those fields blank.

Static DHCP Mapping on LAN

MAC Address	00:15:5d:01:11:12	<input type="button" value="Copy My MAC"/>		
MAC address (6 hex octets separated by colons)				
Client Identifier				
IP Address	172.16.1.3	If an IPv4 address is entered, the address must be outside of the pool. If no IPv4 address is given, one will be dynamically allocated from the pool.		
Hostname	squadsight	Name of the host, without domain part.		
Description	IPS sensor management interface	A description may be entered here for administrative reference (not parsed).		
ARP Table Static Entry	<input checked="" type="checkbox"/> Create an ARP Table Static Entry for this MAC & IP Address pair.			
WINS Servers	WINS 1	WINS 2		
DNS Servers	172.16.1.1	DNS 2	DNS 3	DNS 4
Note: leave blank to use the system default DNS servers - this interface's IP if DNS Forwarder or Resolver is enabled, otherwise the servers configured on the General page.				
Gateway				
The default is to use the IP on this interface of the firewall as the gateway. Specify an alternate gateway here if this is not the correct gateway for the network.				

DNS Resolver

By default, pfSense has the DNS Resolver service enabled. To verify this, navigate to Services > DNS Resolver. Verify that the “Enable DNS resolver” checkbox is checked, and that “All” interfaces are selected for “Network Interfaces” and “Outgoing Network Interfaces”

To confirm what DNS servers pfSense will forward queries to if it can't resolve the domain name, navigate to System > General Setup and view the DNS Server Settings. In my case, I'm using 8.8.8.8 (Google DNS), 4.2.2.2 (Level 3 DNS) and 192.168.1.1 (the Default Gateway/DNS for my physical LAN).

Squid Proxy

pfSense supports utilizing the squid HTTP proxy; all you need to do is install the package for it. Having a proxy installed enables us to have granular control over the HTTP and FTP traffic allowed in and out of the lab network and, at a later date, we could choose to log the requests made and have them sent to our siem vm for review event data. To install squid, navigate to System > Package Manager and click on “Available Packages”. On the search bar, type in “squid” and click the green Install button to the right of the “squid” package.

The screenshot shows the pfSense Package Manager interface. At the top, there are two tabs: "Installed Packages" and "Available Packages", with "Available Packages" being the active tab and highlighted with a red box. Below the tabs is a search bar with the word "squid" typed into the "Search term" field, also highlighted with a red box. The search bar includes a "Search" button and a "Clear" button. The main area is titled "Packages" and lists three packages:

Name	Version	Description	Action
Lightsquid	3.0.4	LightSquid is a high performance web proxy reporting tool. Includes proxy realtime statistics (SQStat). Requires Squid3 package.	+ Install
squid	0.4.23	High performance web proxy cache (3.5 branch). It combines Squid as a proxy server with its capabilities of acting as a HTTP / HTTPS reverse proxy. It includes an Exchange-Web-Access (OWA) Assistant, SSL filtering and antivirus integration via C-ICAP.	+ Install
squidGuard	1.14_3	High performance web proxy URL filter.	+ Install

Each row shows the package name, version, a brief description, and a green "+ Install" button. The "squid" package entry has its name and the "+ Install" button both highlighted with red boxes.

This should kick off the installer. When the installation is done, the installer log in the middle of the screen will read “Success”. To confirm that squid was installed properly, navigate to Services. If “Squid Proxy Server” is an option under Services, the installation was successful. Click on “Squid Proxy Server.” On the page, Click the checkbox next to “Enable Squid Proxy”, then under the Proxy Interface(s) pane, make sure that the “LAN” and “OPT1” interfaces are highlighted. This ensures that the proxy is enabled for both of these networks.

Note that with the firewall rules we are using for the management and IPS networks, you MUST use the proxy for all regular HTTP and FTP traffic. This could make using tools like wget or apt-get for setting up the other virtual machines problematic. In order to use command line applications that use HTTP or FTP for application updates in Linux/BSD environments, make sure that you familiarize yourself with the http_proxy variable and its use for linux/BSD cli tools. I found this great tutorial on how to set the http_proxy variable to ensure that this isn't a problem. <http://www.putorius.net/2012/09/how-to-set-httpproxy-variable-in-linux.html>

Note: This guide assumes the proxy is located on port 8080/tcp. Our squid proxy is located on port 3128/tcp. Keep this in mind!

Defense in Depth for Windows Hosted Hypervisors

For VMware Workstation on Windows, VirtualBox on Windows or Microsoft Client Hyper-V, you'll have created virtual networks that add a virtual network adapter to the Windows host operating system. Each hypervisor has a different name for this network card:

Client Hyper-V calls this interface “vEthernet (vswitch name)” (in our case, vEthernet (Management))

VirtualBox calls this interface “VirtualBox Host Only-Network”

VMware Workstation calls this “VMware Network Adapter vmnetx” (where x is the vmnet or vm network the host is attached to. Usually, the default host-only network is vmnet1)

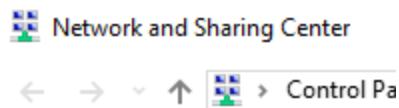
While in the case of all these hypervisors you have the option of creating internal virtual networks or vswitches without adding a virtual network card to the host, it's convenient to add the adapter to the host to be able to access assets in the virtual lab without having to add multiple persistent routes, or firewall rules on the WAN interface in pfSense. Not only that, but if you are running your VM lab on a laptop or an a device that constantly moves from one network to another, you may not always have the ability to control the IP address of your physical system for accessing your virtual assets. Adding a virtual adapter to the host for the Management network (LAN network in pfSense) allows your Windows hypervisor host to directly interact with the ips and siem vm without having to worry about fire rules in pfSense, or adding network routes on the host system. As previously stated, to reach the kali VM on the IPS network, you need a firewall rule on the LAN interface to allow SSH access to the IP address of the kali VM. Additionally, you'll need to add a network route on the Windows host in order to tell the host system where to send its packets to interact with the kali host.

While having this network card directly attached to the management network is convenient, it also means that you are potentially exposing the host system to the devices on the management network. While we have the pfSense firewall to enforce boundaries and segmentation between the various networks, taking additional security measures ensures we are not putting all of our eggs into one basket. The following sections will instruction you on how to unbind various Microsoft network protocols from the virtual network adapter, and how to implement a firewall rule on the windows host firewall that denies inbound connections to the virtual network adapter, but will allow the virtual adapter to make connections to the virtual network as necessary.

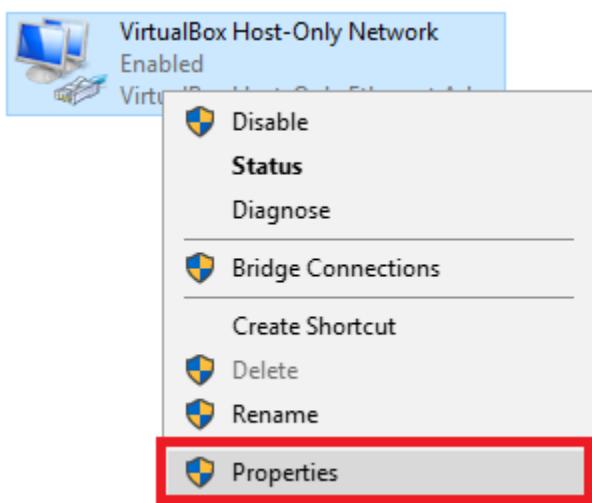
Unbinding Network Protocols on Windows Virtual Adapters

For the sake of simplicity, I will be demonstrating this section using the “VirtualBox Host-Only Network” as the adapter for this tutorial. Substitute this with “vEthernet(Management)” or “VMware Network Adapter vmnet1” as necessary. If there are any special steps or considerations that should be noted for VMware or Client Hyper-V users, I will advice on differences as we come across them.

Navigate to the Network and Sharing center, and click the “Change adapter Settings” option on the left of the screen. Alternatively you could directly navigate to “View Network Connections”.



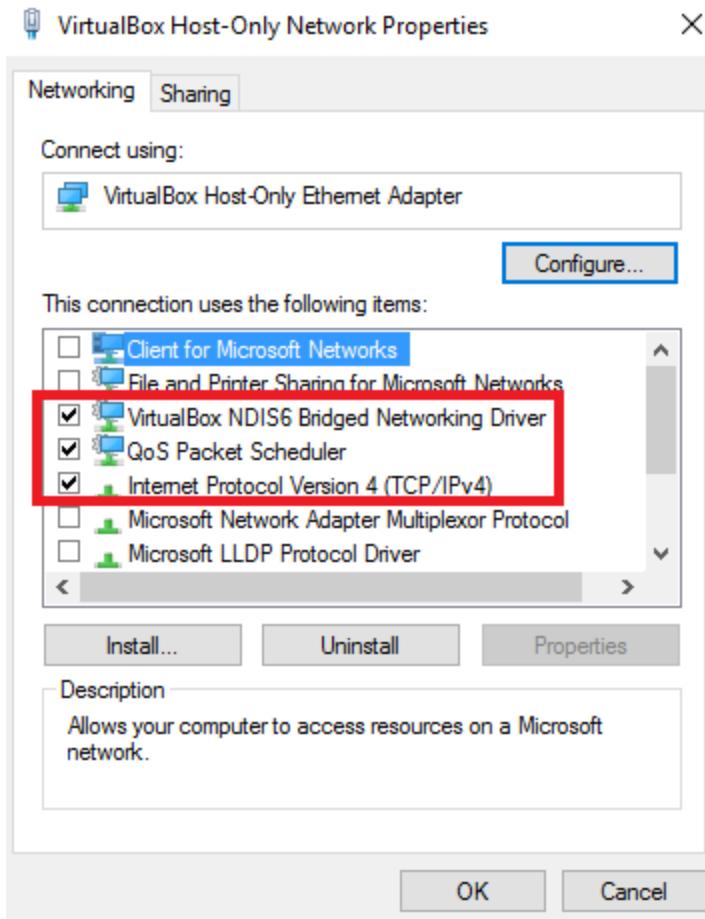
Find the adapter labeled “VirtualBox Host-Only Network”, right click it, and select “Properties”.



On the next window, under the pane entitled “This connection uses the following items:” uncheck everything except for:

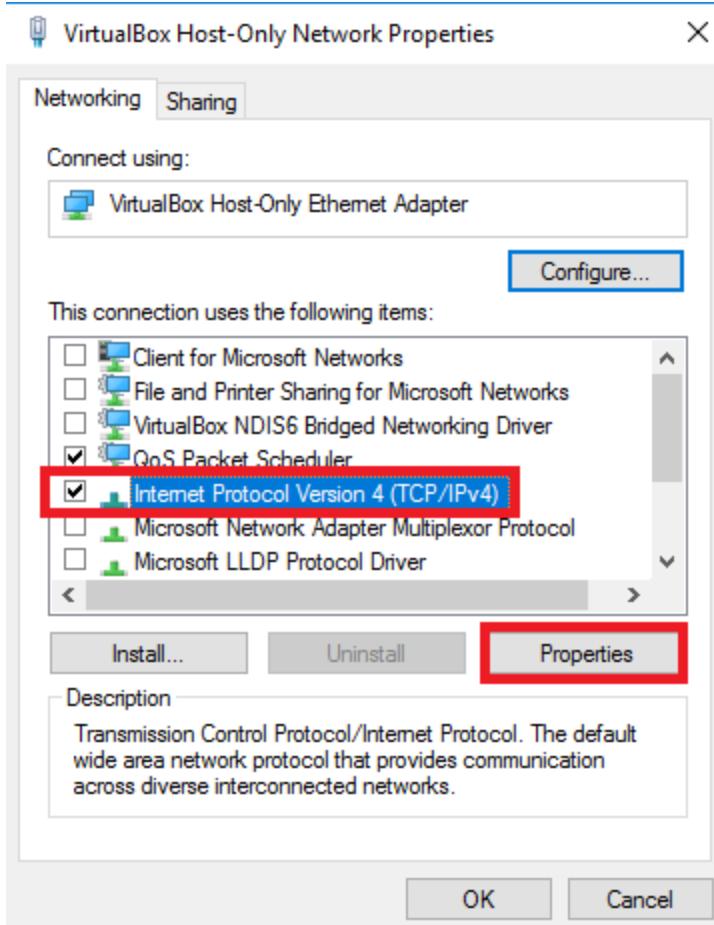
- VirtualBox NDIS6 Bridged Networking Driver
- QoS Packet Scheduler
- Internet Protocol Version 4 (TCP/IPv4)

You may need to scroll down to disable the remaining items.



Note for Client Hyper-V and VMware Workstation Hypervisors: You should only need to enable “QoS Packet Scheduler” and “Internet Protocol Version 4 (TCP/IPv4)”. Everything else here can be set to disabled.

Next, left click on “Internet Protocol Version 4 (TCP/IPv4)” and click on the Properties button below the pane.



First thing we need to do is we need assign this network adapter a static IP address so it can access resources in the host-only network; the “Management” network. For my lab, I use the network 172.16.1.0/24, and I configure the host-only adapter to have the IP address 172.16.1.2, subnet mask of 255.255.255.0 and **NO default gateway**. You can input these settings by clicking the “Use the following IP address:” radio button. Click the “Use the following DNS servers:” radio button and input **NOTHING** in the input boxes.

Internet Protocol Version 4 (TCP/IPv4) Properties

X

General

You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

Obtain an IP address automatically

Use the following IP address:

IP address:

172 . 16 . 1 . 2

Subnet mask:

255 . 255 . 255 . 0

Default gateway:

| . . .

Obtain DNS server address automatically

Use the following DNS server addresses:

Preferred DNS server:

. . .

Alternate DNS server:

. . .

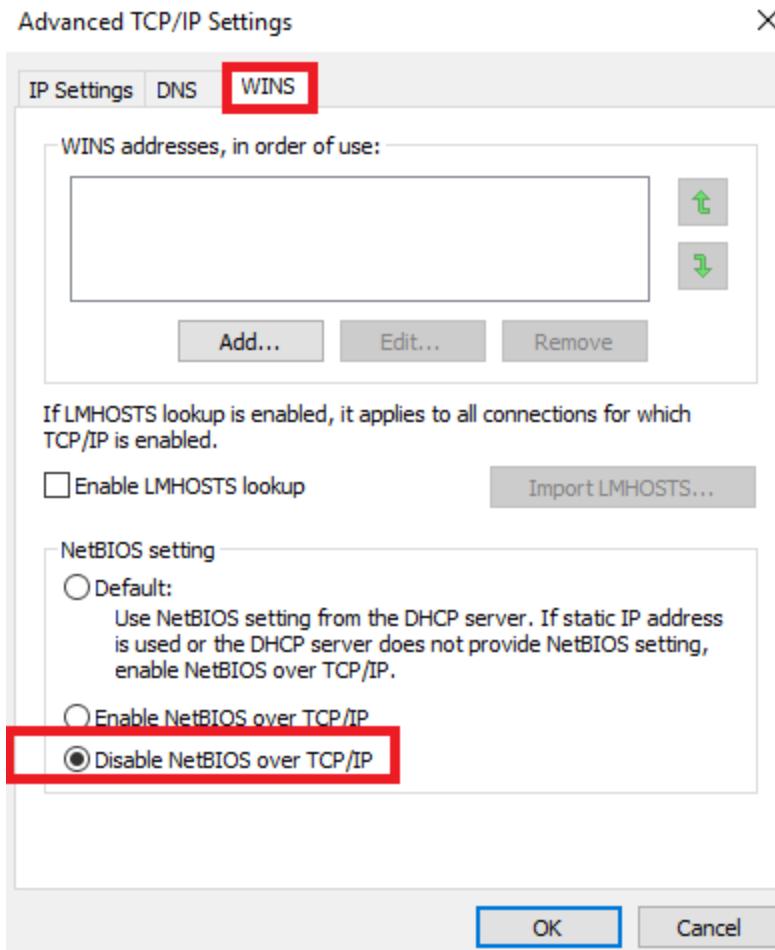
Validate settings upon exit

Advanced...

OK

Cancel

Next, Click the “Advanced...” button. In the new window, click on the “WINS” tab, then at the bottom, under “NetBIOS setting”, click the “Disable NetBIOS over TCP/IP” radio button.

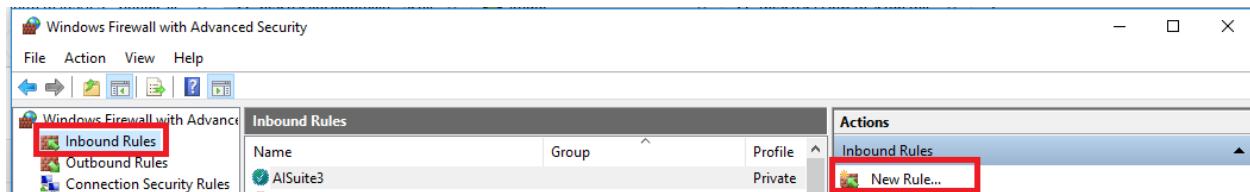


Click OK to close the Advanced TCP/IP Settings window, Click OK to close the Internet Protocol Version 4 (TCP/IPv4) Properties window, and you should be all set here. After you are done here, I would very highly suggest enabling and [configuring Windows Firewall to deny inbound connections to this network card's IP address](#).

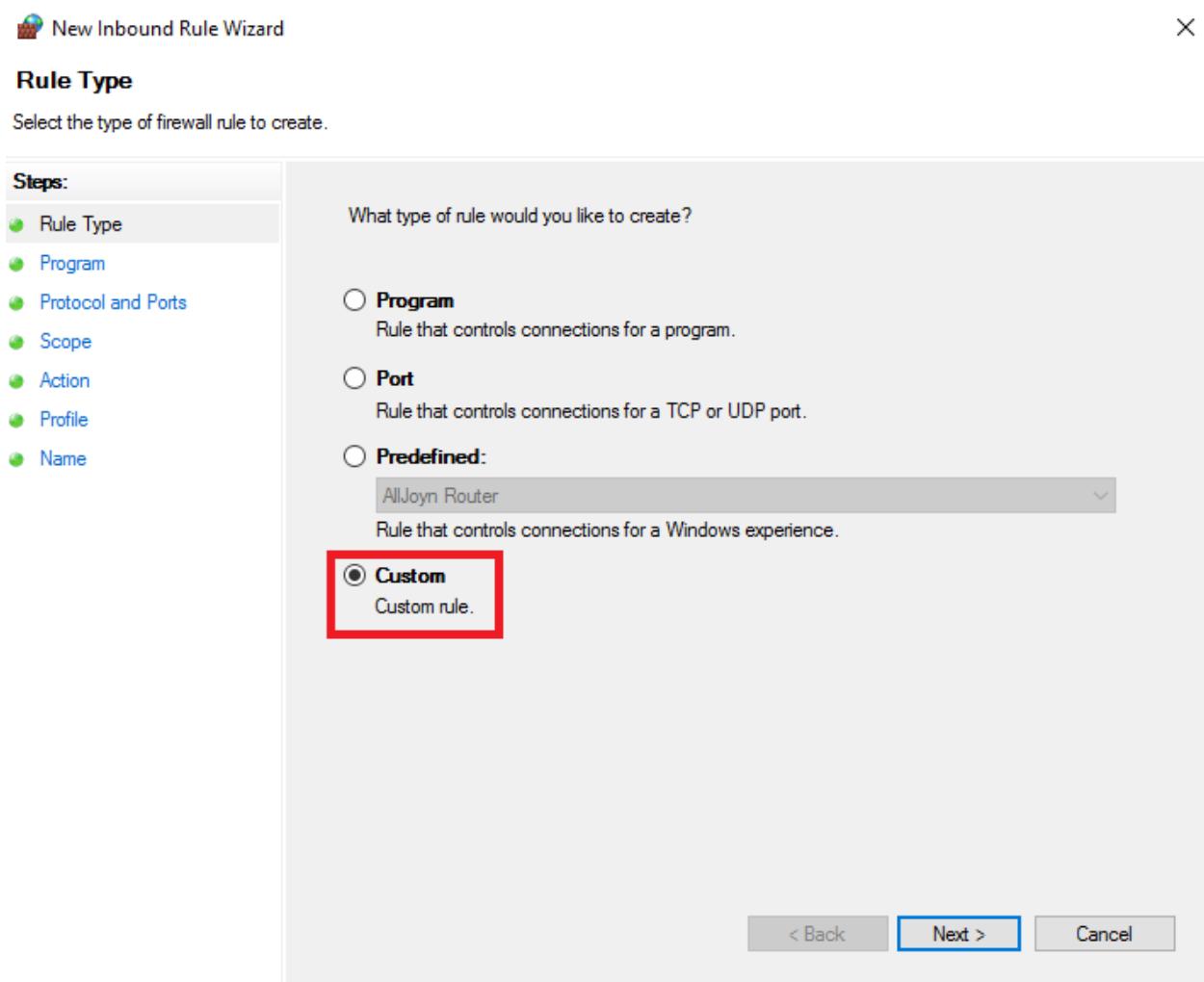
Using Windows Firewall to Limit Exposure of Windows Hypervisor Hosts

As previously established, for hosted hypervisors, we are using the host OS the hypervisor is installed on to manage our virtual machines. In my case, I'm using Windows as the OS I chose to host my hypervisor on. The following is a guide for how to create a firewall rules using the built-in windows firewall. In spite of whatever misgivings you might have, Windows Firewall is actually nice and easy to use; it'll serve our purposes nicely. **This guide is to specifically protect Windows systems that are hosting VMs via a hosted hypervisor that will be used to interact with systems in a virtual network.**

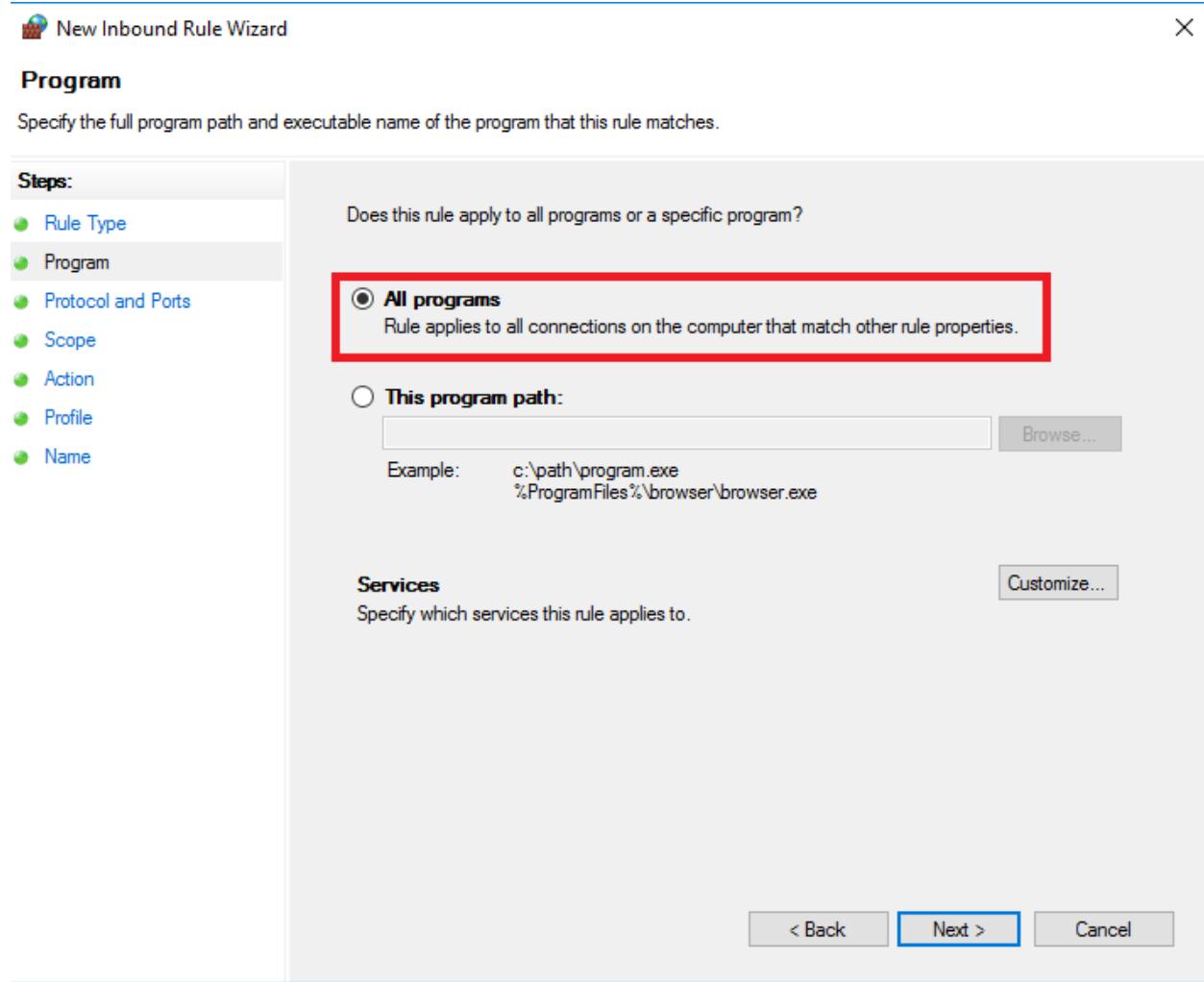
This guide will teach you how to create a firewall rule that will deny any and all traffic inbound towards the IP address we allocated to the virtual NIC (in our case, 172.16.1.2). While the pfSense firewall SHOULD be doing its job enforcing security boundaries, defense in depth means being paranoid. This firewall rule ensures that hosts on the virtual networks cannot initiate ANY connections to 172.16.1.2, but 172.16.1.2 is allowed to initiate, and keep established connections to hosts as necessary. Lets begin by opening “Windows Firewall with Advanced Security”, selecting “Inbound Rules” and then clicking on “New Rule..” to launch the new rule wizard.



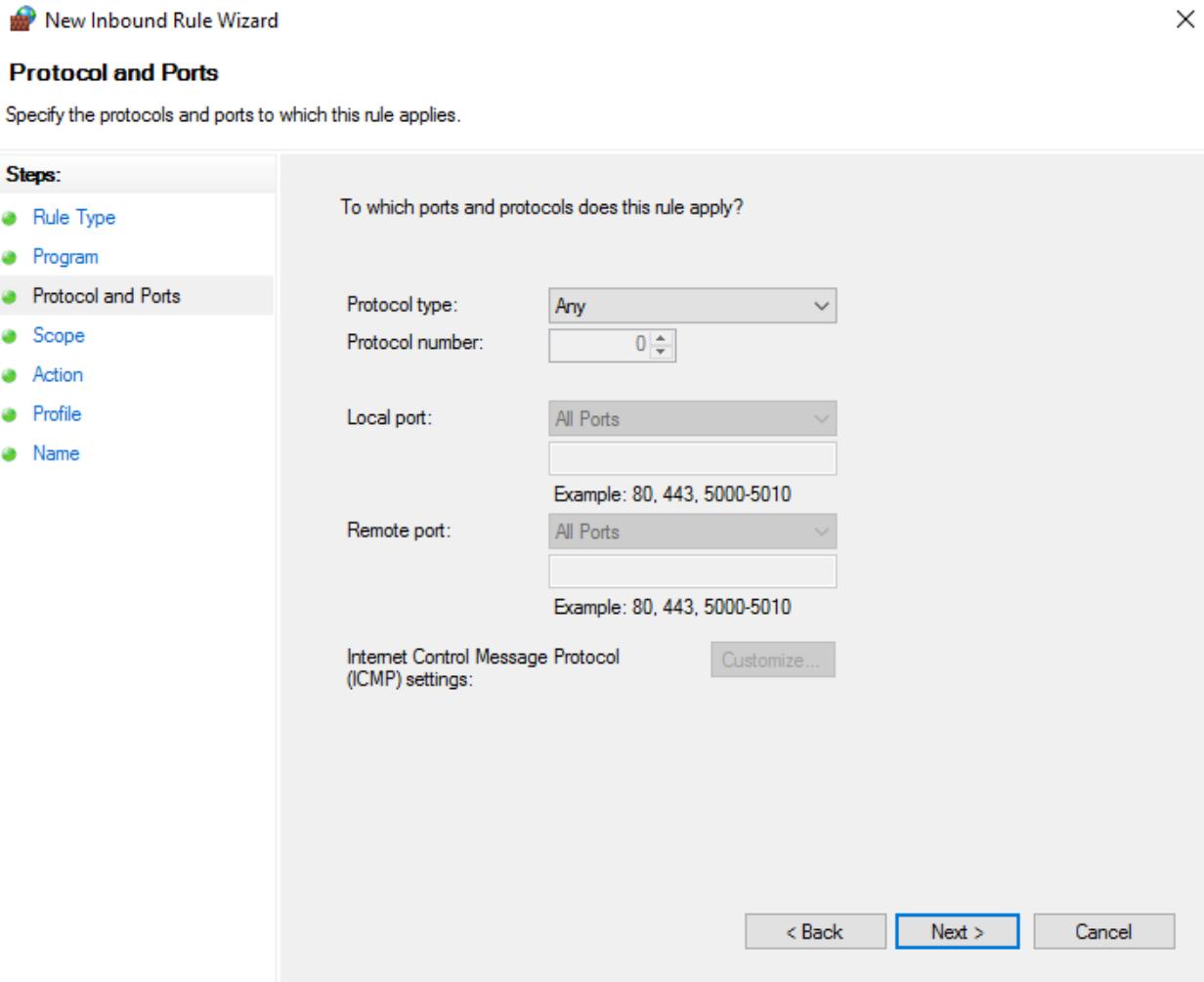
Next, Click the “Custom” radio button, then click next.



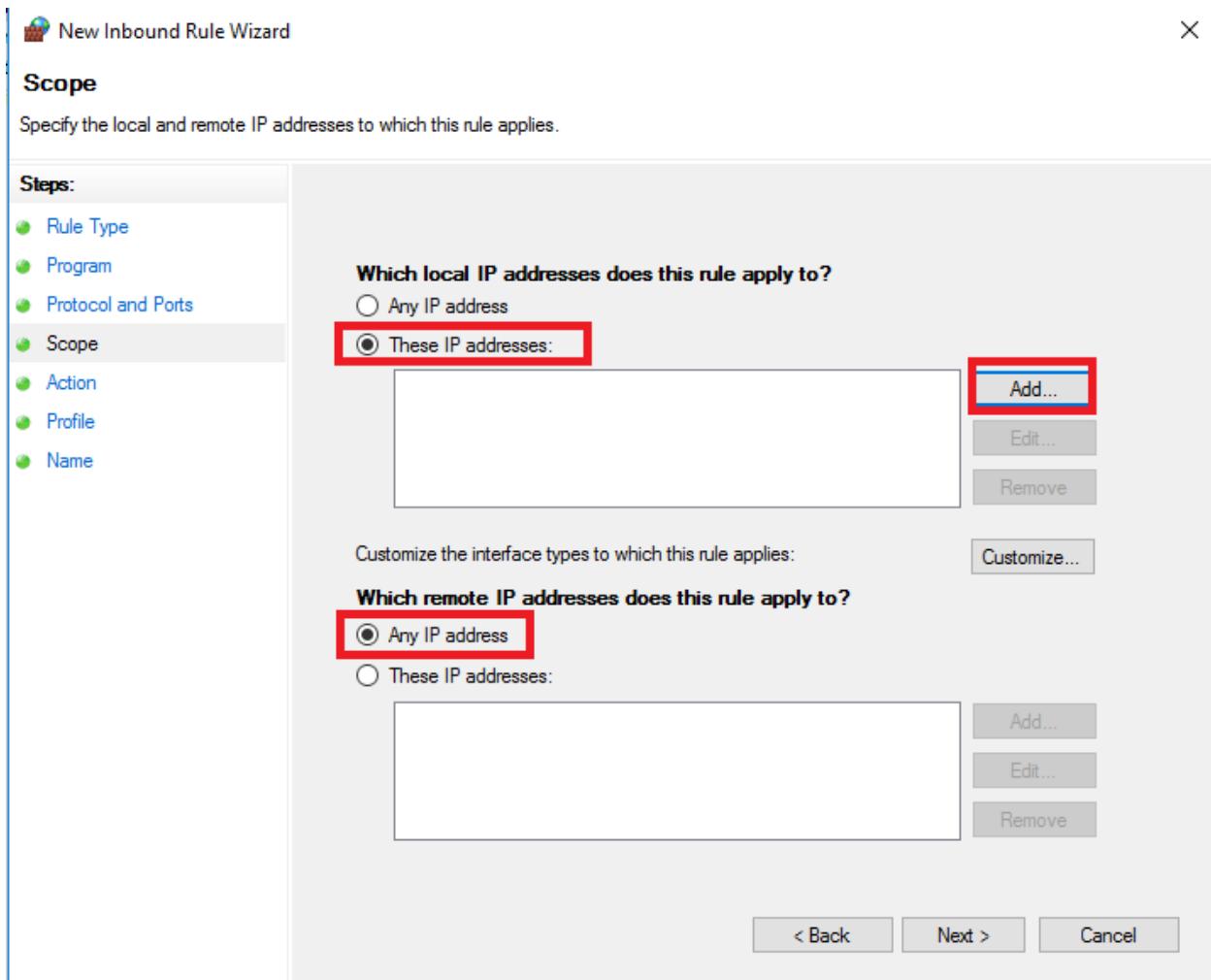
On the next page, click the “All Programs” radio button, then click next.



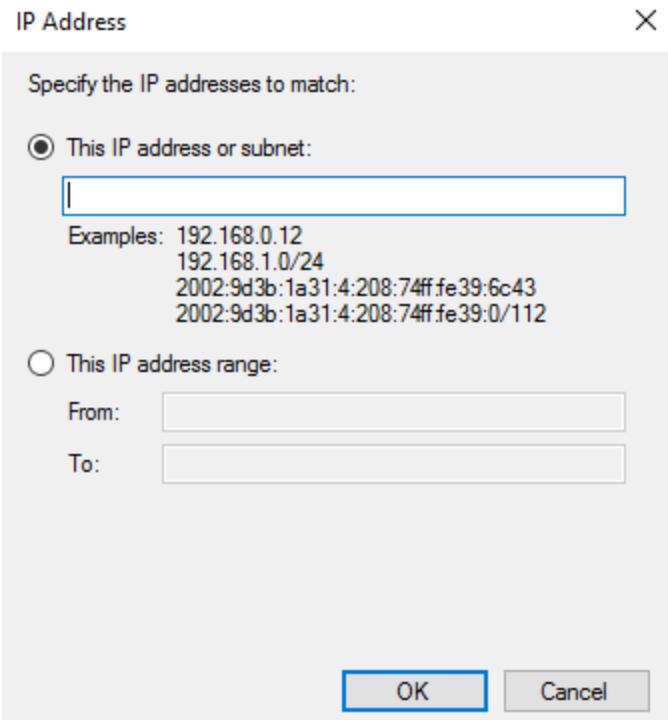
On the next page, click next (we want this rule to apply to ANY protocol).



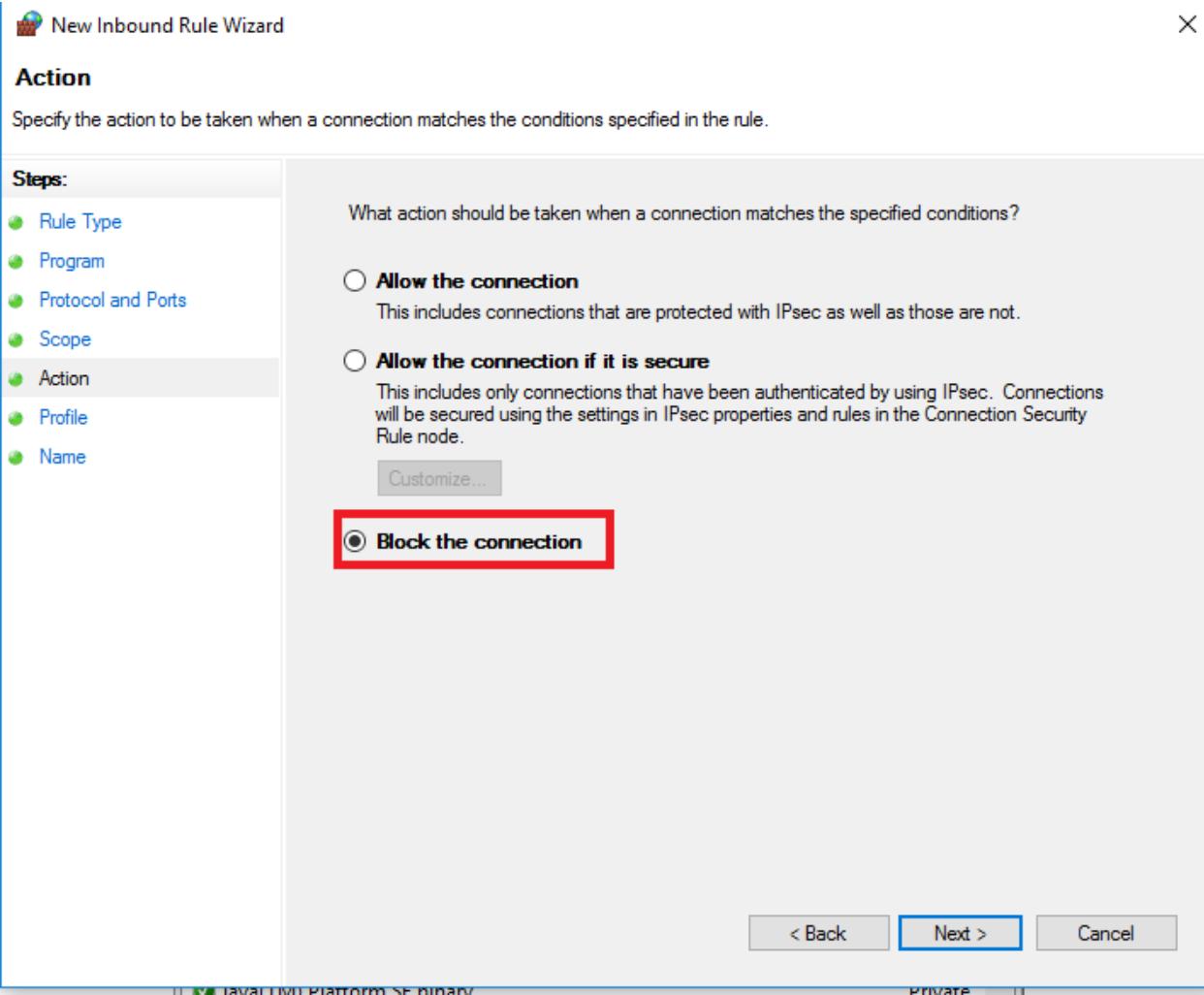
On the next page, under "Which remote IP addresses does this apply to?", ensure the "Any IP address" radio button is clicked. Under "Which local IP addresses does this apply to?" click the "These IP addresses:" radio button, then click the "Add.." button.



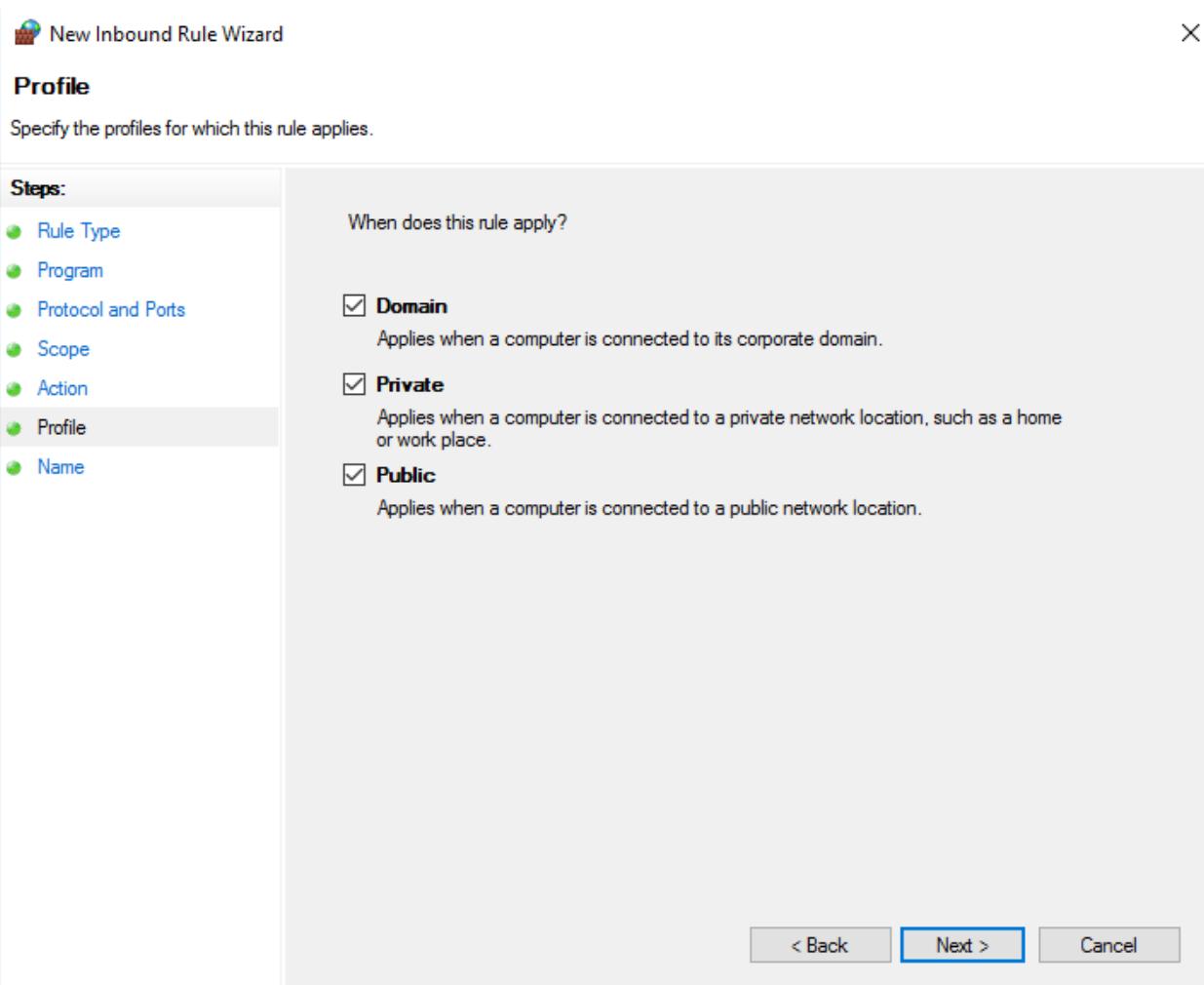
In our case, the IP address we have chosen is 172.16.1.2. Click OK, then click next on the previous window.



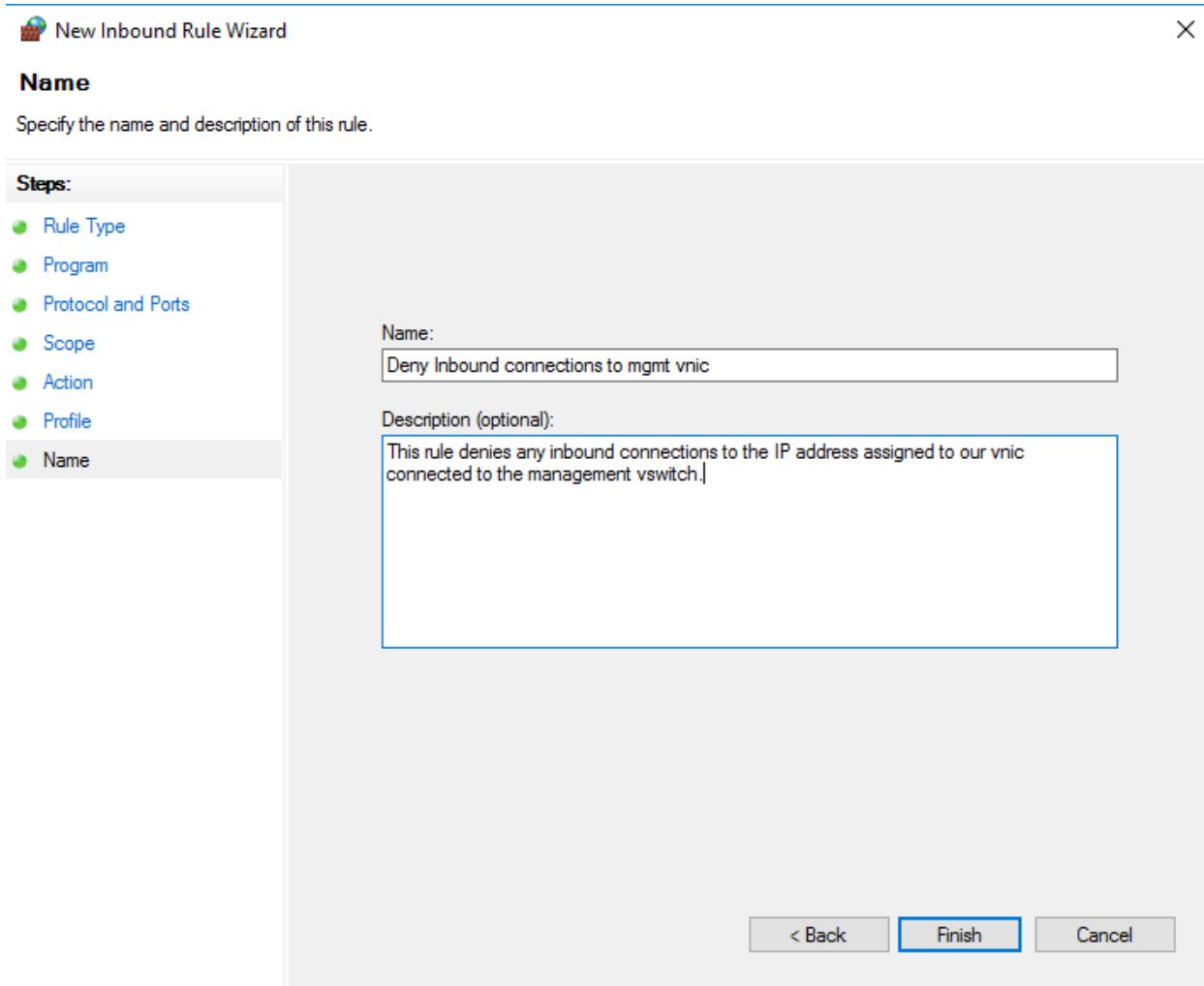
On the next page, click the “Block the connection” radio button and click next.



On the next page, ensure all the checkboxes (domain, private, public) are all checked, and click next.



On the final page of the wizard, give your firewall rule a name, and a description so you don't forget why the rule exists, and click Finish. Make sure to note somewhere that this firewall rule exists.



Automated Patching for Linux Lab VMs

When you create your Ubuntu Linux lab VMs you are given the option to have the OS manage security updates for you. What if you want your linux systems to automatically update ALL of their packages on a regular basis? Proper maintenance and patching of your lab environment on a regular basis is VERY important. I have a very simple script and process for doing this.

updater.sh

The apt command is used to manage software packages on systems that use the .deb package format (In our case, Kali Linux AND Ubuntu Linux use .deb packages and apt for software

package management). Below is a very simple script that runs a series of commands to check for and automatically download the latest updates via the apt-get command:

```
#!/bin/bash
#updater.sh - Weekly update script
#This script checks for the latest updates, downloads them, then reboots the
#system.
#Place this script in /etc/cron.weekly, ensure it is owned by root (chown
root:root /etc/cron.weekly/updater.sh)
#Ensure the script has at least user execute permissions (chmod 700
/etc/cron.weekly/updater.sh)
#If you want updates to be ran once daily or weekly, simply place this script
into /etc/cron.daily or /etc/cron.monthly as you see fit.
export DEBIAN_FRONTEND=noninteractive
apt-get -q update
apt-get -y -q dist-upgrade
logger updater.sh ran successfully. Rebooting system.
init 6
exit 0
```

As the commented lines of this script read, place this script into /etc/cron.daily, weekly, or monthly, depending on whether you want your system to auto-update on a daily, monthly, or weekly basis. This script needs to have been created or otherwise owned by the root user (chown root:root) and the script must as least have use execute file permissions (chmod u+x, or chmod 700) execute successfully.

```
root@ips:~# ls -al /etc/cron.weekly/updater.sh
-rwx----- 1 root root 632 Jan  4 14:28 /etc/cron.weekly/updater.sh
```

When the script runs, the system WILL be rebooted. For this reason, I would NOT, unless it has been agreed upon, deploy this script to a production system that runs anything of any importance. This is a script designed to keep your lab environment secured and up to date, and only your lab environment. The script will also write a line to /var/log/syslog via the logger command to let you know when the update script was ran.

```
Jan  4 14:28:21 ips root: updater.sh ran successfully. Rebooting system.
```

You can use this script to keep the kali, siem, and ips VMs automatically updated, as well as any new VMs you decide to add that use .deb packages and the apt package manager. Hypothetically, one could also modify the script to use the equivalent update commands for yum (the package manager for redhat/RPM-based distros) or other package managers as well.

Remote Lab Management

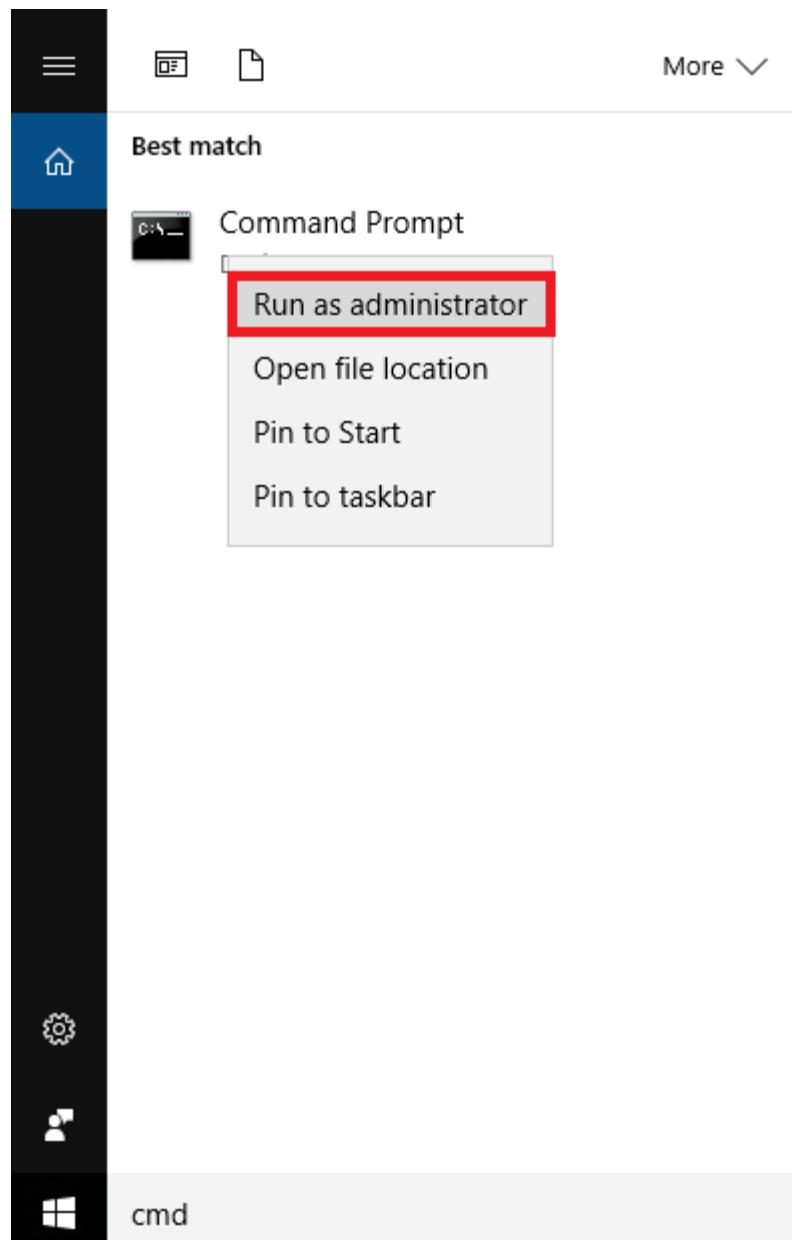
If you're interested in configuring remote management, read on. These are guides that will assist you in setting up SSH access for your lab assets. There are sections dedicated to Configuring remote access on Windows hosted hypervisors, as well as OSX/Unix/Linux hosted hypervisors. There is a separate section that details how to manage remote access to baremetal hypervisors for either Windows or Unix/Linux users.

Windows Remote Access

The following are a list of Windows remote access solutions/configurations to allow users of Windows to remotely administer and interact with their lab systems.

Persistent Static Routes

While our pfSense VM operates as a router to send packets from one network to another, we need a way to tell our Windows system how to get to the 172.16.2.0 network. Enter the “route” command. Route allows you to edit the network routing table in windows. We’re going to create a persistent route to 172.16.2.0 network. Open a command prompt through the start menu or windows search. Right click on the icon for “Command Prompt” and choose the option “Run as administrator”.



Next, run the command:

```
route -p add 172.16.2.0 mask 255.255.255.0 172.16.1.1
```

```
C:\WINDOWS\system32>route -p add 172.16.2.0 mask 255.255.255.0 172.16.1.1
OK!
```

This command instructs windows to route network traffic destined for 172.16.2.0/24 through 172.16.1.1 (The pfSense VM's gateway interface for the management network). The "-p" option makes this route persistent. By default, when the Windows system reboots, any routes added

manually are lost. If you want to confirm you entered the command correctly, run the command:
route print

```
=====
Persistent Routes:
 Network Address      Netmask   Gateway Address   Metric
      172.16.2.0        255.255.255.0       172.16.1.1       1
=====
```

You'll notice that windows has a section called "persistent routes", and our route is added there. You should be able to access VMs in the 172.16.2.0 network, specifically the kali VM, since that is the only VM you should have firewall rules to access (SSH)

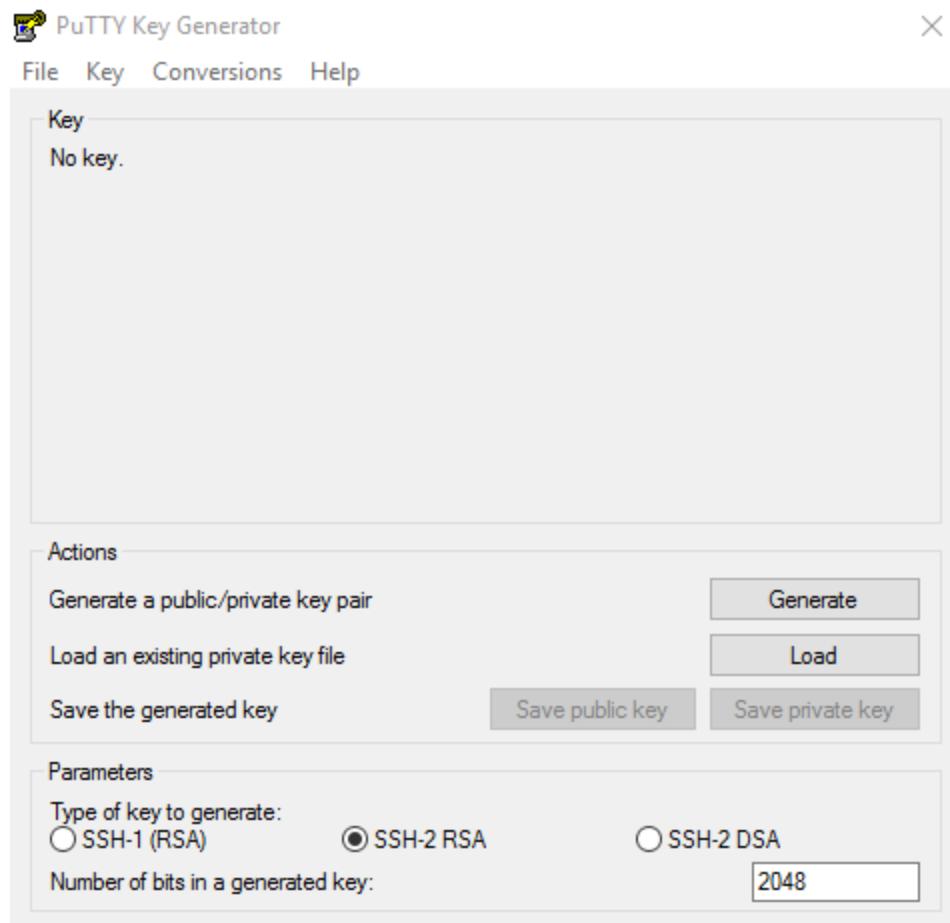
Windows SSH and SCP Software

If you want to use windows to manage your Linux systems over SSH, I recommend downloading and installing the following applications:

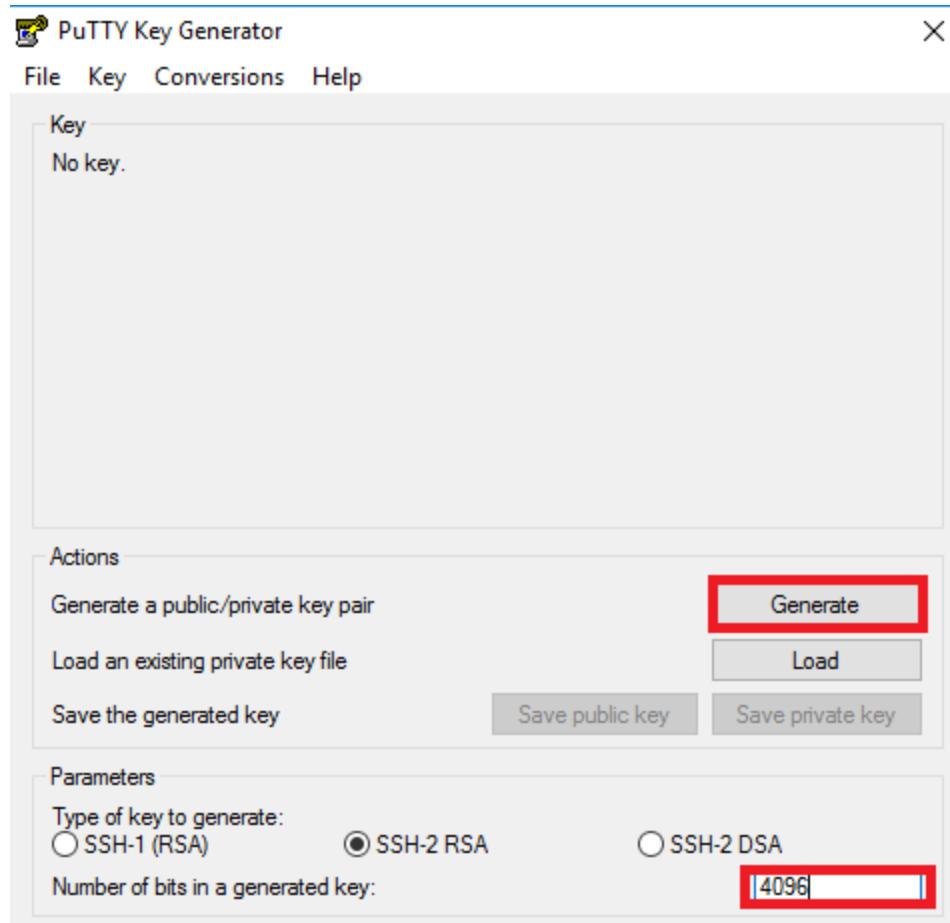
- PuTTYgen (<http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html>)
 - PuTTYgen is a tool for generating SSH public and private keys for doing key-based SSH authentication from Windows SSH clients.
- MRemoteNG (<http://www.MRemoteNG.org/>)
 - MRemoteNG is a tremendously useful application for managing a wide variety of remote access protocols - RDP, SSH (v1 and v2), telnet, rlogin, Citrix ICA, etc.
- WinSCP (<https://winscp.net/eng/download.php>)
 - A graphical SCP client for performing SCP transactions from windows clients.
Also supports key-based authentication.

Generating an SSH key in Windows using PuTTYgen

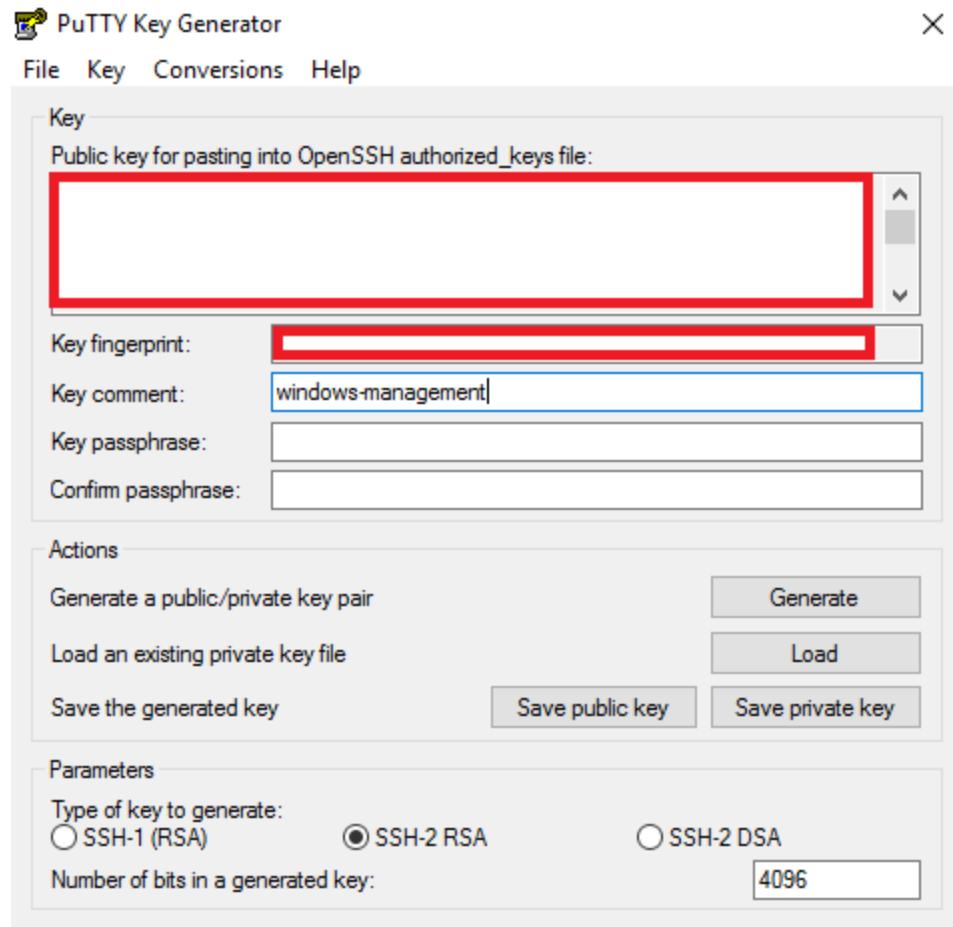
Double-click puttygen.exe to open the application (Please note, if you're running Windows 8 or higher, you may get a Windows SmartScreen "Windows Protected your PC" prompt. Click "More Info", then click the "Run anyway" button). You will be greeted with this screen:



Under “Parameters”, in the “Number of bits in a generated key:” field, change the input box from 2048 to 4096. After doing that, click the “Generate” under “Actions”.



Afterwards, the system will proceed in generating your SSH private and public keys. You'll be greeted with the following:



Feel free to change the key comment to something that makes more sense to you. In my case, I changed it to windows-management to signify that this SSH is for my Windows workstation. PuTTYgen gives you the ability to set a key passphrase for your new SSH key. This means that you **MUST** have your SSH key and you must **ALSO** know the key's passphrase in order to be able to use your SSH key for the lab network. You can also choose to leave the key passphrase field blank. Since this is a lab network environment and should **NOT** be used to host anything of any value or significance, entering a passphrase isn't exactly important; leave it blank. After entering in a key comment, Click the "Save public key". Name the public key something descriptive like "windows-management.pub", and save it.



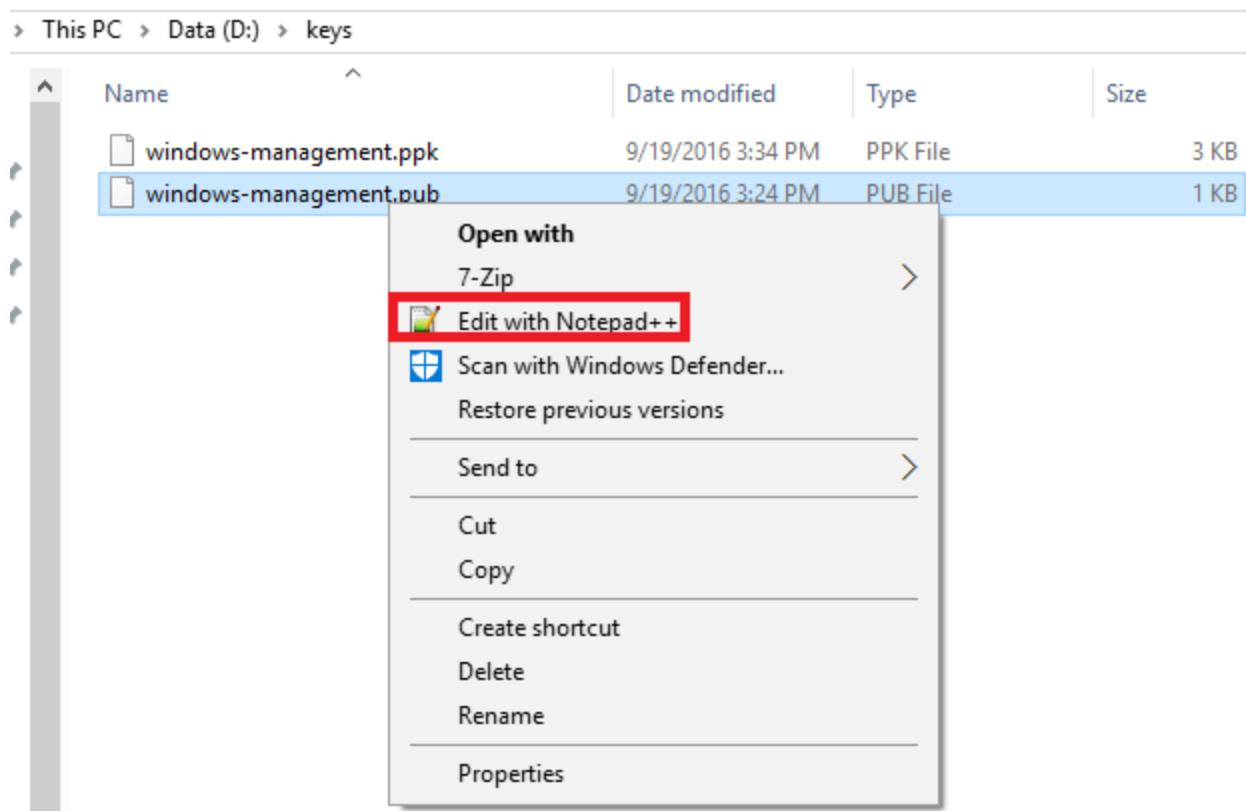
Next, click on the “Save private key”, and save it as well. I would highly suggest saving the private key in the same place you saved the public key.

> This PC > Data (D:) > keys



It is EXTREMELY important that you keep these keys (especially the private key) in a safe location that only you have access to, in order to ensure others do not discover your SSH keys and attempt to abuse them. While this isn't much of a concern with a private lab environment, key security is something to ALWAYS be mindful of on real-world systems if you choose to use key-based SSH authentication.

Download and install notepad++ (<https://notepad-plus-plus.org/>). After doing so, open up windows explorer and navigate to where you saved your public key file. Right click on the file and open it with notepad++.



When you open notepad++ you'll be greeted with the following:

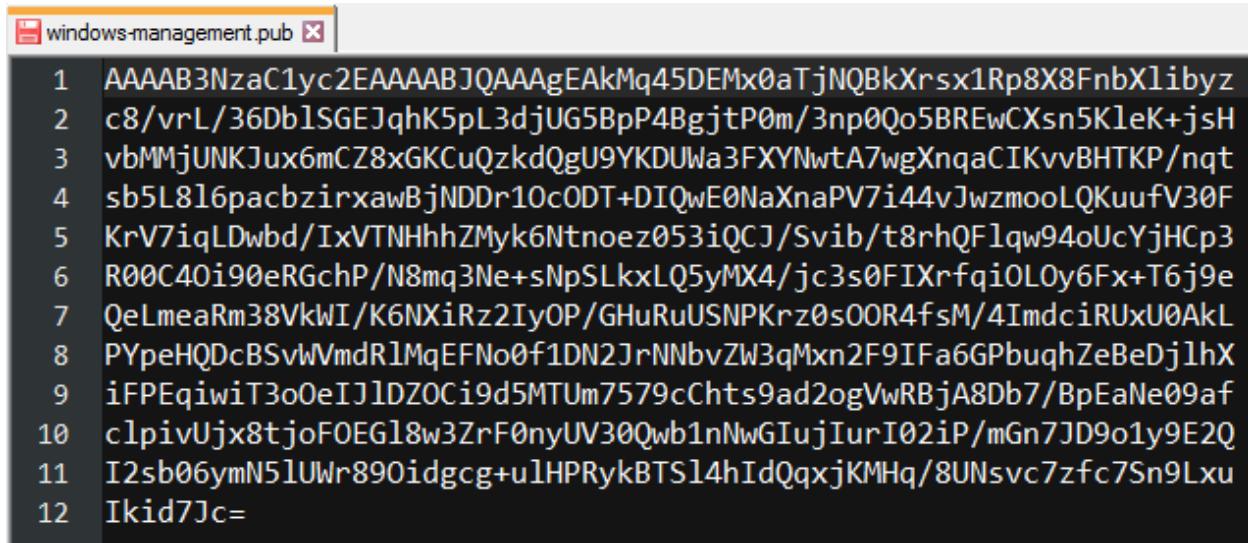
```
1 ----- BEGIN SSH2 PUBLIC KEY -----
2 Comment: "windows-management"
3 AAAAB3NzaC1yc2EAAAQABJQAAgEAkMq45DEMx0aTjNQBkXrsx1Rp8X8FnbXlibyz
4 c8/vrL/36Db1SGEJqhK5pL3djUG5BpP4BgjtP0m/3np0Qo5BREwCXsn5KleK+jsh
5 vbMMjUNKJux6mCZ8xGKCuQzkdQgU9YKDUDa3FXYNwtA7wgXnqaCIKvvBHTKP/nqt
6 sb5L8l6pacbzirxawBjNDDr10c0DT+DIQwE0NaXnaPV7i44vJwzmooLQKuufV30F
7 KrV7iqLDwbd/IxVTNHhhZMyk6Ntnoez053iQCJ/Svib/t8rhQFlqw94oUcYjHCp3
8 R00C40i90eRGchP/N8mq3Ne+sNpSLkxLQ5yMX4/jc3s0FIxrfqjOL0y6Fx+T6j9e
9 QeLmeaRm38VvkWI/K6NXiRz2IyOP/GHuRuUSNPKrz0s00R4fsM/4ImdcirUXu0AkL
10 PYpeHQDcBSvlVmdR1MqEFNo0f1DN2JrNNbvZW3qMxn2F9IFa6GPbuqhZeBeDj1hX
11 iFPEqiwiT3oOeIJ1DZOci9d5MTUm7579cHts9ad2ogVwRBjA8Db7/BpEaNe09af
12 clpivUjx8tjoFOEGl8w3ZrF0nyUV30Qwb1nNwGIujIurI02iP/mGn7JD9o1y9E2Q
13 I2sb06ymN5lUWr890idgchg+ulHPRykBTs14hIdQqxjKMHq/8UNsvc7zfc7Sn9Lxu
14 Ikid7Jc=
15 ----- END SSH2 PUBLIC KEY -----
16
```

Linux and BSD systems expect SSH public keys to be formatted in a very specific way. The key must be on a SINGLE continuous line. Delete the following lines:

- ----- BEGIN SSH2 PUBLIC KEY -----

- Comment: "windows-management"
- ---- END SSH2 PUBLIC KEY ----

This will leave you with the public key on its own.

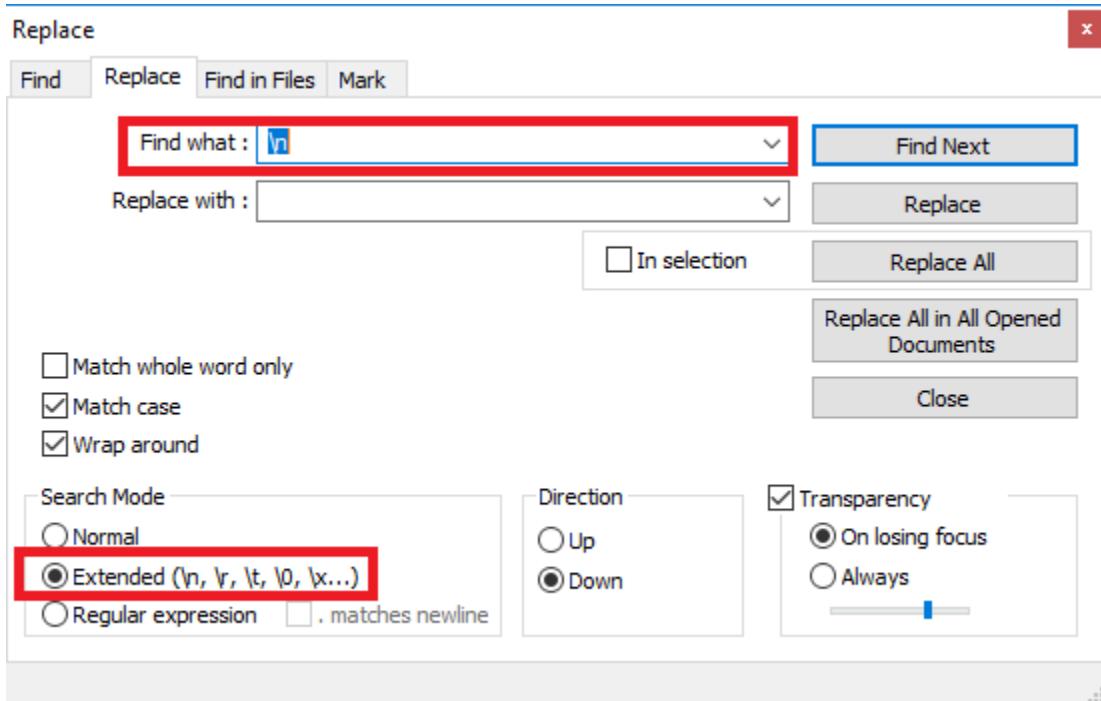


```

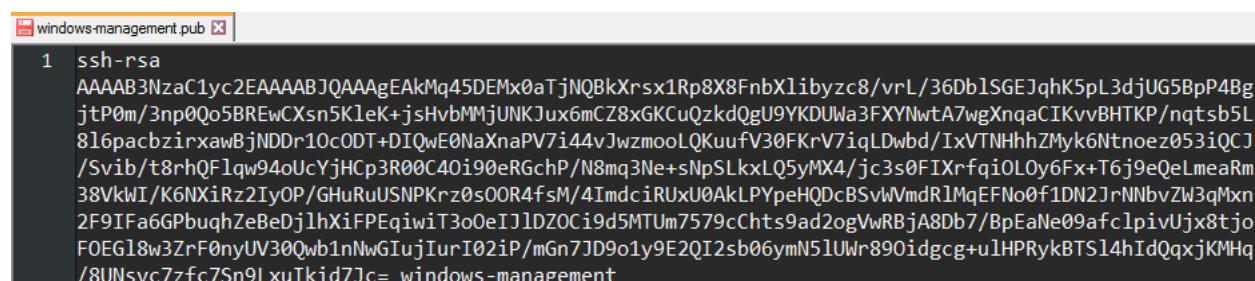
windows-management.pub
1 AAAAB3NzaC1yc2EAAAQABJQAAgEAkMq45DEMx0aTjNQBkXrsx1Rp8X8FnbXlibyz
2 c8/vrL/36Db1SGEJqhK5pL3djUG5BpP4BgjtP0m/3np0Qo5BREwCXsn5KleK+jsH
3 vbMMjUNKJux6mCZ8xGKCuQzkdQgU9YKDUWa3FXYNwtA7wgXnqaCIKvvBHTKP/nqt
4 sb5L8l6pacbzirxawBjNDDr10c0DT+DIQwE0NaXnaPV7i44vJwzmooLQKuufV30F
5 KrV7iqLDwbd/IxVTNHhhZMyk6Ntnoez053iQCJ/Svib/t8rhQFlqw94oUcYjHCp3
6 R00C40i90eRGchP/N8mq3Ne+sNpSLkxLQ5yMX4/jc3s0FIXrfqi0L0y6Fx+T6j9e
7 QeLmeaRm38VkJWI/K6NXiRz2IyOP/GHuRuUSNPKrzs0s00R4fsM/4ImdcIRUXU0AkL
8 PYpeHQDcBSvWVmdR1MqEFNo0f1DN2JrNNbvZW3qMxn2F9IFa6GPbuqhZeBeDj1hX
9 iFPEqiwiT3o0eIJ1DZOCi9d5MTUm7579cChts9ad2ogVwRBjA8Db7/BpEaNe09af
10 clpivUjx8tjoFOEGl8w3ZrF0nyUV30Qwb1nNwGIujIurI02iP/mGn7JD9o1y9E2Q
11 I2sb06ymN5lUWr890idgcg+ulHPRykBTs14hIdQqxjKMHq/8UNsvc7zfc7Sn9Lxu
12 Ikid7Jc=

```

Highlight all of the text left over, and then click Search > Replace.. To open up the find and replace window. Input the following, making sure that the “Extended” radio button is selected under “Search Mode”:

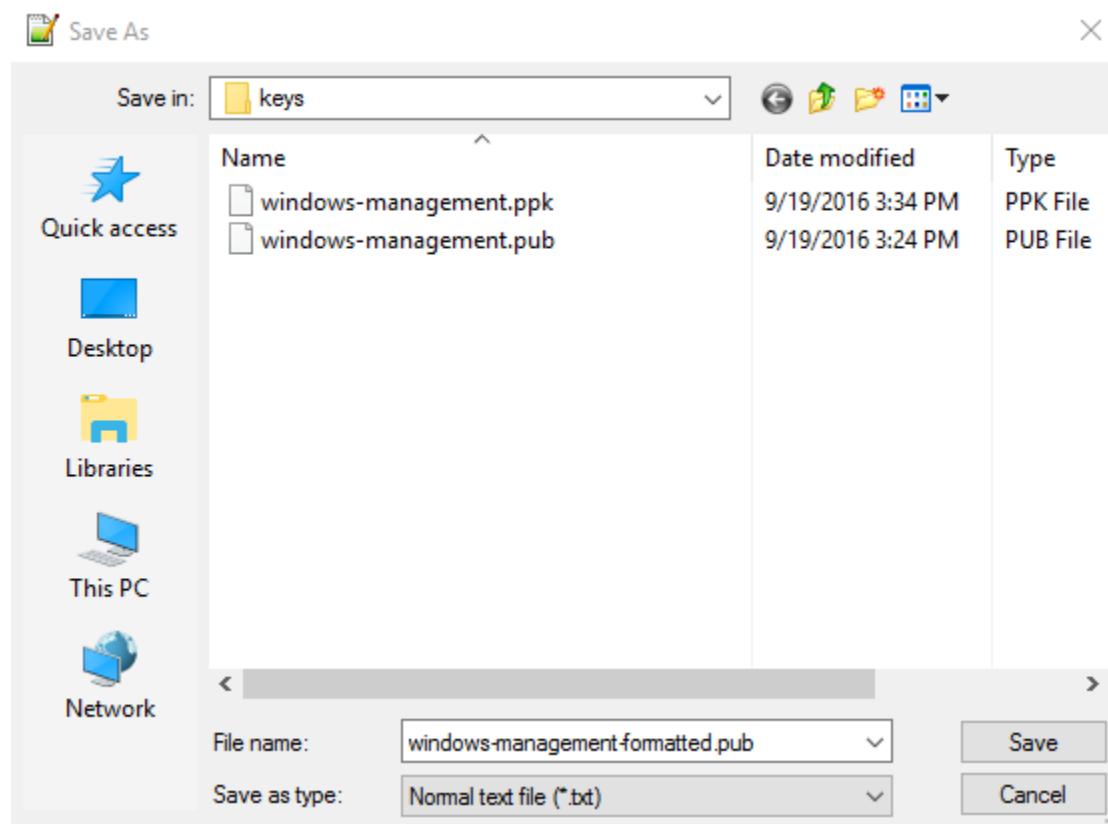


Afterwards, click “Replace All”, then click “Close”. This replaced all of the newline characters in our text file, and placed the key on one continuous line. Now, at the beginning of the line, enter the text “ssh-rsa” (with the space at the end, exactly like it is in quotations), and at the end of the line add the text “windows-management” or whatever comment you want associated with this key (exactly like it is in quotations, with the space between the END of the key and the beginning of your comment). If you did it right, it should look something like this:



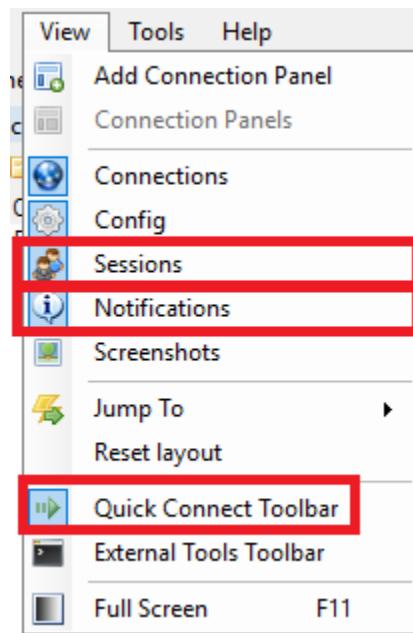
```
windows-management.pub
1 ssh-rsa
AAAAB3NzaC1yc2EAAAQABEAKMq45DEMx0aTjNQBkXrsx1Rp8X8FnbXlibyzc8/vrL/36Db1SGEJqhK5pL3djUG5BpP4Bg
jtP0m/3np0Qo5BREwCXsn5KleK+jshvbMMjUNKJux6mCZ8xGKCuQzkdQgU9YKDULa3FXYNwtA7wgXnqaCTKvvBHTKP/nqtsb5L
816pacbzirxawBjNDDr10c0DT+DIQwE0NaXnaPV7i44vJwzmooLQKuufV30FKrV7iqLDwbd/IxVTNhzhZMyk6Ntnoez053iQCJ
/Svib/t8rhQFlqw94oUcYjhCp3R00C40i90eRGchP/N8mq3Ne+sNpSLkxLQ5yMX4/jc3s0FIXrfqi0LOy6Fx+T6j9eQeLmeaRm
38vkWI/K6NXiRz2IyOP/GHuRuUSNPKrzs0OR4fsM/4ImdcirUXu0AkLPYpeHQdCBSvWmdR1MqEFNo0f1DN2JrNNbvZW3qMxn
2F9IFa6GPbuqhZeBeDjhXiFPEqiwiT3o0eIJ1DZOi9d5MTUm7579cHts9ad2ogVwRBjA8Db7/BpEaNe09afclpivUjx8tjo
FOEG18w3ZrF0nyUV30Qwb1nNwGIujIurI02iP/m6n7JD9o1y9E2QI2sb06ymN51UWr890idgcg+ulHPRykBTS14hIdQqxjKMh
/8UNsvc7zfc7Sn9LxuIkid7Jc= windows-management
```

Now, click File > Save As... and save the file as “windows-management-preformatted.pub”, and click “Save”, making sure to save it to the same place you saved your public and private SSH keys that puttygen generated for you.

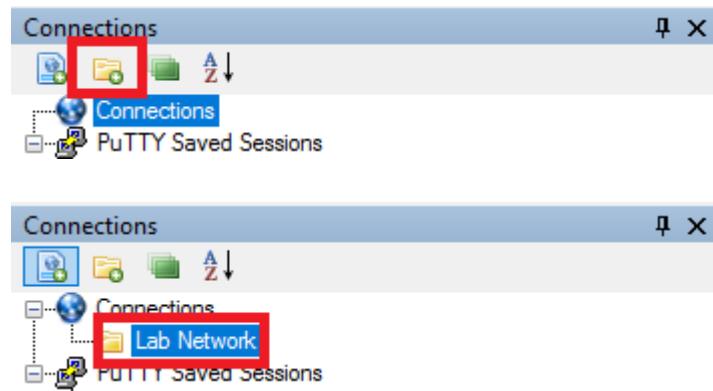


Using MRemoteNG - Connection Files

Open up MRemoteNG. Under the View toolbar, you'll notice a number of highlighted options. Click on Sessions, Notifications and Quick Connect Toolbar to remove these items from view. This will remove a lot of clutter from the interface.



You'll notice there are three main panes in the screen: Connections, Config, and a large, grey window. Click the green folder icon in the connections pane and name the folder "Lab Network".



Click the Lab network folder to highlight it, then click the small icon to the left of the "New Folder" icon (The "New Connection" icon).

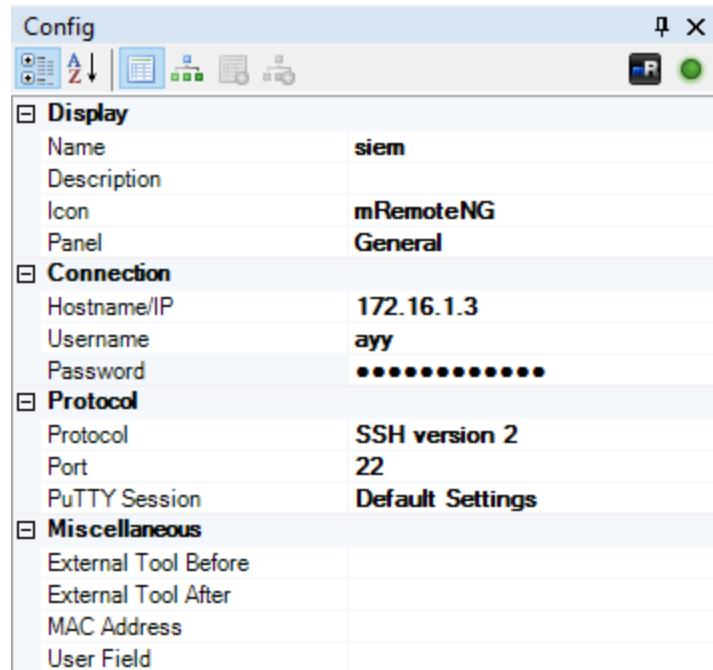


This will create a new connection underneath the “Lab Network” folder. You’ll notice that the Config pane, underneath the Connections pane has a ton of options. I’m going to walk you through doing the first one, and then you’ll do the next two.

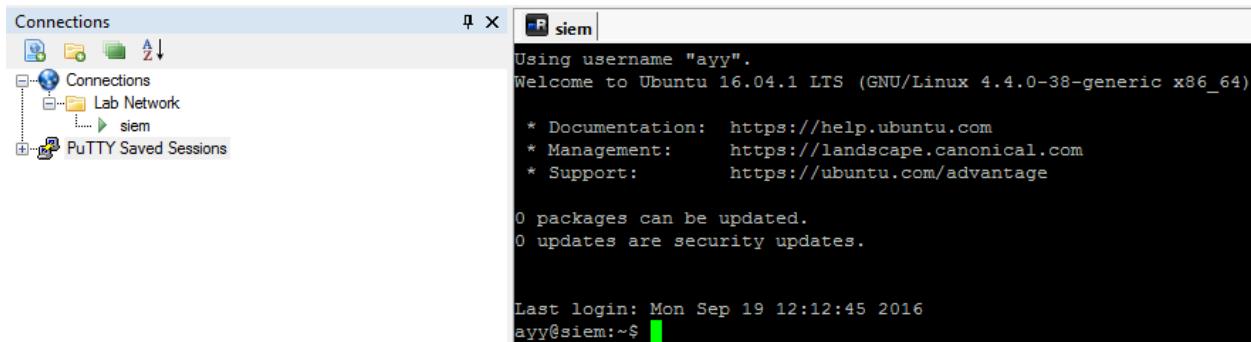
The configuration pane shows the following settings for a new connection:

Display	
Name	New Connection
Description	
Icon	mRemoteNG
Panel	General
Connection	
Hostname/IP	
Username	
Password	
Domain	
Protocol	
Protocol	RDP
Port	3389
Use Console Session	No
Server Authentication	Always connect, even if auth fails
Load Balance Info	
Use CredSSP	Yes
Gateway	
Use Gateway	Never
Appearance	
Resolution	Fit To Panel
Automatic Resize	Yes
Colors	65536 Colors (16-bit)
Cache Bitmaps	Yes

Double click on “New Connection” and input “siem” in the box. Click on “RDP” in the protocol field, then click the drop-down arrow that appears. Change “RDP” to “SSH version 2.” Double click “Hostname/IP” and input “172.16.1.3” in the input field. Double click on “Username” and input the username you used during the Ubuntu 16.04 installation. The, Double click on “Password” to input the password into the password input field.



When you're all done, double click on "siem" in the Connections pane. If the username and password were entered correctly, you should automatically authenticate and log in to the siem VM with no problems.



Now, create two more new connections, under the "Lab network" folder with the following settings:

- Name: ips
 - Protocol: SSH version 2
 - Hostname/IP: 172.16.1.4
 - Username: <username you entered>
 - Password: <password for the username entered>

- Name: kali
 - Protocol: SSH version 2
 - Hostname/IP: 172.16.2.2

- Username: root
- Password: <password you assigned to root>

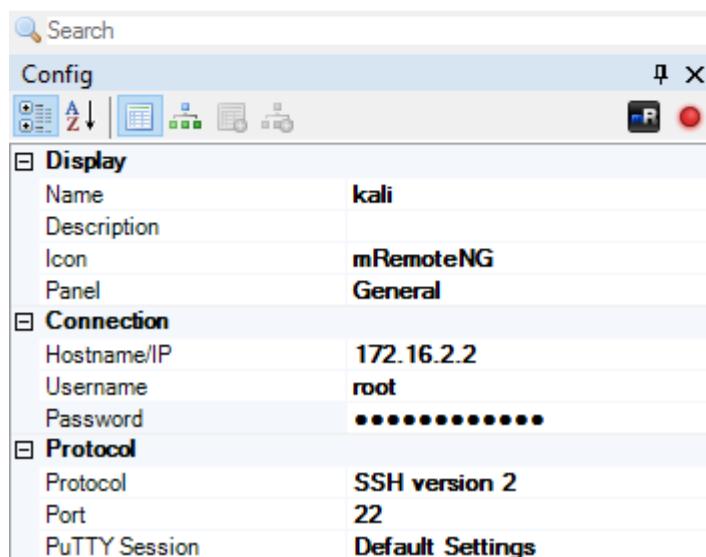
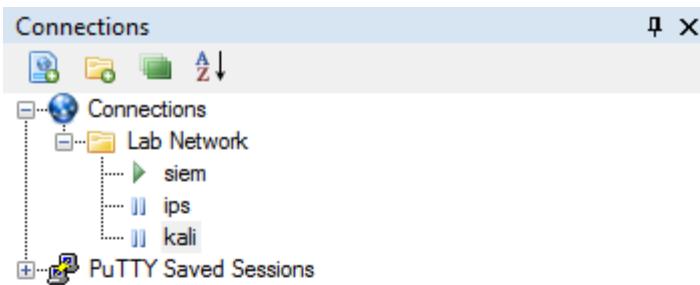
If you did everything correctly, the Connections pane should look like this:

Connections

- Connections
 - Lab Network
 - siem
 - ips
 - kali
- PutTY Saved Sessions

Config

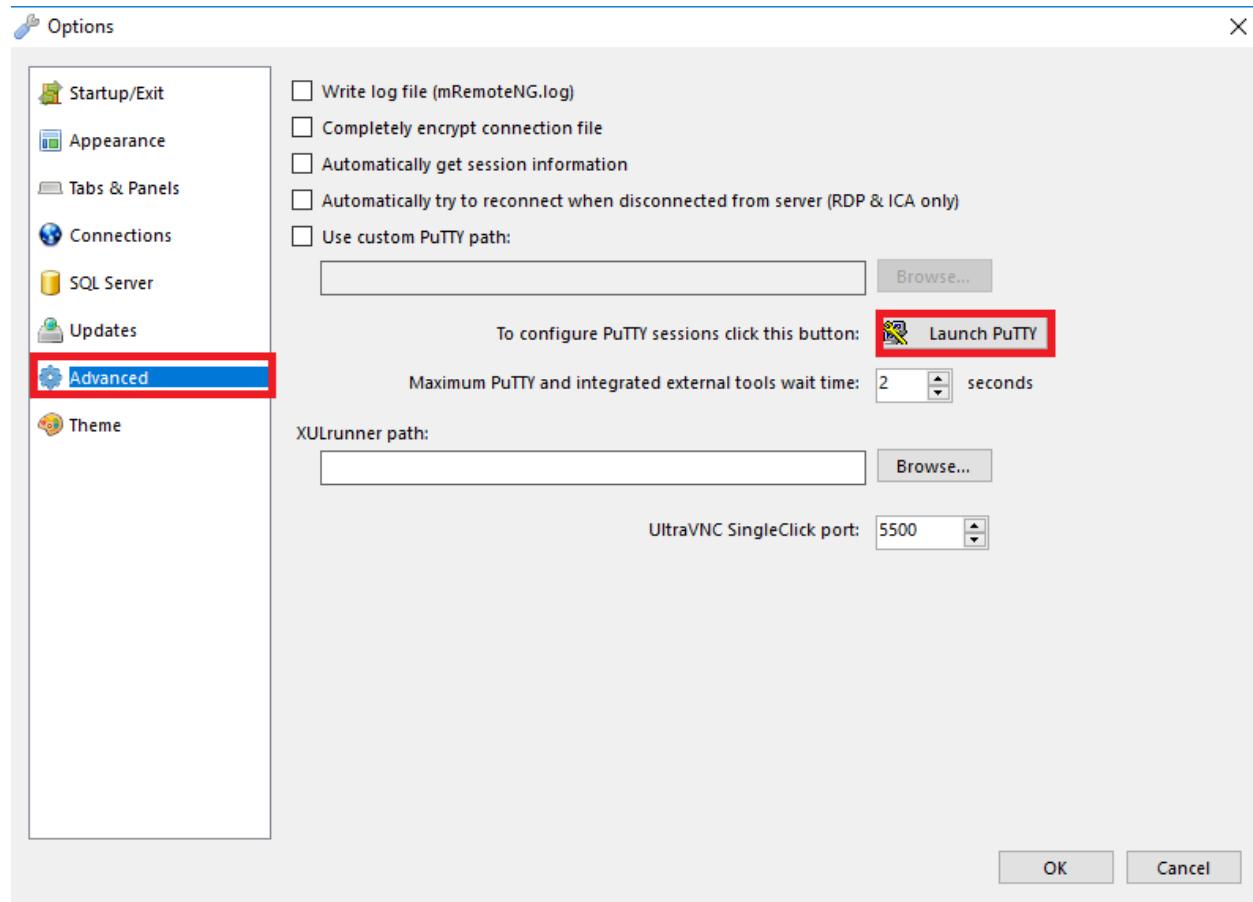
Name	ips
Description	mRemoteNG
Icon	General
Panel	General
Connection	
Hostname/IP	172.16.1.4
Username	ayy
Password	*****
Protocol	
Protocol	SSH version 2
Port	22
PuTTY Session	Default Settings



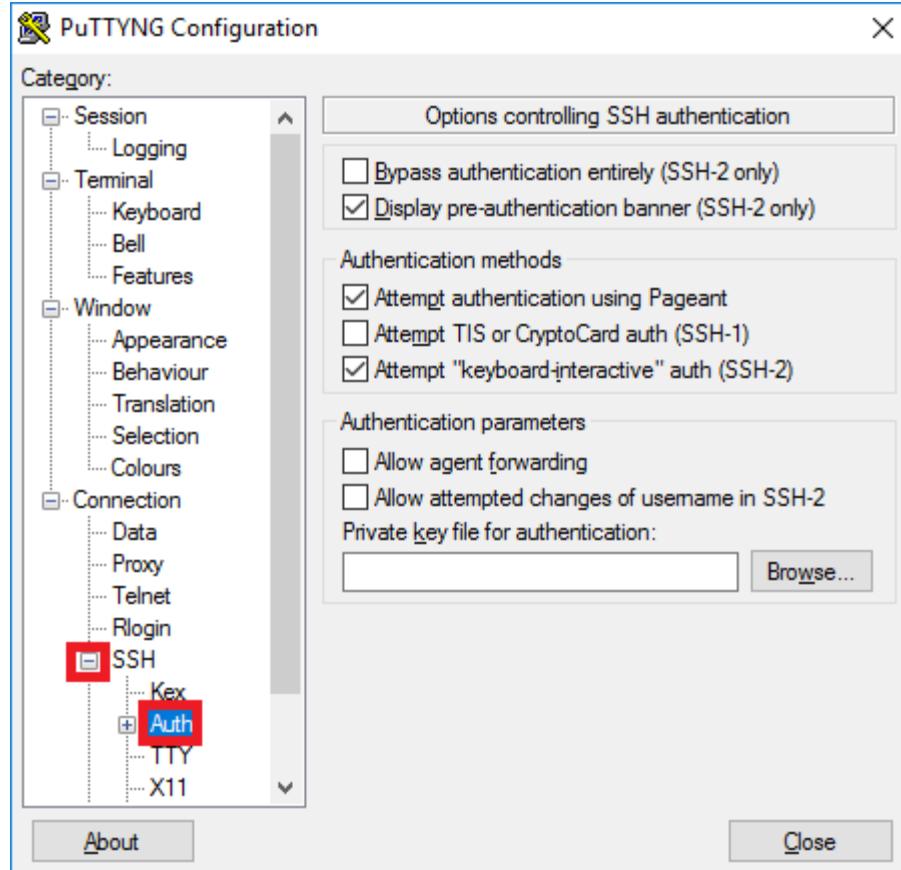
Feel free to test out your SSH connection to the ips VM, but do note that for the time being you will NOT be able to reach the kali VM. Don't worry about this for right now, we're going to fix this shortly.

Using MRemoteNG - PuTTY Saved Sessions

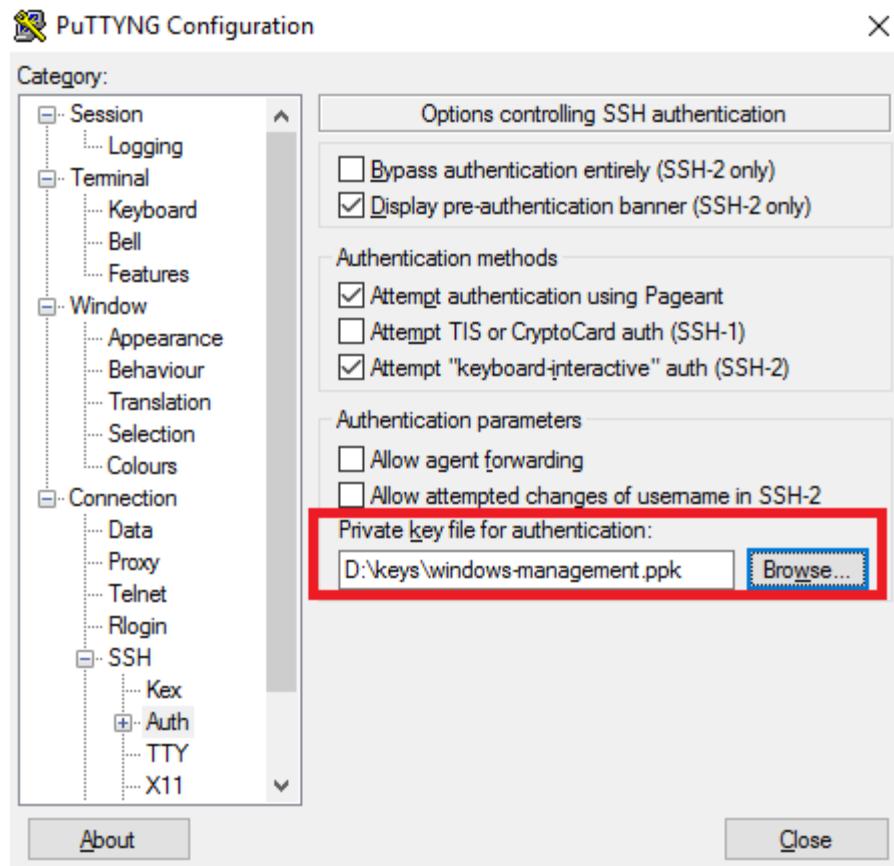
So now that we have SSH connections made, let's make a session profile. Session profiles are going to be what allows us to use our SSH key that we generated with PuTTYgen to authenticate to our lab VMs, instead of using a password. In MRemoteNG, click on Tools > Options. On the Options menu, click on "Advanced" then click the "Launch PuTTY" button.



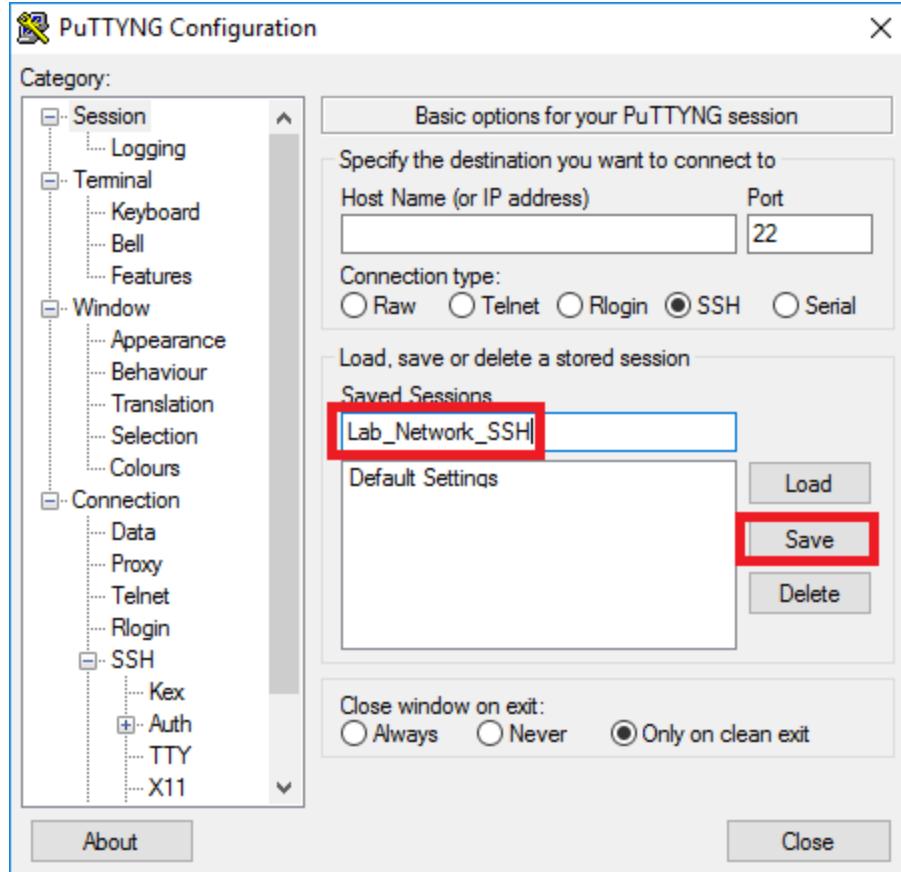
PuTTY opens. Don't worry, we won't be here very long. On the left pane labeled "Category", click on the "+" symbol next to SSH to expand the menu, then click on "Auth" (not the "+" sign, but the text "Auth", right next to it)



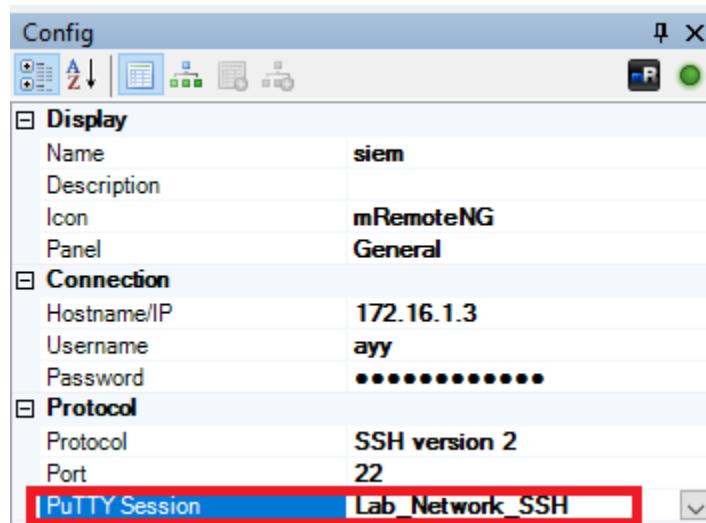
Under “Authentication parameters”, in the “Private key file for authentication:” click the “Browse...” button. An explorer window pops up. Navigate to the SSH **PRIVATE** key that we generated with PuTTYgen earlier.



Click on “Session” under the “Category:” pane again, and under “Load, save, or delete a stored session”, Input “Lab_Network_SSH” into the “Saved Sessions” input box, then click the Save button



Click the close button to exit PuTTY, then click the OK button to close the MRemoteNG Options menu. Back in the MRemoteNG main screen, click on the network connection for “siem” to bring up its configuration in the config pane. Under the Protocol heading, there is an option called “PuTTY Session”. Click on PuTTY Session, then click on the drop-down arrow to select Lab_Network_SSH.



Now, double click on siem to connect to the siem VM. You might notice the text “Server refused our key” in the terminal window. Our SSH connection is trying to send our key to authenticate with the server now. The problem is that the user (“ayy” in my case) we’re attempting to log in as doesn’t have the public key for the private key we are sending in the user’s authorized_keys list. So we’re sending the key and the server is saying “The user isn’t using this key. I don’t care.” Then is immediately falling back to authenticate with the password we have entered in MRemoteNG to successfully authenticate. We’ll be fixing that next.

```
Using username "ayy".
Server refused our key
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-38-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.

Last login: Mon Sep 19 16:56:55 2016 from 172.16.1.2
ayy@siem:~$
```

Remote Lab Management Solutions for Unix/Linux

Enabling Key-Based Authentication in Linux/Unix systems

If you’re continuing on from the previous section, you should have an SSH session opened up as your user account to the siem VM, and you should already be in your user’s home directory. (if you are not, run the command “cd ~”). If not, open up MRemoteNG and create a new session. Once you are logged in, run the following command:

```
mkdir ~/.ssh;chmod 700 ~/.ssh; touch ~/.ssh/authorized_keys;chmod 600
~/.ssh/authorized_keys
```

To verify everything executed properly, run the following command:

```
ls -al ~/.ssh
```

You should have something that looks like this:

```

ayy@siem:~$ mkdir .ssh; chmod 700 .ssh; touch .ssh/authorized_keys; chmod 600 .ssh/authorized_keys
ayy@siem:~$ ls -al .ssh
total 8
drwx---- 2 ayy ayy 4096 Sep 19 19:29 .
drwxr-xr-x 4 ayy ayy 4096 Sep 19 19:29 ..
-rw----- 1 ayy ayy     0 Sep 19 19:29 authorized_keys
ayy@siem:~$ █

```

The first command was actually 4 commands in rapid succession. We created the hidden directory “.ssh”, gave it file permissions that only allows the user who created it access to the directory, the created the empty file “authorized_keys” with the touch command (if the file does NOT already exist, touch creates it), then gave it file permissions to where only our user can read or write to it.

Now, the next part is tricky in that we have to copy the contents of windows-management-formatted.pub to .ssh/authorized_keys in the user’s home directory. There are a few ways to do this.

Key Copy Method 1: echo append to authorized_keys

Open up the file “windows-management-formatted.pub” in notepad++, highlight all of the text in the file, right click on it, then select “Copy”. The contents of our public key are now copied to the windows clipboard.

Back in MRemoteNG, in our SSH session to siem, enter the following command:

```
echo '[right click here]' >> .ssh/authorized_keys
```

Be sure to include the single quotes in the command above!

When we right clicked on the window (note right click only one time), we paste the contents of the clipboard into the terminal window. This should have copied the contents of “windows-management-formatted.pub” in between the single quotes. We’re then telling the terminal to append this key to the .ssh/authorized_keys file.

```

ayy@siem:~$ echo 'ssh-rsa AAAAB3NzaC1yc2EAAAQAAgEAkMq45DEMx0aTjNQBkXrs
sx1Rp8X8FnbXlibyzc8/vrL/36Db1SGEJqhK5pL3djUG5BpP4BgjtP0m/3np0Qo5BREwCXsn5
KleK+jshVbMMjUNKJux6mC28xGKCuQzkdQgU9YKDUWa3FXYNwtA7wgXnqaCIKvvBHTKP/nqts
b5L816pacbzirxawBjNDDr1OcODT+DIQwE0NaXnaPV7i44vJwzmooLQKuuFV30FKrV7iqLDwb
d/IxVTNHhZMyk6Ntnoez053iQCJ/Svib/t8rhQFlqw94oUcYjHCp3R00C4Oi90eRGchP/N8m
q3Ne+sNpSLkxLQ5yMX4/jc3s0FIXrfqiOLOy6Fx+T6j9eQeLmeaRm38VkWI/K6NXiRz2IyOP/
GHuRuUSNPKrz0sOOR4fsM/4ImdcIRUxU0AkLPYpeHQDcBSvWVmdR1MqEFNo0f1DN2JrNNbvZW
3qMxn2F9IFa6GPbuqhZeBeDj1hXiFPEqiwiT3oOeIJ1DZOci9d5MTUm7579cChts9ad2ogVwR
BjA8Db7/BpEaNe09afclpivUjx8tjoFOEGl8w3rF0nyUV30Qwb1nNwGIujIurI02iP/mGn7J
D9o1y9E2QI2sb06ymN51UWr890idgcg+ulHPRykBTs14hIdQqxjKMhq/8UNsvc7zfc7Sn9Lxu
Ikid7Jc= windows-management' >> .ssh/authorized_keys █

```

Next, run the following command:

```
cat .ssh/authorized_keys
```

The cat command reads the contents of a file and spits it back out at the terminal. Your output should look similar to this:

```
ayy@siem:~$ cat .ssh/authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAQAAgEAkMq45DEMx0aTjNQBkXrsx1Rp8X8FnbXlibyzc
8/vrL/36Db1SGEJqhK5pL3djUG5BpP4BgjtP0m/3np0Qo5BREwCXsn5KleK+jshvbMMjUNKJu
x6mCZ8xGKCuQzkdQgU9YKDUWa3FXYNwtA7wgXnqaCIKvvBHTKP/nqtsb5L816pacbzirxawBj
NDDr1OcODT+DIQwE0NaXnaPV7i44vJwzmooLQKuufV30FKrV7iqLDwbd/IxVTNHhhZMyk6Ntn
oez053iQCJ/Svib/t8rhQFlqw94oUcyjHCp3R00C4Oi90eRGchP/N8mq3Ne+sNpSLkxLQ5yMX
4/jc3s0FIXrfqiOLOy6Fx+T6j9eQeLmeaRm38VkWI/K6NXiRz2IyOP/GHuRuUSNPKrz0sOOR4
fsM/4ImdcIRUXu0AkLPYpeHQDcBSvWVmdR1MqEFNo0f1DN2JrNNbvZW3qMxn2F9IFa6GPbuqh
ZeBeDjlhXiFPEqiwIT3oOeIJ1DZOci9d5MTUm7579cChts9ad2ogVwRBjA8Db7/BpEaNe09af
clpivUjx8tjoFOEG18w3ZrF0nyUV30Qwb1nNwGIujIuri02iP/mGn7JD9o1y9E2QI2sb06ymN
51UWr890idgcg+u1HPRykBTs14hIdQqxjKMhq/8UNsvc7zfc7Sn9LxuIkid7Jc= windows-m
anagement
ayy@siem:~$ █
```

Key Copy Method 2: using vi

Vi is a text editor that is older than dirt. It's hard to use, hard to understand, and has arcane invocations that nobody understands. And now you get to struggle with it as your elders did before you. Its a rite of passage.

Open up the file “windows-management-formatted.pub” in notepad++, highlight all of the text in the file, right click on it, then select “Copy”. The contents of our public key are now copied to the windows clipboard.

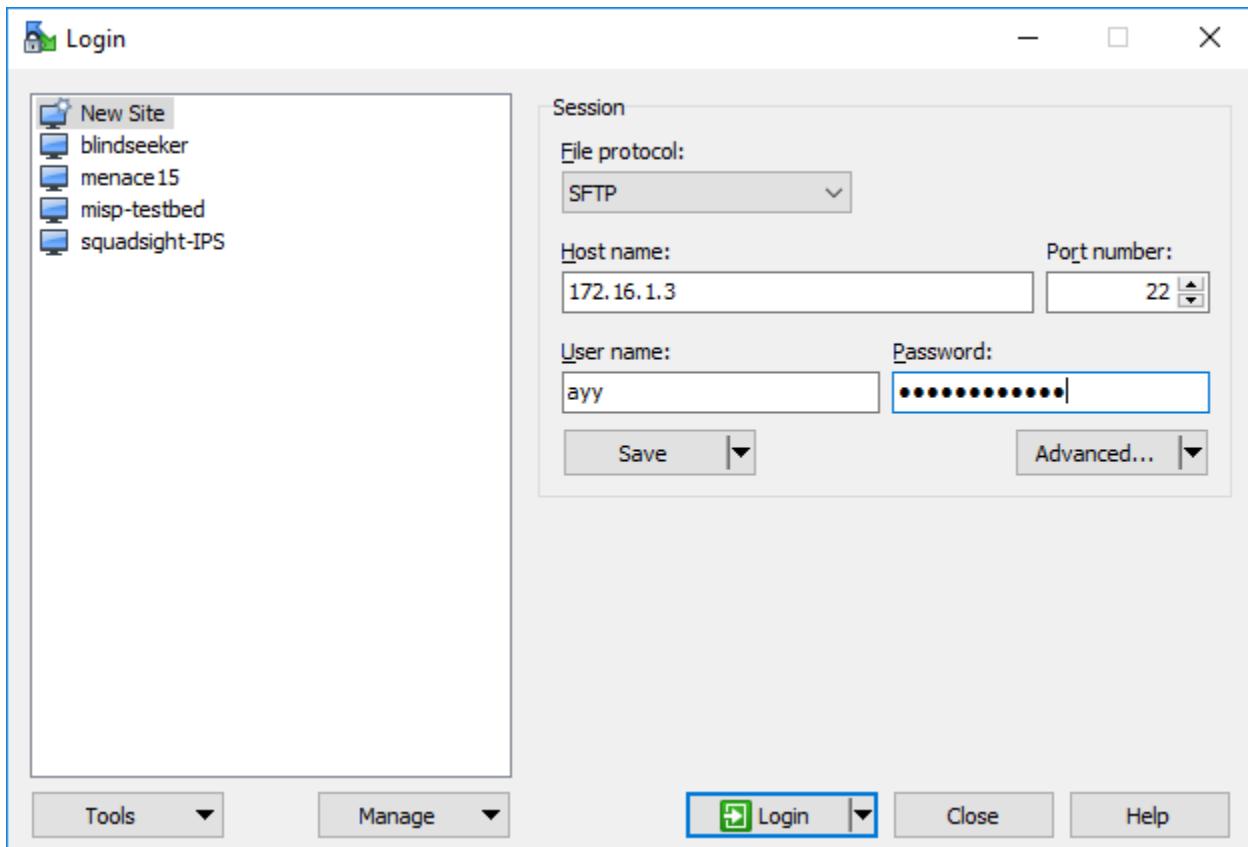
Back in MRemoteNG, in our SSH session to siem, enter the following command:

```
vi ~/.ssh/authorized_keys
```

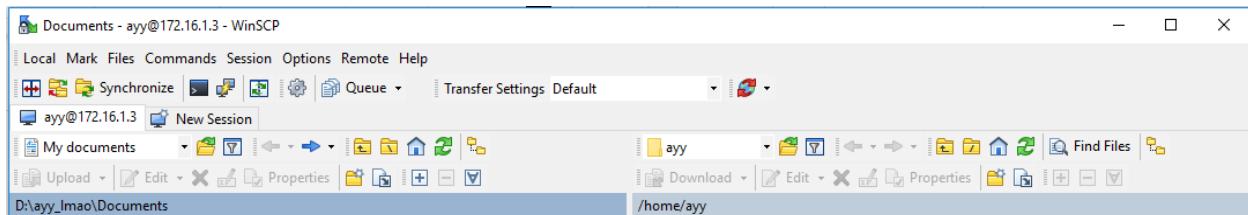
This will open the editor in review mode. There should be nothing in the file. No text, no other keys; nothing. Hit the ‘i’ key to enter ‘insert’ mode. Right click on the terminal window JUST ONCE. This dumps the contents of the windows clipboard into the editor in insert mode. After the contents have been copied into the file, hit the ‘esc’ key then type in ‘:wq!’ to save the file and exit.

Key Copy Method 3: SCP

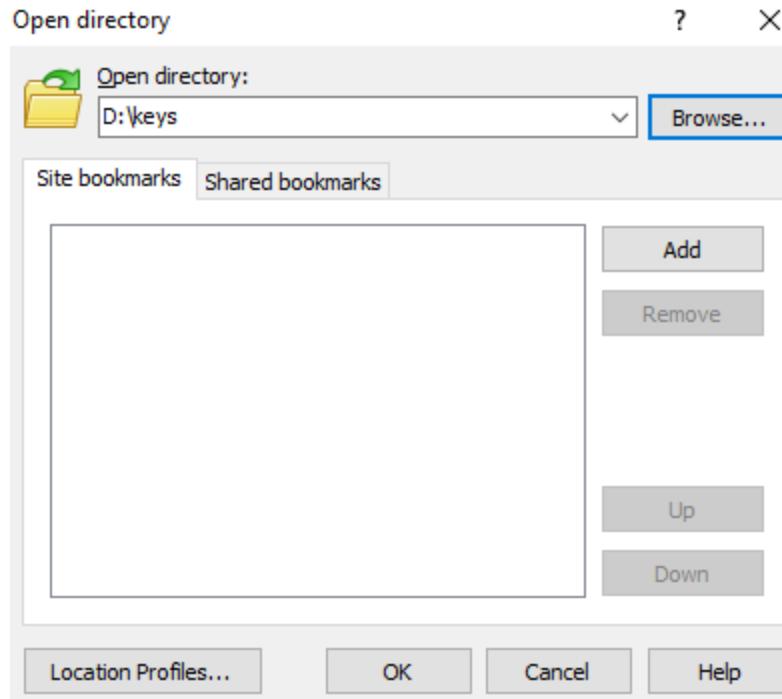
Open WinSCP, and select “New Site” in the left panel. On the right input the host information for the siem VM, then click the login button.



Once you login successfully, you are greeted with with a screen that has two panes: the left pane represents files and locations on the local computer. The right pane represents files and locations on the remote computer.



Double click on the directory path on the left pane (the portion in blue). In the Open directory window that pops up, click the Browse button. Navigate to the directory that holds your SSH public and private keys, then click OK.



The left pane changes to display the contents of the directory. Click and drag the file "windows-management-preformatted.pub" from the left pane to the right pane. If you did it correctly, the file will appear in the right pane.

D:\keys				/home/ayy			
Name	Size	Type	Changed	Name	Size	Changed	Rights
..		Parent directory	9/19/2016 8:11:19 PM	..		9/19/2016 12:10:08 PM	rw-rxr-x
windows-management.ppk	3 KB	PPK File	9/19/2016 3:34:47 PM				rw-rw-r--
windows-management.pub	1 KB	PUB File	9/19/2016 3:24:46 PM	windows-management-formatted.pub	1 KB	9/19/2016 7:37:07 PM	
windows-management-formatted.pub	1 KB	PUB File	9/19/2016 7:37:07 PM				

Exit WinSCP, open MRemoteNG, and open connect to the siem VM. once connected, run:

```
cat windows-management-preformatted.pub
```

```
ayy@siem:~$ cat windows-management-preformatted.pub
ssh-rsa AAAAB3NzaC1yc2EAAAQAAgEAkMq45DEMXo0aTjNQBkXrsx1Rp8X8FnbXlibyzc
8/vrL/36Db1SGEJqhK5pL3djUG5BpP4BgjtP0m/3np0Qo5BREwCXsn5K1eK+jshvbMMjUNKJu
x6mCZ8xGKCuQzkdQgU9YKDUWa3FXYNwtA7wgXnqaCIKvvBHTKP/nqtsb5L816pacbzirxawBj
NDDr1OcODT+DIQwEONaXnaPV7i44vJwzmooLQKuufV30FKrV7iqlDwbd/IxVTNHhhZMyk6Ntn
oez053iQCJ/Svib/t8rhQFlqw94oUcYjHCp3R00C4Oi90eRGchP/N8mq3Ne+sNpSLkxLQ5yMX
4/jc3s0FIXrfqiOLOy6Fx+T6j9eQeLmeaRm38VkWI/K6NXiRz2IyOP/GHuRuUSNPKrzsOOR4
fsM/4ImdcIRUxU0AkLPYpeHQDcBSvWVmdR1MqEFNo0f1DN2JrNNbv2W3qMxn2F9IFa6GPbuqh
ZeBeDjhXiFPEqiwiT3oOeIJ1DZOci9d5MTUm7579cChs9ad2ogVwRBjA8Db7/BpEaNe09af
clpivUjx8tjoFOEGI8w3ZrF0nyUV30Qwb1nNwGIujIuriI02iP/mGn7JD9o1y9E2QI2sb06ymN
51UWr890idgcg+ulHPRykBTs14hIdQqxjKMhq/8UNsvc7zfc7sn9LxuIkid7Jc= windows-m
anagement
ayy@siem:~$
```

This confirms that our file made it over in one piece. Next, run:

```
cat windows-management-formatted.pub >> ~/.ssh/authorized_keys
```

```
ayy@siem:~$ cat windows-management-formatted.pub >> ~/.ssh/authorized_keys
s
```

Making sure it worked

Once you have verified that your key has been successfully copied to .ssh/authorized_keys, type 'exit' to leave the terminal session and log off of the siem VM. Under the Config pane, click on the Password field, and delete the password you put in earlier from this field. Double click on siem in the Connections pane to reconnect to the VM.

```
Using username "ayy".
Authenticating with public key "windows-management"
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-38-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.

Last login: Mon Sep 19 19:51:22 2016 from 172.16.1.2
ayy@siem:~$
```

Now, if you didn't put a password on the SSH key we generated with PuTTYgen, you should be greeted with a login prompt with no need to enter a password. The server is comparing the public and private key to authenticate the user to the siem VM. If the comparison is successful (e.g. the Public key on the system is related to the private key we're sending to authenticate) then you are logged in.

If you DID generate a key passphrase when we generated our SSH keys earlier, you will be prompted to enter the password for that key now.

```
Using username "ayy".
Authenticating with public key "windows-management-auth"
Passphrase for key "windows-management-auth":
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-38-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.

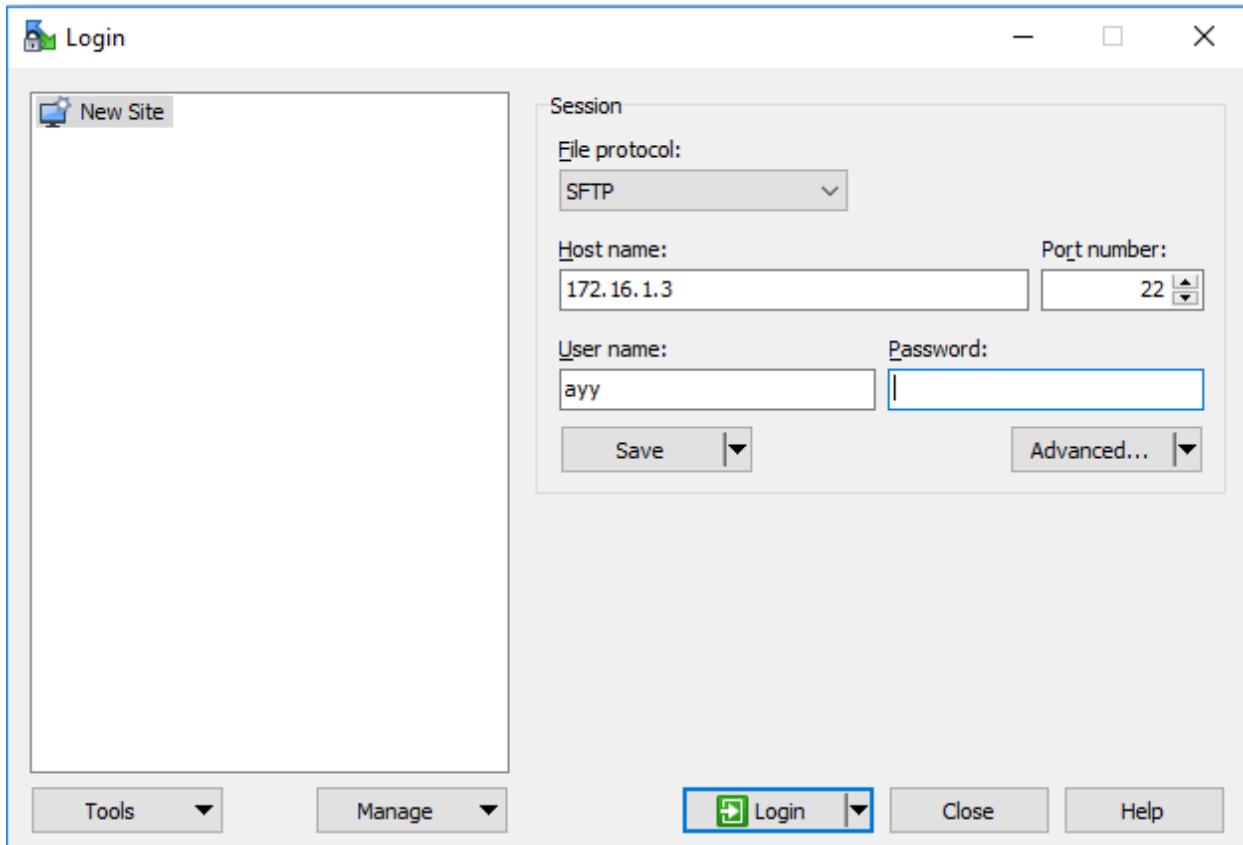
Last login: Mon Sep 19 19:54:12 2016 from 172.16.1.2
ayy@siem:~$
```

How to use Key-Based Authentication with WinSCP

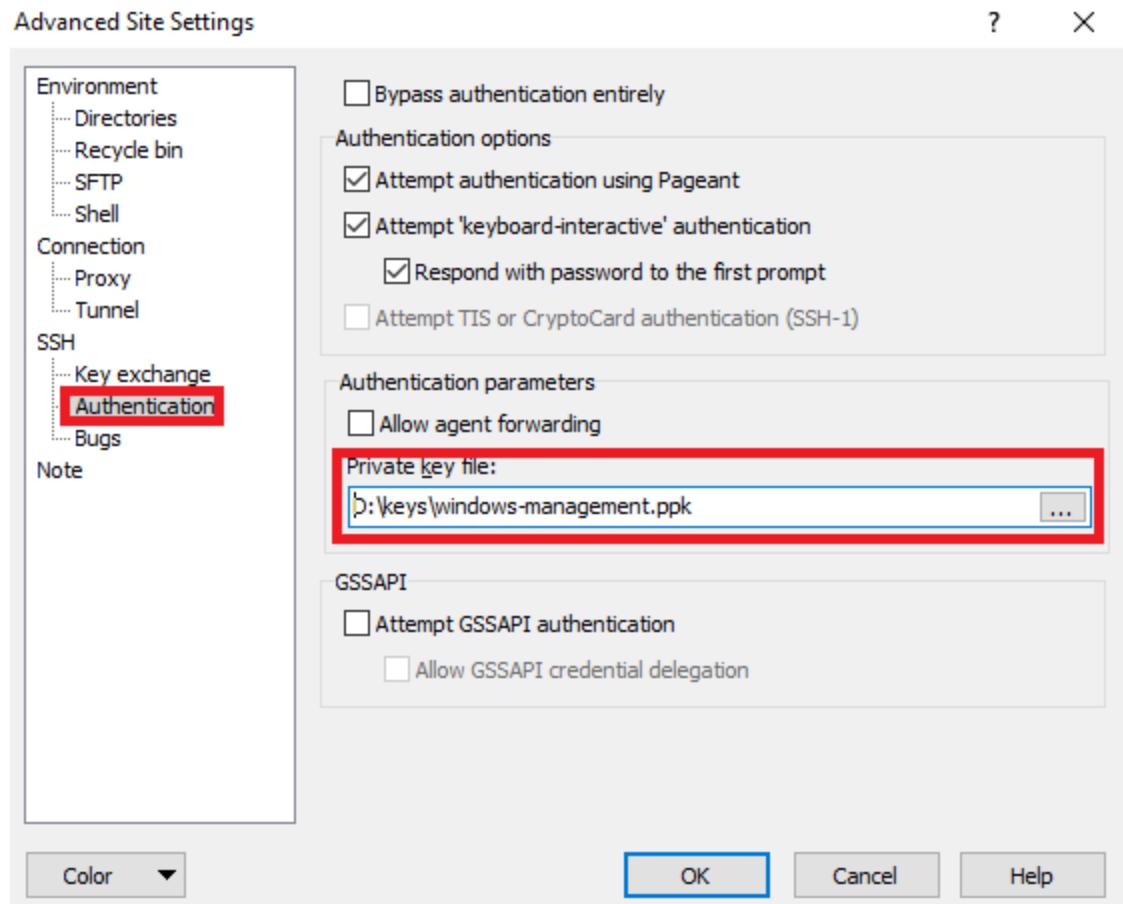
Assuming you followed the instructions above, if you're using windows, you should have, or be able to configure key-based authentication for SSH. SCP is a protocol that relies on using SSH as a secure transport; It is an alternative to FTP. In fact, I had mentioned that it was one of the methods you could use for adding your SSH public key to the Linux servers in our lab.

I'm going to teach you how to configure key-based authentication in WinSCP. I'll guide you through doing this on one VM, and leave it as an exercise to you if you wish to set up SCP to your other lab VMs.

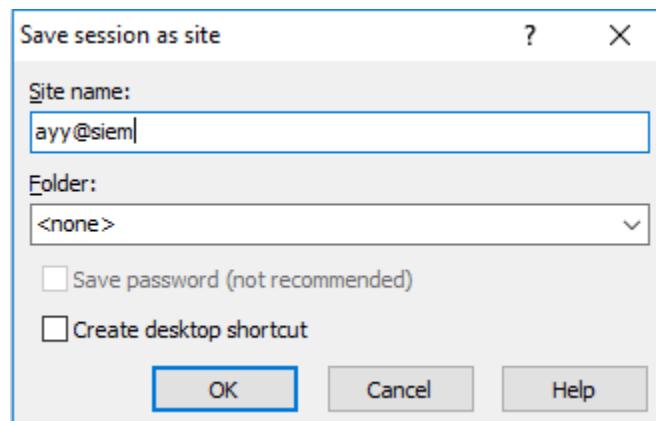
First, open WinSCP. You will be greeted with a pane that has a computer monitor icon and "New Site" highlighted, and a section of the screen labeled Session. We're going to create a session for siem. For the "Host name:" input box, enter 172.16.1.3. Leave the default value of 22 for the "Port number" box. For the "User name:" field, input the user account you configured SSH key-based authentication for on the siem VM (in my case, the user's name was "ayy").



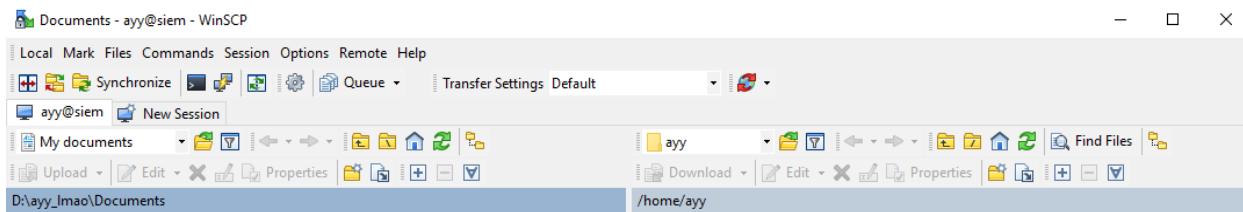
You'll note that I haven't mentioned the "Password:" field in WinSCP yet. That's because we aren't going to use it. After entering the username for the user you enabled key-based authentication for, click on the "Advanced..." button. The "Advanced Site Settings" window pops up. On the left pane, you'll notice the section titled "SSH". Under that section is a sub-menu "Authentication". Click on it. In the middle of the window is a section entitled "Authentication parameters", with an input box titled "Private key file:". Click on the gray ellipses to the right in the input box, and an explorer window pops up. Navigate to the PRIVATE SSH key that corresponds to the public key you already have configured on the host, and select it. Click "OK" in the Advanced Site Settings window to close it.



Next, click the Save button. A new window pops up entitled, “Save session as site”. Change the site name to “[username]@siem”, instead of an IP address, to more easily differentiate what system you will be connecting to. Click OK when you are done.



A new icon should appear on the left pane in the main window double click on it. If everything worked as expected, you should have an SCP session to the siem VM.



Remote Access for Linux and OSX

This section is dedicated to users who are running Linux, BSD, or OSX hosted hypervisors, or workstations you plan on using to establish network access as required (e.g. for both baremetal and/or hosted hypervisor networking).

The following sections will instruct you on how to configure static routes, make these routes persist between reboots, applications and commands I recommend to make remote management even easier, and finally instruct users on how to generate and use SSH keys for fast, easy and secure connectivity to the Linux lab VMs.

Static Routes in Linux and OSX

Almost all Linux, and BSD systems (yes, that includes OSX), include the route command for adding static routes. However, the syntax differs slightly between Linux and BSD variants. Additionally, Linux distros in general are deprecating the route command in favor of the ip command; specifically, `ip route add`.

For Linux systems we'll discuss using the ip command for adding static routes. For BSD and OSX systems, we will discuss using the BSD version of route for adding static routes.

Adding Routes to Linux with the ip Command

The ip command is considered the preferred way to request network information on modern linux systems, as well as add new routes to the system's network routing table. Entering the command:

```
ip route show
```

Will display the current contents of the routing table:

```
default via 172.16.1.1 dev eth0
172.16.1.0/24 dev eth0  proto kernel  scope link  src 172.16.1.4
```

The output indicates that the default gateway is 172.16.1.1 (the IP address of the pfSense VM's management interface). The default gateway is the address a host will forward any traffic that isn't in its local network, in hopes that the gateway knows where to send this traffic. You can also see that it is connected directly to the 172.16.1.0/24 network through its eth0 interface. Let's say we are using a Linux workstation as our hypervisor host, and we want to access the 172.16.2.0/24 network (IPS 1), from our virtual network adapter connected to the 172.16.1.0/24 network (Management). Enter the commands

```
ip route add 172.16.2.0/24 via 172.16.1.1  
ip route show
```

```
:~# ip route add 172.16.2.0/24 via 172.16.1.1  
:~# ip route show  
  
172.16.1.0/24 dev eth0  proto kernel  scope link  src 172.16.1.4  
172.16.2.0/24 via 172.16.1.1 dev eth0
```

Notice the line in the “ip route show” output that shows “172.16.2.0/24 via 172.16.1.1 dev eth0”. The device name (eth0) may change (depending on what version of Linux you run, and the default interface names), but other than that, the output should be more or less the same.

ip route add tells your system that there is a path to get to a certain network through a certain router or gateway system/device, but the firewall on pfSense controls whether or not the traffic will actually be allowed to be sent to systems on that network. For example, If you want to be able to SSH from 172.16.1.2 (the address for your host adapter connected to the management network) to 172.16.2.2 (the kali VM), you need to run ip route add to create a static route to the 172.16.2.0 network, and you will also need to have a firewall rule on the management network interface (LAN network in pfSense) to allow port 22/tcp (SSH) outbound to 172.16.2.2 for the connection to be allowed:

The screenshot shows the pfSense Firewall Rules LAN interface. At the top, there are tabs for Floating, WAN, LAN (which is selected), and OPT1. Below the tabs is a table titled "Rules (Drag to Change Order)". The table has columns for States, Protocol, Source, Port, Destination, Port, Gateway, Queue, Schedule, Description, and Actions. There are 8 rows of rules listed:

States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
<input type="checkbox"/> ✓	6/2.07 MiB	IPv4 TCP	172.16.1.2	*	172.16.1.1	443 (HTTPS)	*	none	Better anti-lockout	🔗 ✍️ 📄 ✖️ trash
<input type="checkbox"/> ✓	0/27 KiB	IPv4 TCP	172.16.1.2	*	172.16.2.2	22 (SSH)	*	none	Manage Kali over SSH	🔗 ✍️ 📄 ✖️ trash
<input type="checkbox"/> ✓	0/1.65 GiB	IPv4 TCP	172.16.1.0/24	*	172.16.1.1	3128	*	none	Allow access to proxy	🔗 ✍️ 📄 ✖️ trash
<input type="checkbox"/> ✓	0/1.32 MiB	IPv4 UDP	172.16.1.0/24	*	172.16.1.1	53 (DNS)	*	none	Allow DNS	🔗 ✍️ 📄 ✖️ trash
<input type="checkbox"/> ✓	0/0 B	IPv4 UDP	172.16.1.0/24	*	172.16.1.1	123 (NTP)	*	none	Allow NTP	🔗 ✍️ 📄 ✖️ trash
<input type="checkbox"/> ✗	0/117 KiB	IPv4 *	*	*	RFC 1918	*	*	none	Deny access to local networks	🔗 ✍️ 📄 ✖️ trash
<input type="checkbox"/> ✓	0/1.00 GiB	IPv4 TCP	172.16.1.0/24	*	*	443 (HTTPS)	*	none	allow HTTPS outbound	🔗 ✍️ 📄 ✖️ trash
<input type="checkbox"/> ✗	0/25.38 MiB	IPv4+6 *	*	*	*	*	*	none	Explicit Deny any/any	🔗 ✍️ 📄 ✖️ trash

At the bottom right of the table are buttons for Add (up and down arrows), Delete, Save, and Separator.

Once you have both a static route and a firewall rule that allows the traffic, you should be able to SSH to the kali VM with no problems, provided you set up the ssh service properly.

Adding Routes to OSX/BSD with the route command

OSX utilizes the route command to add new static routes on-demand. However, if you're used to how the old route command works on Linux, the OSX version of route behaves a little bit differently, because it is based on the BSD version of route. Lets run through how to add a static route to OSX using the BSD version of route.

Note: For those of you running BSD for your management system or hypervisor host, you should be able to follow along as well.

For our example, let's say you are running VMware Fusion as a hosted hypervisor on OSX. Your macbook has a vmnet1 network card that you designated as the "Management" network. Your vmnet1 network card has the static IP address of 172.16.1.2, with a subnet mask of /24, just like the VMware Fusion guide recommends. You want to be able to be able to SSH to a system on the IPS network (172.16.2.0/24) particularly, our kali VM at 172.16.2.2.

First and foremost, You need to have a firewall rule set up on the Management interface of the pfSense VM to allow the connection on port 22/tcp from 172.16.1.2 to 172.16.2.2.

Firewall / Rules / LAN

States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
<input type="checkbox"/> ✓	6/2.07 MiB	IPv4 TCP	172.16.1.2	*	172.16.1.1	443 (HTTPS)	*	none	Better anti-lockout	
<input type="checkbox"/> ✓	0/27 KiB	IPv4 TCP	172.16.1.2	*	172.16.2.2	22 (SSH)	*	none	Manage Kali over SSH	
<input type="checkbox"/> ✓	0/1.65 GiB	IPv4 TCP	172.16.1.0/24	*	172.16.1.1	3128	*	none	Allow access to proxy	
<input type="checkbox"/> ✓	0/1.32 MiB	IPv4 UDP	172.16.1.0/24	*	172.16.1.1	53 (DNS)	*	none	Allow DNS	
<input type="checkbox"/> ✓	0/0 B	IPv4 UDP	172.16.1.0/24	*	172.16.1.1	123 (NTP)	*	none	Allow NTP	
<input type="checkbox"/> ✗	0/117 KiB	IPv4 *	*	*	RFC 1918	*	*	none	Deny access to local networks	
<input type="checkbox"/> ✓	0/1.00 GiB	IPv4 TCP	172.16.1.0/24	*	*	443 (HTTPS)	*	none	allow HTTPS outbound	
<input type="checkbox"/> ✗	0/25.38 MiB	IPv4+6 *	*	*	*	*	*	none	Explicit Deny any/any	

Add
 Add
 Delete
 Save
 Separator

However, if you try to SSH to the kali VM, you'll notice that your connection times out. This is because your mac's routing table does not have a route to the 172.16.2.0/24 network. It will try to send this traffic to your current default gateway (e.g. a local network gateway, a SOHO router, or other network device you use for internet access), and the default gateway won't know how to route the traffic either, so your connection times out, because your system has no idea how to get the packets to kali VM. This is where we use route to modify our system's routing table. In this case, you would run the command

```
route add 172.16.2.0/24 172.16.1.1
```

This tells the system that for any packets destined for the 172.16.2.0/24 network, send them to 172.16.1.1 for them to be forwarded correctly. If you were to attempt to SSH to 172.16.2.2 again (assuming you allowed port 22/tcp through the firewall, and the SSH service was enabled and listening on the kali VM), you would then be able to connect to the system via SSH.

Making Static Routes Persistent

Now, it's one thing to know how to add a static route to Linux, BSD or OSX, but if and when you reboot your system, you will lose any static routes you created. This section details ways you can make your routes persistent, as well as some workarounds for quirks that I discovered in VMWare Fusion on OSX.

Linux and BSD Route Persistence via /etc/rc.local

On most Linux and BSD systems there is a file in /etc you can use called rc.local. This file is a way for you to say “After all the other services on the system are executed, I want you to run these commands.” We can use this to have the system set up our static route(s) every time the system is rebooted. While using rc.local is considered lazy, considering all we want to do is add a static route on boot, it serves our purpose perfectly.

```
root@ips:~# ls -al /etc/rc.local
-rwxr-xr-x 1 root root 306 Jul 19 16:43 /etc/rc.local
root@ips:~# cat /etc/rc.local
#!/bin/sh -e
#
# rc.local
#
# This script is executed at the end of each multiuser runlevel.
# Make sure that the script will "exit 0" on success or any other
# value on error.
#
# In order to enable or disable this script just change the execution
# bits.
#
# By default this script does nothing.

exit 0
```

In the illustration above, I ran ls -al /etc/rc.local, then I used the cat command to display the contents of rc.local. This was to highlight a couple of important features of this file:

1. The file should be owned by root
2. The file needs execute permissions for the root user at a minimum in order for the system to execute it and run the commands contained within
3. The script does nothing by default, but run “exit 0” to exit. You would place any commands you want rc.local to run BEFORE the exit 0 line.
4. If you have to create the file from scratch, the file must begin with the “#!/bin/sh” line, and end with the “exit 0” statement.

By default on Ubuntu Linux (16.04.1 at least), rc.local exists, has the execute permissions set, and looks exactly like it does in the illustration above. All you have to do is add any “ip route add” statements on lines above the “exit 0” line, like so:

```
#!/bin/sh -e
#
# rc.local
#
# This script is executed at the end of each multiuser runlevel.
# Make sure that the script will "exit 0" on success or any other
# value on error.
#
# In order to enable or disable this script just change the execution
# bits.
#
# By default this script does nothing.
ip route add 172.16.2.0/24 via 172.16.1.1
exit 0
```

What if you need to access your virtual networks hosted on a baremetal hypervisor and need to add more than one route? Simply add them above the “exit 0” line (one command per line), and save your changes. Reboot your Linux or BSD host. If you entered the ip route command to rc.local correctly, you should have a static route to 172.16.2.0/24 in your routing table.

```
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-45-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.

Last login: Fri Nov  4 09:18:03 2016 from 172.16.1.2
root@ips:~# ip route show

172.16.1.0/24 dev eth0  proto kernel  scope link  src 172.16.1.4
172.16.2.0/24 via 172.16.1.1 dev eth0
```

Note: The process of using /etc/rc.local is more or less the same for BSD users, however if you are using BSD, you will need to use the [route](#) command instead of ip route add. Also be aware that the /etc/rc.local method doesn't exactly work for OSX systems.

OSX Route Persistence on Hosted Hypervisors

I discovered a unique issue using OSX and VMWare Fusion when I was working on this guide (I am not entirely sure if this issue affects VirtualBox users on OSX, but I know this is a problem with VMWare Fusion on OSX). If VMWare Fusion is not running, the virtual interfaces and networks that one creates in the application simply cease to exist in OSX until you run VMWare

Fusion again. This is why when you restart VMWare Fusion, or reboot OSX, you have to run sudo ifconfig vmnet2 to reset your IP address on the management virtual network. Don't understand? Pay attention:

```
Tony's-MacBook-Pro:~ trobinson$ ps -ef | grep vmware; ifconfig -a | grep vmnet2
  0 9933      1  0  8:19PM ??          0:00.00 /Applications/VMware Fusion.app/Contents/
Library/vmnet-dhcpd -s 6 -cf /Library/Preferences/VMware Fusion/vmnet1/dhcpd.conf -lf /var
/db/vmware/vmnet-dhcpd-vmnet1.leases -pf /var/run/vmnet-dhcpd-vmnet1.pid vmnet1
  0 9942      1  0  8:19PM ??          0:00.00 /Applications/VMware Fusion.app/Contents/
Library/vmnet-dhcpd -s 6 -cf /Library/Preferences/VMware Fusion/vmnet8/dhcpd.conf -lf /var
/db/vmware/vmnet-dhcpd-vmnet8.leases -pf /var/run/vmnet-dhcpd-vmnet8.pid vmnet8
  0 9944      1  0  8:19PM ??          0:00.09 /Applications/VMware Fusion.app/Contents/
Library/vmware-usbarbitrator --kext /Applications/VMware Fusion.app/Contents/Library/kexts
/vmioplug.kext
  501 9956 9442  0  8:19PM ttys000    0:00.00 grep vmware
vmnet2: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
Tony's-MacBook-Pro:~ trobinson$ ps -ef | grep vmware; ifconfig -a | grep vmnet2
  501 10012 9442  0  8:20PM ttys000    0:00.00 grep vmware
Tony's-MacBook-Pro:~ trobinson$
```

These two illustrations run the following commands:

```
ps -ef | grep vmware
ifconfig -a | grep vmnet2
```

The first illustration shows you a couple of processes that VMWare Fusion runs in the background to support various functions. The last line shows us any lines in ifconfig -a that have the text vmnet2, the interface we used to have our hypervisor host connect to the management virtual network. We can see in that very last line of output that vmnet2 does exist and OSX recognizes the virtual network interface.

In the next illustration, we confirm that vmware is not running. The output for ifconfig -a | grep vmnet2 returns no output; vmnet2 simply disappears. There is no way we can create routes to networks and gateways that do not exist. The route command requires OSX to know how to reach a particular gateway before we can send packets to it destined for other networks. So how do we work around this? We're going to make a simple shell script that you can run after starting VMWare Fusion.

```
flightcheck.sh
```

Log in to OSX and open up your favorite text editor. I prefer to open up the terminal application and use vi. Create a file in your home directory (~/) called flightcheck.sh and input the following:

```

#!/bin/bash
#This script is meant for VMWare Fusion Professional users.
#This script assumes that vmnet2 exists and is configured to be our
management virtual network.
echo "Checking for root privs.."
if [ $(whoami) != "root" ]; then
    echo "This script must be ran with sudo or root privileges."
    exit 1
else
    echo "We are root."
fi
ifconfig vmnet2 172.16.1.2 netmask 255.255.255.0
if [ $? -eq 0 ]; then
    echo "vmnet2 interface IP set to 172.16.1.2"
else
    echo "Could not set IP address of vmnet2 interface. Does vmnet2 exist?
Is VMWare Fusion running?"
    exit 1
fi
route add 172.16.2.0/24 172.16.1.1
if [ $? -ne 0 ]; then
    echo "Could not create route to 172.16.2.0/24. Does the network exist?
Is VMWare Fusion running? Is the pfSense/Gateway VM running?"
    exit 1
fi
exit 0

```

Save the script when you are done, and change the file to have execution permissions at a minimum (e.g chmod 700 ~/flightcheck.sh). Start up VMWare Fusion, and then start up all of your virtual machines. After your VMs have finished booting, open a terminal and run the command:

```
sudo ~/flightcheck.sh
```

You should be prompted to enter your password. This is so we can run the commands in our script with root privileges. Otherwise, none of the commands will work. This is what it looks like when the script completes successfully:

```
Tony's-MBP:~ trobinson$ sudo ~/flightcheck.sh  
Password:  
Checking for root privs..  
We are root.  
vmnet2 interface IP set to 172.16.1.2  
add net 172.16.2.0: gateway 172.16.1.1  
Tony's-MBP:~ trobinson$ █
```

OSX route persistence for Baremetal Hypervisors

As we know from years and years of Apple marketing, they like to *think different*. That different thinking isn't always better and leads to having to discover the Apple way of doing things. Apple uses a BSD-like core, but a lot of the functionality built into OSX is its own amalgamation of *different*.

I tried a number of tricks a few years of doing systems administration of Unix-like and Linux systems that I was led to believe might actually work in OSX for doing route persisted in some sort of a fully automated manner on reboot. As previously established, `rc.local` doesn't exactly work on OSX. I also tried creating a crontab entry as root to run a script at boot and that failed. THEN I tried making a `launchd` (Launch Daemon) script to run a shell script on boot and THAT failed. It turns out that running commands as root to rebuild a routing table is really complicated. That, combined with the fact that I don't know nearly enough about the dark inner workings of OSX to try and fix this. So heres the deal:

I made a modified version of the `flightcheck.sh` script we used for setting up networking in the case of hosted hypervisors running locally on OSX. This modified script won't automatically run at boot, but I've provided with instructions on how to run it as root on-demand to fix your static routes as necessary. If you can find a way to run this script as the root user automatically at boot with no need to enter a password, then great.

`flightcheckBM.sh`

Use your favorite text editor and create the file `flightcheckBM.sh` in your home directory (e.g. `~/flightcheckBM.sh`) with the following content:

```
#!/bin/bash  
#this script can help automate building static routes to virtual networks  
hosted on a baremetal hypervisor.
```

```

echo "Checking for root privs.."
if [ $(whoami) != "root" ]; then
    echo "This script must be ran with sudo or root privileges."
    exit 1
else
    echo "We are root."
fi
route add 172.16.1.0/24 192.168.1.22
if [ $? -ne 0 ]; then
    echo "Could not create route to 172.16.1.0/24. Does the network exist?
Is ESXi running? Is the pfSense/Gateway VM running?"
    exit 1
fi
route add 172.16.2.0/24 192.168.1.22
if [ $? -ne 0 ]; then
    echo "Could not create route to 172.16.2.0/24. Does the network exist?
Is ESXi running? Is the pfSense/Gateway VM running?"
    exit 1
fi
exit 0

```

Save the script when you are done, and change the file to have execution permissions at a minimum (e.g chmod 700 ~/flightcheckBM.sh).

Note: 192.168.1.22 was the IP address of the pfSense WAN interface on my local network, you will need to adjust this ip address in the route commands above as necessary for your home or office network.

When you are ready, open a terminal and run the command:

```
sudo ~/flightcheckBM.sh
```

You should be prompted to enter your password. This is so we can run our commands to modify the routing table as the root user; only root can modify the routing table on our system. This is what it looks like when the script completes successfully:

```
Tonys-MBP:~ trobinson$ sudo ~/flightcheckBM.sh  
Password:  
Checking for root privs..  
We are root.  
add net 172.16.1.0: gateway 192.168.1.22  
add net 172.16.2.0: gateway 192.168.1.22  
Tonys-MBP:~ trobinson$
```

The ssh and scp terminal Applications

Most Unix/Linux systems have the command line utilities ssh and scp already installed as core utilities, installed with the OS by default. This means that almost every Linux distro, BSD variant (and by extension, every OSX install) will have them by default. Being a minimalist at heart, I like to live off the land and use applications that are available by default when and if possible, so using the ssh and scp commands suits me just fine. For the novice user not use to the command-line environment, the syntax for ssh and/or scp may take some getting use to, but like everything else practice (and time) makes perfect.

As mentioned above, the man pages will usually provide you all the information you need for using a command including all of the unique options available for advanced usage. For sake of completion however, I will show you how to interact with the lab VM siem using the ssh and SCP command line clients. For this exercise, we are going to assume that you are using a Linux, BSD, or OSX system with a hosted hypervisor, with a virtual network card connected to the management network, with the ip address 172.16.1.2 though, the process is mostly the same on a hosted hypervisor, provided you have static routes and firewall rules configured as necessary.

To start an ssh session to the siem VM, all you need to do is type:

```
ssh 172.16.1.3
```

Upon connection, the server will ask what username you wish to login as, and/or what password you'd like to use. To make things slightly faster, your basic ssh command can be modified to:

```
ssh [username]@172.16.1.3
```

In the command above, replace [username] with the user you wish to log in as. For instance, if I want to log in as the user “ayy”, the command is:

```
ssh ayy@172.16.1.3
```

If done correctly, you should be given a prompt to enter a password for the user you wish to connect as. If you entered the username and password correctly, you're greeted with a shell session on the system.

The scp program allows you to transfer files to and from a target system. The syntax varies slightly, depending on whether or not you're trying to download a file from the target system, or sent a file to the target system. For example:

```
scp /Users/lmao/data.txt ayy@172.16.1.3:/home/ayy/data.txt
```

In the example above, we are attempting to send the file "data.txt" located at /Users/lmao/data.txt and trying to transfer it to 172.16.1.3 as the user "ayy" to the path /home/ayy/data.txt. If the connection to 172.16.1.3 is successful, scp will ask for the password for the user "ayy". If you enter the password correctly, the file will be transferred. Now here's another example:

```
scp ayy@172.16.1.3:/home/ayy/results.pdf /Users/lmao/results.pdf
```

This command we're attempting to copy the "results.pdf" file from 172.16.1.3, in the directory /home/ayy, authenticating as the user "ayy", and copy the file to /Users/lmao/results.pdf on your local machine.

iTerm2 and Terminator

While typically on Unix/Linux systems, the scp and ssh commands are enough to suit my needs, there are a pair of terminal programs -- one for OSX, and another for Linux systems that I highly recommend using for the wonderful features they have.

The OSX application iTerm2 is an application that replaces the built-in iTerm application for OSX. While there are a lot of neat features that I don't use on a regular basis, the features that I use (and love) include the ability to create multiple tabs for multiple terminal sessions, and the ability to take a single window/tab and split it vertically and/or horizontally to fit multiple SSH sessions into a single application window. Visit <https://www.iterm2.com/> if you're interested.

Terminator is essentially the Unix/Linux equivalent with a comparable featureset. Most distros have terminator as a part of their distro's included packages (I know for certain that Kali Linux and Ubuntu provide it via apt/aptitude by default, if nothing else). Find out more at <https://gnometerinator.blogspot.com/p/introduction.html>

Generating ssh keys using ssh-keygen

In addition to having ssh and scp clients available as core functionality, most Linux, Unix, and BSD variants also have the key generation program ssh-keygen. The ssh-keygen program is used to generate public and private keys to be used as a method of authentication for ssh connections; Generating a public and private key, and configuring the SSH server you normally connect to to use your public key can allow you to use that public key to authenticate to the SSH server instead of using a password. Additionally, you can put a password on your SSH private key, and utilize that password as a second factor of authentication in order to connect to the SSH server in question.

Running ssh-keygen is as simple as just typing

ssh-keygen

```
Tony's-MBP:~ trobinson$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/Users/trobinson/.ssh/id_rsa):
Created directory '/Users/trobinson/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /Users/trobinson/.ssh/id_rsa.
Your public key has been saved in /Users/trobinson/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:heLkIAmwQFMBgwlKbXYUadsIKFTUgYOL1JQ50QuBbgI trobinson@Tonys-MBP.blindseeker.com
The key's randomart image is:
+---[RSA 2048]---+
|&%/==++|
|E%oX = . |
|O O B * . .|
|o+ o B o .|
|o o S |
| |
| |
| |
+---[SHA256]---+
```

By default, when you run ssh-keygen, the directory .ssh is created in the current user's home directory, and the files id_rsa and id_rsa.pub are created. The file id_rsa is the private key, while the file id_rsa.pub on the other hand, is the public key. The information contained in id_rsa.pub has to be copied to any system you want to use SSH key-based authentication on (We'll be going over how to do this in just a moment), while the data in the id_rsa file must be kept secure and private at all costs. When you run ssh-keygen, the application will ask you if you wish to set

a password for your SSH private key. Leaving the password blank is an option, if you don't want to have to enter a password when you use your key for authentication, however if you don't set a password, anyone who acquires your id_rsa file will be able to ssh to any system you set up key-based authentication on. By default ssh-keygen will generate RSA keys that are 2048 bits in length. If you want to change this to a higher number, for stronger keys, (e.g. 4096) use the option "-b 4096" when running ssh-keygen.

Regardless of whether or not you decide to set a password with ssh-keygen, I cannot stress the importance protecting your id_rsa file enough. If someone manages to get a hold of that file (and it's not password-protected, or they also have the password), then they can log in to any system you set up key-based authentication for as your user; they can essentially become you. On the other hand, you need to have the contents of the id_rsa.pub file on hand for any systems you want to set up key based authentication for. The contents of this file need to be copied to to file "authorized_keys" in the .ssh directory of the user you want to use key-based authentication for. I'll be demonstrating a couple of ways you can do this in just a moment, but for now, run ls -al ~/.ssh and verify that both id_rsa, and id_rsa.pub have been generated on your workstation.

```
Tonys-MBP:~ trobinson$ ls -al .ssh
total 40
drwx----- 7 trobinson staff 238 Sep  6 06:57 .
drwxr-xr-x+ 26 trobinson staff 884 Dec  2 13:48 ..
-rw-r--r--  1 trobinson staff 1133 Sep  6 06:57 authorized_keys
-rw-r--r--  1 trobinson staff   39 May 23 2016 config
-rw-----  1 trobinson staff 3326 May 23 2016 id_rsa
-rw-r--r--  1 trobinson staff  759 May 23 2016 id_rsa.pub
-rw-r--r--  1 trobinson staff 3123 Dec  1 16:49 known_hosts
Tonys-MBP:~ trobinson$
```

The alias Command

The alias command, like most the commands mentioned so far is available by default in most Linux, Unix and BSD systems. What alias allows you to do is define a sort of "shortcut" for running certain commands. For example, we type the command alias in an ssh session on the siem VM, and we can see a couple of aliases already configured by default

```
alias egrep='egrep --color=auto'
alias fgrep='fgrep --color=auto'
alias grep='grep --color=auto'
alias l='ls -CF'
alias la='ls -A'
alias ll='ls -alF'
alias ls='ls --color=auto'
```

So why are we interested in the command? We can use what are called “resource files” to make permanent aliases to tell the terminal/shell every time we type a certain aliased command, to open an SSH session to a remote host.

```
root@siem:~# alias ips='ssh ayy@172.16.1.4'
root@siem:~# ips
The authenticity of host '172.16.1.4 (172.16.1.4)' can't be established.
ECDSA key fingerprint is SHA256:oAK0mdjyPE5+2/mnwvZFnM0dbIKHz1n0i3xXZef/9k.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '172.16.1.4' (ECDSA) to the list of known hosts.
ayy@172.16.1.4's password:
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-38-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

1 package can be updated.
0 updates are security updates.

*** System restart required ***
Last login: Mon Sep 19 12:14:39 2016
ayy@ips:~$
```

In this way, we can sort of create connection profiles and more easily automate connecting to our lab VMs. The problem with the alias above is that as soon as the terminal session is closed, the alias is lost.

The way around that is to use the shell resource files in your user’s home directory to give them persistence. By default, on many Linux and Unix systems, the BASH (Bourne Again SHell) shell is the default shell used for command-line interactions (BSD derivatives usually have KSH (Kourne SHell) configured as the default OS X is an exception, and uses the BASH shell by default). If you are running a Linux/Unix/BSD system where BASH is not the default, I highly recommend installing BASH and making it the default shell for your user. Consult documentation for the OS of your choice on how to do this. Usually in each user’s directory there is file called “.bashrc” or “.bash_profile” (If neither of these files exist in your user’s home directory, feel free to create either file using a text editor of your choosing). These are files that are used to customize various settings in the BASH shell to a user’s liking. When you log in, and the bash shell is the default shell for your system (or say, if you use a graphical interface and start a terminal application), if these files exist in your user’s home directory, they are automatically read, and their settings are applied. As we saw in the screen capture above, there are several aliases that are configured by default, at least on Ubuntu systems, and more than likely, configured through .bashrc or .bash_profile. So that means if you have aliases you want to remain persistent, you can define them in either file. We can simply do so by appending our alias to the end of either file. Try running the following:

```
cd ~/;cp ~/.bash_profile ~/.bash_profile_bak;echo "alias ips='ssh ayy@172.16.1.4'" >> ~/.bash_profile
```

```
Tonys-MBP:~ trobinson$ cd ~/;cp ~/.bash_profile ~/.bash_profile_bak; echo "alias ips='ssh ayy@172.16.1.4'" >> ~/.bash_profile  
cp: /Users/trobinson/.bash_profile: No such file or directory
```

What this series of commands does is change directories back to the current user's home directory, makes a copy of .bash_profile (if it exists) and names it .bash_profile_bak (in case we do something that "breaks" .bash_profile, we can restore from our backup), then appends our alias command (sets the "ips" command to open an ssh session to 172.16.1.4 as the user "ayy"). If "ayy" is not a user on your ips VM, substitute with the correct username as necessary) to the end of the .bash_profile file, or if the file doesn't exist, creates the file with your alias in it. If everything went right you should see a line like this when you run the command (from your user's home directory)

```
cat .bash_profile
```

```
Tonys-MBP:~ trobinson$ cat .bash_profile  
alias ips='ssh ayy@172.16.1.4'
```

Note: if ~/.bash_profile does NOT exist, the command "cp ~/.bash_profile ~/.bash_profile_bak;" will print an error message stating "No such file or directory". There is no need to worry about this, especially if cat ~/.bash_profile displays the alias correctly.

Our alias should be appended to the end of the .bash_profile now. If you want to test the alias now to make sure it works, Try the following:

```
source ~/.bashrc; alias
```

```
Tonys-MBP:~ trobinson$ source ~/.bash_profile; alias  
alias ips='ssh ayy@172.16.1.4'
```

If you don't get any errors, and you see your new alias listed among the other system default aliases (if there are any defined), then everything worked fine. The source command tells the shell to execute the contents of the file and apply the settings found within to the current shell. In a nutshell, this applies our configuration settings without forcing us to logout, if we don't want to. To verify your new aliased command actually makes an SSH connection properly, simply type

```
ips
```

On the command line. You should be prompted for a password. After entering the password for the the “ayy” user on the ips VM (or the user you set up on the ips VM), you should be connected.

```
Tony's-MBP:~ trobinson$ ips
ayy@172.16.1.4's password:
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-51-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.

Last login: Thu Dec  1 16:19:34 2016 from 172.16.1.2
ayy@ips:~$
```

If you want to create aliases for the other virtual machines, it should be fairly simple. Simply reuse the instructions above, replacing the IP address, alias name, and username as required. The way the command is formatted above, if .bashrc exists, it simply appends your new aliases to the bottom of the file, instead of overwriting them.

Enabling Key-Based Authentication in Unix/Linux Systems

If you haven’t ran ssh-keygen on your Linux, Unix, or OSX host yet already, please do so now. If you’re continuing on from the previous section, you should have an SSH session opened up as your user account to the ips VM, and you should already be in your user’s home directory. (if you are not, run the command “cd ~”). If not, open up an SSH session to the IPS VM at 172.16.1.4. Once you are logged in, run the following command:

```
mkdir ~/.ssh;chmod 700 ~/.ssh; touch ~/.ssh/authorized_keys;chmod 600
~/.ssh/authorized_keys
```

```
ayy@ips:~$ mkdir ~/.ssh; chmod 700 ~/.ssh; touch ~/.ssh/authorized_keys;chmod 600
~/.ssh/authorized_keys
```

To verify everything executed properly, run the following command:

```
ls -al ~/.ssh
```

You should have something that looks like this:

```
ayy@ips:~$ ls -al ~/.ssh
total 8
drwx----- 2 ayy ayy 4096 Dec  2 13:38 .
drwxr-xr-x  4 ayy ayy 4096 Dec  2 13:38 ..
-rw-----  1 ayy ayy    0 Dec  2 13:38 authorized_keys
```

The first command was actually 4 commands in rapid succession. We created the hidden directory “.ssh”, gave it file permissions that only allows the user who created it access to the directory, the created the empty file “authorized_keys” with the touch command (if the file does NOT already exist, touch creates it), then gave it file permissions to where only our user can read or write to it.

Key Copy Method 1: echo append to authorized_keys

You can use your favorite graphical text editor, or in the command line, using the command line (open another terminal, or if you’re using iTerm2/terminator, open another tab) and run

```
cat ~/.ssh/id_rsa.pub
```

Highlight all of the text in the file from beginning to end and hit **ctrl + c** (or on a mac **meta + c**) to copy its contents to your clipboard.

```
Tony's-MBP:~ trobinson$ cat ~/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1n32EAAAQABAA91QDNu5GavH6nP0TGdKFmmzBJdA/NYQ/coDaAbcRBXlWR
twFTAjExUKv0A/5D4ftoXBIsSjSSvxJxyVAZas0QIlmJU+8lMroaZ/fAI48d7kUM0YhzcMxIbvjap0V
vMkizTXsxq0Uus4w840aI9rFLCkj03wPZbYtA5U13qHjBAqWbz0G7DckvlClky0kQZLUBS1TI59hjTo0
xN3veIykp9KYT1JF8FwRZp4V9pWbT/9lCrcLGWmVZzxc4+V5iBnAW+IQuVA53ESGTYJlPyI4TVoJR7iV
UTamZ2GPpeP43Ge81CkkxA5UPVcnn0XPAkSl0zZP50W21xWJHjg1U3y8cdne9rpLV0dtZ95mogw3e0v5
zKyEfRq9KnC0ZrfiINBWWn1t+0lYbeR7PZ1dTuh0tZM1eC8se9yq6TGEc90Xku8mVAXfPpaB/rt9Nf1H
t6XlzLns5jx2BUf76/jjzESRwzj2XyL8uzFZ7m6EgDH1BrZmLU5GU0jjws8WqWoh01qmeWqfB1fwD3mJ
5t7h094/kKFQotQ2SHLRroyE0BCn3fqrlUJJp+VgBlHb0ahVBwzpWyIg4twPHvLNYDDxw2qU+fepcQ1W
D1PMNZJsa//asu/y7+07q+771Iql9wzcBT30A0nzS4yJ07/TuqFsG0iETNPP2E300ALuu7JhrIpJrzQ6
Rw== trobinson@Tonys-MacBook-Pro.local
```

Back in our SSH session to the ips VM, enter the following command:

```
echo “[ctrl + v or meta + v (on a mac)]” >> ~/.ssh/authorized_keys
```

Be sure to include the double quotes in the command above!

```
ayy@ips:~$ echo "ssh-rsa AAAAB3NzaC1n32EAAAQABAA91QDNu5GaVH6nPoTGdKFmmzBJdA/NYQ/coDaAbcRBXlWRtwFTAjExUKv0A/5D4ftoXBIsSjSSvxJxyVAZas0QI1mJU+81MroaZ/fAI48d7kUM0YhzcMxIbvjap0VvvMkizTXsxq0Uus4w840aI9rFLCkj03wPZbYtA5U13qHjBAqWbz0G7Dckv1C1ky0kQZLUBS1TI59hjTo0xN3veIykp9KYT1JF8FwRZp4V9pWbT/91CrcLGWmVZzxc4+V5iBnAW+IQuVA53ESGTYJ1PyI4TVoJR7iVUTamZ2GPpeP43Ge81CkxxA5UPVcnn0XPAkS10zZP50W21xWJHjg1U3y8cdne9rpLV0dtZ95mogw3e0v5zKyEfRq9KnC0ZrfiINBWWn1t+01YbeR7PZ1dTu0tZM1eC8se9yq6TGEc90Xku8mVAXfPpaB/rt9Nf1Ht6X1zLns5jx2BUf76/jjzESRwjz2XyL8uzFZ7m6EgDH1BrZmLU5GU0jjws8WqWoh01qmeWqfB1fwD3mJ5t7h094/kKFQotQ2SHLRroyE0BCn3fqrLUJJp+VgBLHb0ahVBwzpWlyIg4twPHvLNYDDxw2qU+fepcQLWD1PMNZJsa//asu/y7+07q+771Iql9wzcBT30A0nzS4yJ07/TuqFsG0iETNPP2E300ALuu7JhrIpJrzQ6Rw== trobinson@Tonys-MacBook-Pro.local" >> ~/.ssh/authorized_keys
```

When you run `ctrl+v` (or in OSX meta + v), this dumps the current contents of your clipboard. In our case, this is the content of `id_rsa.pub` file. This should have copied the contents of “`id_rsa.pub`” in between the single quotes. We’re then telling the terminal to append this text to the `.ssh/authorized_keys` file. Next, run the following command:

```
cat ~/.ssh/authorized_keys; wc -l
```

The `cat` command reads the contents of a file and reads it back out to the terminal, while the `wc -l` command normally reads all the words of a file. The “-l” option is being used to count the number of lines in the `authorized_keys` file. It is VERY important that all of the contents of `id_rsa.pub` got written to the `authorized_keys` file on a single line. There can be no newlines in public keys entered into the `authorized_keys` file. If there are, then they need to be fixed. Each key entered into the `authorized_keys` file should only register as a single line in the file, and considering this is the first public key we’ll be entering into the file, it should be the ONLY line in the file. Your output should look similar to this:

```
ayy@ips:~$ cat ~/.ssh/authorized_keys; wc -l ~/.ssh/authorized_keys
ssh-rsa AAAAB3NzaC1n32EAAAQABAA91QDNu5GaVH6nPoTGdKFmmzBJdA/NYQ/coDaAbcRBXlWRtwFTAjExUKv0A/5D4ftoXBIsSjSSvxJxyVAZas0QI1mJU+81MroaZ/fAI48d7kUM0YhzcMxIbvjap0VvvMkizTXsxq0Uus4w840aI9rFLCkj03wPZbYtA5U13qHjBAqWbz0G7Dckv1C1ky0kQZLUBS1TI59hjTo0xN3veIykp9KYT1JF8FwRZp4V9pWbT/91CrcLGWmVZzxc4+V5iBnAW+IQuVA53ESGTYJ1PyI4TVoJR7iVUTamZ2GPpeP43Ge81CkxxA5UPVcnn0XPAkS10zZP50W21xWJHjg1U3y8cdne9rpLV0dtZ95mogw3e0v5zKyEfRq9KnC0ZrfiINBWWn1t+01YbeR7PZ1dTu0tZM1eC8se9yq6TGEc90Xku8mVAXfPpaB/rt9Nf1Ht6X1zLns5jx2BUf76/jjzESRwjz2XyL8uzFZ7m6EgDH1BrZmLU5GU0jjws8WqWoh01qmeWqfB1fwD3mJ5t7h094/kKFQotQ2SHLRroyE0BCn3fqrLUJJp+VgBLHb0ahVBwzpWlyIg4twPHvLNYDDxw2qU+fepcQLWD1PMNZJsa//asu/y7+07q+771Iql9wzcBT30A0nzS4yJ07/TuqFsG0iETNPP2E300ALuu7JhrIpJrzQ6Rw== trobinson@Tonys-MacBook-Pro.local
1 /home/ayy/.ssh/authorized_keys
```

Key Copy Method 2: using vi

The `vi` text editor that is older than dirt, hard to use, hard to understand, and has arcane invocations that nobody understands. And now you get to struggle with it as your elders did before you. It’s a rite of passage in the Unix/Linux world.

Similar to what we did with method one, you will either need to open the id_rsa.pub file in a text editor of your choosing, or use the cat command on the command line to dump it to the terminal. High all of the text the file contains, and copy it to your clipboard via ctrl + c or meta + c.

```
Tonys-MBP:~ trobinson$ cat ~/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1n32EAAAQABAA91QDNu5GaVH6nP0TGdKFmmzBJdA/NYQ/coDaAbcRBXlWR
twFTAjExUKv0A/5D4ftoXBIsSjSSvxJxyVAZas0QI1mJU+8lMroaZ/fAI48d7kUM0YhzcMxIbvjap0Vy
vMkizTXsxq0Uus4w840aI9rFLCkj03wPZbYtA5U13qHjBAqWbz0G7DckvlClky0kQZLUBS1TI59hjTo0
xN3veIykp9KYT1JF8FwRZp4V9pWbT/91CrcLGWmVZzxc4+V5iBnAW+IQeVA53ESGTYJ1PyI4TVoJR7iV
UTamZ2GPpeP43Ge81CkkxA5UPVcnn0XPAkSl0zZP50W21xWJHjg1U3y8cdne9rpLV0dtZ95mogw3e0v5
zKyEfRq9KnCOZrfiINBWWn1t+0lYbeR7PZ1dTuh0tZM1eC8se9yq6TGEc90Xku8mVAXfPpaB/rt9Nf1H
t6X1zLns5jx2BuF76/jjzESRwzj2XyL8uzFZ7m6EgDH1BrZmLU5GU0jjws8WqWoh01qmeWqfB1fwD3mJ
5t7h094/kKFQotQ2SHLRroyE0BCn3fqrLUJJp+VgBlHb0ahVBwzpWyIg4twPHvLNYDDxw2qU+fepcQ1W
D1PMNZJsa//asu/y7+07q+771Iql9wzcBT30A0nzS4yJ07/TuqFsG0iETNPP2E300ALuu7JhrIpJrzQ6
Rw== trobinson@Tonys-MacBook-Pro.local
```

Back in our SSH session to the ips VM, enter the following command:

```
vi ~/.ssh/authorized_keys
```

```
ayy@ips:~/.ssh$ vi ~/.ssh/authorized_keys |
```

This will open the editor in review mode. There should be nothing in the file. No text, no other keys; nothing.

A screenshot of a terminal window showing a blank file in Vim. The screen is mostly black with a vertical green bar on the left. The status bar at the bottom shows the text "0,0-1 All".

```
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
--No lines in buffer--
```

Hit the 'i' key to enter 'insert' mode. Hit **ctrl +v** (meta + v in OSX) just once, to paste the contents of the clipboard in insert mode. After the contents have been copied into the file, hit the 'esc' key then type in

```
:wq!
```

to save the file and exit.

Next, run the following command:

```
cat ~/.ssh/authorized_keys; wc -l
```

```
ayy@ips:~$ cat ~/.ssh/authorized_keys; wc -l ~/.ssh/authorized_keys
ssh-rsa AAAAB3NzaC1n32EAAAQABAAA91QDNu5GaVH6nP0TGdKFmmzBJdA/NYQ/coDaAbcRBXlWR
twFTAjExUkV0A/5D4ftoXBIsSjSSvxJxyVAZas0QIlmJu+8LMroaZ/fAI48d7kUM0YhzcMxIbvjap0Vy
vMkizTXsxq0Uus4w840aI9rFLCkj03wPZbYtA5U13qHjBAqWbz0G7DckvLC1ky0kQZLUBS1TI59hjTo0
xN3veIykp9KYT1JF8FwRZp4V9pWbT/91CrcLGWmVZzxc4+V5iBnAW+IQuVA53ESGTYJlPyI4TVoJR7iV
UTamZ2GPpeP43Ge81CkkxA5UPVcnn0XPAkS10zZP50W2lxWJHjg1U3y8cdne9rpLV0dtZ95mogw3e0v5
zKyEfRq9KnCOZrfiINBWWh1t+0lYbeR7PZ1dTuh0tZM1eC8se9yq6TGEc90Xku8mVAXfPpaB/rt9Nf1H
t6X1zLns5jx2BuF76/jjzESRwzj2XyL8uzFZ7m6EgDHLBrZmLU5GU0jjws8WqWoh01qmeWqfB1fwD3mJ
5t7h094/kKFQotQ2SHLRroyE0BCn3fqrLUJJp+VgBlHb0ahVBwzpWyIg4twPhvLNYDDxw2qU+fepcQLW
D1PMNZJsa//asu/y7+07q+771Iql9wzcBT30A0nzS4yJ07/TuqFsG0iETNPP2E300ALuu7JhrIpJrzQ6
Rw== trobinson@Tonys-MacBook-Pro.local
1 /home/ayy/.ssh/authorized_keys
```

Just like with method 1 above, these commands are to ensure that the key got copied correctly and there is only a single line in the authorized_keys file.

Key Copy Method 3: SCP

Method 3 involves using the scp application to upload a copy of the id_rsa.pub file to the ips VM. From there, we can take the contents of the id_rsa.pub file and simply append them to the existing authorized_keys file. On your Linux, Unix, or OSX host, run the following command:

```
scp ~/.ssh/id_rsa.pub ayy@172.16.1.4:~/id_rsa.pub
```

```
Tony's-MBP:~ trobinson$ scp ~/.ssh/id_rsa.pub ayy@172.16.1.4:~/id_rsa.pub  
ayy@172.16.1.4's password:  
id_rsa.pub                                              100%   759      0.7KB/s   00:00
```

Replace “ayy” in the command above with the name of the user you set up on the ips VM. This command tells scp to copy the id_rsa.pub from our local system, and put it in the user ayy’s home directory on the ips VM.

Next, if you don’t already have an SSH session on the ips VM, SSH to the ips VM as the user you set up on the VM. Run the command

```
ls -al ~/
```

To verify that there is a copy of the id_rsa.pub file in the user’s home directory.

```
ayy@ips:~$ ls -al ~/  
total 44  
drwxr-xr-x 4 ayy  ayy  4096 Dec  3 18:10 .  
drwxr-xr-x 3 root root 4096 Dec  1 16:10 ..  
-rw----- 1 ayy  ayy  2510 Dec  3 18:09 .bash_history  
-rw-r--r-- 1 ayy  ayy   220 Dec  1 16:10 .bash_logout  
-rw-r--r-- 1 ayy  ayy  3771 Dec  1 16:10 .bashrc  
drwx----- 2 ayy  ayy  4096 Dec  1 16:10 .cache  
-rw-r--r-- 1 ayy  ayy   759 Dec  3 18:10 id_rsa.pub  
-rw-r--r-- 1 ayy  ayy   655 Dec  1 16:10 .profile  
drwx----- 2 ayy  ayy  4096 Dec  3 18:09 .ssh
```

From here, run the command

```
cat ~/id_rsa.pub >> ~/.ssh/authorized_keys
```

```
ayy@ips:~$ cat ~/id_rsa.pub >> ~/.ssh/authorized_keys
```

Next, run the following command:

```
cat ~/.ssh/authorized_keys; wc -l
```

```
ayy@ips:~$ cat ~/.ssh/authorized_keys; wc -l ~/.ssh/authorized_keys
ssh-rsa AAAAB3NzaC1n32EAAAQABAAA91QDNu5GaVH6nP0TGdKFmmzBJdA/NYQ/coDaAbcRBX1WR
twFTAjExUkV0A/5D4ftoXBIsSjSSvxJxyVAZas0QI1mJU+81MroaZ/fAI48d7kUM0YhzcMxBvjaP0Vy
vMkizTXsxq0Uus4w840aI9rFLCkj03wPZbYtA5U13qHjBAqWbz0G7DckvlClky0kQZLUBS1TI59hjTo0
xN3veIykp9KYT1JF8FwRZp4V9pWbT/91CrcLGwmVZzxc4+v5iBnAW+IqeVA53ESGTYJ1PyI4TVoJR7iV
UTamZ2GPpeP43Ge81CkkxA5UPVcnn0XPAkS10zZP50W21xWJHjg1U3y8cdne9rpLV0dtZ95mogw3e0v5
zKyEfRq9KnC0ZrfiINBWWn1t+01YbeR7PZ1dTu0tZM1eC8se9yq6TGEc90Xku8mVAXfPpaB/rt9Nf1H
t6X1zLns5jx2BUf76/jjzESRwzj2XyL8uzFZ7m6EgDH1BrZmLU5GU0jjws8WqWoh01qmeWqfB1fwD3mJ
5t7h094/kKFQotQ2SHLRroyE0BCn3fqrlUJJp+VgBlHb0ahBVwzpWyIg4twPHvLNYDDxw2qU+fepcQLW
D1PMNZJsa//asu/y7+07q+771Iql9wzcBT30A0nzS4yJ07/TuqFsG0iETNPP2E300ALuu7JhrIpJrzQ6
Rw== trobinson@Tonys-MacBook-Pro.local
1 /home/ayy/.ssh/authorized_keys
```

Just like with the other methods, this verifies that the key copied correctly and is intact. If the key is not on a single line, or mal-formed, you will need to correct the errors. After you have verified that the key has been added to the authorized_keys file correctly, run the command

```
rm -rf ~/id_rsa.pub
```

```
ayy@ips:~$ rm -rf ~/id_rsa.pub
```

On the ips VM to remove the uploaded public key, or if you'd like, you can keep the key on the system until after you have tested key-based authentication to verify everything is working as intended.

Making Sure it worked

So you finally have your public key copied to the ips VM. Now, we have to ensure that our key copied correctly, disconnect from any SSH sessions to the ips VM, and simply try to reconnect to the ips VM. At this point, you should just be able to type

```
ips
```

And your host system should have an alias set up to open an SSH session to your ips VM. If you didn't set up aliases, or don't want to bother right now, simply run

```
ssh ayy@172.16.1.4
```

If everything was copied correctly and key-based auth was configured correctly, you should have a session on the ips VM without ever having to enter a password (or, if you configured a password for your SSH key, the ssh session WILL ask for the id_rsa password when connecting - this is the password you entered for your SSH key.)

Note for OSX users: If you set up a key for your id_rsa, and have ever entered it previously, OSX will add this password to your system's keyring. Without going in to too many details, the OSX keyring is an OS-integrated credential manager.

If you want to see whether or not OSX is taking your id_rsa (SSH key) password from the keyring, or if you are trying to troubleshoot issues with your key-based authentication failing, try this out:

Run the command ayy@172.16.1.4 -vv (or you can use your ips alias and simply run "ips -vv")

When you add the option "-vv", this runs the ssh command in very verbose mode. This shows you a lot of what is going on in the background when you run SSH. It can generate a lot of output. Let's take a look at what a successful key authentication looks like take a look at this snippet:

```
debug2: service_accept: ssh-userauth
debug1: SSH2_MSG_SERVICE_ACCEPT received
debug1: Authentications that can continue: publickey,password
debug1: Next authentication method: publickey
debug1: Offering RSA public key: /Users/trobinson/.ssh/id_rsa
debug2: we sent a publickey packet, wait for reply
debug1: Server accepts key: pkalg rsa-sha2-512 blen 535
debug2: input_userauth_pk_ok: fp SHA256:mucXnvrZy1a7JRH4fxSWRhY4we6PPUZZ6LbQANu4w
ao
debug2: using passphrase from keychain
debug1: Authentication succeeded (publickey).
Authenticated to 172.16.1.4 ([172.16.1.4]:22).
```

You can see in the output above, that we send our SSH key to the system. And that the server at 172.16.1.4 accepts the key. Immediately after that you can see the line "using passphrase from keychain". This is the OSX keychain that I referenced before; the system saved my password for my SSH key and applied it automatically.

If you get anything other than "Authentication succeeded (publickey)." in your verbose SSH output, then that means key-based auth for SSH failed. If you get a password prompt for the user "ayy" (or the user you configured for the ips VM), then key-based authentication failed, and you will need to do some troubleshooting. Here are some things to consider in order to help you troubleshoot:

Did you copy ALL the contents of the id_rsa.pub file?
Did you copy the contents of the id_rsa.pub file to ~/.ssh/authorized_keys?
Did you ensure that file permissions for the ~/.ssh directory were 700, and authorized_keys was set to 600 via the “chmod” command? (Note that many versions of Linux will REFUSE to use key-based authentication if the .ssh directory and files therein are NOT secured properly)
When you copied id_rsa.pub to the authorized_keys file, were the entire contents of id_rsa.pub placed on a single line?
When you configured your alias, or ran your ssh command to ips (172.16.1.4) did you specify the correct username in your ssh command (e.g. ssh ayy@172.16.1.4, if your user's name is “ayy”)?

Using key-based authentication with the SCP command

If you configured key-based authentication for SSH as a certain user, and you attempt to SCP to or from a system in which you enabled key-based authentication for, then the SCP command should automatically be able to utilize the SSH key. To demonstrate, I created the file foo.txt on the ips VM, that simply contains the text, “This is a test.”

```
ayy@ips:~$ file foo.txt
foo.txt: ASCII text
ayy@ips:~$ cat foo.txt
This is a test.
```

Next, I ran the command scp ayy@172.16.1.4:~/foo.txt ~/foo.txt

The command copied the file effortlessly. The ssh key also allows you to upload files from your local system to the ipm VM just as effortlessly. On my OSX system, I create the file bar.txt, that contains the text “This is also a test.”

```
Tony-MBP:~ trobinson$ scp ayy@172.16.1.4:~/foo.txt ~/foo.txt
foo.txt                                         100%   16      0.0KB/s  00:00
Tony-MBP:~ trobinson$ cat foo.txt
This is a test.
```

Next, I ran the command scp ~/bar.txt ayy@172.16.1.4:~/bar.txt

```
Tony-MBP:~ trobinson$ scp ~/bar.txt ayy@172.16.1.4:~/bar.txt
bar.txt                                         100%   21      0.0KB/s  00:00
```

Once again, the file copies effortlessly, if the SSH key has been configured properly on the target host, with the target user.

```
Tony's-MBP:~ trobinson$ ips
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-51-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.

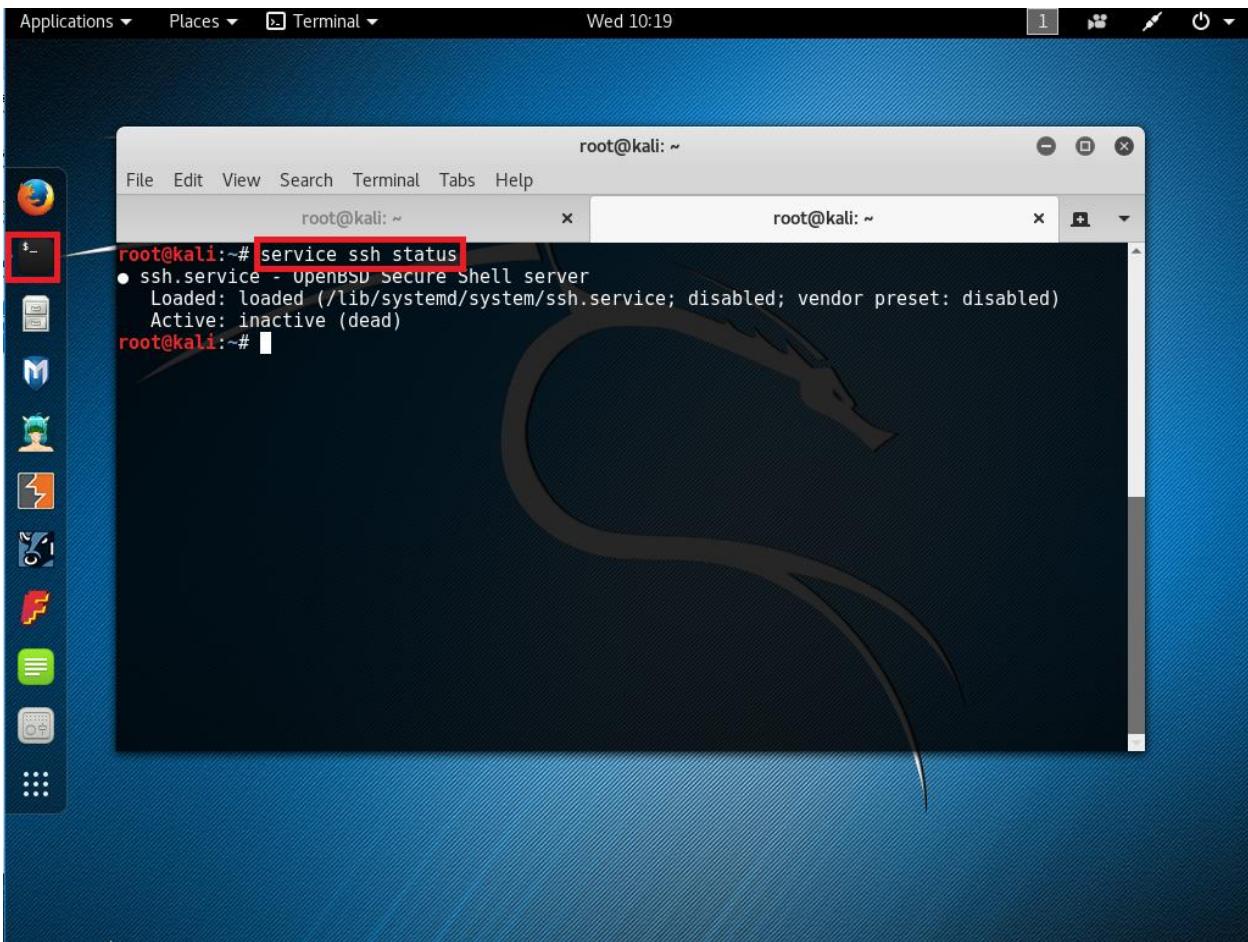
Last login: Sat Dec  3 19:22:52 2016 from 172.16.1.2
ayy@ips:~$ cat bar.txt
This is also a test.
ayy@ips:~$
```

How to Enable SSH on Kali Linux

For one reason or another, Kali Linux installs with SSH disabled by default. It is believed this is by design in order to reduce exposure and possible attack surface for a system with a vast array of offensive security tools. Not to mention the fact, the only and default user for Kali is the “root” user; it is generally frowned upon to SSH in as the root user.

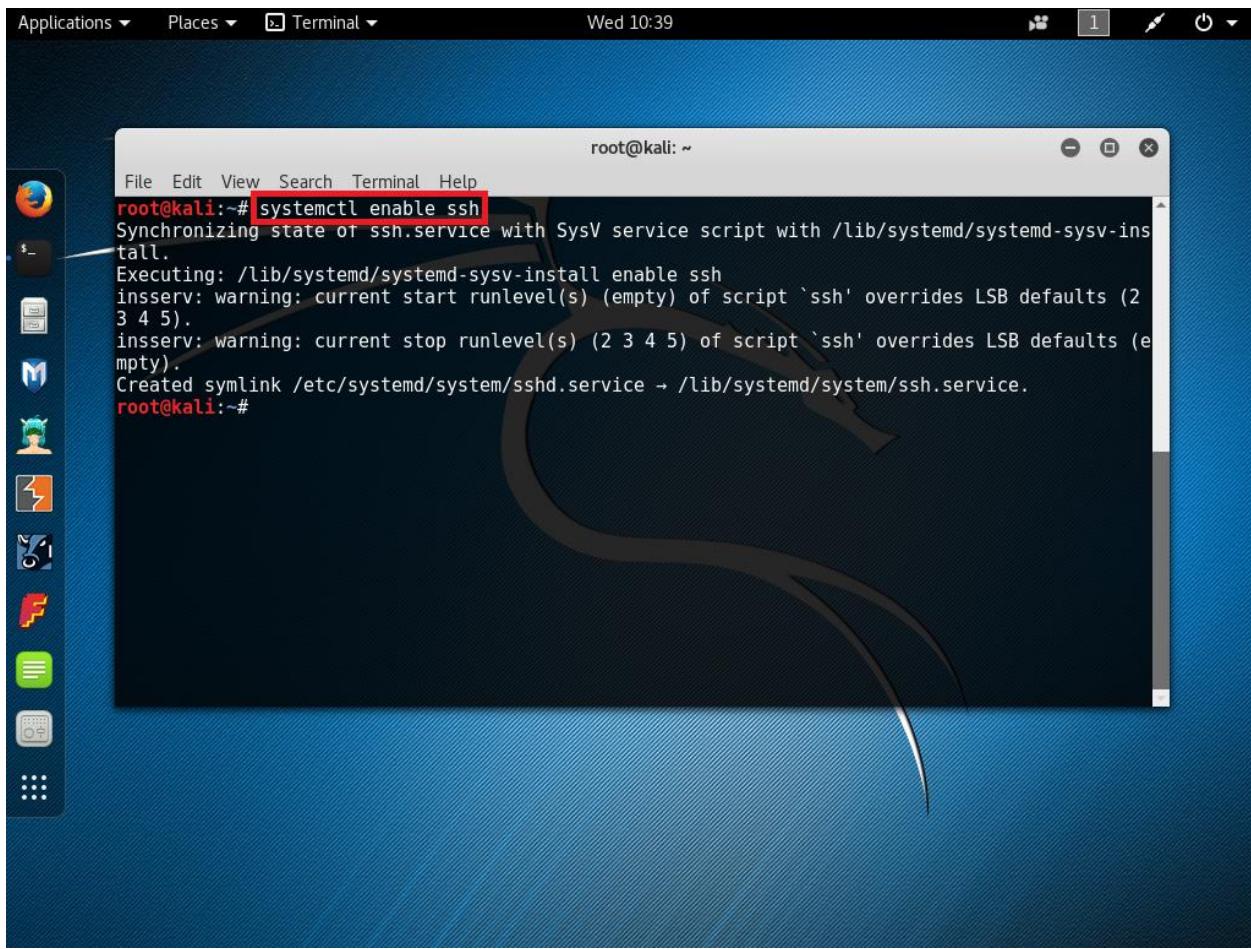
Power on your kali VM, and log in. Open up the terminal application and enter:

```
service ssh status
```



You'll notice that SSH is disabled in the illustration above. Run the following command:

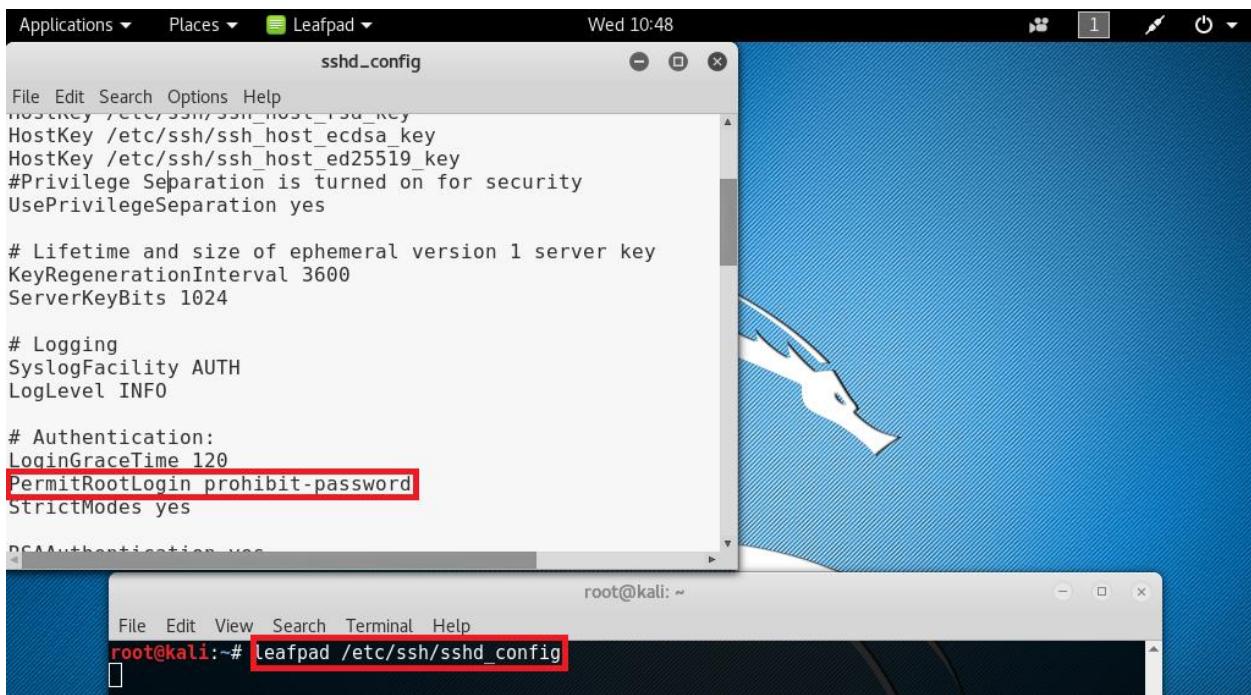
```
systemctl enable ssh
```



This command tells systemd, the system services manager that we want ssh to start on boot. Next, we need to make a temporary change to the file /etc/ssh/sshd_config. Before we do that, back up sshd_config by running the following command:

```
cp /etc/ssh/sshd_config ~/sshd_config_bak
```

Next, we need to edit the sshd_config file. Open /etc/ssh/sshd_config with your favorite editor (Kali Linux includes "Leafpad", a graphical editor if you don't want to struggle with vi). Find the line "PermitRootLogin prohibit-password"



Edit the text from “prohibit-password” to “yes”. This will allow the root user to be able to authenticate over SSH with just a password. This is meant to be TEMPORARY. Save the file and exit the editor.

```
sshd_config
File Edit Search Options Help
HostKey /etc/ssh/ssh_host_ecdsa_key
HostKey /etc/ssh/ssh_host_ed25519_key
#Privilege Separation is turned on for security
UsePrivilegeSeparation yes

# Lifetime and size of ephemeral version 1 server key
KeyRegenerationInterval 3600
ServerKeyBits 1024

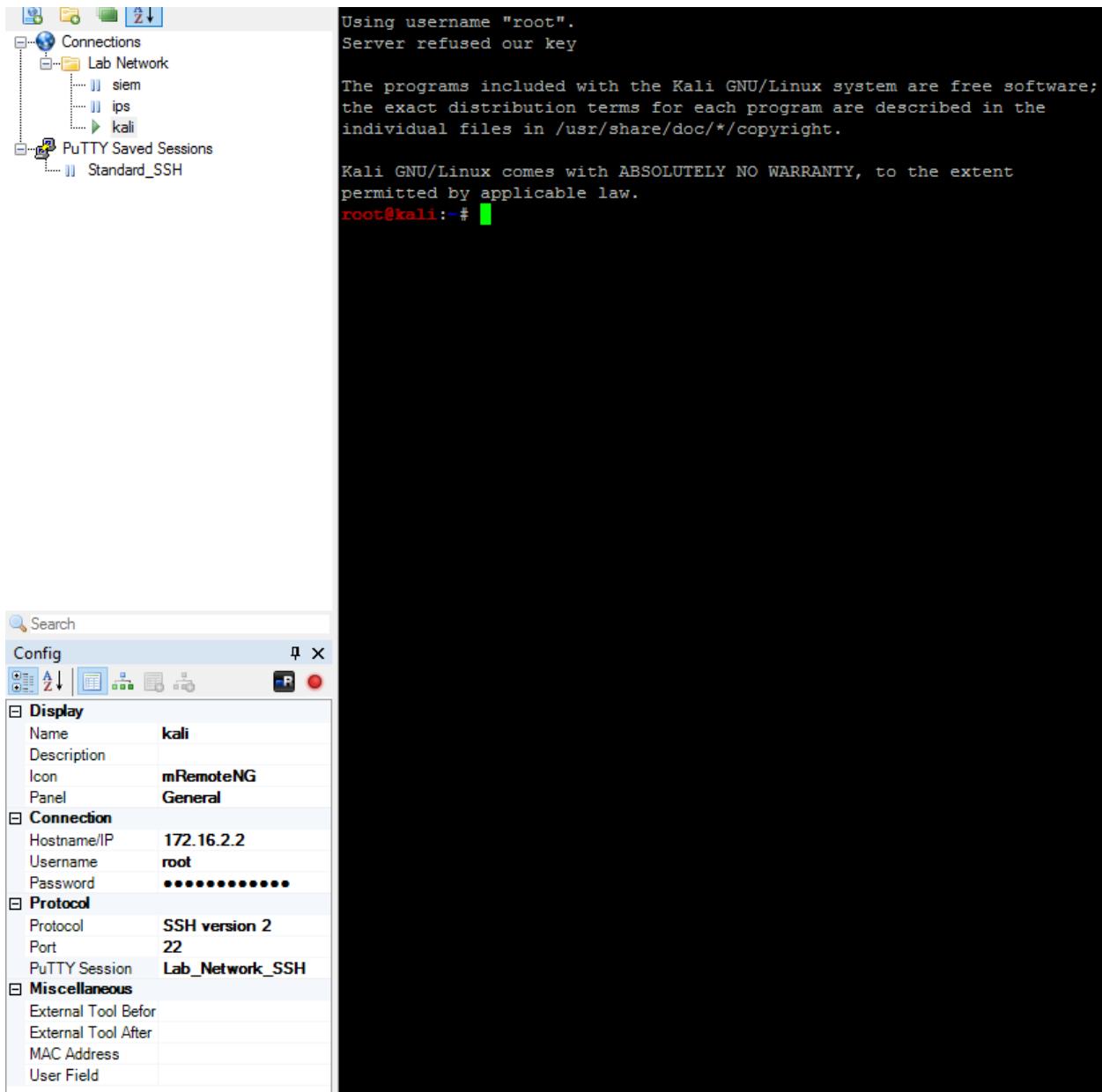
# Logging
SyslogFacility AUTH
LogLevel INFO

# Authentication:
LoginGraceTime 120
PermitRootLogin yes
StrictModes yes

RSAAuthentication yes
```

Now, we need to reboot the VM. You can use your hypervisor's power controls to reset the VM, or you can restart the VM from inside the OS by whatever means you're most comfortable (e.g. init 6, reboot, the shutdown button on the desktop, etc.)

While the system is rebooting, If you're on Windows, open MRemoteNG and create a new SSH version 2 connection to 172.16.2.2 with the PuTTY profile we made earlier to enable key-based authentication.



If you're using Linux or OSX, simple open a new terminal window or tab, and run

```
ssh root@172.16.2.2
```

Then enter the root user's password to get a session on the kali VM.

So now, that you have an SSH session on the kali linux host (If you don't, check to make sure you have port 22 open from 172.16.1.1 to 172.16.2.2 on the LAN network firewall policy on the pfSense firewall, and that your system is able to route to 172.16.2.0 -- check your routing tables). Depending on whether your hypervisor host, or workstation of choice is running Windows, Linux, Unix, or OSX, You'll want to follow the directors for [enabling key-based](#)

[authentication for your Windows hypervisor host/workstation](#), or [enabling key-based authentication for your Linux, Unix, or OSX hypervisor host/workstation](#). Copy your SSH public key to the `~/.ssh/authorized_keys` on the kali VM using any method you see fit. Afterwards, verify that key-based authentication is working for the kali VM. In mremoteng, this is as simple as disconnecting the SSH session, and removing the root user's password from the MRRemoteNG, then attempting to connect. In the case of Linux, Unix, or OSX systems, disconnect your current SSH session, then reconnect using the command

```
ssh root@172.16.2.2
```

If you get a session without having to enter root's password, everything is working perfectly. Now, we need to revert the `sshd_config` file:

```
mv /root/sshd_config_bak /etc/sshd_config
```

At this point, you should be able to SSH in as root, and password authentication for the root account should be prohibited.

Enabling, and securing root SSH

Depending on who you talk to, people believe that you should never log in to the root account on a system, and that root account access over SSH should never ever happen. This is why by default, the root account is disabled on the Ubuntu Linux distribution, and most other linux distros require you to create a user level account as part of the installation. On real-world systems, the root user account should be used sparingly, and SSH as root should never be enabled unless absolutely necessary.

However, this is a lab environment, *OUR* lab environment, you're the only user here, the lab network has a VERY strict access control to limit our risks, and you take snapshots of your VMs. I argue that for a personal lab environment SSH as root with proper security in place is perfectly fine. **Just be careful, and know the risks.** This section is entirely optional and is only for the extremely lazy.

I'm going to show you how to enable SSH as root on the siem VM (running Ubuntu), then show you how to disable password authentication for SSH entirely; for both root and non-root users. From there, you can decide if you want to apply these settings to your other Linux VMs. Disabling password authentication for SSH totally removes any possibility of brute force attacks; the SSH server will drop the connection if an attacker is trying to use a password for authentication.

Note: If you completed the section "[How to Enable SSH on Kali Linux](#)", then you're already using SSH as the root user, and you should have already configured kali to use your SSH key

for authentication. If you plan on keeping up with that setup, I very highly recommend reading on for disabling password authentication over SSH)

Adding your SSH public key to root's authorized_keys file

Using either MRemoteNG on Windows, or the ssh command on Linux, Unix, or OSX systems, SSH to the siem VM as the user you created during initial setup (in my case, I used the "ayy" user) , and run the following command to run gain access to a root shell:

```
sudo su -
```

Sudo is a special command on Unix/Linux systems. If your user exists in the /etc/sudoers file, you are allow to run certain commands with root privileges (so long as you have the current user's password). Since the first user you create in Ubuntu is in /etc/sudoers and there are no restrictions on the commands the user can run via sudo, we can then immediately tell sudo to run "su -" as root. The su command allows you to log in from your current shell session as root. Normally, you need the root password to do this. Since we just ran sudo before su, the system sees us as already being root and just spawns a root shell for us. This allows us to login as root, even on Ubuntu, where the account is restricted.

```
Using username "ayy".
Authenticating with public key "windows-management"
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-38-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.

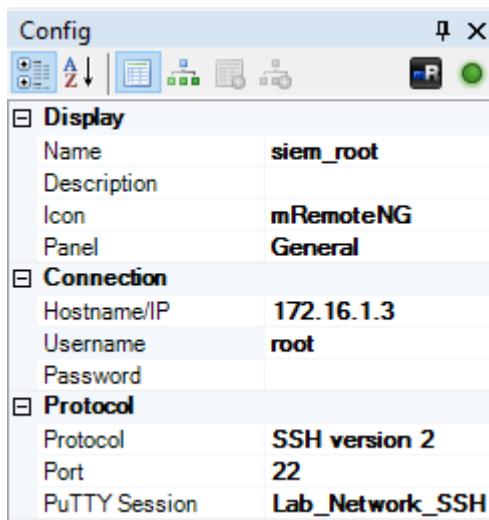
Last login: Wed Sep 21 22:28:03 2016 from 172.16.1.2
ayy@siem:~$ sudo su -
[sudo] password for ayy:
root@siem:~# id
uid=0(root) gid=0(root) groups=0(root)
root@siem:~# █
```

We now have to add our SSH public key to /root/.ssh/authorized_keys. Depending on whether your hypervisor host, or workstation of choice is running Windows, Linux, Unix, or OSX, You'll want to follow the directors for [enabling key-based authentication for your Windows hypervisor host/workstation](#), or [enabling key-based authentication for your Linux, Unix, or OSX hypervisor host/workstation](#). This step usually boils down to making the .ssh directory in root's home directory, and adding our SSH public key to /root/.ssh/authorized_keys, while ensuring that file

permissions for the .ssh directory and authorized_keys file are set to where only the root user can access them.

```
root@siem:~# mkdir /root/.ssh;chmod 700 /root/.ssh;touch /root/.ssh/authorized_keys;
chmod 600 /root/.ssh/authorized_keys; ls -al /root/.ssh
total 8
drwx----- 2 root root 4096 Sep 21 22:37 .
drwx----- 3 root root 4096 Sep 21 22:37 ..
-rw----- 1 root root     0 Sep 21 22:37 authorized_keys
root@siem:~#
```

Now, add your SSH public key to authorized_keys, through the means you are most comfortable with. After you add your key, if you are on Windows open MRemoteNG, and add a new connection. Use the following settings:



If everything went well, you should be logged in as root.

For Linux, Unix, or OSX systems, run the command

```
ssh root@172.16.1.3
```

To verify that your SSH key is functioning properly. **Regardless of whether or not you are using Windows or Linux/Unix/OSX hosts, you need to make sure that your SSH keys are working properly for the next part of this exercise.**

```
Tony's-MBP:~ trobinson$ ssh root@172.16.1.3
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-31-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.

*** System restart required ***

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

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

root@siem:~#
```

Note: If you are using Linux, Unix or OSX, you may want to review the section on using [the alias command](#) and and/or modify existing aliases in your .bashrc, or .bash_profile files to simply SSH into the siem VM as root instead. You can edit .bashrc or .bash_profile with your favorite text editor, or you can simply create and append a new, unique alias to denote this alias logs you into the siem VM as root.

```
Tony's-MBP:~ trobinson$ echo "alias rootsiem='ssh root@172.16.1.3'" >> ~/.bash_profile
Tony's-MBP:~ trobinson$ source ~/.bash_profile; alias
alias ips='ssh ayy@172.16.1.4'
alias rootsiem='ssh root@172.16.1.3' [red box]
alias siem='ssh ayy@172.16.1.3'
```

```
Tonys-MBP:~ trobinson$ root@siem  
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-31-generic x86_64)  
  
 * Documentation: https://help.ubuntu.com  
 * Management: https://landscape.canonical.com  
 * Support: https://ubuntu.com/advantage  
  
0 packages can be updated.  
0 updates are security updates.  
  
*** System restart required ***  
Last login: Sat Dec  3 20:30:43 2016 from 172.16.1.2  
root@siem:~#
```

Disabling password authentication entirely via sshd_config

In spite of Ubuntu disabling local login as root, sshd allows users to log in as root so long as they have their public SSH key in root's authorized_keys file. This is because of the "PermitRootLogin without-password" directive in the /etc/sshd_config. (You may notice in some other distributions such as Kali Linux, that the "PermitRootLogin" setting is set to "prohibit-password". This operates the same as "without-password" for all intents and purposes). If for some reason PermitRootLogin is set to anything other than "without-password", or "prohibit-password", modify the PermitRootLogin directive to either of those settings.

```
root@siem:~# cat /etc/ssh/sshd_config | grep PermitRootLogin  
PermitRootLogin without-password
```

Additionally, we need to edit the "PasswordAuthentication" directive in /etc/ssh/sshd_config. Change these lines from:

```
# Change to no to disable tunneled clear text passwords  
#PasswordAuthentication yes
```

To:

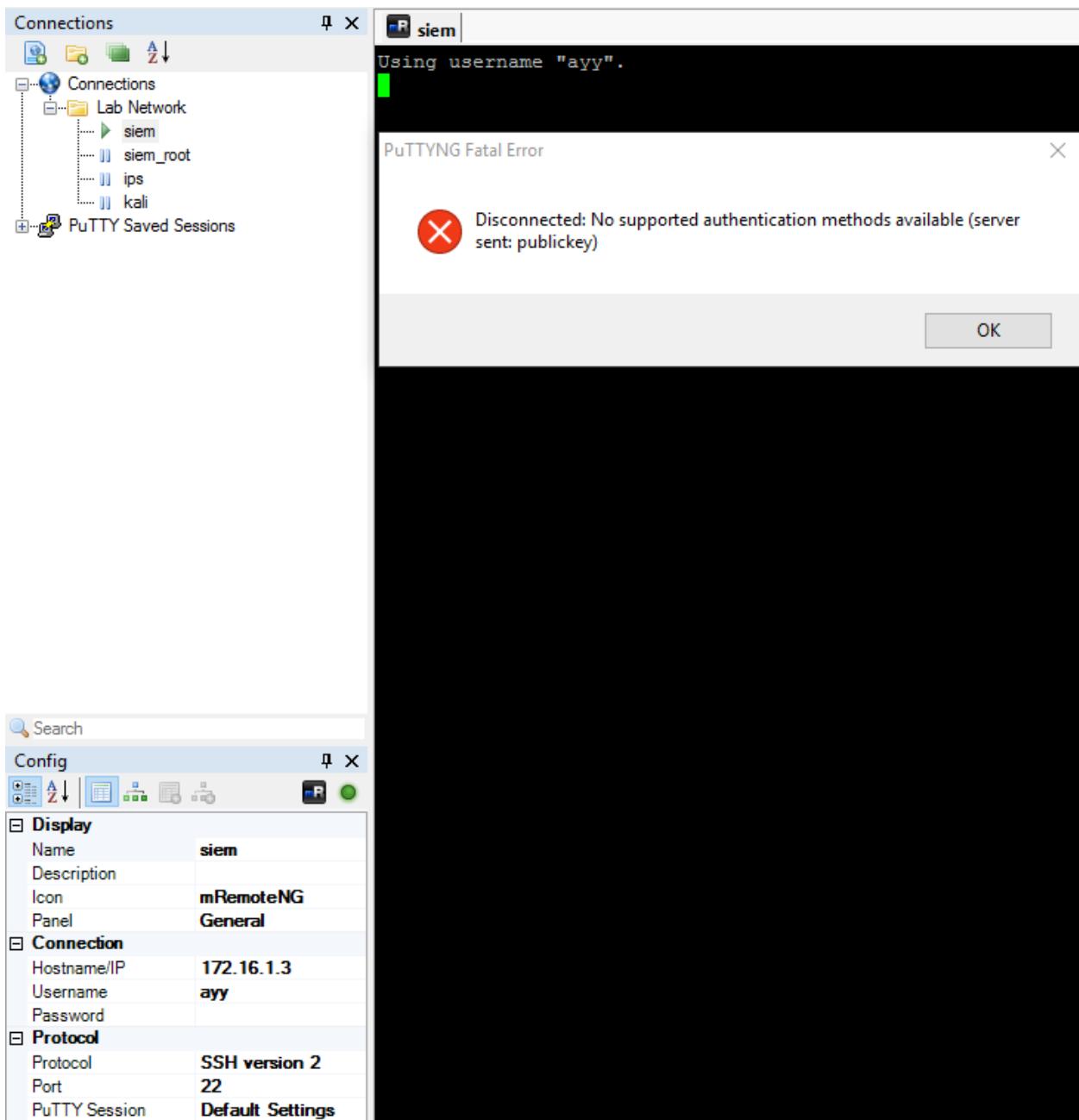
```
# Change to no to disable tunnelled clear text passwords
PasswordAuthentication no
```

The Octothorpe (or “hash mark”) at the beginning of a line is used in many configuration files to denote that the line is a comment that doesn’t need to be parsed. In this case, we’re removing comment line, and changing PasswordAuthentication from “yes” to “no”. **Remember this in the event you ever have a need to revert these changes.**

Save the /etc/ssh/sshd_config file, then restart ssh through the command:

```
service ssh restart
```

This setting disabled the use of passwords entirely to authenticate SSH sessions. This means that if you enable this option, the only means you have to connect to the VM over SSH is if you are using your SSH key. **This also affects other user accounts on the system as well:**



In the illustration above, I attempted to log in to the user 'ayy' using the default PuTTY session (not using key authentication), and the connection is dropped immediately. This provides an extra layer of security, but also means you must keep control of your SSH private and public keys, and ensure they remain secure. If you lose the keys, the only access you will have to the VM(s) you implement this on is through your hypervisor's console connection to the VM.

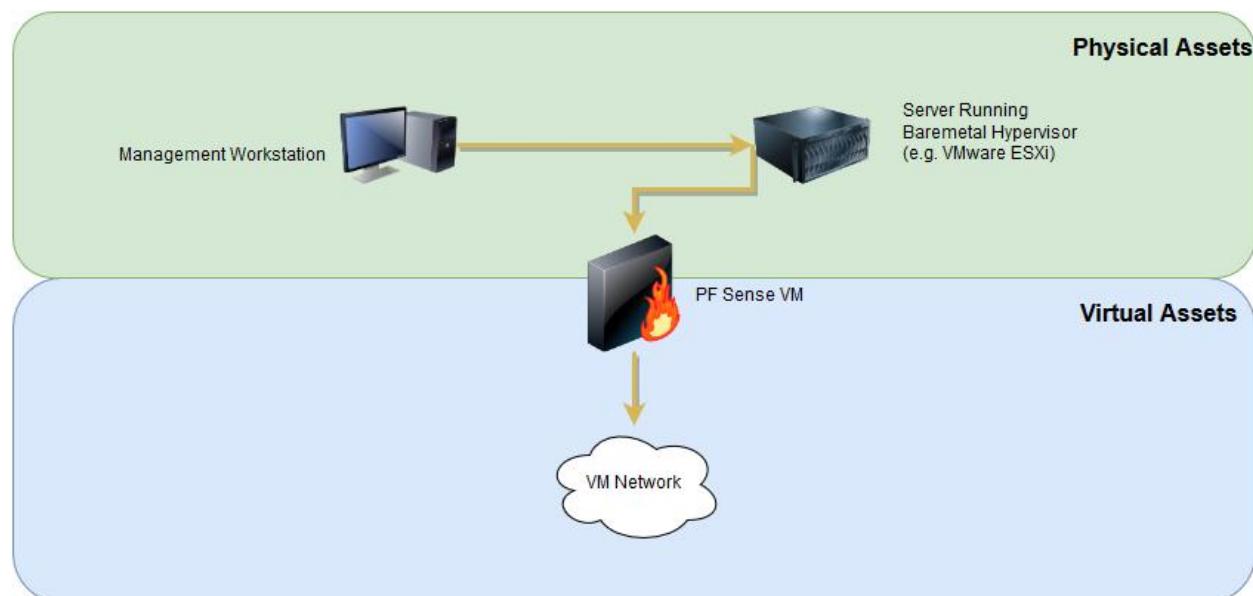
Note: If you enabled key-based authentication for the root user, that means that you can also use key-based authentication for the Linux/Unix "scp" command for file transfers, or you can create/modify your sites in WinSCP on Windows to authenticate as the root user.

I'm going to give you the same general warning that I gave you on the section with enabling SSH via the root account: It's a nice-to-have feature, fast, convenient, and all of that, but best practice in the real-world is to transfer files via a non-root user, if possible and elevate your privileges to the root user only as necessary. That being said, this is our lab, we've taken fairly strong precautions against shooting ourselves in the foot, we're not running anything mission-critical, and we're not storing any sensitive data. So, if you want to SCP stuff as root, have fun, know the risks involved, and **be careful**.

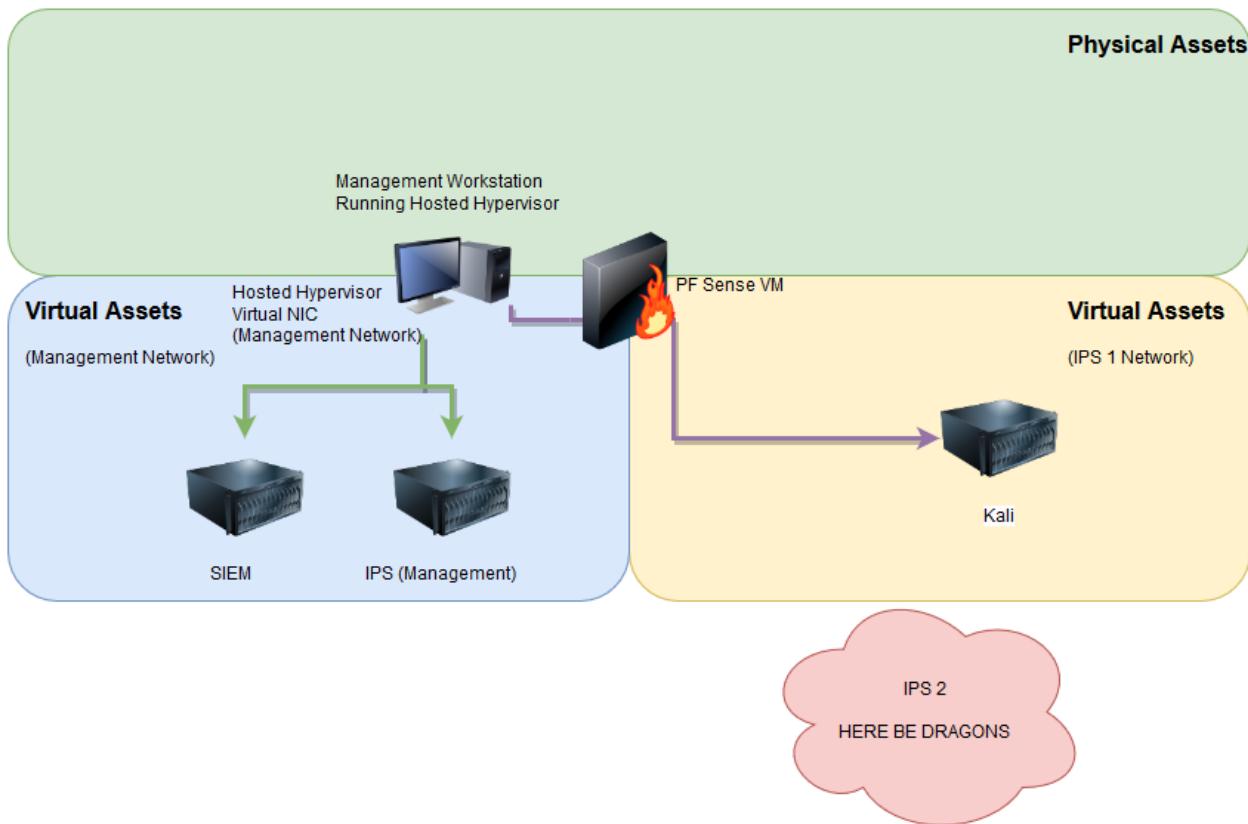
Network Design Factors When Working with Baremetal Hypervisors

While this book has provided a great deal instruction on how to create virtual machines and set up virtual network infrastructure, almost all of the hypervisors described in this document are hosted hypervisors, with the exception of VMware ESXi. While creating the virtual networks and VMs are similar enough to hosted hypervisor counterparts, managing and maintaining remote access to VMs hosted on a baremetal hypervisor ends up being a little bit different, and requires a little more network setup than working with VMs hosted locally on a hosted hypervisor.

Consider this illustration:



This illustration reflects a lab environment with a management workstation, and a system running a baremetal hypervisor, hosting the lab network, and all the VMs. As you can see by the orange line, your management workstation, a separate physical system, needs network access enabled in order to use the VMs hosted on the server running the baremetal hypervisor. Pretty obvious stuff, right? Let's compare this how networking is generally configured on our hosted hypervisors. Please review this diagram:



In the diagram above, our management workstation (a physical asset) is also running the hosted hypervisor, allowing us to create a virtual network card (a virtual asset), directly connected to the management virtual network. We can access systems on the management virtual network directly over what is considered a local network with no need to create any static routes or firewall rules.

In the case of a baremetal hypervisor, your management workstation does not have a virtual network interface directly connected to the management network; our management workstation is considered a remote system. This means that if you want to remotely manage your lab VMs behind the pfSense Firewall using SSH, RDP, or other remote access protocols. You will need to ensure that you have static routes in place for any networks behind the pfSense firewall (e.g. 172.16.1.0/24 for the management network, and 172.16.2.0/24 for the IPS network), as well as firewall rules configured on the WAN interface of the pfSense firewall to allow your external management workstation to access the VM on correct TCP port(s).

I'm going to walk you through how to setup static routes, then how to create necessary firewall rules on the pfSense firewall to enable remote access.

Prereqs

This guide assumes you are running ESXi 6.x or higher with the latest updates, ESXi is configured as instructed in the VMWare VSphere Hypervisor setup guide (meaning all the virtual networks, and VMs have already been created, and you have access to the web interface of your pfSense VM), and that the management and IPS virtual networks are set to 172.16.1.0/24 and 172.16.2.0/24 respectively.

This guide also assumes that the management interface of the ESXi server, The WAN interface of the pfSense Firewall VM, and the workstation you'll be using to manage your baremetal hypervisor and VMs all either have static IP addresses or static DHCP leases so where the IP addresses of these systems will NOT change. **At an absolute minimum, you need to be able to set up a static IP address or static DHCP allocation for your ESXi server's management interface, the WAN interface of your pfSense VM, and an additional "jump box" VM or physical system located on the same network that your management system can route its requests through.** Visit the section on Dealing with DHCP to learn how to create jump boxes and route your management system's traffic through them to interact with your VMs. For purposes of sheer laziness, I'll be using Windows 10 as the OS for my management workstation, but will make sure those of you running Linux or OSX workstations aren't left out.

Creating static routes

On my home network, I used the 192.168.1.0/24 network for my physical machines. The pfSense VM was given a static DHCP allocation and is located at 192.168.1.22. With this information, we can create static routes to the 172.16.1.0/24 and 172.16.2.0/24 networks as necessary. If you're using windows 7 or greater, open an elevated command prompt and enter the following commands:

```
route -p add 172.16.1.0 mask 255.255.255.0 192.168.1.22  
route -p add 172.16.2.0 mask 255.255.255.0 192.168.1.22
```

Replace the ip address 192.168.1.22 with the WAN IP address of the pfSense firewall VM as necessary. You can then run:

```
route print
```

Persistent Routes:				
Network Address	Netmask	Gateway Address	Metric	
172.16.1.0	255.255.255.0	192.168.1.22	1	
172.16.2.0	255.255.255.0	192.168.1.22	1	

If you have entries similar to the illustration above, you have successfully added static routes correctly.

If you're running OSX or a Linux distro as your management workstation, check out the Remote Access guide. Specifically if you're using Linux, pay attention to the sections detailing [the ip command](#), while OSX and BSD users should read up on the [route command](#). After you have your routes configured, you'll need a method to ensure that they persist on reboot. Linux and BSD users should check out the section on using [/etc/rc.local](#). OSX users can read up on the section [OSX Route Persistence for Baremetal Hypervisors](#).

Creating Firewall Rules

You'll need to log in to the web interface of your pfSense router. You'll need to log in from the WAN interface. If you followed the ESXi setup guide, you should have entered the IP address of the external workstation you will be using to manage the environment. Didn't do that? Check out the [Network Configuration](#) section for the pfSense VM for how to get yourself out of that mess, or if you need to add another management workstation, etc.

Once you have logged in to the web interface of the pfSense VM, navigate to Firewall > Rules and ensure that the WAN interface is selected. All of the firewall rules we'll be adding for remote access will be added to the WAN interface. In our case we have to add three firewall rules. Each of them is going to be for port 22/tcp (SSH). The destination IP addresses (use the "single host or alias" option) are 172.16.1.3, 172.16.1.4, and 172.16.2.2 -- the siem, ips, and kali VMs respectively. The source IP address for each rule should be your management workstation -- in my case 192.168.1.22 (again, use the "single host or alias" option). When you are done, your firewall rule set on the WAN interface should look something like this:

Rules (Drag to Change Order)										
States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
× 0/21 Kib	*	Reserved Not assigned by IANA	*	*	*	*	*	*	Block bogon networks	
✓ 8/3.61 MIB	IPv4 TCP	192.168.1.17	*	192.168.1.22	443 (HTTPS)	*	none		Easy Rule: Passed from Firewall Log View	
✓ 0/0 B	IPv4 TCP	192.168.1.17	*	172.16.1.3	22 (SSH)	*	none		Allow SSH access to SIEM VM	
✓ 0/0 B	IPv4 TCP	192.168.1.17	*	172.16.1.4	22 (SSH)	*	none		Allow SSH access to ips VM	
✓ 0/0 B	IPv4 TCP	192.168.1.17	*	172.16.2.2	22 (SSH)	*	none		Allow SSH access to kali VM	
✗ 0/0 B	IPv4+6	*	*	*	*	*	*	none	explicit deny any/any rule	

The new rules should be placed AFTER pass rule allow your management workstation access to the web interface, but BEFORE the explicitly deny any/any rule at the bottom of the rule list (if you elected to create one). If you need to change the rule order, you can drag and drop the rule columns to change the order in which they are evaluated.

The only obvious way to verify if the firewall rules are working is to actually try connecting via SSH, after you have finished setting up all three of these VMs (If you want to learn how to set up SSH on the kali VM, refer to the guide for [enabling SSH on the kali VM](#) in the remote access section of this book). If you haven't already, download an SSH client for the OS you will be using to manage your VMs (e.g. mremoteng for Windows, the native ssh client for OSX, Linux and/or BSD) and simply try authenticating as the user you set up as a part of the OS installation (or the user "root" if you set up SSH on kali):

```
Using username "ayy".
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-31-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.

*** System restart required ***
Last login: Thu Dec 22 15:06:56 2016
ayy@siem:~$ 
```

```
Using username "ayy".
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-31-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

110 packages can be updated.
60 updates are security updates.

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

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ayy@ips:~$ 
```

```

Using username "root".
Server refused our key

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.
root@kali:~#

```

As you can see in the illustrations above, I was able to connect and authenticate successfully to each session. At this point, I would recommend reading the rest of the remote access guide, either for [Windows](#) or [Linux/BSD/OSX](#) hosts, and setting up key-based authentication, as well as disabling password authentication for ssh entirely (if you use key-based auth). Additionally, if you're using a Windows workstation, you may want to read the Section under Defense in Depth for Windows Hosted Hypervisors, [Using Windows Firewall to Limit Exposure of Windows Hypervisor](#). As an abundance of caution, you'll probably want to make sure that you block ANY and all connections initiated from the 172.16.1.0/24, and 172.16.2.0/24 networks.

New Inbound Rule Wizard

Scope

Specify the local and remote IP addresses to which this rule applies.

Steps:

- Rule Type
- Program
- Protocol and Ports
- Scope**
- Action
- Profile
- Name

Which local IP addresses does this rule apply to?

Any IP address
 These IP addresses:

172.16.1.0/24
 172.16.2.0/24

Add... Edit... Remove...

Customize...

Which remote IP addresses does this rule apply to?

Any IP address
 These IP addresses:

172.16.1.0/24
 172.16.2.0/24

Add... Edit... Remove...

[< Back](#) [Next >](#) [Cancel](#)

Dealing with DHCP

DHCP can lead to remote access problems when dealing with VMs hosted on a baremetal hypervisor. Setting up static routes and firewall rules requires the IP addresses that you're routing to, and the source/destination addresses. As a part of the network setup portion for all of the hypervisors in this guide, we configured static DHCP mappings for all the VMs in our lab networks. This was to make sure that the IP addresses of each of our VMs always remained the same, even after reboot, without having to set IP addresses, default gateways, or DNS information manually on each virtual machine. As mentioned above, if at all possible, your management workstation, ESXi server, and WAN interface of the pfSense firewall should either have static DHCP mappings based on the mac addresses of these systems, or static IP addresses configured. Without knowing what your home or office network looks like, I can say with some measure of confidence that most network devices (even SOHO routers provided by most ISPs) allow you to configure static DHCP mappings or "IP Address Reservations".

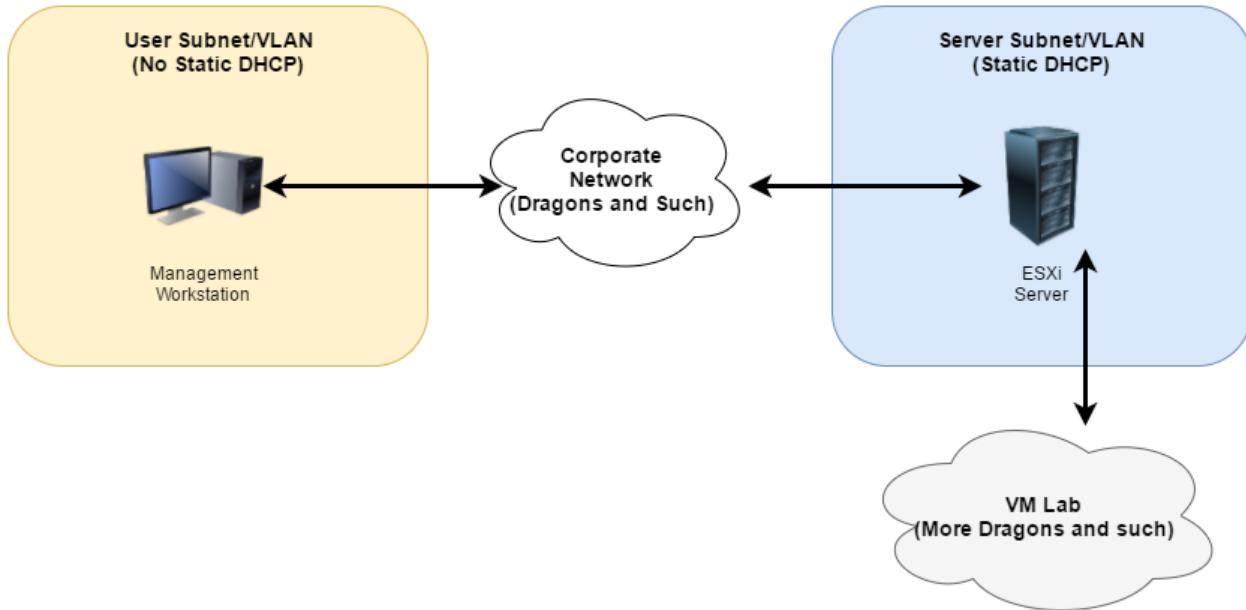
DHCP Connection Settings

Hostname:	pfSense
IP Address:	192 . 168 . 1 . 22
MAC Address:	00 : 0c : 29 : 49 : 38 : 3e
<input checked="" type="checkbox"/> Static Lease Type	
Apply Cancel	

The screen capture above is from my home network router; an Actiontec router my ISP provided me. Notice the option to set the DHCP lease type to Static. If this thing can do static DHCP allocations, I'm sure whatever you are using can do it as well! Consult with the device documentation or your ISP's tech support department if you need help. As an alternative, you can check to see what the DHCP IP address pool is for the network(s) your workstation, and ESXi Server and/or pfSense VM are connected to, and simply statically configure an IP address in the same local network, but outside of the DHCP address pool. For example, my home network has a SOHO router my ISP provides. The network is 192.168.1.0/24. The DHCP pool is 192.168.1.2 - 192.168.1.100. This means that for my workstation, the ESXi server, and the WAN interface on my pfSense router, I can configure a static IP address anywhere from 192.168.1.101 - 192.168.1.254 for these three systems. Bear in mind that if you have to do this, you will need to manually set the default gateway for these systems, as well as the DNS server to use.

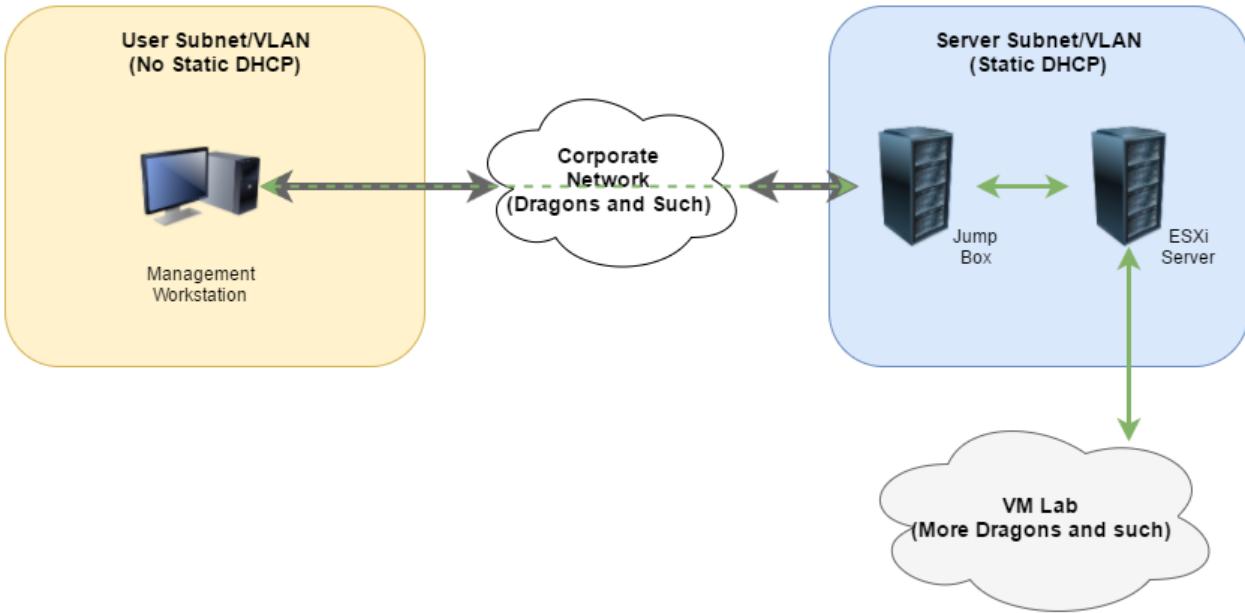
Jump Boxing

What are you supposed to do when your workstation resides in a network where static configuration or static DHCP allocations are not available?



In most enterprise networks, user systems usually reside in a subnet and/or vlan that is configured to use DHCP. Unless you are VERY good friends with your network admins, you probably won't be able to get a static DHCP mapping for your workstation. There may be technical reasons, or it may just be a policy. However, on the server subnet, the network and system admins have static DHCP allocations as an option. You can install a small embedded system (e.g. a raspberry pi), or create another small VM connected to the server subnet, like the ESXi server's management interface, and the pfSense VM's WAN interface. We can then set up firewall rules to let this system into the virtual network, and set up our static routes on this system, and redirect the traffic from our management workstation through this system. A system used to forward and broker your connections to another network is known as a jump box.

Note: Jump boxes are also referred to as proxies or redirectors. For the most part, these names are interchangeable. They all refer to a system that you use as a connection broker to forward your connections to another destination you cannot or do not want to interact with directly. In high integrity/high security networks, jump boxes are used by IT and Information Security staff to access assets in other network segments securely. The jump box acts as both a choke point, and a way to audit connections to sensitive assets.



In the diagram above, the grey lines denote a tunnel connection from our management workstation to the jump box located on the same subnet/vlan where our ESXi Server and VMs are. The green lines to the ESXi server (The pfSense VM on the ESXi server) and the VM lab network are connections from our management workstation that were either initiated by the jump box system, or forwarded from the jump box system to the destination in the VM lab.

Using a Raspberry Pi as a Jump Box

I have an older Raspberry Pi Model B (version 1) from some time ago, so I'll be writing this guide using this hardware, specifically. This process should be nearly identical for other newer raspberry pi models (with the exception of the raspberry pi zero - you'll need to use a USB OTG adapter to connect a USB to ethernet adapter to give your pi zero an internet connection). This guide also assumes that you'll be using wired ethernet to connect to your raspberry pi jump box, so if you want to use wi-fi, you're on your own (Though there shouldn't be too much of a difference, really).

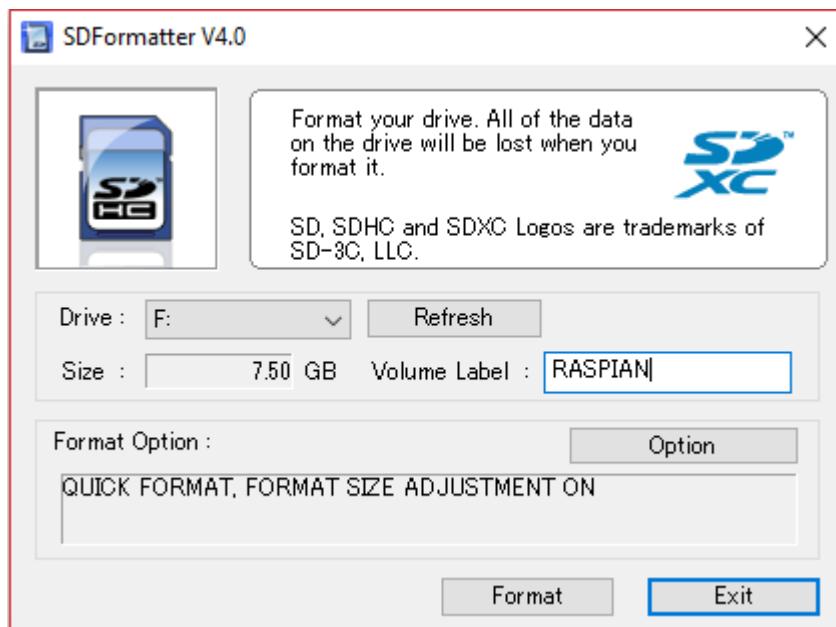
Why go through the trouble of setting up a physical jump box? Having a physical jump box has some advantages to it that you should consider. A lot of newer servers and server motherboards have small, embedded systems integrated into the server chassis often referred to as LOM, IPMI or some variation. These systems, so long as the server is plugged in and getting power can be remotely accessed to provide a variety of functions you would normally need physical access to do, such as physically power the server on or off, or redirecting console output on the server over the network. Having a separate physical jump box allow us to connect to these systems, in addition to access our VM lab systems. If the power on the server goes out, and you can't physically access the server, having a physical jump box at least lets you try to troubleshoot the issue remotely. While you don't have to use a raspberry pi, the raspberry pi

doesn't require much in the way of power; its primary power source is a micro USB connector. You can choose to connect a USB to micro USB cable to the server itself to get power from the USB controller on the server, or you can use an AC adapter to supply power to the raspberry pi from a power outlet. In either case, the raspberry pi requires very little power to operate, making it an ideal physical jump box.

Installing the Raspian Image to your Raspberry Pi

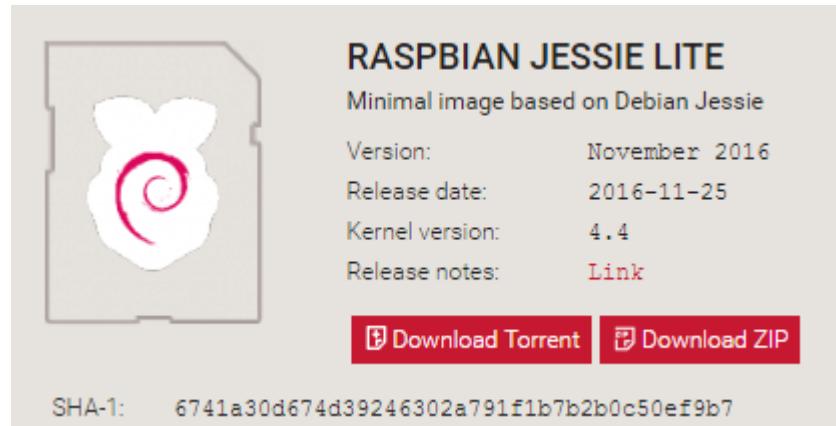
The fine folks at raspberrypi.org have a great guide for doing initial setup of your raspberry pi, and we'll be using their guide for installing raspbian, an rpi (shorthand for raspberry pi) optimized version of debian that we'll be installing to an SD card we will use to boot our rpi. You can use any SD card you'd like, so long as it is 4GB or larger. In my case I had an 8GB SD card lying around, so I used that.

The first thing they recommend doing is reformatting your SD card. The instructions for doing this are located at <https://www.raspberrypi.org/documentation/installation/noobs.md>. In my case, I will be using the Windows instructions that recommend download the SD card formatter tool from sdcard.org/downloads, and reformatting the card (please note that the link above has instructions for formatting your SD card on both OSX and Linux, using the SD card formatter tool for OSX, and gparted for Linux users, respectively). Be sure to configure the "Format size adjustment option to "ON" in order to resize the SD card partition to use the entire SD card if necessary.

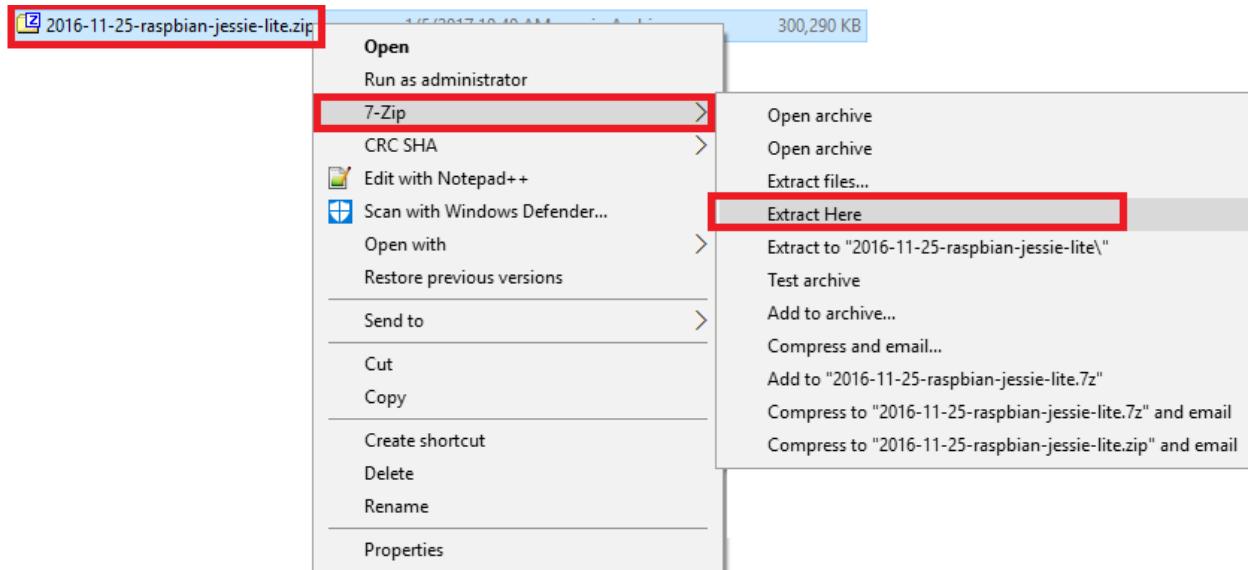


Note: IF you are using an SDXC card (these are defined as SD cards over 32GB in size), there are special instructions on how to format these SD cards for Windows, Linux, and OSX at https://www.raspberrypi.org/documentation/installation/sdxc_formatting.md

Next, visit <https://www.raspberrypi.org/downloads/raspbian/> and download the latest version of raspian. The current version of raspian (as of writing) is “Jessie”. You’ll want to download the lite version of the current raspian image. Why lite? Because outside of the initial setup here, you’ll probably never be connecting a keyboard and monitor to your rpi (unless something breaks), so anything that can be done to save drive space, RAM and CPU cycles is worthwhile.

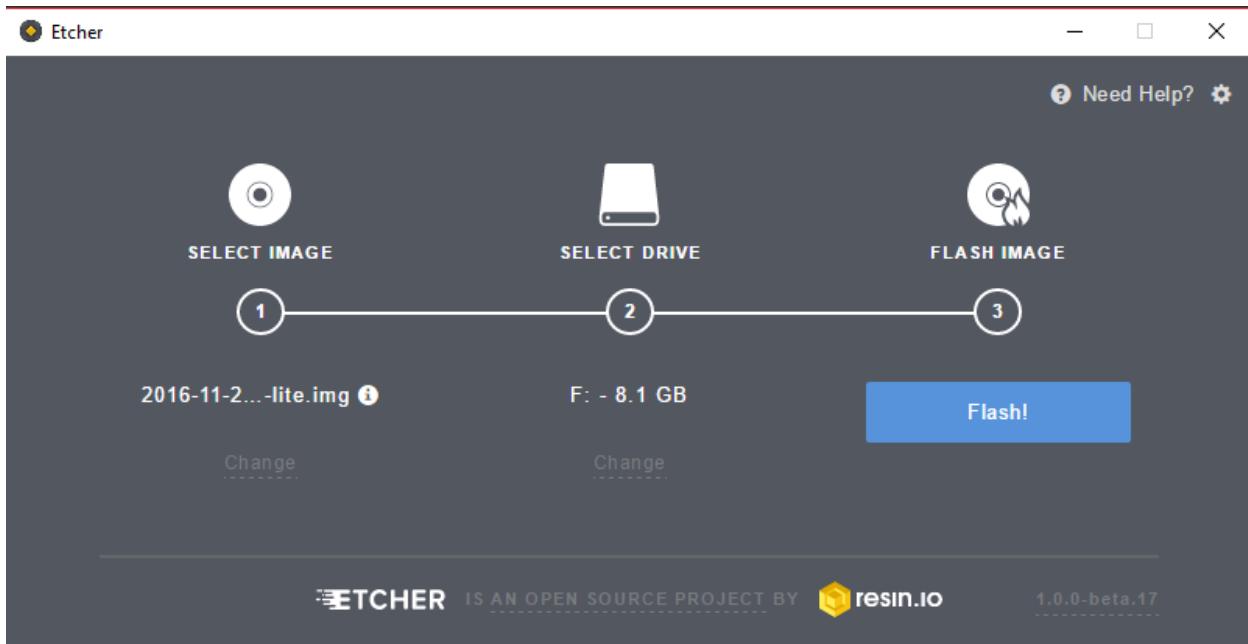


After you’ve downloaded the image, you’ll need to unzip it before we write it to the SD card. OSX and Linux users should use unzip, while Windows users should use 7-zip or another decompression utility to unzip the .img file.



Next, you need to write the image to the sd card. There are a number of image writing utilities for a variety of operating systems. The official instructions on raspberrypi.org instruct you to use the dd command and disk management tools for OSX and Linux and the Win32DiskImager utility for Windows. However, I discovered a tool called “Etcher”, from etcher.io that is available for Windows, Linux, and OSX that fits the bill nicely. Simply select the unzipped image file,

select the destination disk (the SD card), and click the flash button to write the image to the drive and verify it after writing it.



Now comes the fun part. Install the SD card into your raspberry pi, connect a USB keyboard, ethernet (make sure the network connection is on the same network that the ESXi server and pfSense VM's WAN interface is on), and HDMI video to your rpi, then connect USB power to your rpi to power it on.





```

Starting Trigger Flushing of Journal to Persistent Storage...
Starting LSB: Switch to ondemand cpu governor (unless shift key is pressed)
Starting LSB: Prepare console...
[ OK ] Started Create Volatile Files and Directories.
[ OK ] Started Tell Plymouth To Write Out Runtime Data.
[ OK ] Started Trigger Flushing of Journal to Persistent Storage.
Starting Update Utmp about System Boot/Shutdown...
[ OK ] Started Update Utmp about System Boot/Shutdown.
[ OK ] Started LSB: Switch to ondemand cpu governor (unless shift key is pressed)
[ OK ] Started LSB: Prepare console.
Starting LSB: Set console font and keymap...
[ OK ] Reached target Sound Card.
[ OK ] Started LSB: Set console font and keymap.
[ OK ] Started LSB: Raise network interfaces...
[ OK ] Reached target System Initialization.
[ OK ] Listening on Avahi mDNS/DNS-SD Stack Activation Socket.
[ OK ] Listening on D-Bus System Message Bus Socket.
[ OK ] Reached target Sockets.
[ OK ] Reached target Timers.
[ OK ] Reached target Basic System.
Starting LSB: Apply config from /boot/os_config.json...
Starting LSB: Resize the root filesystem to fill partition...
Starting dhcpcd on all interfaces...
Starting Regular background program processing daemon...
[ OK ] Started Regular background program processing daemon.
Starting Regenerate SSH host keys...
Starting Login Service...
Starting LSB: Autogenerate and use a swap file...
Starting LSB: triggerhappy hotkey daemon...
Starting Avahi mDNS/DNS-SD Stack...
Starting D-Bus System Message Bus...
[ OK ] Started D-Bus System Message Bus.
[ OK ] Started Avahi mDNS/DNS-SD Stack.
Starting System Logging Service...
[ OK ] Started LSB: triggerhappy hotkey daemon.
[ OK ] Started Login Service.
[ OK ] Started System Logging Service.
[ OK ] Started LSB: Resize the root filesystem to fill partition.
[ OK ] Started LSB: Apply config from /boot/os_config.json.
[ OK ] Started LSB: Autogenerate and use a swap file.
[ OK ] Started regenerate_ssh_host_keys.service.
[ OK ] Started dhcpcd on all interfaces.
[ OK ] Reached target Network.
Starting /etc/rc.local Compatibility...
Starting Permit User Sessions...
[ OK ] Reached target Network is Online.
Starting LSB: Start NTP daemon...
[ OK ] Started /etc/rc.local Compatibility.
[ OK ] Started Permit User Sessions.
[ OK ] Started LSB: Start NTP daemon.
Starting Hold until boot process finishes up...
Starting Terminate Plymouth Root Screen...
Raspbian GNU/Linux 8 raspberrypi ttys1
raspberrypi login: -

```

If you see a login prompt when you have everything connected, your raspberry pi is just about ready to go.

Configuring Raspian

The default credentials to log in to raspian are username “pi” and password “raspberry”. Log in via the console (keyboard and monitor). We need to make sure that the rpi system got an IP address, verify the system’s MAC address for making a static DHCP mapping, and enable the OpenSSH service is running for us to use this system as a jump box.

Once you have logged in, run the command `ifconfig -a` to display information about the interfaces on the rpi. Be sure to note your rpi’s IP address, and MAC address (labeled `Hwaddr` in the `ifconfig` output).

```

pi@raspberrypi:~ $ ifconfig -a
eth0      Link encap:Ethernet HWaddr b8:27:eb:b3:77:a6
          inet addr:192.168.1.10  Bcast:192.168.1.255  Mask:255.255.255.0
                      inet6 addr: fe80::59cc:372e:d8f8:25f7/64 Scope:Link
                         UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
                         RX packets:7233 errors:0 dropped:0 overruns:0 frame:0
                         TX packets:1372 errors:0 dropped:0 overruns:0 carrier:0
                         collisions:0 txqueuelen:1000
                         RX bytes:9848871 (9.3 MiB)  TX bytes:127484 (124.4 KiB)

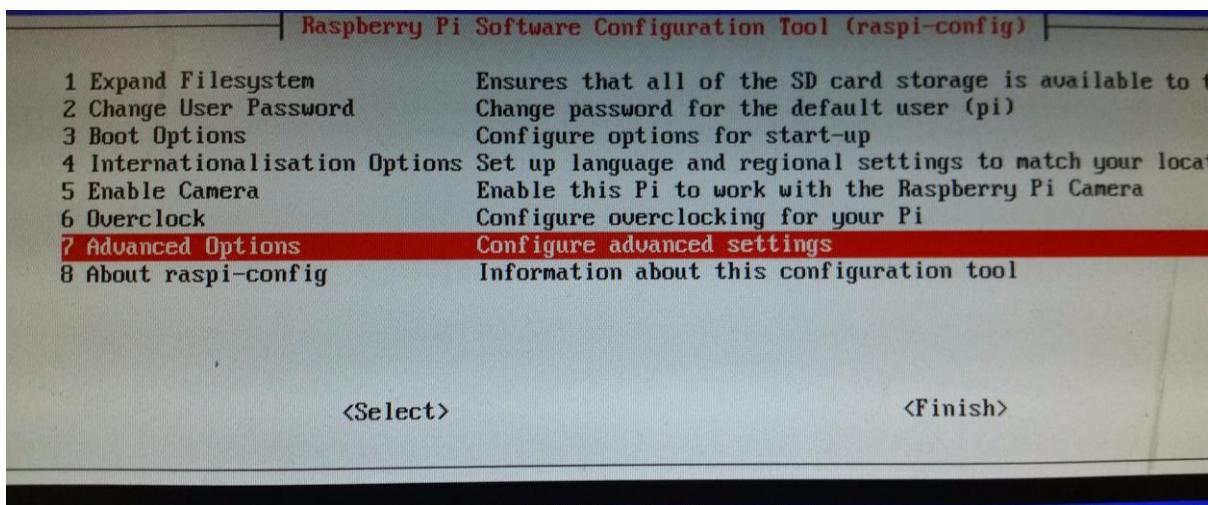
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
                         UP LOOPBACK RUNNING  MTU:65536  Metric:1
                         RX packets:0 errors:0 dropped:0 overruns:0 frame:0
                         TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
                         collisions:0 txqueuelen:1
                         RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

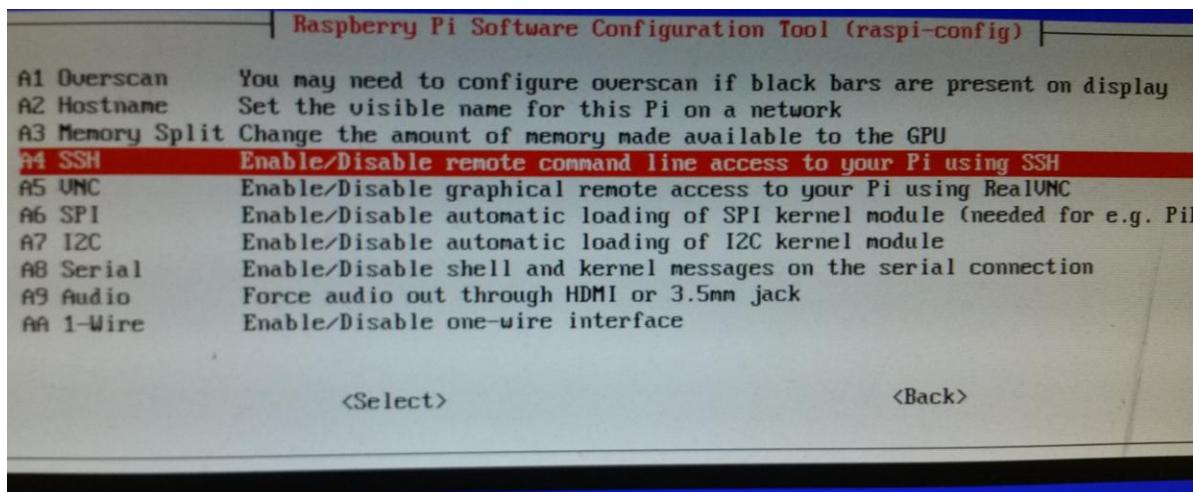
pi@raspberrypi:~ $

```

We can see that the rpi has an IP address of 192.168.1.10, and a MAC address of b8:27:eb:b3:77:a6.

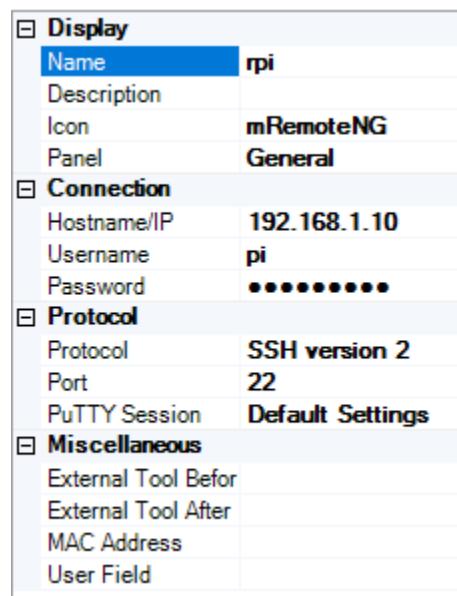
Next, run the command `sudo raspi-config` and a menu will pop up with a variety of different options available. Select option 7 (Advanced Options), then select option A4 (SSH Enable/Disable remote command line access to your Pi using SSH) to enable the SSH service.





After enabling the SSH server, hit escape to exit the menu and gain access to the command line again. Next, we have to test the SSH server and confirm we can log in.

Using the IP address you recorded from the ifconfig output, SSH to the raspberry pi using your SSH client of choice (e.g. Windows mremoteng, Linux/OSX the ssh command). Verify that you can log in with the system's default username and password.



```

Using username "pi".

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

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Thu Jan  5 18:08:33 2017 from new-host-3.blindseeker.com

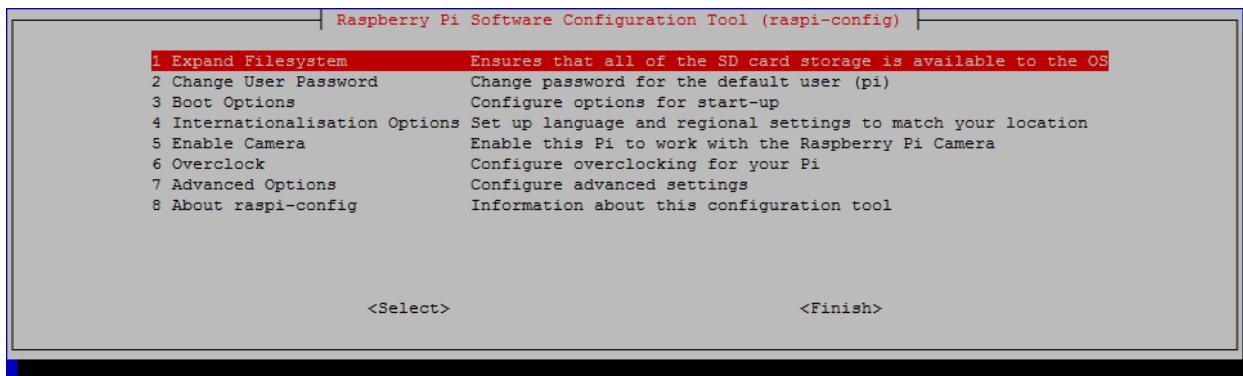
SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set a new password.

pi@raspberrypi:~ $ █

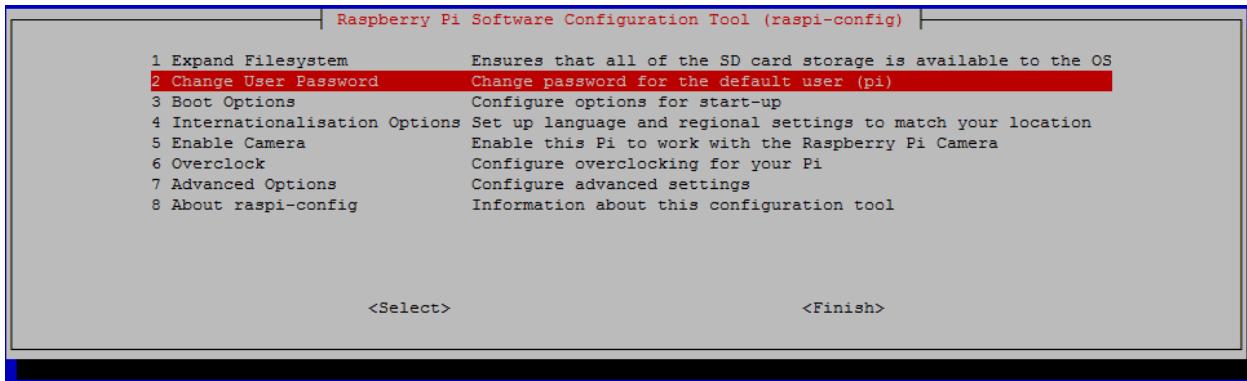
```

Note: Notice above that the system tells you that you have NOT changed the default password for the pi user. We're going to fix this in just a moment.

Now that you have remote access to your raspberry pi, there are some things we need to do before moving on. Run the command `sudo raspi-config` to open the raspberry pi configuration menu again. Select option 1 (Expand Filesystem). This will expand the raspian image to utilize all the space available on the SD card in order help prevent the system from running out of space.



When you select this option, the system informs you that the partition will be re-sized on the next reboot. So now, let's select option 2 (Change User Password) and change the pi user's password to something a bit more secure. We'll be disabling password authentication via SSH in any case however, it is good security hygiene to never leave default credentials enabled on any system you use.



When you are done, hit the escape key. The system will ask if you want to reboot now. Select yes, and give the system some time to go through the reboot. Remember to use the new password you assigned to the pi user.

Note: It is **very** important that you set a complex password for the pi user. The pi user has the ability to use sudo and run commands as root without a password. Take extra care to secure this account's password, and ensure that you run through the section on [disabling password authentication for SSH](#).

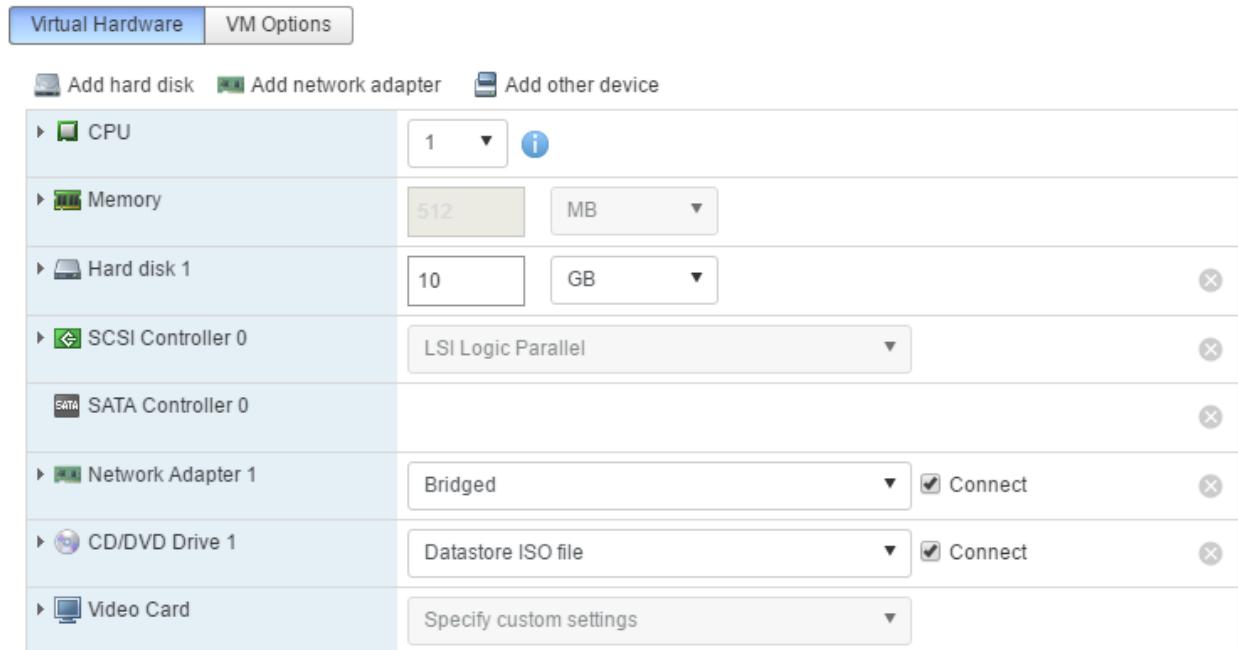
Once you have logged back into the rpi over SSH, you'll want to update the system and its packages (e.g. `export DEBIAN_FRONTEND=noninteractive; apt-get update; apt-get -y dist-upgrade`). I also highly recommend utilizing the `update.sh` script from the [Automated Patching for Linux Lab VMs](#) section for ensuring that your rpi jump box continues to stay up to date with the latest patches and security updates. At this point, you can safely remove the video and keyboard to make your rpi jump box "headless".

```
pi@raspberrypi:~$ sudo su -
root@raspberrypi:~# export DEBIAN_FRONTEND=noninteractive;apt-get update;apt-get -y dist-upgrade
```

After you have finished updating your rpi system, review the [Setting Up Your Jump Box](#) section for next steps.

Creating a Jump Box VM

On your ESXi server, create a small Ubuntu Linux 64-bit VM with 512mb of RAM, 10GB of hard drive space (thick provisioned, attached to SATA Controller 0 (0:0), and a single virtual network card connected to the bridged network.



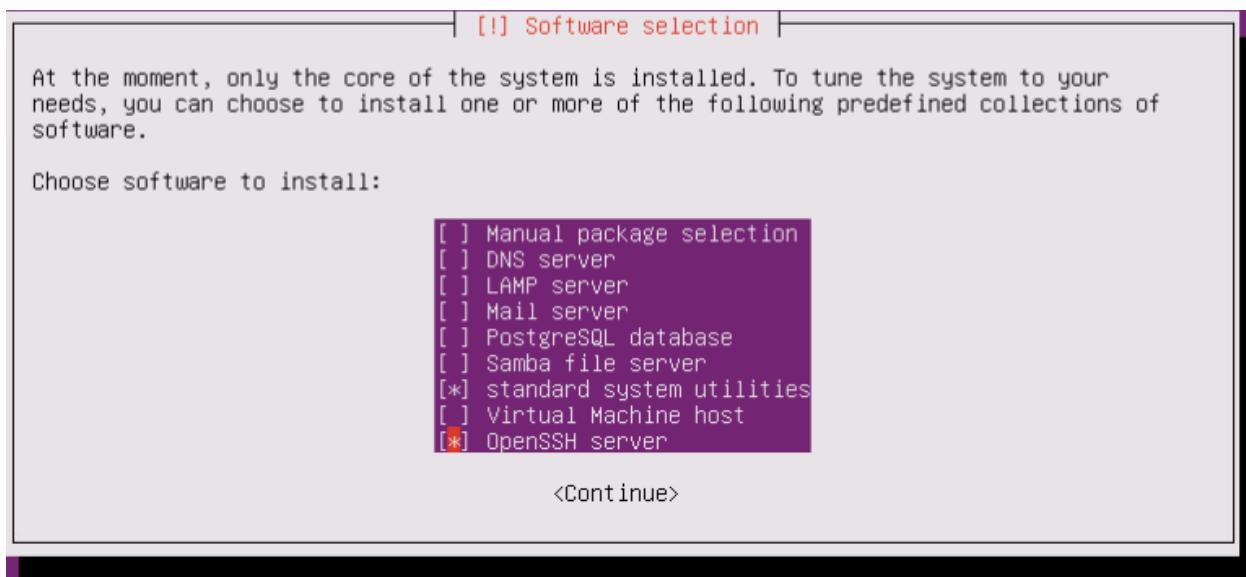
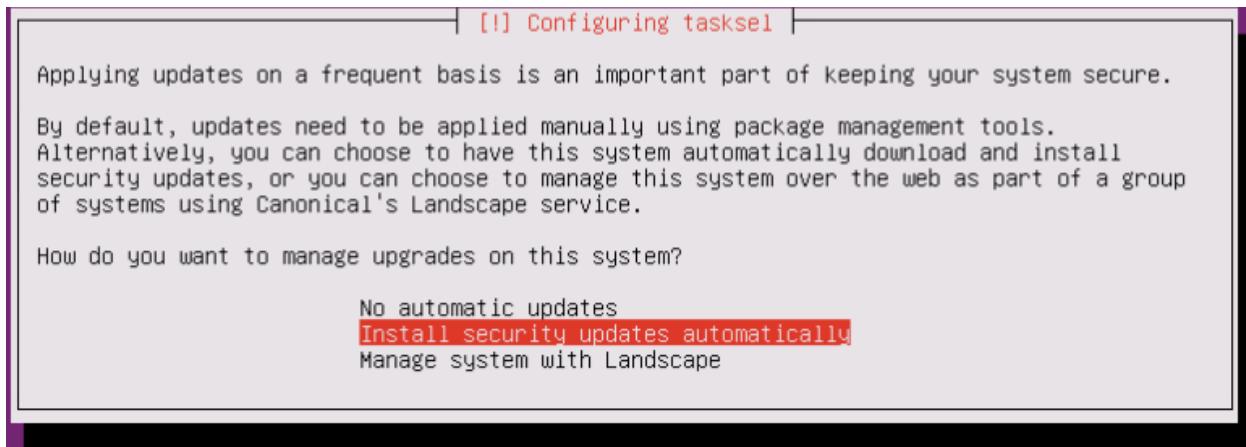
Just like the rest of our lab Virtual lab Linux systems, install Ubuntu Server 16.04 (Use the Ubuntu 16.04.1 ISO you previously uploaded to the datastore).



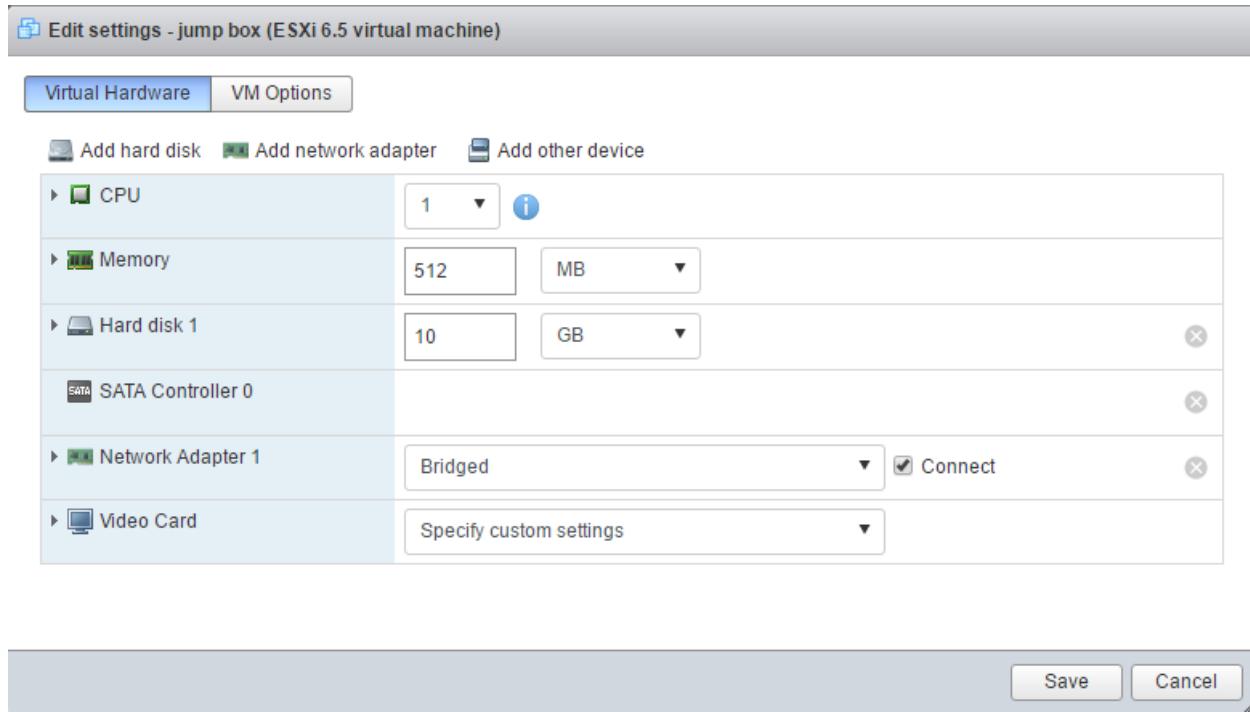
While the installer is running, check the hardware configuration of your VM from the ESXi web interface, and record the MAC address of your VM, if you plan on configuring a static DHCP allocation for your jump box.

▼ Network adapter 1	
Network	Bridged (Connected)
Connected	Yes
MAC address	00:0c:29:c1:ea:4a

Follow the same OS installation instructions as you followed for the other Ubuntu systems on your ESXi host -- configure the host to receive security updates automatically, and install the OpenSSH Server package (along with the “standard system utilities” package)



After the installation is complete, power down the VM, and edit the hardware settings. Make sure that the USB controller, CD/DVD drive, and the SCSI controller have all been removed.



Power the VM back on, login to your new VM, and download the latest updates for it. (e.g. sudo su - to become root, then run export DEBIAN_FRONTEND=noninteractive;apt-get -q update;apt-get -y -q dist-upgrade)

```
ayy@jumpbox:~$ sudo su -
[sudo] password for ayy:
root@jumpbox:~# export DEBIAN_FRONTEND=noninteractive;apt-get -q update; apt-get -y -q dist-upgrade
```

Run ifconfig -a to display the IP address that was assigned to your system as well as confirm your system's mac address.

```
ayy@jumpbox:~$ ifconfig -a
ens160      Link encap:Ethernet HWaddr 00:0c:29:c1:ea:4a
              inet addr:192.168.1.8 Bcast:192.168.1.255 Mask:255.255.255.0
                      inet6 addr: fe80::20c:29ff:fe29:c1ea/64 Scope:Link
                         UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
                         RX packets:92486 errors:0 dropped:0 overruns:0 frame:0
                         TX packets:33643 errors:0 dropped:0 overruns:0 carrier:0
                         collisions:0 txqueuelen:1000
                         RX bytes:183726131 (183.7 MB) TX bytes:2399738 (2.3 MB)

lo          Link encap:Local Loopback
              inet addr:127.0.0.1 Mask:255.0.0.0
              inet6 addr: ::1/128 Scope:Host
                         UP LOOPBACK RUNNING MTU:65536 Metric:1
                         RX packets:160 errors:0 dropped:0 overruns:0 frame:0
                         TX packets:160 errors:0 dropped:0 overruns:0 carrier:0
                         collisions:0 txqueuelen:1
                         RX bytes:11840 (11.8 KB) TX bytes:11840 (11.8 KB)
```

Afterwards, using your SSH client of choice (e.g. mremoteng on windows, ssh command on Linux/OSX) confirm that you are able to connect to your jump box VM over SSH using the username and password you created during the OS installation.

```
Using username "ayy".
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-57-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.

Last login: Thu Jan  5 15:13:45 2017 from 192.168.1.17
ayy@jumpbox:~$
```

While you're here, you may also want to implement the updater.sh script mentioned in the [Automated Patching for Linux Lab VMs](#) section to ensure your jump box is always up to date with the latest patches. After you are finished, complete the [Setting Up Your Jump Box](#) section.

Other Physical Jump Boxes

If you don't have a raspberry pi available, you can easily substitute it with any spare hardware you have lying around. Follow along with instructions from the [Creating a Jump Box VM](#) section (ignoring the ESXi configuration portions), and simply install Ubuntu Server 16.04 on your physical jump box.

The major differences to note are that you need to physically plug in your jump box to the same physical network the ESXi server and pfSense VM's WAN interface reside on, and that you will need to wait until the operating system is installed to check the MAC address of your system for making a static DHCP allocation. Other than that, the process is more or less identical.

Preparing Your Jump Box for Service

This section guides you through how to finish setting up your jump box and management workstation to access your VM lab. This portion of the guide borrows very heavily from other material in this book, namely, a vast majority of the Remote Lab Management guide. For example, setting up ssh on Windows or Linux/OSX/BSD management workstations (including recommended applications, generating SSH public/private key pairs, copying your public keys to other systems for key-based authentication, and finally, disabling password authentication as a countermeasure to prevent unauthorized access), and setting up persistence static routes on our Linux jump box. We will also be covering setting up static DHCP allocations, creating firewall rules to allow your jump box to access the lab VMs you created, and finally, a guide on using the jump box to forward your traffic to the lab VMs via SSH Tunneling (also known as TCP Forwarding).

Configuring Static DHCP Allocations

While it was mentioned earlier that one could configure static IP address, default gateway and DNS manually, I highly recommend configuring your device to utilize a static DHCP allocation, if at all possible, as opposed to static IP addressing. How you acquire static DHCP allocations on your network depends on where you're setting up your lab (home vs work), and the network equipment you're using. In my case, my router on my home network allows me to configure static DHCP allocations via an option called "IP Address Distribution". I can then display DHCP connections, and change "Dynamic" DHCP connections to "static":

Hostname:	jumpbox
IP Address:	192.168.1.8
MAC Address:	00:0c:29:c1:ea:4a
<input checked="" type="checkbox"/> Static Lease Type	
Apply Cancel	

I was able to confirm that my host, named "jumpbox" with the mac address we recorded earlier was assigned the IP address 192.168.1.8. All I had to do was check the "Static Lease Type" checkbox and click Apply.

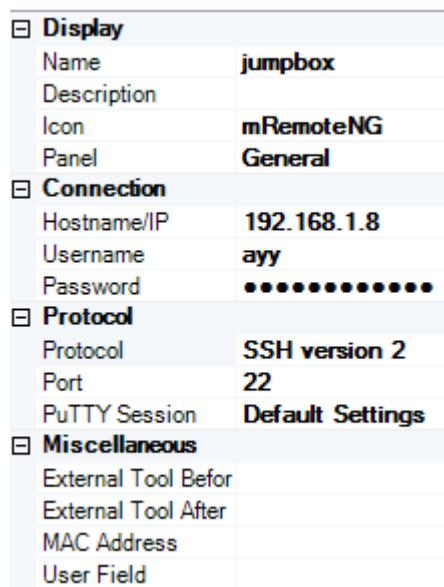
Enabling Key-Based Authentication for your Jump Box

The next step is to configure this system for remote access over SSH; specifically, copying your public SSH key to the host, and disabling password authentication on the jump box via `/etc/sshd_config`.

Windows

If you're using Windows as your management workstation you will use to connect to your jump box, follow the [Windows Remote Access](#) guide, skipping over the section on Persistent static routes. Specifically:

1. You should have already used mremoteng to log in via the user you set up during the OS installation for your jump box. If you have not, download and install mremoteng, Connect to the jump box, and log in

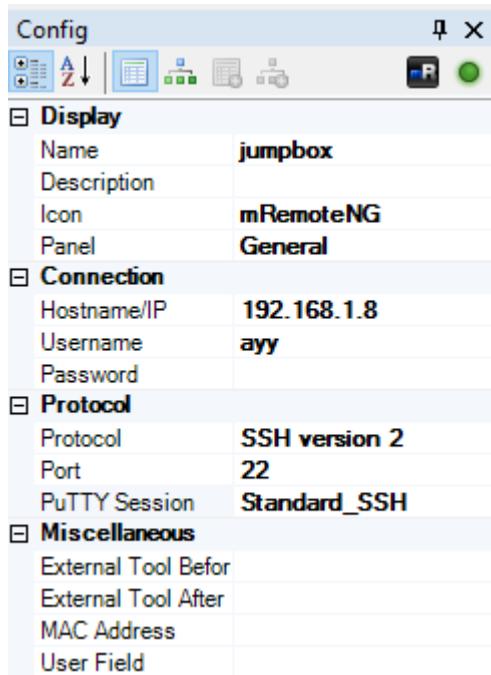


2. You should already have an SSH public/private key pair generated via puttygen. If you have not done so already, follow the Windows Remote Access guide and generate an SSH public/private key pair via puttygen. Add your system's public key to the `~/.ssh/authorized_keys` file on the jump box

```
ayy@jumpbox:~$ mkdir ~/.ssh;chmod 700 ~/.ssh;touch ~/.ssh/authorized_keys;chmod 600 ~/.ssh/authorized_keys
ayy@jumpbox:~$ vi ~/.ssh/authorized_keys
```

3. Modify your PuTTY Session and mremoteng connection profile to log in to the jump box with your ssh key, and verify that key-based authentication is working (Make sure to remove the password from your mremoteng connection profile). If you need assistance

with this, the Windows Remote Access guide has detailed instructions for creating a new PuTTY Session that utilizes your SSH key.



```
Using username "ayy".
Authenticating with public key "rsa-key-20140415"
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-31-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.

*** System restart required ***
Last login: Wed Jan  4 15:53:27 2017 from 192.168.1.17
ayy@jumpbox:~$
```

4. I **highly** recommend that you disable password authentication entirely for your jump box due to its exposure to a physical network. If you need guidance on how to do this, refer to the [Disabling Password Authentication Entirely via sshd_config](#) section of the Remote Lab Management guide. Become the root user (`sudo su -`) and modify `/etc/sshd_config` to disable password authentication, then restart the ssh service.

```
ayy@jumpbox:~$ sudo su -
root@jumpbox:~# vi /etc/ssh/sshd_config
root@jumpbox:~# cat /etc/ssh/sshd_config | egrep "^\$PasswordAuthentication"
PasswordAuthentication no
root@jumpbox:~# service ssh restart
root@jumpbox:~#
```

At this point your Windows workstation should be able to connect to the jump box with no password required.

Linux/OSX/BSD

If you're using Linux, OSX, or BSD as your management workstation you will use to connect to your jump box, follow the [Remote Access for Linux and OSX](#) guide, skipping over the sections on static routes. Specifically, you should:

1. You should have already used ssh to log in via the user you set up during the OS installation for your jump box. If you haven't already, create an alias for your jump box (e.g. echo "alias jumpbox='ssh ayy@192.168.1.8'" >> ~/.bash_profile; source ~/.bash_profile). And connect to it with the new alias to make sure it works.

```
Tonys-MacBook-Pro:~ trobinson$ vi ~/.bash_profile
Tonys-MacBook-Pro:~ trobinson$ source ~/.bash_profile
Tonys-MacBook-Pro:~ trobinson$ jumpbox
The authenticity of host '192.168.1.8 (192.168.1.8)' can't be established.
ECDSA key fingerprint is SHA256:fFA22l1QSoJ7tM+0T4KnB5VaKBbjHcFztAc4zHocJg.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.1.8' (ECDSA) to the list of known hosts.
ayy@192.168.1.8's password:
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-57-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.

Last login: Thu Jan  5 15:25:33 2017 from 192.168.1.17
ayy@jumpbox:~$
```

2. Add your system's public key to the user's `~/.ssh/authorized_keys` file

```
ayy@jumpbox:~$ mkdir ~/.ssh;chmod 700 ~/.ssh;touch ~/.ssh/authorized_keys;chmod  
600 ~/.ssh/authorized_keys  
ayy@jumpbox:~$ vi ~/.ssh/authorized_keys  
ayy@jumpbox:~$
```

3. Try reconnecting to the jumpbox with the alias you made before to see if key authentication is working properly.

```
Tonys-MacBook-Pro:~ trobinson$ jumpbox  
Enter passphrase for key '/Users/trobinson/.ssh/id_rsa':  
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-57-generic x86_64)  
  
 * Documentation: https://help.ubuntu.com  
 * Management: https://landscape.canonical.com  
 * Support: https://ubuntu.com/advantage  
  
0 packages can be updated.  
0 updates are security updates.  
  
Last login: Thu Jan  5 21:09:23 2017 from 192.168.1.3  
ayy@jumpbox:~$
```

4. I **highly** recommend that you disable password authentication entirely for your jump box due to its exposure to a physical network. If you need guidance on how to do this, refer to the [Disabling Password Authentication Entirely via sshd_config](#) section of the Remote Lab Management guide. Become the root user (`sudo su -`) and modify `/etc/sshd_config` to disable password authentication, then restart the ssh service.

```
root@jumpbox:~# vi /etc/ssh/sshd_config  
root@jumpbox:~# cat /etc/ssh/sshd_config | egrep "^\$PasswordAuthentication"  
PasswordAuthentication no  
root@jumpbox:~# service ssh restart  
root@jumpbox:~#
```

At this point your Linux/OSX/BSD workstation should be able to connect to the jump box with no password required.

Adding Static Routes to your Jump Box

So now that we have verified that we can connect to the jump box from our workstation, we now need to set up the jump box to access the VM lab network. This section will be borrowing material from Remote Access for Linux and OSX, since we'll be configuring a Linux system to access our VM lab. Specifically, we'll be using the sections on [the ip command](#), and [/etc/rc.local](#) for persisting static routes on reboot.

Connect to your jump box over ssh, and use the `sudo su -` command to become the root user. Using vi or the command line text editor of your choice, you will need to modify the file `/etc/rc.local`, and add our commands to establish static routes to the 172.16.1.0/24 and 172.16.2.0/24 networks.

```
root@jumpbox:~# vi /etc/rc.local
root@jumpbox:~# cat /etc/rc.local
#!/bin/sh -e
#
# rc.local
#
# This script is executed at the end of each multiuser runlevel.
# Make sure that the script will "exit 0" on success or any other
# value on error.
#
# In order to enable or disable this script just change the execution
# bits.
#
# By default this script does nothing.

ip route add 172.16.1.0/24 via 192.168.1.22
ip route add 172.16.2.0/24 via 192.168.1.22

exit 0
root@jumpbox:~#
```

In the screen capture above, 192.168.1.22 is the IP address of the WAN interface of the pfSense system on my network. Adjust that ip address as necessary for your network. After you have finished editing the `/etc/rc.local` file as root, reboot your jump box, then reconnect to it. Next, run the command `ip route show` to display the contents of the routing table.

```
Using username "ayy".
Authenticating with public key "rsa-key-20140415"
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-57-generic x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.

Last login: Thu Jan  5 21:42:43 2017 from 192.168.1.17
ayy@jumpbox:~$ ip route show
default via 192.168.1.1 dev ens160
172.16.1.0/24 via 192.168.1.22 dev ens160
172.16.2.0/24 via 192.168.1.22 dev ens160
192.168.1.0/24 dev ens160 proto kernel scope link src 192.168.1.8
ayy@jumpbox:~$
```

As you can see above, rc.local successfully added static routes to our VM lab networks.

Adding Firewall Rules and SSH tunnels to allow access to the VM lab networks

During the VMWare VSphere Hypervisor Setup guide (aka ESXi), the [network configuration](#) for pfSense portion of the guide, there is an initial walk-through on how to gain access to the pfSense web interface from the WAN interface of the firewall. That involved using the pfctl and easyrule commands from the console of the pfSense VM to temporarily let your management workstation in on the web UI, then creating a firewall rule to persist your access to the firewall's web interface. Additionally, there is the section of this guide titled [Creating Firewall Rules](#) for your management workstation to enable SSH access to your lab VMs.

Depending on whether or not your management workstation can reach the pfSense WebConfigurator, you may simply need to create new firewall rules for your jump box system. Otherwise, you'll have to use pfctl -d from the pfSense VM's console to temporarily disable the firewall and create pass rules to allow your jump box VM access to the WebConfigurator and VM lab.

In addition to creating firewall rules for your jump box to access the WebConfigurator and lab VMs, you'll also need to configure SSH for directing traffic from your management workstation to the pfSense WebConfigurator and lab VMs through your jump box system.

I Can Still Access the pfSense WebConfigurator with my Management Workstation

Log in to the pfSense WebConfigurator, then navigate to Firewall > Rules, and if it isn't already highlighted, select the "WAN" interface.

Rules (Drag to Change Order)										
States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
X	0/166 KiB	*	Reserved Not assigned by IANA	*	*	*	*	*	Block bogon networks	⚙️
▢	✓ 3/1.13 MiB	IPv4 TCP	192.168.1.17	*	192.168.1.22 443 (HTTPS)	*	none	Easy Rule: Passed from Firewall Log View	🔗 ✍️ 🕒 trash	
▢	✓ 0/0 B	IPv4 TCP	192.168.1.17	*	172.16.1.3 22 (SSH)	*	none	Allow SSH access to SIEM VM	🔗 ✍️ 🕒 trash	
▢	✓ 0/844 KiB	IPv4 TCP	192.168.1.17	*	172.16.1.4 22 (SSH)	*	none	Allow SSH access to ips VM	🔗 ✍️ 🕒 trash	
▢	✓ 0/0 B	IPv4 TCP	192.168.1.17	*	172.16.2.2 22 (SSH)	*	none	Allow SSH access to kali VM	🔗 ✍️ 🕒 trash	
▢	✗ 0/7.40 MiB	IPv4+6 *	*	*	*	*	none	explicit deny any/any rule	🔗 ✍️ 🕒 trash	

⬆️ Add
⬇️ Add
trash Delete
💾 Save
➕ Separator

ⓘ

As you can see from the screen capture above, I have a number of rules that allow my windows workstation (192.168.1.17) access to the pfSense WebConfigurator over HTTPS, and several of the lab VMs over port 22 (SSH). We're going to add a new set of rules to allow the jump box system (192.168.1.8) to access to these same systems. Modify the firewall rules for the WAN interface by adding 4 new pass rules -- one to access the pfSense WAN IP address (e.g. 192.168.1.22 in my case) over port 443(https) and 3 rules to access the kali (172.16.2.2), siem (172.16.1.3), and ips (172.16.1.4) VMs over port 22(SSH). These rules must be placed ABOVE the explicitly deny any/any rule (if you chose to create one. By default pfSense denies any traffic on an interface that is not explicitly allowed). Your firewall rules for the WAN interface should look like this:

Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description
*	Reserved Not assigned by IANA	*	*	*	*	*		Block bogon networks
IPv4 TCP	192.168.1.17	*	192.168.1.22	443 (HTTPS)	*	none		Easy Rule: Passed from Firewall Log View
IPv4 TCP	192.168.1.17	*	172.16.1.3	22 (SSH)	*	none		Allow SSH access to SIEM VM
IPv4 TCP	192.168.1.17	*	172.16.1.4	22 (SSH)	*	none		Allow SSH access to ips VM
IPv4 TCP	192.168.1.17	*	172.16.2.2	22 (SSH)	*	none		Allow SSH access to kali VM
IPv4 TCP	192.168.1.8	*	192.168.1.22	443 (HTTPS)	*	none		Allow Bastion Host to access pfSense web UI
IPv4 TCP	192.168.1.8	*	172.16.1.3	22 (SSH)	*	none		Allow Bastion Host to access SIEM VM over SSH
IPv4 TCP	192.168.1.8	*	172.16.1.4	22 (SSH)	*	none		Allow Bastion Host to access IPS VM over SSH
IPv4 TCP	192.168.1.8	*	172.16.2.2	22 (SSH)	*	none		Allow Bastion Host to access Kali VM over SSH
IPv4+6 *	*	*	*	*	*	none		explicit deny any/any rule

At this point you can log out of the WebConfigurator from your management workstation. Don't delete the firewall rules for your management workstation yet, and don't take a new snapshot of the pfSense VM just yet, because we need to verify that the rules are working as intended. Go to the section [TCP Forwarding and You](#), then [Testing your Dynamic Tunnels with FoxyProxy](#). After you have verified that your dynamic tunnel is Working and you have logged on from the IP address of your jump box, you can then proceed to [Testing Your Forward Tunnels](#).

After you have finished testing connectivity to all of your lab assets using your jump box system, Use the icon that looks like a trash bin, and delete the rules referring the IP address of your management workstation (e.g. in the above illustration, delete all the rules containing the IP address 192.168.1.17). Your final ruleset should look something like this:

Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description
*	Reserved Not assigned by IANA	*	*	*	*	*		Block bogon networks
IPv4 TCP	192.168.1.8	*	192.168.1.22	443 (HTTPS)	*	none		Allow Bastion Host to access pfSense web UI
IPv4 TCP	192.168.1.8	*	172.16.1.3	22 (SSH)	*	none		Allow Bastion Host to access SIEM VM over SSH
IPv4 TCP	192.168.1.8	*	172.16.1.4	22 (SSH)	*	none		Allow Bastion Host to access IPS VM over SSH
IPv4 TCP	192.168.1.8	*	172.16.2.2	22 (SSH)	*	none		Allow Bastion Host to access Kali VM over SSH
IPv4+6 *	*	*	*	*	*	none		explicit deny any/any rule

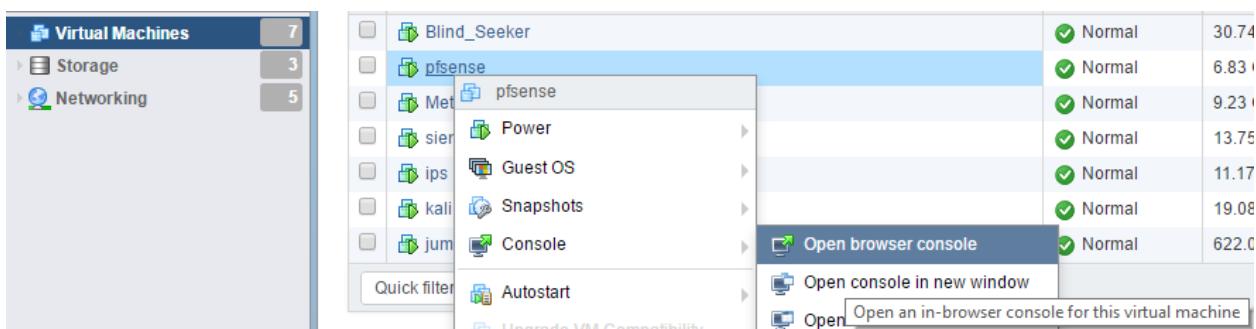
Once you have finished modifying the firewall rules for the WAN interface, save your changes and log out of the WebConfigurator.

If everything has gone well so far, your jump box should be ready for service.

I Have Lost Access to the pfSense WebConfigurator UI

So you no longer have access to the WebConfigurator interface, either due to misconfiguration, or due to DHCP giving your workstation a new IP address. So long as your management workstation can access the ESXi web interface or you can utilize the VMWare vSphere Client, no problem!

Login to the ESXi web interface. Click on Virtual Machines in the Navigator panel on the left side of the browser window, right click on the pfSense VM, click on Console, then click “Open browser console”. This opens a console session to the pfSense VM in your web browser, same as when we needed to initially configure the VM.



```
FreeBSD/amd64 (pfSense.bastion.local) (ttyv0)

*** Welcome to pfSense 2.3.2-RELEASE-p1 (amd64 full-install) on pfSense ***

WAN (wan)      -> em0          -> v4/DHCP4: 192.168.1.22/24
LAN (lan)      -> em1          -> v4: 172.16.1.1/24
OPT1 (opt1)    -> em2          -> v4: 172.16.2.1/24

0) Logout (SSH only)          9) pfTop
1) Assign Interfaces          10) Filter Logs
2) Set interface(s) IP address 11) Restart webConfigurator
3) Reset webConfigurator password 12) PHP shell + pfSense tools
4) Reset to factory defaults   13) Update from console
5) Reboot system               14) Enable Secure Shell (sshd)
6) Halt system                 15) Restore recent configuration
7) Ping host                   16) Restart PHP-FPM
8) Shell

Enter an option:
```

Select option 8 to access the shell interface for this system. Enter the command `pfctl -d` to completely disable the pfSense firewall. This is the fastest and easiest way for us to regain control to the firewall. Don't worry, we'll be re-enabling the firewall in just a moment.

```
[2.3.2-RELEASE] [root@pfSense.bastion.local]# pfctl -d
pf disabled
```

At this point, the firewall is disabled. We need to establish a dynamic tunnel and proxy off of our jump box to continue. Please proceed to the section [TCP Forwarding and You](#), then [Testing your Dynamic Tunnels with FoxyProxy](#). Confirm that you can login to the pfSense WebConfigurator from the WAN interface of the pfSense VM. Navigate to Firewall > Rules, and if it isn't already highlighted, select the "WAN" interface. We're going to add a new set of rules to allow the jump box system (192.168.1.8) to access to these same systems. Modify the firewall rules for the WAN interface by adding 4 new pass rules -- one to access the pfSense WAN IP address (e.g. 192.168.1.22 in my case) over port 443(https) and 3 rules to access the kali (172.16.2.2), siem (172.16.1.3), and ips (172.16.1.4) VMs over port 22(SSH). These rules must be placed ABOVE the explicitly deny any/any rule (if you chose to create one). By default pfSense denies any traffic on an interface that is not explicitly allowed). Depending on whether or not you added firewall rules to allow the (presumably now invalid) IP address of your management workstation access to your lab network, your firewall rules on the WAN interface may look like this:

Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description
*	Reserved Not assigned by IANA	*	*	*	*	*		Block bogon networks
IPv4 TCP	192.168.1.17	*	192.168.1.22	443 (HTTPS)	*	none		Easy Rule: Passed from Firewall Log View
IPv4 TCP	192.168.1.17	*	172.16.1.3	22 (SSH)	*	none		Allow SSH access to SIEM VM
IPv4 TCP	192.168.1.17	*	172.16.1.4	22 (SSH)	*	none		Allow SSH access to ips VM
IPv4 TCP	192.168.1.17	*	172.16.2.2	22 (SSH)	*	none		Allow SSH access to kali VM
IPv4 TCP	192.168.1.8	*	192.168.1.22	443 (HTTPS)	*	none		Allow Bastion Host to access pfSense web UI
IPv4 TCP	192.168.1.8	*	172.16.1.3	22 (SSH)	*	none		Allow Bastion Host to access SIEM VM over SSH
IPv4 TCP	192.168.1.8	*	172.16.1.4	22 (SSH)	*	none		Allow Bastion Host to access IPS VM over SSH
IPv4 TCP	192.168.1.8	*	172.16.2.2	22 (SSH)	*	none		Allow Bastion Host to access Kali VM over SSH
IPv4+6	*	*	*	*	*	none		explicit deny any/any rule
*								

Using the icon that looks like a trash bin, simply delete the rules referring the (old) IP address of your management workstation (e.g. in the above illustration, delete all the rules containing the IP address 192.168.1.17). Your final ruleset should look something like this:

Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description
*	Reserved Not assigned by IANA	*	*	*	*	*		Block bogon networks
IPv4 TCP	192.168.1.8	*	192.168.1.22	443 (HTTPS)	*	none		Allow Bastion Host to access pfSense web UI
IPv4 TCP	192.168.1.8	*	172.16.1.3	22 (SSH)	*	none		Allow Bastion Host to access SIEM VM over SSH
IPv4 TCP	192.168.1.8	*	172.16.1.4	22 (SSH)	*	none		Allow Bastion Host to access IPS VM over SSH
IPv4 TCP	192.168.1.8	*	172.16.2.2	22 (SSH)	*	none		Allow Bastion Host to access Kali VM over SSH
IPv4+6	*	*	*	*	*	none		explicit deny any/any rule
*								

Once you have finished modifying the firewall rules for the WAN interface, save your changes and log out of the WebConfigurator. When you saved the firewall configuration, pfSense should

have re-enabled the firewall. However if you want to be 100% sure, login to the ESXi web interface and open a console session to the pfSense VM. Select option 8 to open a root shell on pfSense, then run the command `pfctl -e`.

```
[2.3.2-RELEASE][root@pfSense.bastion.local]/root: pfctl -e  
pfctl: pf already enabled
```

If you got the output above, the firewall has already been re-enabled. If not, then you did save your firewall rule changes. You'll probably need to run `pfctl -d` again, and reconfigure the firewall rules on the WAN interface again.

If everything has gone well so far, proceed to the [Testing Your Forward Tunnels](#) section. If your forward tunnels are working properly, your jump box should be ready for service.

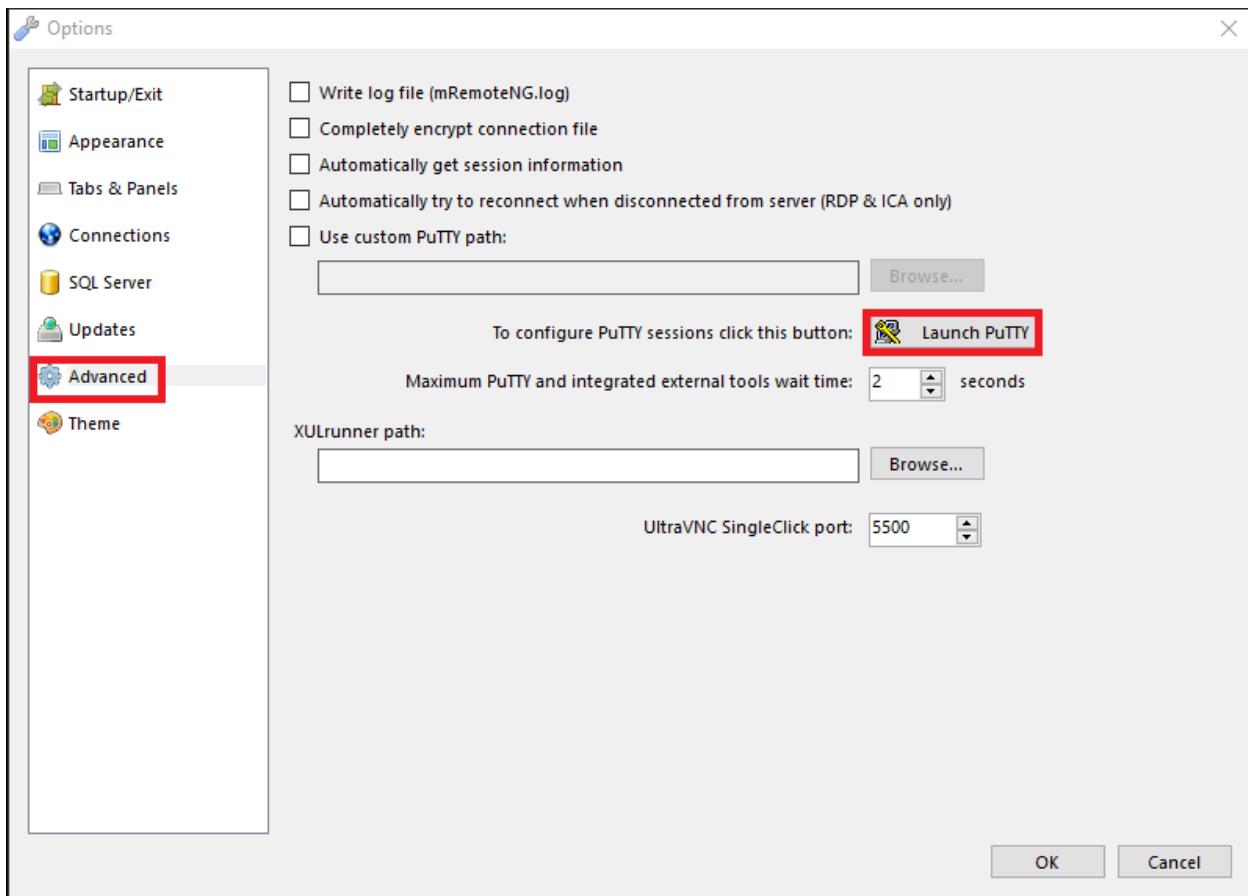
TCP Forwarding and You

Most SSH servers have an option available called "TCP Forwarding". A lot of professionals also refer to this as "SSH Tunneling". The idea is that SSH can be made to forward TCP network connections, not unlike a proxy to allow TCP connections to appear as though they are being sent from the SSH server itself. This functionality is the core of what allows our jump box to operate as a jump box for accessing our VM lab.

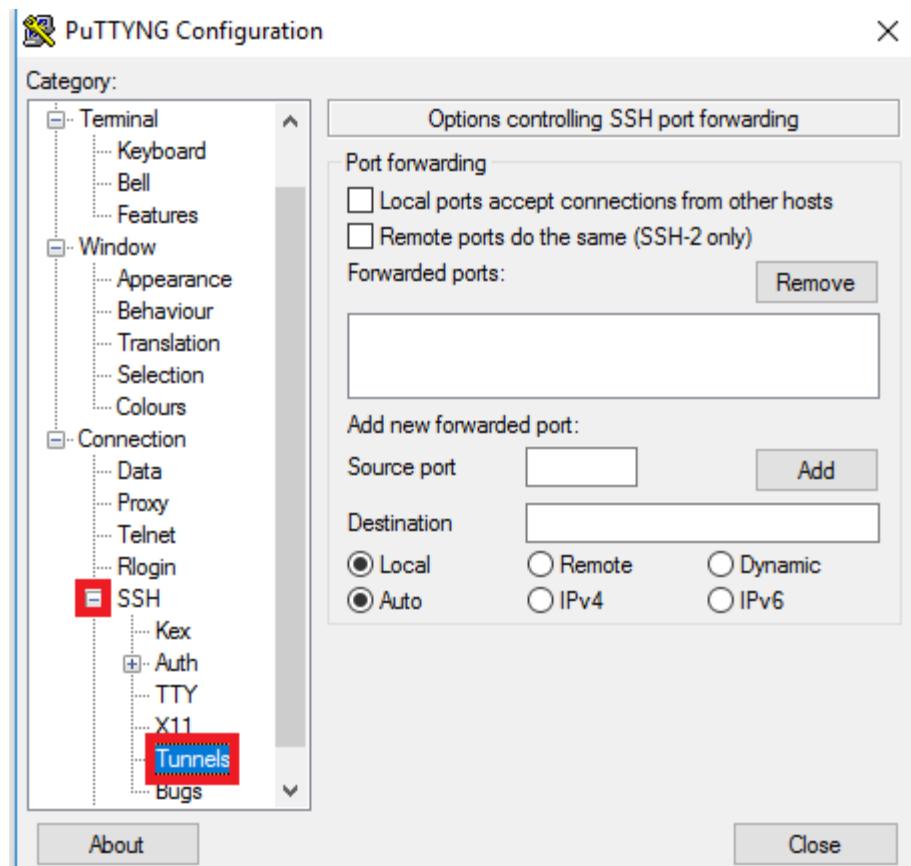
You will need to make some modifications to the SSH sessions you created to access your bastion host earlier. If you are using Windows as your management workstation, review the [Windows SSH Tunnels section](#). Linux/OSX/BSD management workstation users should review the [Linux/BSD/OSX SSH Tunnels](#) section, instead. These sections will teach you how to create Dynamic SSH tunnels for forwarding web traffic through your jump box, and Forward SSH tunnels for forward SSH (or other protocols) connections to the lab VMs through the jump box as well.

Windows SSH Tunnels

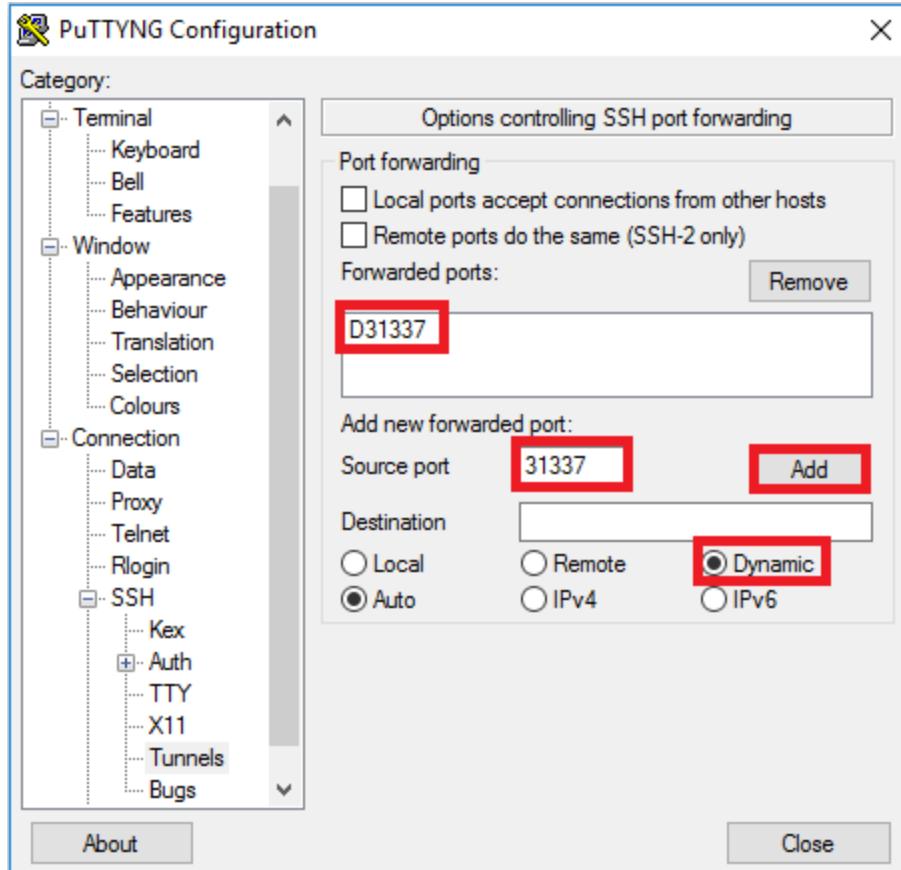
If you created mremoteng SSH sessions earlier to use key-based authentication, most of these set-up instructions will seem VERY familiar. Open up the mremoteng program, then on the menu bar at the top of the screen, select Tools > Options to open the options menu. Once in the Options menu, click "Advanced", then click on the "Launch PuTTY" button.



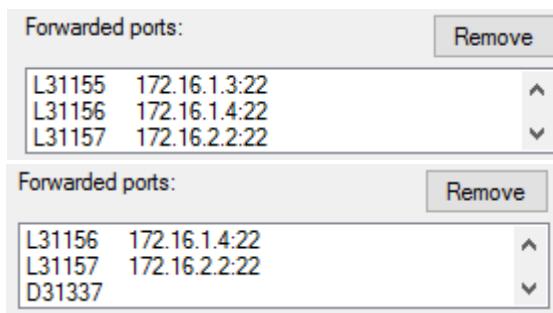
This should open up the PuTTY executable that mremoteng uses for establishing SSH connections. Under the Category pane, click on the plus sign next to “SSH”, then click on “Tunnels”.



In the input box labeled “Source port”, enter a random TCP port to bind on your Windows workstation above TCP port 1024. I usually pick a port number above 30000, like 31337, or another random, high-numbered TCP port. Leave the input box labeled “Destination” blank, and finally click the radio button next to “Dynamic”, then click the “Add” button. If you did it right, You will see the text “D31337 (or the source port you chose) appear in the message box labeled “Forwarded ports:”



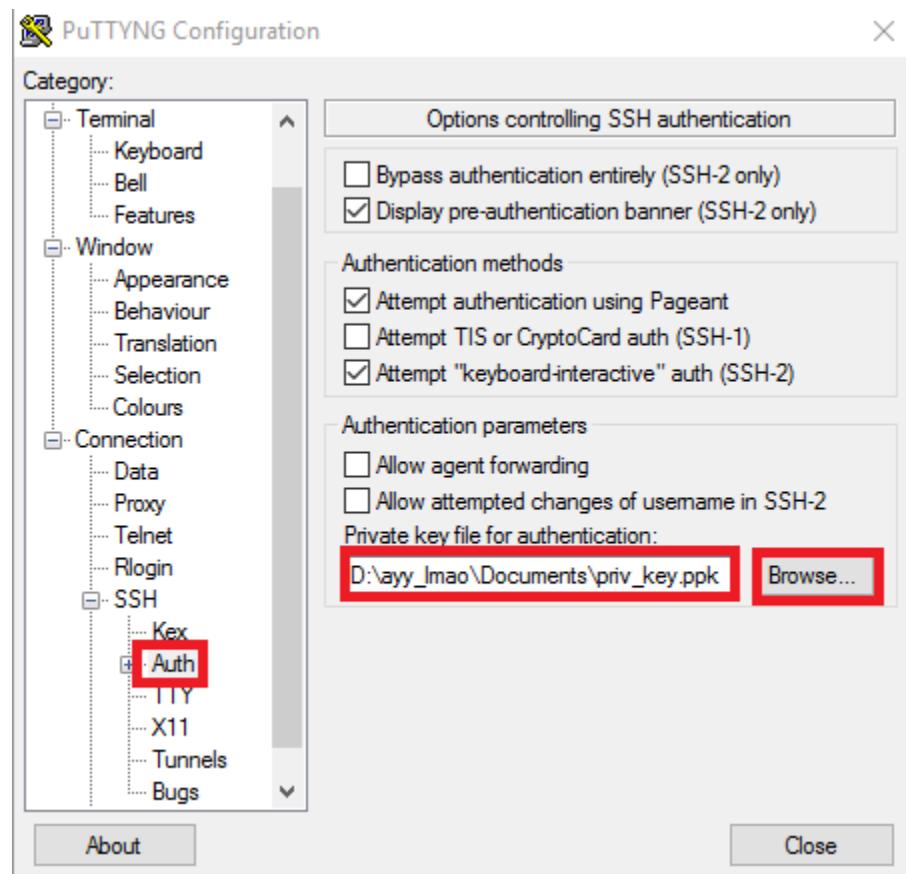
While we're here, we're going to add three more tunnels to access our Lab VMs easily. Enter another random TCP Source port in the "Source port" input box; for instance, I chose 31155. Then, input "172.16.1.3:22" into the "Destination" input box. Make sure that the "Local" radio box is selected, then click the "Add" button. Repeat this process twice changing the source port to 31156 and destination to "172.16.1.4:22", and finally 31157 and 172.16.2.2:22. Your "Forwarded ports:" field should look like this:



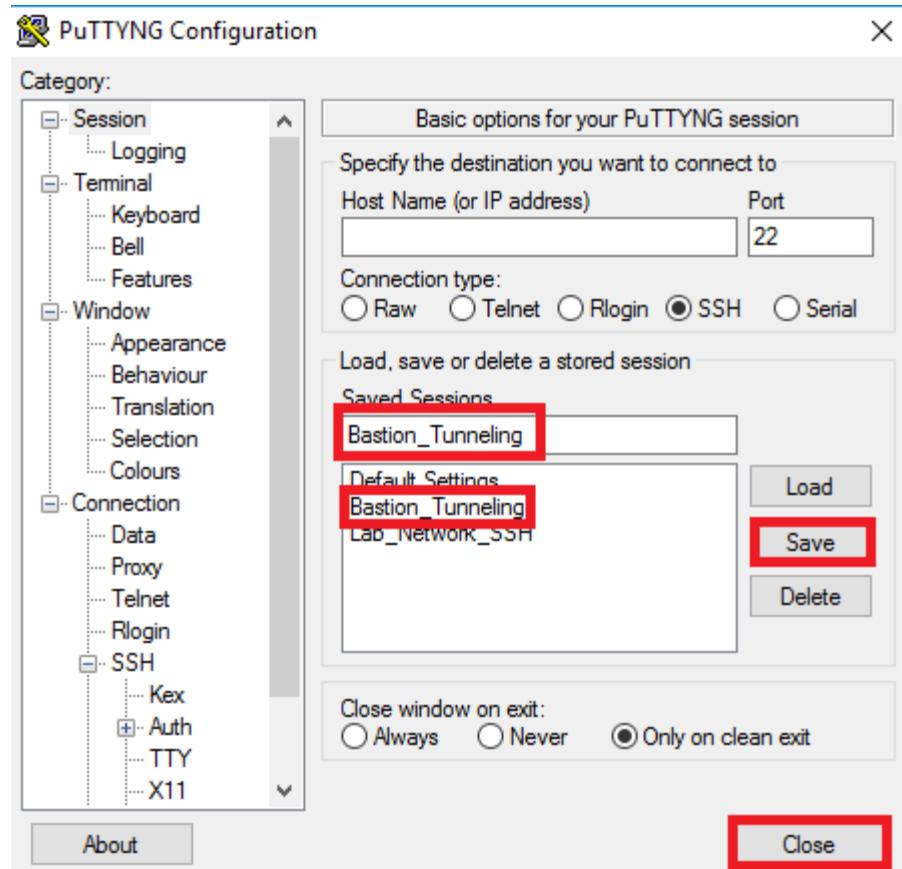
Note: D31337 should be there, simply scroll down in the forwarded ports field to reveal it. You should have **FOUR** entries in total.

This configuration creates a PuTTY session with one dynamic TCP tunnel, and three forward connect TCP tunnels. The dynamic tunnel allows us to use our jump box as a proxy and use its network connection to interact with the pfSense WebConfigurator, while the forward tunnels allow us to send SSH connection requests to our lab VMs, through our jump box.

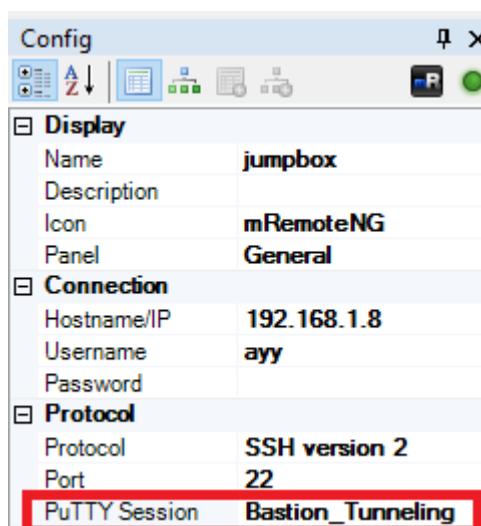
Next, click on the text labeled “Auth” in the category pane, then click the browse button next to the input box labeled “Private key file for authentication” browse to the private key ppk file you created with puttygen earlier, and select it. The input box should populate with the directory path to your private key.



Finally, scroll up on the Category pane and click “Session”. Under the input box labeled “Saved Sessions” input something to describe this PuTTY connection profile I chose “Bastion_Tunneling”, then click the save button. You can then click “Close” button to exit out of PuTTY.



Close the mremoteng options menu and return to the main screen. Click on the mremoteng session you created for your bastion host. In my case, the session is named “jumpbox”. In the config portion, under Protocol, there is an option called “PuTTY Session”. Click on the name of the current PuTTY Session. A small box with a downward facing arrow appears. Click on it to get a drop-down menu listing all the PuTTY Sessions mremoteng is configured with. Select the one we just made, (Bastion_Tunneling) to select it.



Finally, click File > Save Connection File to save your changes. Double click on your newly modified mremoteng session to see that it works and you are able to authenticate.

```
Using username "ayy".
Authenticating with public key "rsa-key-20140415"
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-57-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.

Last login: Mon Jan  9 14:24:45 2017 from 192.168.1.17
ayy@jumpbox:~$ 
```

Next, open up a command prompt on your Windows host, and run netstat -ano | findstr 127.0.0.1:31

```
C:\Users\ayy_lmao>netstat -ano | findstr 127.0.0.1:31
TCP    127.0.0.1:31155      0.0.0.0:0          LISTENING      9132
TCP    127.0.0.1:31156      0.0.0.0:0          LISTENING      9132
TCP    127.0.0.1:31157      0.0.0.0:0          LISTENING      9132
TCP    127.0.0.1:31337      0.0.0.0:0          LISTENING      9132
```

What this does is show us all strings from the netstat command that contain 127.0.0.1:31, and since all of our tunnels open a port on 127.0.0.1 (localhost) on ports 31155-57 and port 31337, we search for “31” since its the common denominator. When you are finished here, you can proceed to [Testing your Dynamic Tunnels with FoxyProxy](#).

Linux/BSD/OSX SSH Tunnels

Earlier, we created an SSH session our jump box and made it semi-permanent by simply adding an alias to our user’s ~/.bashrc or ~/.bash_profile configuration. Before you make any change to your .bashrc or .bash_profile again, I highly recommend making a copy by running cp ~/./.bash_profile ~/./.bash_profile_bak; cp ~/./.bashrc ~/./.bashrc_bak just to cover your bases and make sure you have something to recover your changes from.

Next, using your favorite text editor, you need to modify the alias entry you have for your jumpbox system to:
alias jumpbox='ssh -D31337 -L31155:172.16.1.3:22 -L31156:172.16.1.4:22 -L31157:172.16.2.2:22 ayy@192.168.1.8'

This alias creates one dynamic (-D) TCP tunnel, and three Local (-L) forward connect TCP tunnels. The dynamic tunnel allows us to use our bastion host as a proxy and use its network connection to interact with the pfSense WebConfigurator, while the forward tunnels allow us to send SSH connection requests to our lab VMs, through our bastion host.

Save the changes to your .bash_profile or .bashrc file. Then either use the source command (e.g. source ~/.bashrc OR source ~/.bash_profile) to reload the configuration, or simply reboot your management workstation, then verify you can connect to your jump box.

```
Tony's-MacBook-Pro:~ trobinson$ vi ~/.bash_profile
Tony's-MacBook-Pro:~ trobinson$ cat ~/.bash_profile | grep jumpbox
alias jumpbox='ssh -D31337 -L31155:172.16.1.3:22 -L31156:172.16.1.4:22 -L31157:172.16.2.2:22 ayy@192.168.1.8'
Tony's-MacBook-Pro:~ trobinson$ source ~/.bash_profile
Tony's-MacBook-Pro:~ trobinson$ jumpbox
Enter passphrase for key '/Users/trobinson/.ssh/id_rsa':
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-57-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.

Last login: Mon Jan  9 15:16:24 2017 from 192.168.1.3
ayy@jumpbox:~$
```

If you're using OSX/BSD, open a terminal session, and run netstat -anp tcp | grep LISTEN | grep 127.0.0.1.31

```
Tony's-MacBook-Pro:~ trobinson$ netstat -anp tcp | grep LISTEN | grep 127.0.0.1.31
tcp4      0      0  127.0.0.1.31157      *.*
tcp4      0      0  127.0.0.1.31156      *.*
tcp4      0      0  127.0.0.1.31155      *.*
tcp4      0      0  127.0.0.1.31337      *.*
```

This command runs netstat -anp tcp (which looks for JUST TCP connections). We then use the grep command to look for lines that contain “LISTEN” as in that port is listening for connections. We then use grep again to look for any lines within those lines that contain 127.0.0.1.31. This specifically looks for TCP sessions listening for connections on 127.0.0.1 (localhost). All of the TCP ports we defined in our ssh command begin with “31” so our grep command will match on 31155-31157 and 31337. Since the command has four entries, we know that our tunnels are all functioning normally.

Linux users should run netstat -tlpn | grep 127.0.0.1:31 to return similar results. Just like with OSX/BSD systems, you are expecting to have four lines returned -- 127.0.0.1:31155-57, and 127.0.0.1:31337.

Note: In the alias above, the ports 31337, 31155-57 can be **any TCP port above 1024**, so long as the port is NOT currently in-use by the operating system and the port chose for each -L or -D option is NOT the same. As a reminder, you may need to modify the line of your ssh alias from ayy@192.168.1.8 to the username and IP address you configured for your environment.

When you are finished here, proceed to [Testing your Dynamic Tunnels with FoxyProxy](#).

Testing your Dynamic Tunnels with FoxyProxy

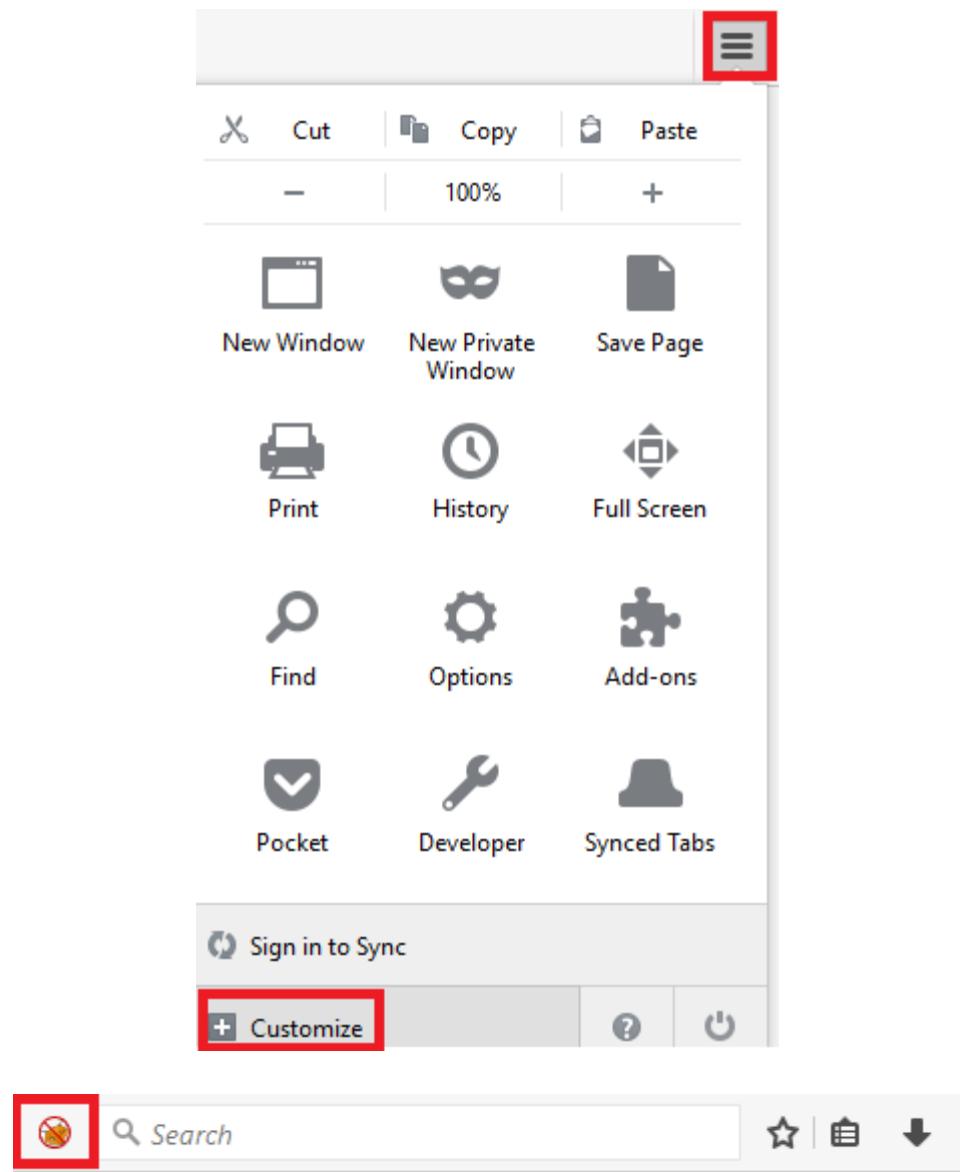
Before we move on, on your management workstation, We need to configure your web browser for using the bastion host as a proxy. I recommend using either Google Chrome or Mozilla FireFox web browsers, since they both support the browser extension, "FoxyProxy". Next, you will need to install the extension. FoxyProxy allows you to create proxy configurations/sessions that can be instantly applied to your web browser at any time. These will allow us to use our management workstation to access the pfSense WebConfigurator via the dynamic SSH tunnel we establish on our bastion host.

You can download Mozilla FireFox at <https://www.mozilla.org/en-US/firefox/desktop/>

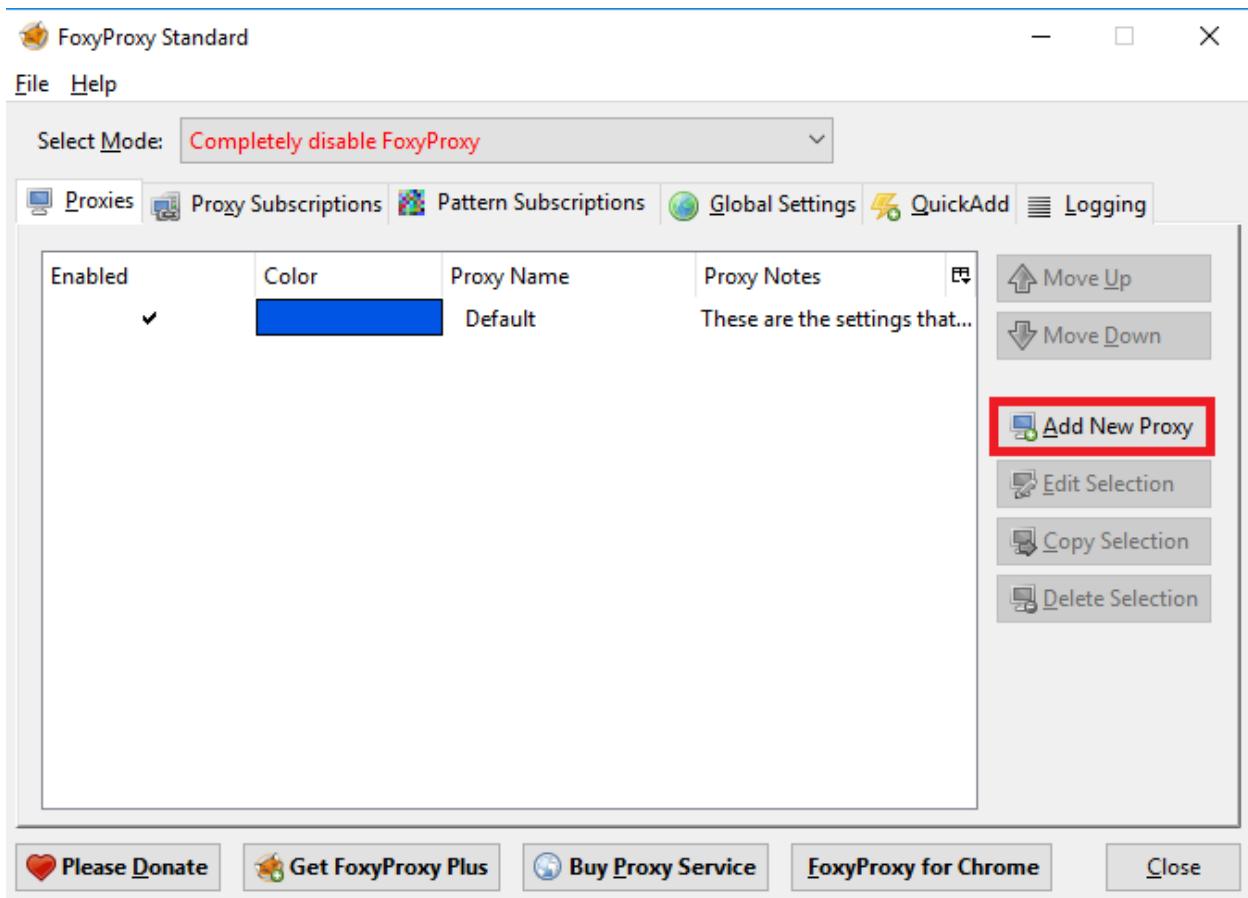
You can download Google Chrome at <https://www.google.com/chrome/>

You can download FoxyProxy at <https://getfoxyproxy.org/downloads/>

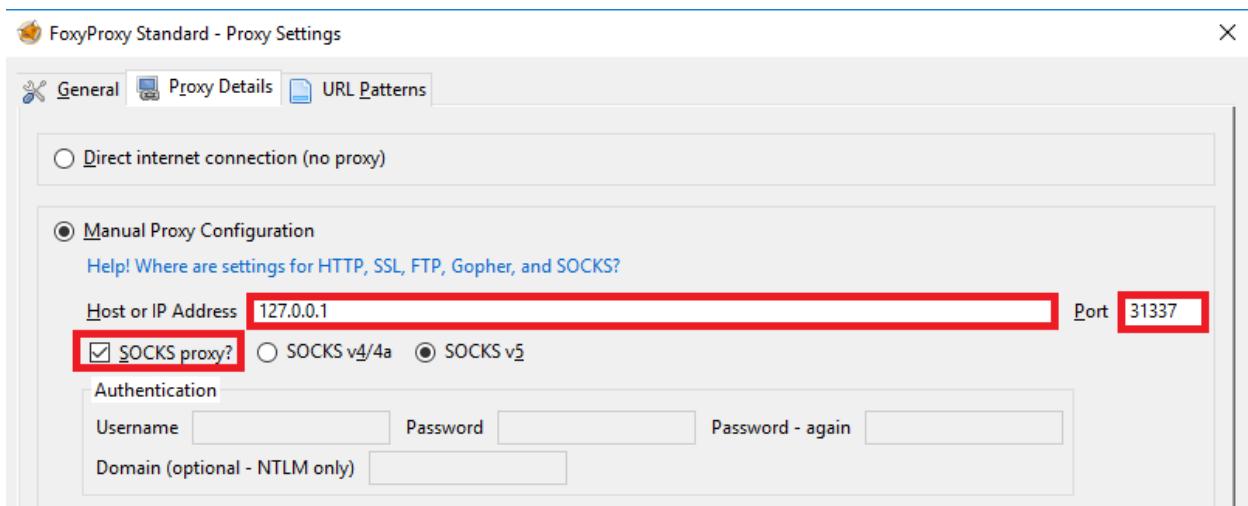
Open your favorite web browser with FoxyProxy installed, for this exercise, I am using Firefox on Windows, but the process should be more or less the same using other web browsers on other platforms -- find the the FoxyProxy settings to add a new proxy, and follow along. If you haven't already (and you're using FireFox) I highly recommend adding the icon for FoxyProxy to the web bar in FireFox. This can be done by clicking the settings icon > Customize, then dragging the FoxyProxy icon to the address bar.



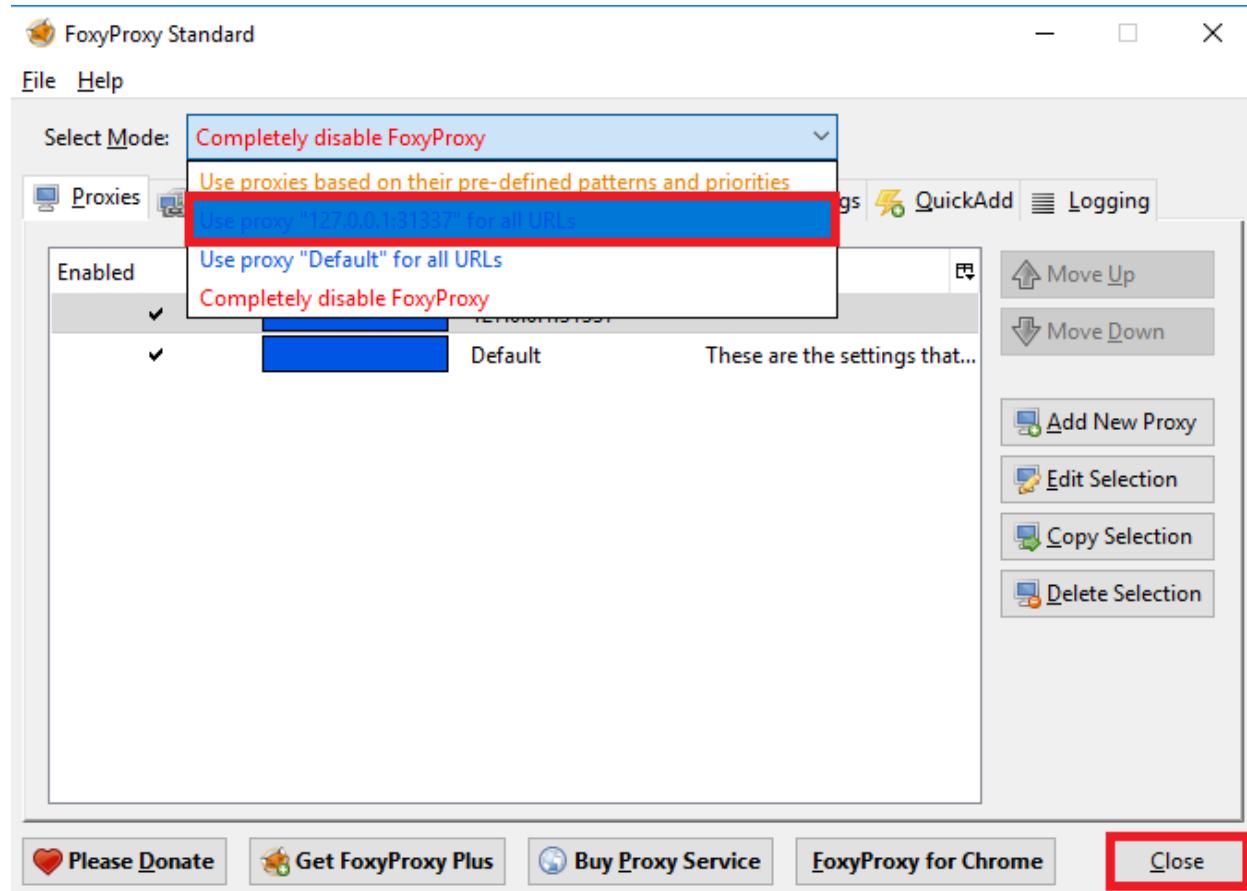
Click on the FoxyProxy icon (or otherwise enter the configuration settings in your browser for FoxyProxy), then in the new window that opens up, click the “Add New Proxy” button.



On the next window, ensure that the “Manual Proxy Configuration” radio button is selected. In the “Host or IP Address” input box, enter “127.0.0.1”. In the “Port” input box you MUST use the same port you used when creating your dynamic tunnel via ssh -D or mremoteng. In my case, this is port 31337. Finally, click the checkbox labeled “SOCKS proxy?” to check the box. Finally, click OK to save the new proxy.



Clicking OK will bring you back to the main menu. There is a drop-down named “Select Mode:” whose default value is set to “Completely disable FoxyProxy”. If you click on this text, a drop-down menu appears. Click on the value that says “Use proxy “127.0.0.1:31337” for all URLs”, then click the “Close” button.



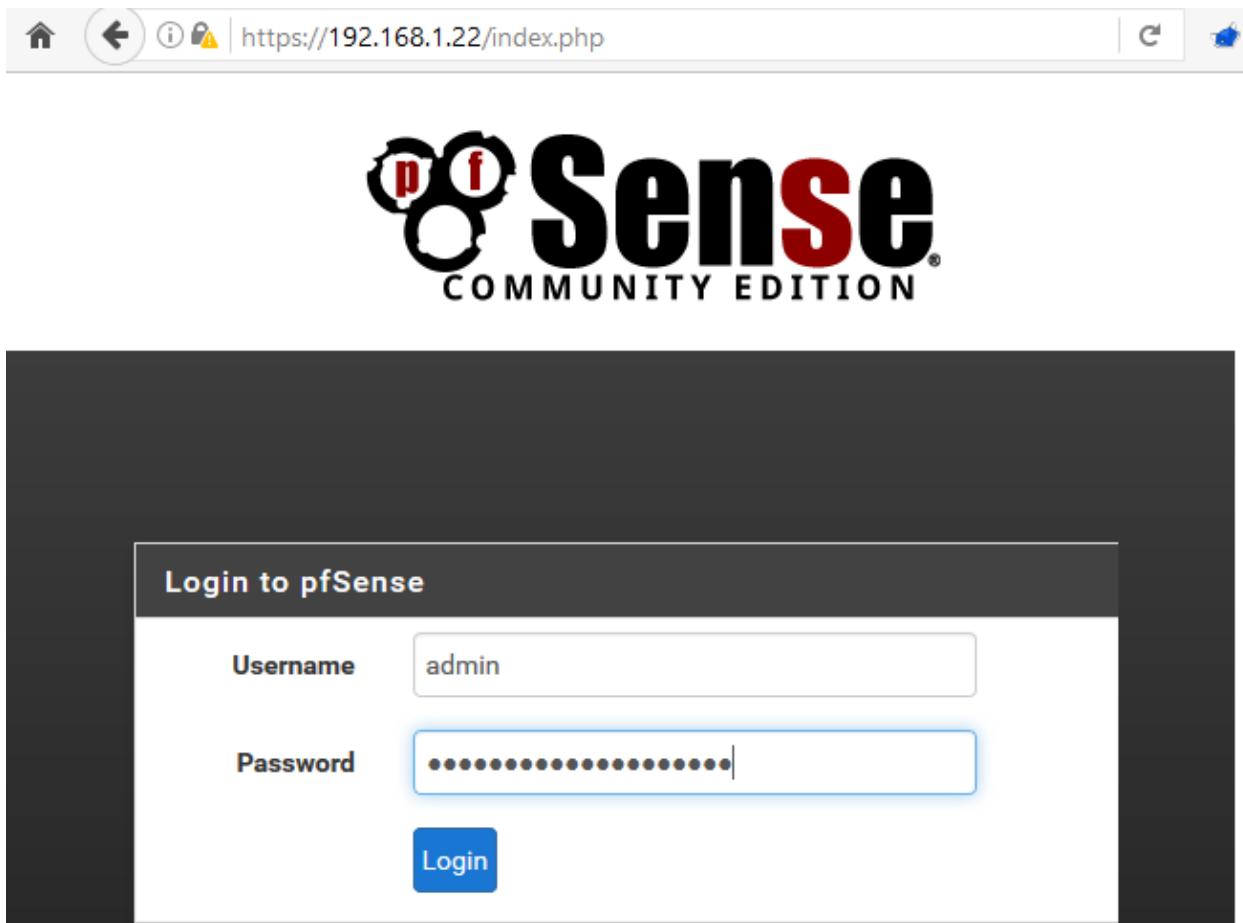
Note: When FoxyProxy says “Use proxy “x.x.x.x:xxxxxx” for all URLs”, it really means what it says. **All of your web traffic will be funneled through your jump box.** I would highly suggest experimenting with Firefox and Chrome profiles/sessions, and Creating a separate person/profile that you use exclusively for accessing the pfSense web UI through the jump box, if you don’t want your jump box to be proxying all of your web traffic.

For more on adding profiles to Chrome, visit
<https://support.google.com/chrome/answer/2364824?co=GENIE.Platform%3DDesktop&hl=en>
Chrome grants you the ability swap profiles directly in the web browser.

Firefox users should create a new profile via the profile manager, and get familiar with firefox’s -ProfileManager and -no-remote command line options. Go here for more information
https://developer.mozilla.org/en-US/docs/Mozilla/Command_Line_Options

Now we have to actually test and verify that the Dynamic Tunnel/Proxy is working. Start up an SSH session to your jump box VM using the newly modified connection profile or alias that creates the dynamic and forward tunnels we set up just a moment ago. This should be obvious, but **you need to have an active connect to the jump box system in order to use it as a proxy/tunnel system.**

Try connecting to the IP address of the WAN interface on your pfSense system. If you created a firewall rule to allow your jump box to access the pfSense VM (or used pfctl -d) you should get access to the web prompt. Login as normal.



If you are able to log in, then your dynamic tunnel should be working. If you're like me and you REALLY want to make sure your tunnel is working, login to the ESXi web interface, and open up a console session to your pfSense VM.

```
Message from syslogd@pfSense at Jan 10 16:13:10 ...
pfSense php-fpm[270]: /index.php: Successful login for user 'admin' from: 192.16
8.1.8
```

In my case, my jump box system's IP address is 192.168.1.8, so that means that my dynamic tunnel/proxy is working fine.

```
ayy@jumpbox:~$ ifconfig ens160
ens160      Link encap:Ethernet HWaddr 00:0c:29:c1:ea:4a
              inet addr:192.168.1.8 Bcast:192.168.1.255 Mask:255.255.255.0
              inet6 addr: fe80::20c:29ff:fea4a/64 Scope:Link
                  UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
                  RX packets:45315 errors:0 dropped:0 overruns:0 frame:0
                  TX packets:16686 errors:0 dropped:0 overruns:0 carrier:0
                  collisions:0 txqueuelen:1000
                  RX bytes:46893671 (46.8 MB) TX bytes:38564143 (38.5 MB)
```

If you get an error along the lines of “Proxy Server is refusing connections”, “Connection Timed out”, “Connection Refused”, or an IP address that is NOT your jump box’s IP address (e.g. your management workstation’s IP address) appears in the pfSense console logs, please do the following before panicking:

Make sure you have an active SSH session to your jump box. None of your tunnels will work without an active SSH session to your jump box.

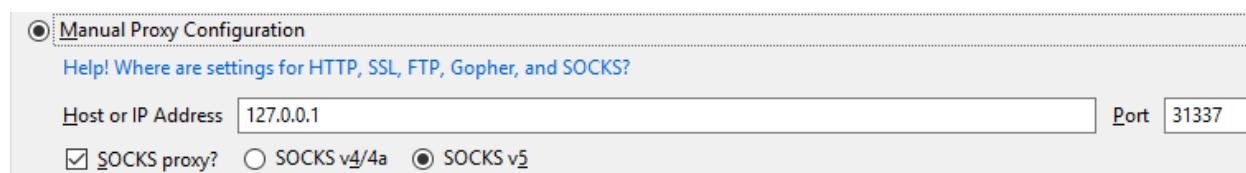
```
ayy@jumpbox:~$
```

While you are connected, on your management workstation, run `netstat -ano | findstr 127.0.0.1:31337` | `findstr LISTEN` (Windows) or `netstat -anp tcp | grep 127.0.0.1.31337` | `grep LISTEN` on OSX/BSD or `netstat -tln | grep 127.0.0.1:31337` | `grep LISTEN` on Linux. Note that if you specified another TCP port, replace 31337 with the TCP port you used to make your dynamic tunnel instead. This ensures the dynamic tunnel is up and ready to accept connections from your management workstation.

```
C:\Users\ayy_lmao>netstat -ano |findstr 127.0.0.1:31337 | findstr LISTEN
      TCP      127.0.0.1:31337          0.0.0.0:0          LISTENING      5304
```

```
Tony's-MacBook-Pro:~ trobinson$ netstat -anp tcp | grep 127.0.0.1.31337
      tcp4       0       0  127.0.0.1.31337        *.*          LISTEN
```

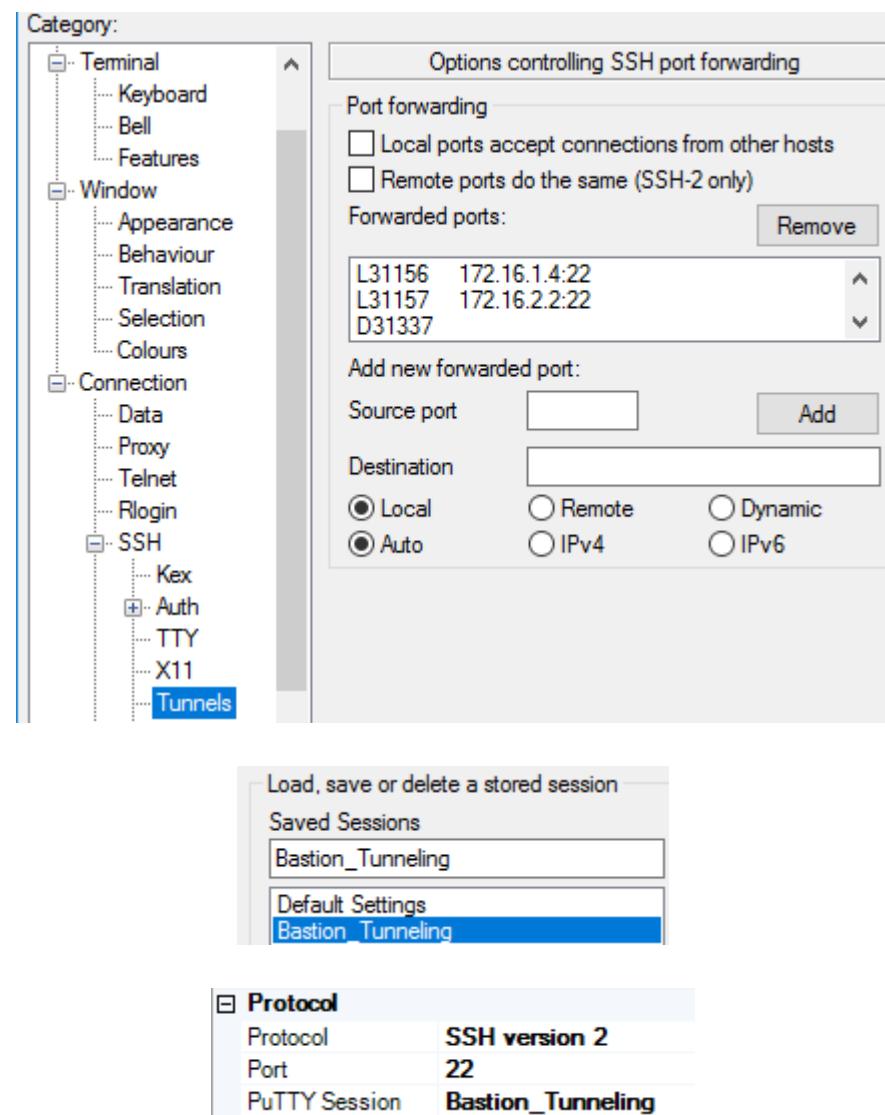
Verify that you created the FoxyProxy session successfully. If you had to use a different port for your Dynamic SSH tunnel other than port 31337, then you should specify that port and the IP address “127.0.0.1” for your FoxyProxy session. Make sure that the “SOCKS proxy?” checkbox is checked as well.



Verify that FoxyProxy is configured to “Use proxy “127.0.0.1:31337” for all URLs.” You have to be using the profile actively for FoxyProxy to send traffic through it.



If you’re on Windows, Verify that you created your PuTTY Session with a dynamic tunnel to port 31337, or the alternate TCP port you specified for your dynamic tunnel (e.g. D31337 under the SSH > Tunnels Category), that you SAVED the PuTTY Session, and that you APPLIED that saved PuTTY session to your mremoteng session to your jump box system.



If you’re on Linux/OSX/BSD, verify that your alias was created properly (see the alias in the illustration below, changing the TCP port 31337 as necessary) and added to `~/.bashrc` or `~/.bash_profile`, ensure that you sourced `~/.bashrc` or `~/.bash_profile` properly. Finally try connecting to your jump box via SSH again, using your alias.

```
Tony's-MacBook-Pro:~ trobinston$ vi ~/.bash_profile
Tony's-MacBook-Pro:~ trobinston$ cat ~/.bash_profile | grep jumpbox
alias jumpbox='ssh -D31337 -L31155:172.16.1.3:22 -L31156:172.16.1.4:22 -L31157:172.16.2.2:22 ayy@192.168.1.8'
Tony's-MacBook-Pro:~ trobinston$ source ~/.bash_profile
Tony's-MacBook-Pro:~ trobinston$ jumpbox
Enter passphrase for key '/Users/trobinston/.ssh/id_rsa':
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-57-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.

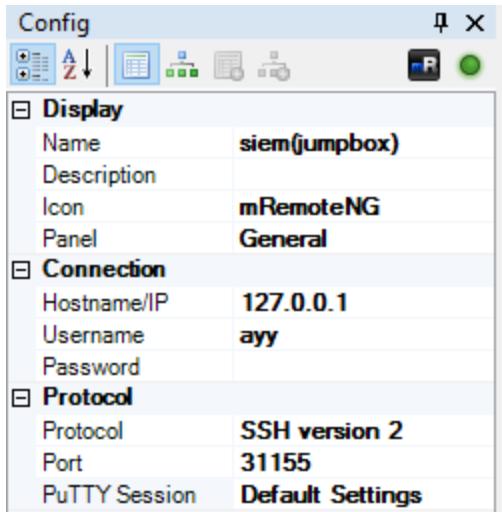
Last login: Mon Jan  9 15:16:24 2017 from 192.168.1.3
ayy@jumpbox:~$
```

Testing Your Forward Tunnels

In order to test whether or not the forward tunnels are working and able to forward SSH requests to your lab VMs, obviously, you will have had to create the lab VMs, given them the correct IP addresses, and have configured the SSH service to be listening. Additionally, you need to have an active SSH session to your jump box, with your forward listeners configured. Let's assume that I have three forward TCP listeners set up on my management workstation; 31155 forwards to 172.16.1.3:22 (SSH on the SIEM VM), 31156 forwards to 172.16.1.4:22 (SSH on the IPS VM), and 31157 forwards to 172.16.2.2:22 (SSH on the kali VM, assuming you [enabled SSH on the kali VM](#)).

Windows

If you're using Windows, Create a new mremoteng connection profile. Name the profile "siem(jumpbox)". Make it SSH version 2. Enter 127.0.0.1 as the IP address, utilize the username you configured when you first created the siem VM. Under the protocol section, change the TCP port number to 31155 (or the TCP port you configured for your forward connect to 172.16.1.3:22). For now, you can set the PuTTY Session to "Default Settings".



With your SSH session to your jump box (With your tunnels enabled) still active, double click on your new connection, and try to connect.

```
Using username "ayy".
ayy@127.0.0.1's password:
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-57-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.

*** System restart required ***
Last login: Tue Jan 10 19:03:42 2017 from 192.168.1.8
ayy@siem:~$ hostname -f
siem.local
ayy@siem:~$ ifconfig -a
ens160    Link encap:Ethernet HWaddr 00:0c:29:b8:2e:24
          inet addr:172.16.1.3 Bcast:172.16.1.255 Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:feb8:2e24/64 Scope:Link
            UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
            RX packets:75497 errors:0 dropped:0 overruns:0 frame:0
            TX packets:40493 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1000
            RX bytes:98730752 (98.7 MB)  TX bytes:2835482 (2.8 MB)
```

Repeat this process and create a connection profile for 127.0.0.1 on port 31156 and port 31157 in order to make SSH connections to the IPS and kali VMs respectively.

Display	Display
Name	ips(jumpbox)
Description	
Icon	mRemoteNG
Panel	General
Connection	Connection
Hostname/IP	127.0.0.1
Username	ayy
Password	
Protocol	Protocol
Protocol	SSH version 2
Port	31156
PuTTY Session	Default Settings
Display	Display
Name	kali(jumpbox)
Description	
Icon	mRemoteNG
Panel	General
Connection	Connection
Hostname/IP	127.0.0.1
Username	root
Password	
Protocol	Protocol
Protocol	SSH version 2
Port	31157
PuTTY Session	Default Settings

```
Using username "ayy".
ayy@127.0.0.1's password:
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-57-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.

*** System restart required ***
Last login: Wed Jan 11 11:07:47 2017 from 192.168.1.8
ayy@ips:~$ hostname -f
ips.local
ayy@ips:~$ 
```

```
Using username "root".
root@127.0.0.1's password:

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.
Last login: Wed Jan 11 11:21:07 2017 from 192.168.1.8
root@kali:~# hostname -f
kali.local
root@kali:~# 
```

Linux/OSX/BSD

With your SSH session to your jump box (With your tunnels enabled) still active, you'll need to open another terminal session and run the command
`ssh -p 31155 ayy@127.0.0.1`

```
Tonys-MBP:~ trobinson$ ssh -p 31155 ayy@127.0.0.1
ayy@127.0.0.1's password:
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-59-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.

Last login: Tue Jan 24 09:58:28 2017 from 192.168.1.8
ayy@siem:~$ hostname -f
siem.local
ayy@siem:~$
```

After you confirm that you are able to log in, repeat the process and connect to port 31156 (ssh -p 31156 ayy@127.0.0.1) and port 31157 (ssh -p 31157 ayy@127.0.0.1) to verify connectivity to the ips and kali VMs respectively.

```
Tony's-MBP:~ trobinson$ ssh -p 31156 ayy@127.0.0.1
ayy@127.0.0.1's password:
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-59-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.
```

```
Last login: Tue Jan 24 09:58:28 2017 from 192.168.1.8
ayy@ips:~$ hostname -f
ips.local
ayy@ips:~$ █
```

```
Tony's-MBP:~ trobinson$ ssh -p 31157 root@127.0.0.1
root@127.0.0.1's password:
```

```
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.
```

```
Last login: Tue Jan 24 10:15:05 2017 from 192.168.1.8
root@kali:~# hostname -f
kali.local
root@kali:~# █
```

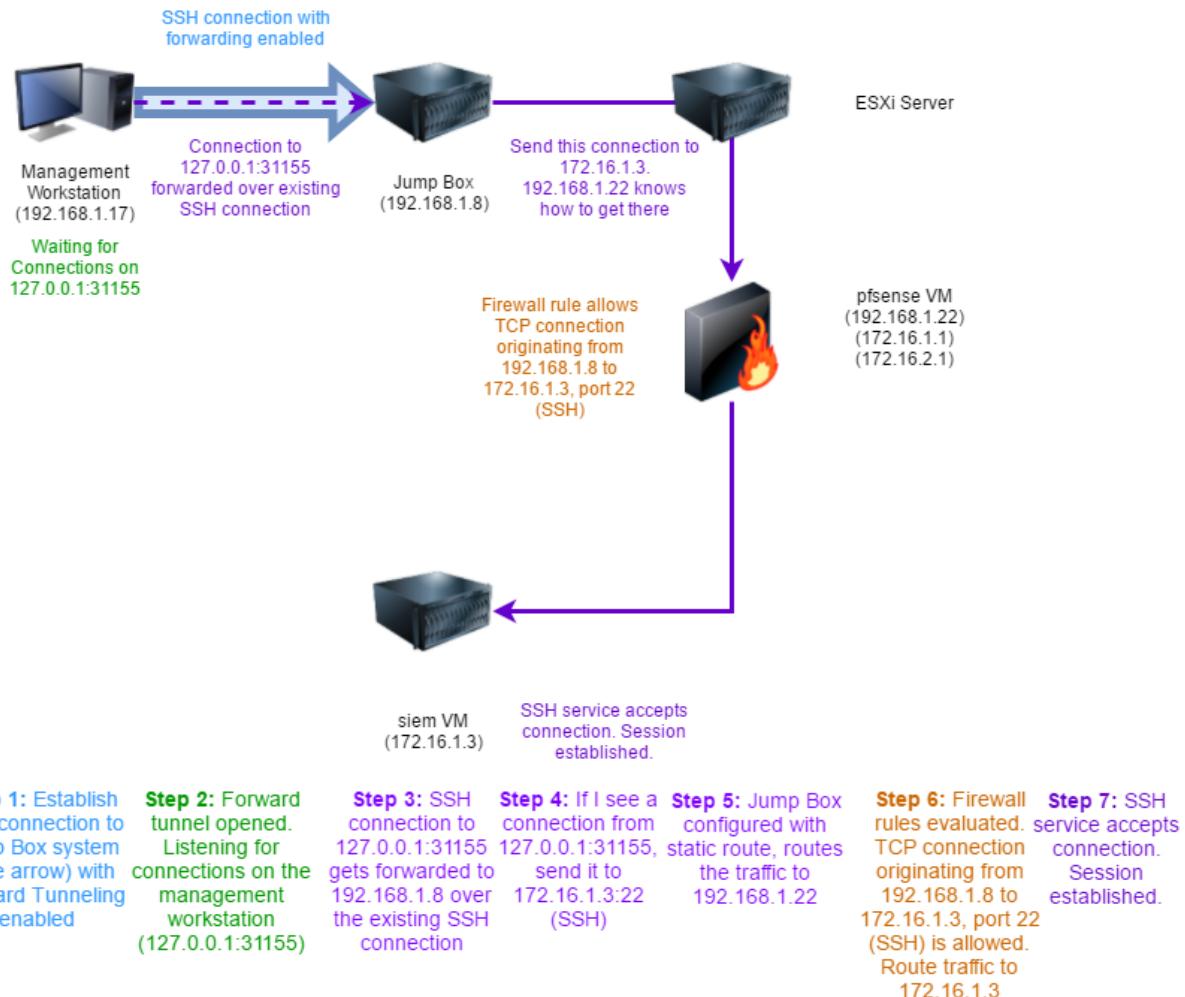
While you're on your OSX/BSD/Linux system, consider adding aliases to make connecting to your jumpbox, and VMs tunneled through the jumpbox easier.

```
alias jumpbox='ssh -p 19342 -D31337 -L31155:172.16.1.3:22 -L31156:172.16.1.4:22
-L31157:172.16.2.2:22 ayy@192.168.1.8'
alias siem='ssh -p 31155 ayy@127.0.0.1'
alias ips='ssh -p 31156 ayy@127.0.0.1'
alias kali='ssh -p 31157 root@127.0.0.1'
```

What? How?

I just connected to 127.0.0.1, how am I connected to the siem VM? SSH forward tunnels are dark magic.

You connected to the jump box system at 192.168.1.8 with your forward and dynamic tunnels configured. These tunnels open listening network ports (in our case, TCP port 31155) on your management workstation, specifically on its loopback IP address, 127.0.0.1. We then create an SSH connection to 127.0.0.1 on port 31155. The listener accepts our connection, and forwards our TCP data over our already existing SSH connection to the jump box on the other end. The jump box then forwards the connection to 172.16.1.3 on TCP port 22 (SSH). This makes it appear as though the connection is originating from our jump box at 192.168.1.8. Since the jump box has static routes to the 172.16.1.0/24 network via 192.168.1.22, and we have a firewall rule that allows 192.168.1.8 to connect to 172.16.1.3 on port 22/TCP, the connection is forwarded to the siem VM and allowed, granting us a successful connection.



Closing Note on Jump Boxing

Lets talk about key-based authentication, and managing your SSH keys with jump boxes/bastion hosts.

Key-Based Authentication

Just like with hosted hypervisors, you can configure key-based authentication for accessing the jumpbox, or your tunneled SSH sessions to the lab VMs. Key-based authentication with SSH tunnels works just as well regardless of whether you are using Windows (PuTTY/mRemoteNG) or Linux/OSX/BSD.

```
Using username "root".
Authenticating with public key "rsa-key-20140415"

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.
Last login: Tue Jan 24 11:20:14 2017 from 192.168.1.8
root@kali:~# less .ssh/authorized_keys
root@kali:~# █
```

```
Tony's-MBP:~ trobinson$ ssh -p 31157 root@127.0.0.1
Enter passphrase for key '/Users/trobinson/.ssh/id_rsa':
```

```
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.
```

```
Last login: Tue Jan 24 10:15:15 2017 from 192.168.1.8
```

```
root@kali:~# █
```

If you want to establish key-based authentication , simply follow the [Windows Remote Access](#) or [Linux/OSX/BSD](#) Remote Access guides, specifically, the portions on ssh key generation, and copying your SSH keys to the jumpbox and lab VMs.

Note: Depending on whether or not you value security or redundancy, you may wish to generate an SSH key on your jump box, and copy the public key of your jump box system to the kali, siem, and ips VMs as well. There are those that say it is good practice to NOT generate or store SSH keys on a system that is meant to act as a jump box, and instead argue that tunnel your SSH connections and leaving/using SSH keys on your local workstation is recommended to reduce the possibility of attackers gaining access to poorly secured SSH keys on a jump box/bastion host system. On the other hand, have the ability to directly SSH to your lab VMs from an SSH session on your jump box system provides some degree of redundancy in the event you can't get SSH tunnels or TCP forwarding to work as expected. A compromise between these two would be to password protect any SSH keys you generate and use that will be stored on the jump box system.

In addition to enabling key-based authentication, you can also choose to [disable password authentication](#) to the jumpbox and lab VMs (which I very highly recommend), and/or [enable key-based authentication as root](#) for both the jumpbox and lab VMs, if you're so inclined. Typically, I don't enable key-based authentication as root on the jump box system because TCP forwarding (what allows us to make tunnels to the lab VMs) doesn't require root privileges.

IPS Installation Guide

At this point you should have a mostly functional lab environment. You should have firewall rules configured, and 5 mostly functional Virtual Machines created. This section pertains primarily to the 'ips' VM. Before getting started, I'm going to recommend completing the [Remote Lab Management](#) section. Enabling SSH and SCP access for the ips sensor will make system management and configuration much easier. Additionally, your hypervisor network and/or the ips VM (depending on the hypervisor you chose), **must be configured to allow promiscuous mode**, and in some cases (Client Hyper-V, ESXi) MAC spoofing/forged transmits. If you followed the instructions laid out for your hypervisor of choice, you should be all set; I went into painstaking detail to make sure these details got covered.

Currently the most popular rule/signature based IDS/IPS solutions are Snort, (<http://www.snort.org>) and Suricata (<http://www.suricata-ids.org>). To ensure I'm being fair (in spite of having used Snort most of my career) I'm going to instruct you on how to install both solutions. Snort or Suricata can be used to supply an AF_PACKET, fail-close bridge for your lab network. Both systems should operate nearly the same, the only difference is in the rules one solution provides by default as compared to the other.

Installing and configuring Snort (via Autosnort)

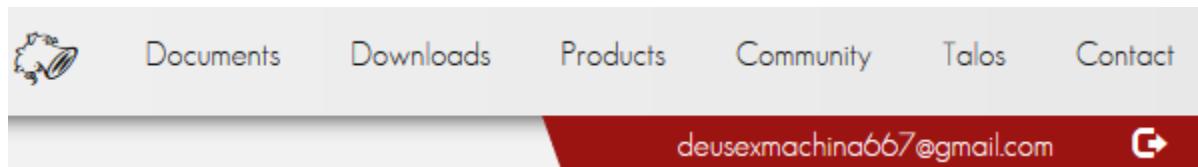
I have a small github project that I have worked on over the years called "Autosnort" (I'm not one for particularly creative names). The scripts (I had multiple versions, for different Linux distributions) were designed to install the prerequisites required to compile and run Snort including the DAQ - Data AcQuisition libraries, the pulledpork rule manager, the alert spooler

Barnyard2, as well as configuring an alert interface of the user's choice (usually a web-based interface with a database backend that was fed by barnyard2, but the scripts also support SGUIL, as well as directly SYSLOG output).

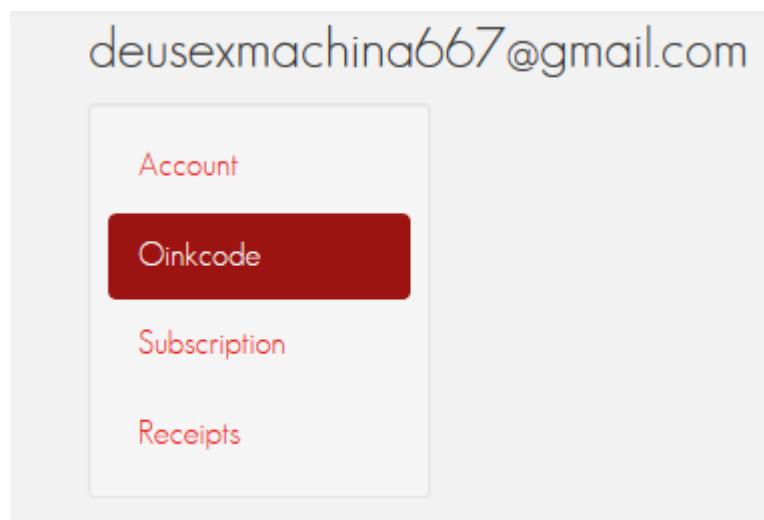
I made a special edition version of Autosnort for this book. It implements some important changes compared to the standard script. First and foremost, it's very stripped down. The script only installs Snort and pulledpork. Additionally, the script assumes that we will be running our sensor in inline mode, using AFPACKET which is exactly what we want for passing traffic between the IPS 1 and IPS 2 networks. The script does NOT install barnyard2 or any kind of an output interface for viewing alerts, because in the next section, we will be detailing how to setup Splunk and forward our IPS logs to splunk for review.

To get started, first, I'm going to have you visit snort.org. Register for an account on this site. If you don't want to use your real email address, you can probably use 10-minute mail or another throwaway email account to register. Be sure to save the login information.

When you log in, click on your email address in the upper right corner. This will bring you to your account settings page.



On the left side of the page under your e-mail address, click on oink code, and a new page will open, displaying the oink code in the center of the screen. Copy the string of letters and numbers; you'll need it momentarily.



When you have your oinkcode, log in to your ips VM as root (over SSH if you did the Remote Lab Management configuration exercises) or log in and run “sudo su -” and put in your user’s password, to get a root shell. This script mucks with a lot of system settings, so **running the script as root is a requirement**.

Once you are logged in as root, Use the command

```
git clone https://github.com/da667/Autosnort
```

If for some reason, you do not have the git command available (it should have been installed by default), run “apt-get -y install git” to install it.

```
root@ips:~# git clone https://github.com/da667/Autosnort
Cloning into 'Autosnort'...
remote: Counting objects: 767, done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 767 (delta 0), reused 0 (delta 0), pack-reused 760
Receiving objects: 100% (767/767), 2.20 MiB | 0 bytes/s, done.
Resolving deltas: 100% (418/418), done.
Checking connectivity... done.
root@ips:~# cd Autosnort/Autosnort\ -\ Ubuntu/AVATAR/
root@ips:~/Autosnort/Autosnort - Ubuntu/AVATAR# ls
autosnort-ubuntu-AVATAR.sh  full_autosnort.conf  snortd
root@ips:~/Autosnort/Autosnort - Ubuntu/AVATAR# vi full_autosnort.conf
```

After downloading Autosnort, run “ifconfig -a” identify which interfaces are available for use on your system. Different hypervisors user different names for the network interfaces. In my case, the network interface I used for establishing an SSH connection was “eth0”. I also had eth1, and eth2. Eth0 should be connected to the “Management” network, while eth1, and eth2 should be connected to the IPS1 and IPS2 networks. Check the “HWaddr” field for each of the network cards and compare them with your notes for the ips sensor to confirm which network interface is connected to which virtual network.

```

root@ips:~/Autosnort/Autosnort - Ubuntu/AVATAR# ifconfig -a
eth0      Link encap:Ethernet HWaddr 00:15:5d:01:11:22
          inet addr:172.16.1.4 Bcast:172.16.1.255 Mask:255.255.255.0
          inet6 addr: fe80::215:5dff:fe01:1122/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          RX packets:2247 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1644 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2541744 (2.5 MB) TX bytes:142231 (142.2 KB)

eth1      Link encap:Ethernet HWaddr 00:15:5d:01:11:23
          BROADCAST MULTICAST MTU:1500 Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)

eth2      Link encap:Ethernet HWaddr 00:15:5d:01:11:24
          BROADCAST MULTICAST MTU:1500 Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)

```

Change directories into Autosnort/Autosnort - Ubuntu/AVATAR. In this directory, there should be four files. One of these files is named “full_autosnort.conf”. You’ll need to edit this file and change a couple of settings. Using whatever text editor you are most comfortable with, edit the following settings:

- snort_iface_1: by default, this is set to “eth1”, the second listed network interface. In my case, I didn’t have to change this.
- snort_iface_2: by default, this is set to “eth2”, the third listed network interface. Again, in my case, I did not have to change this.
- o_code: by default this is blank. This line should contain the oinkcode you copied from snort.org earlier.

```

##full_autosnort configuration file##
# The options below are configuration options that n
t is for, why it needs to be set, and valid configur
# After Autosnort completes and you confirm everythi
# But in all seriousness, after the script completes

##snort_basedir##
# This option is sets the directory where you would
# Do not place any trailing slashes (/) at the end o
# See the default setting for a valid example
##Options##
# Any valid unix directory path, Autosnort will crea
#default setting: /opt/snort
snort_basedir=/opt/snort

##snort_iface_1##
# This is the name of the first interface you will b
# This option MUST be set.
# Example:
# snort_iface_1=eth1
#default setting: snort_iface_1=eth1
snort_iface_1=eth1

##snort_iface_2##
# This is the name of the second interface you will
# This option MUST be set.
# Example:
# snort_iface_2=eth2
#default setting: snort_iface_2=eth2
snort_iface_2=eth2

##o_code##
# This setting is the oink code that will be used by
# You MUST input a valid oink code for the script to
# This can be a registered user oink code, or VRT ru
# If you have no idea what an oink code is, or how t
# After registering your account, and logging in und
# Example:
# o_code=2426170067b2e110c1f3fddee444118fcc15180f0
# the above is not a valid oink code; do not use it.
o_code=2426170067b2e110c1f3fddee444118fcc15180f0

```

After you are done, run the command “bash autosnort-ubuntu-AVATAR.sh” to begin running the autosnort installer.

Note: If you installed and configured the SQUID HTTP proxy per the [Network Configuration - Core Network Services](#) guide, you need to run the following commands for Autosnort to work:

```
export HTTP_PROXY=http://172.16.1.1:3128
export http_proxy=http://172.16.1.1:3128
```

```
export HTTPS_PROXY=
export https_proxy=
```

These commands set the http_proxy, HTTP_PROXY, HTTPS_PROXY, and https_proxy variables, which are required for command line utilities to be aware of the fact that a proxy is available and required to gain external network access. If you don't set these variables, you may notice that the autosnort script hangs at the phase where it attempts to download the latest version of Snort from snort.org, or that the portion of the script that uses pulledpork to download rules for snort fails entirely.

The script keeps you notified as to what tasks it is running at any given time. Please be patient and let it run its course. When the script is finished running, your VM will reboot.

```
root@ips:~/Autosnort/Autosnort - Ubuntu/AVATAR# bash autosnort-ubuntu-AVATAR.sh
[*] Checking for config file...
[*] Found config file.
[*] Checking for root privs...
[*] We are root.
[*] Performing apt-get update and upgrade (May take a while if this is a fresh install)...
[*] System updates successfully completed.
[*] Installing base packages: libdumbnet-dev ethtool build-essential libpcap0.8-dev libpcap3-dev bison flex autoconf libtool libarchive-tar-perl libcrypt-ssleay-perl libwww-perl...
```

Note: If you installed and configured the SQUID HTTP proxy, you need to run the following commands for autosnort to work: export http_proxy=http://172.16.1.1:3128 export https_proxy=""

If for some reason, the script fails to run, the log file is located in /var/log/autosnort_install.log and should contain clues behind the failure. After identifying the fix, you can try to run the script again, or revert the VM to a previous snapshot, apply the fix, snapshot the VM again, then attempt to run the script again. Upon completion, the script will reboot the virtual machine. Log back into the ips VM and run the command "ps -ef | grep snort"

```
root@ips:~# ps -ef | grep snort
snort      1124      1  0 21:06 ?        00:00:00 /opt/snort/bin/snort -D -u snort -g snort -c /opt/snort/etc/snort.conf -Q --daq afpacket --daq-mode inline -i eth1:eth2
```

The output above shows that snort is running out of /opt/snort/bin with the -D (daemonize), -u snort, -g snort (user and group to drop privileges to), -c /opt/snort/etc/snort.conf (points to the configuration file to use), -Q (inline mode) --daq afpacket (tells snort what DAQ module to use) -daq-mode inline (tells snort to run our DAQ module with support for inline mode) -i eth1:eth2 (the interface pair to use for running snort inline). If your output looks like this, then snort should be running and passing traffic just fine. If you need to restart/reload snort for any reason (e.g. adding/removing rules, changing configuration options, etc), the init script "snortd" exists and supports the stop, start, restart, and status options for managing the snort process. If you want to confirm that snort is running properly, proceed to the "Testing your IPS bridge" section.

Installing and configuring Suricata (via Autosuricata)

Not unlike with Snort, I also made a script that prepares Suricata in a similar matter named "Autosuricata" (Again, I'm not terribly creative with names). Just like with Autosnort, it needs to be ran as root, and installs a very stripped down Suricata installation. Unlike Autosnort,

Autosuricata does NOT require you to have an oink code, unless you have an emerging threats pro subscription that you would like to utilize for your lab environment.

Log in to your ips VM as root (over SSH if you did the Remote Lab Management configuration exercises) or log in and run “sudo su -” and put in your user’s password, to get a root shell. This script mucks with a lot of system settings, so **running the script as root is a requirement**.

Once you are logged in as root, Use the command

```
git clone https://github.com/da667/Autosuricata
```

If for some reason, you do not have the git command available (it should have been installed by default), run “apt-get -y install git” to install it.

```
root@ips:~# git clone https://github.com/da667/Autosuricata
Cloning into 'Autosuricata'...
remote: Counting objects: 22, done.
remote: Compressing objects: 100% (13/13), done.
remote: Total 22 (delta 6), reused 22 (delta 6), pack-reused 0
Unpacking objects: 100% (22/22), done.
Checking connectivity... done.
root@ips:~# cd Autosuricata/
AutoSuricata - Deb/.git/
root@ips:~# cd Autosuricata/AutoSuricata\ -\ Deb/AVATAR/
```

After downloading Autosuricata, run “ifconfig -a” identify which interfaces are available for use on your system. Different hypervisors user different names for the network interfaces. In my case, the network interface I used for establishing an SSH connection was “eth0”. I also had eth1, and eth2. Eth0 should be connected to the “Management” network, while eth1, and eth2 should be connected to the IPS1 and IPS2 networks. Check the “HWaddr” field for each of the network cards and compare them with your notes for the ips sensor to confirm which network interface is connected to which virtual network.

```

root@ips:~/Autosnort/Autosnort - Ubuntu/AVATAR# ifconfig -a
eth0      Link encap:Ethernet HWaddr 00:15:5d:01:11:22
          inet addr:172.16.1.4 Bcast:172.16.1.255 Mask:255.255.255.0
          inet6 addr: fe80::215:5dff:fe01:1122/64 Scope:Link
            UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
            RX packets:2247 errors:0 dropped:0 overruns:0 frame:0
            TX packets:1644 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1000
            RX bytes:2541744 (2.5 MB) TX bytes:142231 (142.2 KB)

eth1      Link encap:Ethernet HWaddr 00:15:5d:01:11:23
          BROADCAST MULTICAST MTU:1500 Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)

eth2      Link encap:Ethernet HWaddr 00:15:5d:01:11:24
          BROADCAST MULTICAST MTU:1500 Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)

```

Change directories into Autosuricata/AutoSuricata - Deb/AVATAR. In this directory, there should be five files. One of these files is named “full_autosuricata.conf”. You’ll need to edit this file and change a couple of settings. Using whatever text editor you are most comfortable with, edit the following settings:

- suricata_iface_1: by default, this is set to “eth1”, the second listed network interface. In my case, I didn’t have to change this.
- suricata_iface_2: by default, this is set to “eth2”, the third listed network interface. Again, in my case, I did not have to change this.
- o_code: by default this is blank. This field is optional. You only need to enter a valid oink code if you are using the Emerging Threats Pro ruleset.

```
#full_autosuricata configuration file##
# The options below are configuration options
# that is for, why it needs to be set, and valid
# After Autosuricata completes and you can
# But in all seriousness, after the script

##suricata_iface_1##
# This is the name of the first interface
# This option MUST be set.
# Example:
# suricata_iface_1=eth1
#default setting: suricata_iface_1=eth1
suricata_iface_1=eth1

##suricata_iface_2##
# This is the name of the second interface
# This option MUST be set.
# Example:
# suricata_iface_2=eth2
#default setting: suricata_iface_2=eth2
suricata_iface_2=eth2

##o_code##
# This setting is the oink code that will
# You only need to fill out this option if
# Example:
# o_code=2426170067b2e110c1f3fdee444118fc0
# the above is not a valid oink code; do not
# use it
o_code=
```

After you are done, run the command “bash autosuricata-deb-AVATAR.sh” to begin running the autosnort installer.

Note: If you installed and configured the SQUID HTTP proxy per the [Network Configuration - Core Network Services](#) guide, you need to run the following commands for Autosnort to work:

```
export HTTP_PROXY=http://172.16.1.1:3128
export http_proxy=http://172.16.1.1:3128
export HTTPS_PROXY=
export https_proxy=
```

These commands set the http_proxy, HTTP_PROXY, HTTPS_PROXY, and https_proxy variables, which are required for command line utilities to be aware of the fact that a proxy is available and required to gain external network access. If you don't set these variables, you may notice that the script hangs and fails when it is attempt to download suricata to compile it.

The script keeps you notified as to what tasks it is running at any given time. Please be patient and let it run its course. When the script is finished running, your VM will reboot.

```
root@ips:~/Autosuricata/AutoSuricata - Deb/AVATAR# bash autosuricata-deb-AVATAR.sh
[*] Checking for config file..
[*] Found config file.
[*] Checking for root privs..
[*] We are root.
[*] Performing apt-get update and upgrade (May take a while if this is a fresh install)...
```

If for some reason, the script fails to run, the log file is located in /var/log/autosuricata_install.log and should contain clues behind the failure. After identifying the fix, you can try to run the script again, or revert the VM to a previous snapshot, apply the fix, snapshot the VM again, then attempt to run the script again. Upon completion, the script will reboot the virtual machine. Log back into the ips VM and run the command “ps -ef | grep suricata”

```
root@ips:~# ps -ef | grep suricata
root      1128      1  1 21:44 ?        00:00:07 /usr/local/bin/suricata -D -c /usr/local/etc/suricata/suricata.yaml --af-packet
```

The output above shows that suricata is running out of /usr/local/bin with the -D (daemonize), -c /usr/local/etc/suricata/suricata.yaml (points to the configuration file to use), --af-packet (instructs Suricata to operate in AFPACKET mode. Since we created an af-packet.yaml file in /usr/local/etc/suricata/af-packet.yaml, and the suricata.yaml uses an “include:” statement to use this yaml file, suricata knows to use af-packet bridging between eth1 and eth2 via the full_autosuricata.conf file). If your output looks like this, then suricata should be running and passing traffic just fine. If you need to restart/reload suricata for any reason (e.g. adding/removing rules, changing configuration options, etc), the init script “suricatad” exists and supports the stop, start, restart, and status options for managing the suricata process. If you want to confirm that suricata is running properly, proceed to the “Testing your IPS bridge” section.

Testing your IPS Bridge

Power on your kali VM and your metasploitable 2 VMs. Remember that kali is located on the ips1 network, while metasploitable 2 is connected to the ips2 network. If you recall correctly. Kali should have an IP address of 172.16.2.2, and Metasploitable 3 (which hasn’t been connected to the network yet) should have a static DHCP mapping to 172.16.2.3. Log in to the metasploitable virtual machine via the default username and password of “msfadmin”. Then run:

```
ping -c 4 172.16.2.2
```

```
msfadmin@metasploitable:~$ ping -c 4 172.16.2.2
PING 172.16.2.2 (172.16.2.2) 56(84) bytes of data.
64 bytes from 172.16.2.2: icmp_seq=1 ttl=64 time=8.66 ms
64 bytes from 172.16.2.2: icmp_seq=2 ttl=64 time=1.19 ms
64 bytes from 172.16.2.2: icmp_seq=3 ttl=64 time=8.37 ms
64 bytes from 172.16.2.2: icmp_seq=4 ttl=64 time=0.760 ms
```

If you get results similar to the output above, then that means the metasploitable VM on IPS 2 is able to reach the kali VM on the IPS 1 network. Next, log in to the kali VM and try the following:

```
ping -c 4 172.16.2.3
curl 172.16.2.3
```

If you get output similar to the illustration above, then congratulations, your AF_PACKET bridge, virtual networking, promiscuous mode (and/or forged transmit settings) are all correct, and the kali linux VM on the IPS 1 network can reach metasploitable 2 on the IPS 2 network. **At this point, I would very highly recommend creating another snapshot of the IPS VM**, since it is now in a known good working configuration.

Splunk Installation Guide

Splunk is going to be the SIEM or Security Intrusion Events Manager we'll be installing on the siem VM for our lab. While there are other choices for open-source SIEMs (e.g. ELK -- Elastic Search, Logstash, Kibana), Splunk is dead-easy to set up, has extremely robust extension/app support, a large user community, and will more than fit the bill for our small lab environment.

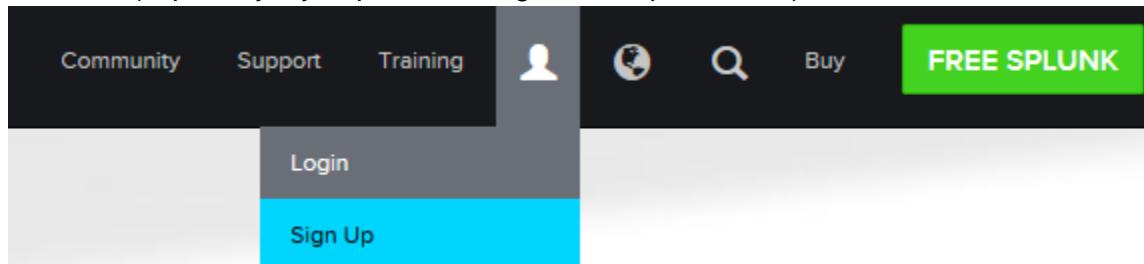
Splunk isn't truly a SIEM in and of itself, but a log aggregation system that, through a huge log parsing ecosystem has been beaten into submission

The one concern most critics of Splunk have are the limitations on the amount of data you can log/use every day. By default, without any licensing, you can collect up to 500mb of logs per day. Anything beyond that requires a license, and commercial licensing for splunk tends to be very pricey. However, Splunk has a dev license program. For absolutely no cost to you, you can request access to a development license that allows you up to 10GB of data per day, and the license is good for one year. However it is VERY important to mention that this license is meant for training, development of tools, and personal skill development and should NOT be used in a production environment due to possible legal and ethical concerns. The bottom line is that you should make use of the development license for training if necessary, but **do not abuse the privilege.**

Initial setup (server installation)

Depending on how large of an installation you have, Splunk can consist of several components such as the "Search Head" (the user interface used to query collected logs), Multiple Indexers (responsible for organizing and storing collected logs), and a variety of forwarders (Universal Forwarders, Heavy Forwarders, Syslog servers, etc. used to collect logs off the systems we wish to monitor). Since we have a small lab, our configuration will consist of one server installation where both the index and search head will be installed (our siem VM), and a single universal forwarder (our ips VM) with either the Splunk or Suricata applications configured so we can retrieve IDS events from the search head.

Start by visiting www.splunk.com. You'll want to hover over the icon that looks like a person and select "Sign Up". If you're paranoid, you can choose to fill out fake information for most of the fields on the page, however you need to associate your account with a somewhat permanent e-mail address (especially if you plan on using a developer license).



Create Your Account

Already have a Splunk.com account? [Log in here.](#)

First Name	<input type="text"/>
Last Name	<input type="text"/>
Email Address	<input type="text"/> Required for validation
Company	<input type="text"/>
Country	<input type="text" value="United States"/>
State	<input type="text" value="Please choose a state..."/>
Zip/Postal Code	<input type="text"/>
Phone	<input type="text"/> Numbers only, no special characters
Username	<input type="text"/>
Password	<input type="text"/> Must be at least 8 characters
Confirm Password	<input type="text"/>

I have read, understood and hereby accept the [Splunk Websites Terms and Conditions of Use](#) and [Splunk Privacy Policy](#).

[Create Your Account](#)

Once you have signed up and logged in, you'll want to navigate to Products > Core Products > Splunk Enterprise. Click the free download button.

Splunk® Enterprise

See the forest and the trees

Splunk Enterprise makes it simple to collect, analyze and act upon the untapped value of the big data generated by your technology infrastructure, security systems and business applications—giving you the insights to drive operational performance and business results.

[Free Download](#)

[Try our Free Cloud Trial](#)

On the next page, you'll see a button to click for Linux.

Download Splunk Enterprise

Splunk Enterprise is the leading platform for real-time operational intelligence. When you download Splunk Enterprise for free, you get a Splunk Enterprise license for 60 days that lets you index up to 500 megabytes of data per day.

When the free trial ends, you can convert to a perpetual Free license or purchase an Enterprise license to continue using the expanded functionality designed for multi-user deployments.

- ✓ Troubleshoot application problems and investigate security incidents in minutes
- ✓ Avoid service degradation or outages
- ✓ Deliver compliance at lower cost
- ✓ Gain new business insights

[Windows](#)

[Linux](#)

[Solaris](#)

[Mac OS](#)

If you click on it, a pop-up will appear asking you to select which file to download. As of right now, the current version of splunk enterprise is version 6.5.0. You'll want to click on the file that ends in ".deb"

Download Splunk Enterprise For Linux

OS version

2.6+ kernel Linux distributions (64-bit)

[Release Notes](#)

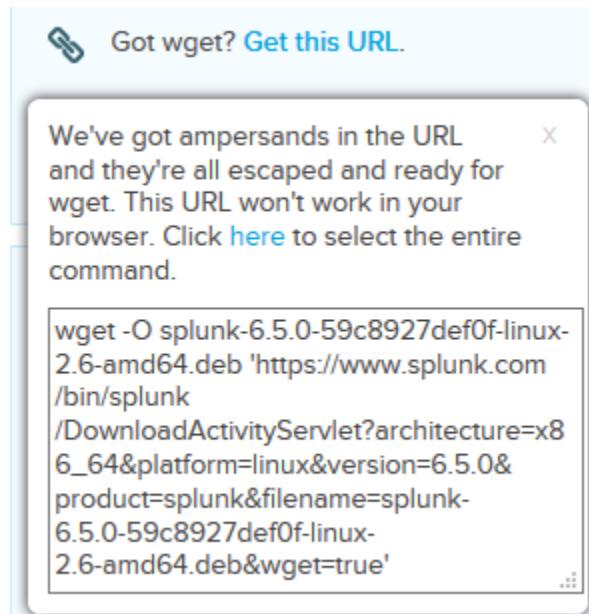
[Splunk-6.5.0-59c8927def0f-linux-2.6-x86_64.rpm](#)

[Splunk-6.5.0-59c8927def0f-linux-2.6-amd64.deb](#)

[Splunk-6.5.0-59c8927def0f-Linux-x86_64.tgz](#)

Click on the ".deb" file sends you to a splash page that thanks you for downloading splunk. Your web browser will open a dialogue box asking you where you want to download this file. If you choose to download the file in this manner, you will need to have SCP access to copy the file from the system you are browsing the splunk site with to your siem VM. Otherwise, click cancel on the download file dialogue. On the splash page thanking you for download splunk, look for

the text that says “Got wget? Get this URL”. If you click on it, a text box pops up with a wget command you can copy to an SSH session on the siem VM for it to download the file directly from splunk.com, either that, or you can copy and paste the entire command into a text editor application and copy the wget command into the siem VM’s console.



Log into the siem VM as root, or use “sudo su -” and enter your user’s password to become root. Copy the wget command you got from splunk.com (don’t worry if you’re downloading a newer version of Splunk), and enter it into the console or your SSH session.

```
root@siem:~# wget -O splunk-6.5.0-59c8927def0f-linux-2.6-amd64.deb 'https://www.splunk.com/bin/splunk/DownloadActivityServlet?architecture=x86_64&platform=linux&version=6.5.0&product=splunk&filename=splunk-6.5.0-59c8927def0f-linux-2.6-amd64.deb&wget=true'
--2016-11-07 13:25:55-- https://www.splunk.com/bin/splunk/DownloadActivityServlet?architecture=x86_64&platform=linux&version=6.5.0&product=splunk&filename=splunk-6.5.0-59c8927def0f-linux-2.6-amd64.deb&wget=true
Resolving www.splunk.com (www.splunk.com) ... 52.85.142.160, 52.85.142.23, 52.85.142.109
,
...
Connecting to www.splunk.com (www.splunk.com) |52.85.142.160|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://download.splunk.com/products/splunk/releases/6.5.0/linux/splunk-6.5.0-59c8927def0f-linux-2.6-amd64.deb [following]
--2016-11-07 13:25:57-- https://download.splunk.com/products/splunk/releases/6.5.0/linux/splunk-6.5.0-59c8927def0f-linux-2.6-amd64.deb
Resolving download.splunk.com (download.splunk.com) ... 52.85.142.151, 52.85.142.68, 52.85.142.102, ...
Connecting to download.splunk.com (download.splunk.com) |52.85.142.151|:443... connected
.
HTTP request sent, awaiting response... 200 OK
Length: 221512580 (211M) [application/octet-stream]
Saving to: 'splunk-6.5.0-59c8927def0f-linux-2.6-amd64.deb'

-splunk-6.5.0-59c8927def0f-linux-2.6-amd64.deb  24%[=====>]   50.75M  9.78MB/s    eta 27s
```

When the download is completed, you should have a file in root's directory that starts with "splunk" and ends in ".deb". In my case, the file was named "splunk-6.5.0-59c8927def0f-linux-2.6-amd64.deb". Now we have to use dpkg, the Debian (and Ubuntu) package manager to install this .deb software package. The command

```
dpkg -i [filename]
```

tells dpkg to try and install the .deb file you specify. In my case, the dpkg command would look like

```
dpkg -i splunk-6.5.0-59c8927def0f-linux-2.6-amd64.deb
```

```
root@siem:~# dpkg -i splunk-6.5.0-59c8927def0f-linux-2.6-amd64.deb
Selecting previously unselected package splunk.
(Reading database ... 91636 files and directories currently installed.)
Preparing to unpack splunk-6.5.0-59c8927def0f-linux-2.6-amd64.deb ...
Unpacking splunk (6.5.0) ...
```

You should get the output "complete" and the command prompt should return. At this point, splunk is installed, but now, we need to start it. Splunk gets installed to "/opt/splunk", run the command

```
/opt/splunk/bin/splunk start --accept-license
```

This command runs accepts the splunk usage license and performs all of the initial setup tasks to get splunk up and running. After the command runs, you should receive this notification if everything proceeded normally.

```
If you get stuck, we're here to help.
Look for answers here: http://docs.splunk.com

The Splunk web interface is at http://siem:8000
```

Sage advice to be had here. This guide is meant to get you through the initial setup and confirm that data is coming in to splunk normally. Beyond that however.. You will have to learn how to use splunk. That is where docs.splunk.com comes into place. Below that, the message says the web interface is up and running on port TCP 8000. If you are running a hosted hypervisor, your hypervisor host should be able to directly connect to http://172.16.1.3:8000. If you are using a baremetal hypervisor, you may need to create firewall rules on pfSense to allow your management workstation to connect to the splunk web interface on the siem VM. Before we do that however, there is one more thing to do on the console/SSH session before we move on. Run the commands

```
chown -R splunk:splunk /opt/splunk
/opt/splunk/bin/splunk enable boot-start -user splunk
```

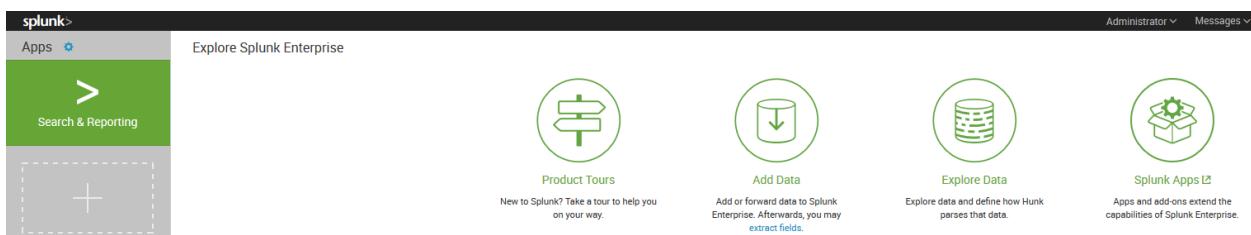
```
root@siem:/opt/splunk# /opt/splunk/bin/splunk enable boot-start -user splunk
Init script installed at /etc/init.d/splunk.
Init script is configured to run at boot.
```

This installs an init script to ensure that the splunk service starts up on system boot as the “splunk” service user, instead of root. In this way, if you have to reboot the system for updates, maintenance, etc. The splunk service should restart on boot automatically.

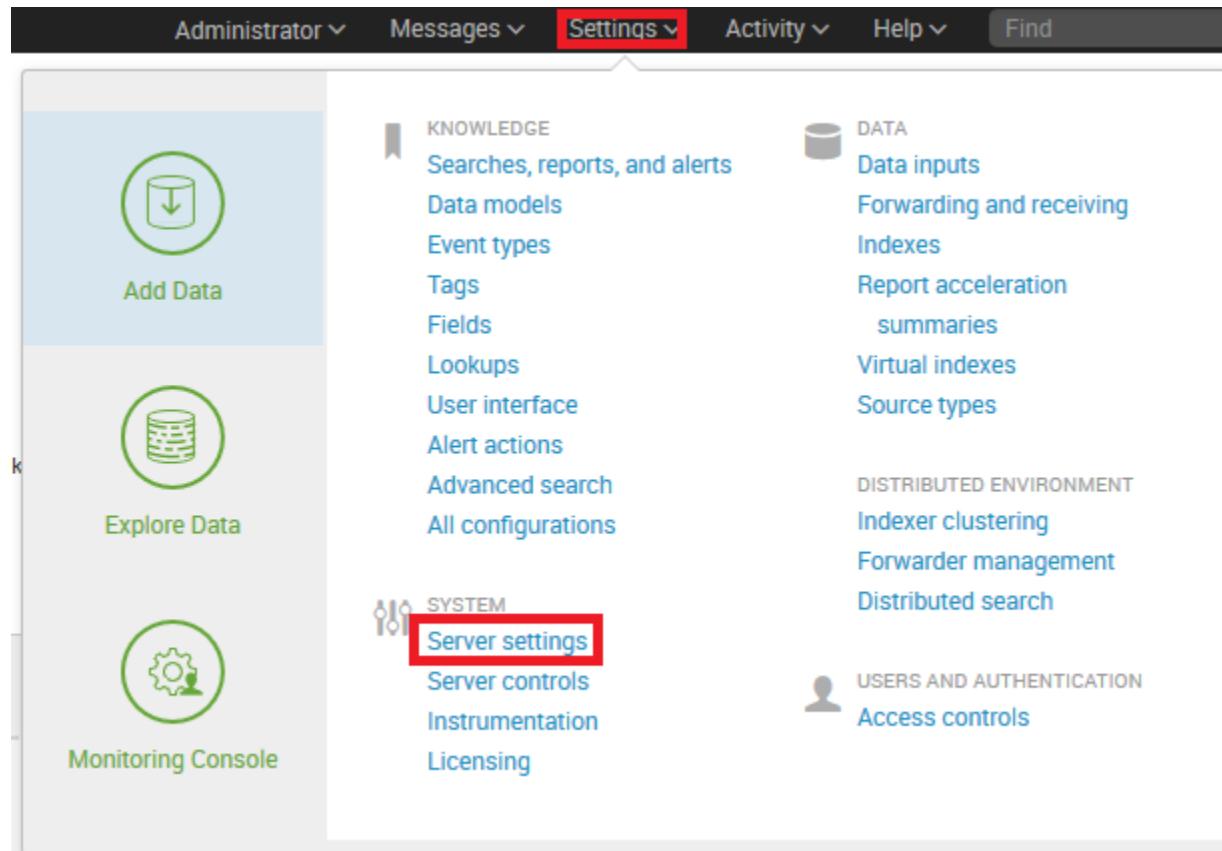
In your favorite web browser, browser to <http://172.16.1.3:8000> and you should be greeted with a login prompt.



As the text implies, since this is our first time logging in, we will use the username “admin” and the password “changeme” to log in. Splunk makes you select a new password for the admin user. Input a password, and keep it somewhere safe. After changing the password, you will be logged in. A small pop-up box will ask you if you wish to submit data to make splunk better, as well as a checkbox that says you agree to send license usage data to splunk. Check both of these boxes; and click ok. Finally, you’re greeted by the default splash page for Splunk.



Congratulations, Splunk is up and running. But before we're done here there is one more change we need to make. In the upper right corner of the screen, click on Settings. A sub-menu appears. In that sub-menu, click on "Server settings".



On the new page that pops up, click on "General settings".

Server settings

Manage system settings including ports, host name, index path, email server, and system logging.

[General settings](#) (highlighted)

[Login background](#)

[Email settings](#)

[Server logging](#)

[Deployment client](#)

[Search preferences](#)

On this page, there are two settings you need to be aware of, the “Splunk Web” and the “Index Settings”. Splunk Web settings control how the web interface operates. Change the setting “Enable SSL (HTTPS) in Splunk Web?” to “Yes”. Below that, be aware that the Index Settings are set to stop splunk of storage space on this system falls below 5GB of free space. You can adjust that if you’d like, but for our purposes, this is fine.

Splunk Web

Run Splunk Web
 Yes No

Enable SSL (HTTPS) in Splunk Web?
 Yes No

Web port *
8000

App server ports
8065

Port number(s) for the python-based application server to listen on. Use comma-separated list to specify more than one port number.

Session timeout *
1h

Set the Splunk Web session timeout. Use the same notation as relative time modifiers, for example 3h, 100s, 6d.

Index settings

Default host name
siem

Sets the host field value for all events coming from this server.

Path to indexes
/opt/splunk/var/lib/splunk

Pause indexing if free disk space (in MB) falls below *

5000

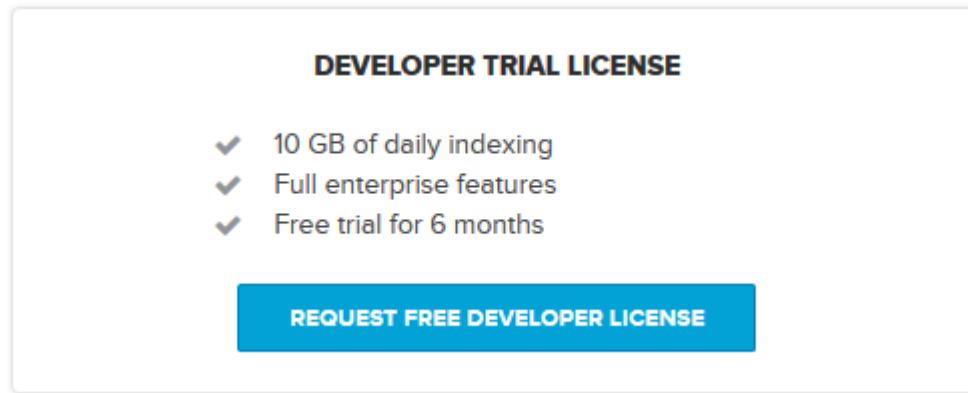
After you’ve enabled SSL access, click the green “Save” button on the bottom of the page. To test things out, exit the splunk web interface and reboot the siem VM. After the reboot is complete, try logging back in to the splunk web interface at <https://172.16.1.3:8000>. If the login attempt fails, you can log into the siem VM console and run “service splunk status” to get a status report usually this includes an error message. The first time I did this, I had the service set to start as the splunk user, and did not grant ownership to /opt/splunk as the splunk user and group. This was resolved by running chown -R splunk:splunk /opt/splunk. On the next reboot, the service started successfully and everything was working fine.

(Optional) Requesting and Implementing a Splunk Dev License

Note: While this section may be optional, I highly recommend following through with this, as the 10GB per day data limit will definitely come in handy, especially if you decide to expand your lab network and/or want to perform log and statistical analysis in the future.

As previously mentioned, by default with Splunk enterprise you get to store up to 500MB of data in Splunk per day. However you can request access to a developer license to increase that limit to 10GB of data per day. Be aware that developer licenses are meant for home or testing environments and are NOT intended for commercial or production use. Frankly I think its awesome that Splunk allows this sort of flexibility, essentially handing these licenses out like candy for anyone that wants to run splunk at home and learn how to run it and/or develop cool tools for it, but if you've ever heard the saying that it only takes one bad apple to spoil the bunch, all it takes is a number of license abuse cases and Splunk may not offer the program any longer. So what I'm getting at here is act ethically and responsibly.

In your favorite web browser, navigate to <https://splunkbase.splunk.com/develop/>. You'll need to log in with your [splunk.com](#) account. Click on the "request free developer license" button.



On the next page, under "2. Get your developer license." click on the link "request your developer license." At this point, you have to wait to get your license. When/if you are approved, you'll get an e-mail to your inbox

Splunk License <license@splunk.com>

to me

Aug 31

**** THIS MESSAGE IS SENT FROM AN UNMONITORED MAILBOX. DO NOT REPLY TO THIS MESSAGE ****

Hello,

Thank for requesting a Splunk Developer Trial license. We want to ensure that you have all of the support and resources you need to be successful developing with Splunk. Get started material, downloads, documentation, code samples and tutorials can be found at <http://dev.splunk.com>. You can get the latest updates by following us on Twitter: <https://twitter.com/splunkdev>

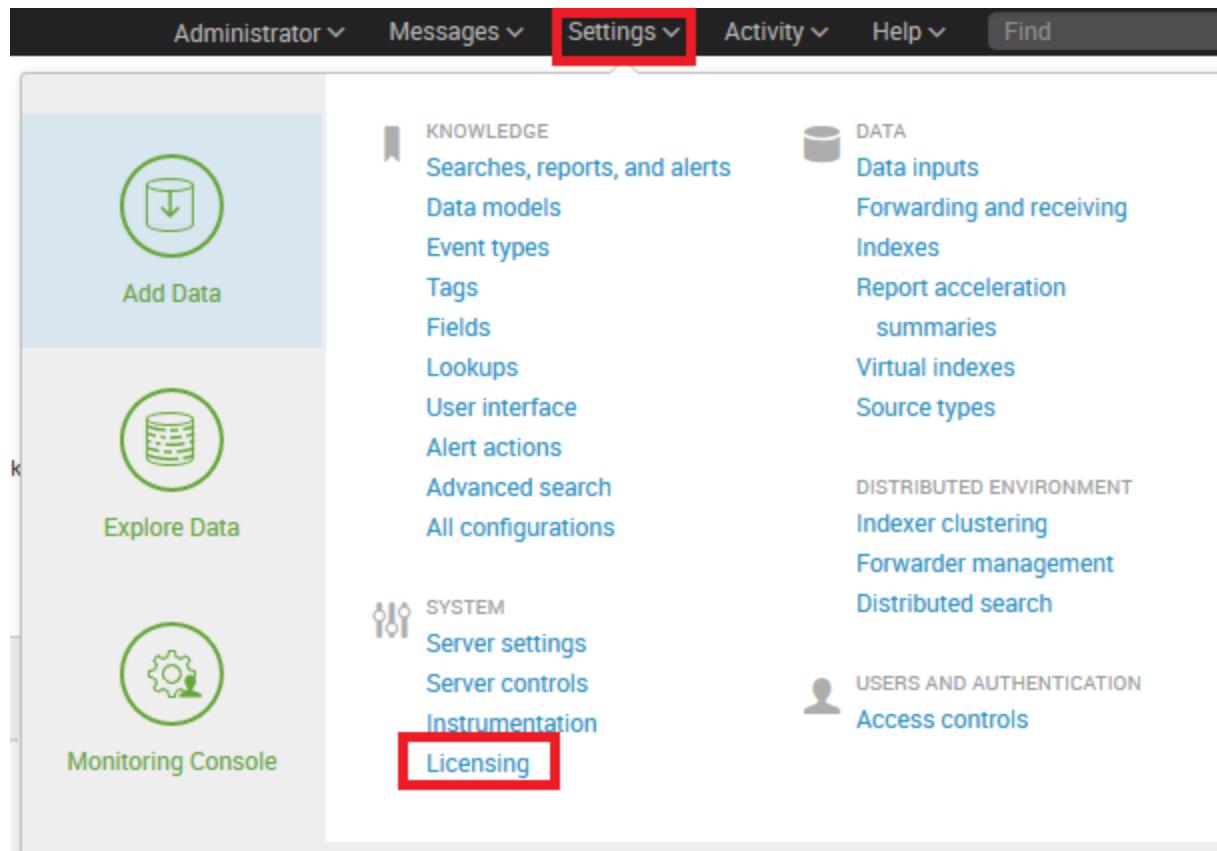
Here are some additional resources:

Python SDK - <http://dev.splunk.com/view/python-sdk/SP-CAAAEBB>
Java SDK - <http://dev.splunk.com/view/java-sdk/SP-CAAAECN>
JavaScript SDK - <http://dev.splunk.com/view/javascript-sdk/SP-CAAAECM>
Ruby SDK - <http://dev.splunk.com/view/ruby-sdk/SP-CAAAEFQ>
PHP SDK - <http://dev.splunk.com/view/php-sdk/SP-CAAAEJM>
C# SDK - <http://dev.splunk.com/view/csharp-sdk/SP-CAAAEPK>
Splunk's web framework - <http://dev.splunk.com/view/web-framework/SP-CAAER6> & the web framework toolkit: <http://apps.splunk.com/app/1613>
Dev Tools: Splunk Plug-in for Eclipse & Java Monitoring - <http://dev.splunk.com/view/tools/SP-CAAAEQ2>

We are always interested in learning more about your use case to use in a SplunkLive and don't hesitate to let us know if you have questions and/or feedback at devinfo@splunk.com

License Details:
Product: Splunk Developer Personal License NOT FOR RESALE
Size: 10 GB
Expiration Date: March 3, 2017, 11:12am

Attached to the message you receive should be a splunk license file, "splunk.license". Download this file, then navigate to your splunk search head (<https://172.16.1.3:8000>) and log back in as admin. Under the settings menu, click on "Licensing".



On the licensing page, click the "Add license" button.

Licensing

This server is acting as a standalone license server

Trial license group

[Change license group](#)

This server is configured to use licenses from the Trial I

[Add license](#)

[Usage report](#)

Alerts

Licensing alerts notify you of excessive indexing warnin

Current

- No licensing alerts

Permanent

- No licensing violations

Local server information

Indexer name	siem
License expiration	Jan 6, 2017 1:36:41 PM
Licensed daily volume	500 MB
Volume used today	0 MB (0% of quota)
Warning count	0
Debug information	All license details All indexer details

Click on the browse button, and browse to where you stored the “splunk.license” file. Click the green “Install” button.

Add new license

Learn more about your license options at the [licensing section](#) on splunk.com.

To install a license, upload a license file here (license files end with .license):

splunk.license

Or, copy & paste the license XML directly...

Cancel
Install

The next page asks you to restart Splunk to apply the license. You have no data indexed currently so there's no reason to not select restart now. Restart Splunk and log back in as admin again. The licensing page will have updated, and should now display the development license that you acquired.

Splunk Developer Personal License NOT FOR RESALE stack [Learn more](#)

Licenses	Volume	Expiration	Status
Splunk Developer Personal License NOT FOR RESALE	10,240 MB	Mar 3, 2017 2:12:31 PM	valid
Effective daily volume		10,240 MB	
Pools	Indexers	Volume used today	
auto_generated_pool_enterprise		0 MB / 10,240 MB	Edit Delete
<i>No indexers have reported into this pool today</i>			

At this point, the Splunk service on the siem VM is all but set up. **Take a snapshot of the siem VM**, and let's move on to setting up a Universal Forwarder on the ips VM.

Universal Forwarder Setup

The Universal Forwarder is responsible for sending logs from a given system to selected heavy forwarder or indexer for collection and organization. For our lab, we'll be downloading the Universal Forwarder package for Linux, installing it on our ips VM, and configuring it to send the IDS logs from Snort or Suricata (whichever you chose to install) to splunk for processing.

First, using your favorite web browser, navigate to [splunk.com](#), and log in with your account. On the main page, all the way at the bottom of the page, under the "Free Trials and Downloads" list, click on the link entitled "Universal Forwarder".

FREE TRIALS AND DOWNLOADS

Splunk Enterprise

Splunk Cloud

Splunk Light Cloud Service

Splunk Light Software

Splunk Enterprise Security

Splunk IT Service Intelligence

Splunk Mobile Access

Splunk Universal Forwarder

Not unlike with the Splunk Enterprise installation you have options for a variety of operating systems that the universal forwarder can be installed on.

Splunk Universal Forwarder

Universal Forwarders provide reliable, secure data collection from remote sources and forward that data into Splunk (Enterprise, Light, Cloud or Hunk) for indexing and consolidation. They can scale to tens of thousands of remote systems, collecting terabytes of data with minimal impact on performance.

- ✓ Tagging of metadata (source, sourcetype and host)
- ✓ Configurable throttling and buffering
- ✓ Data compression
- ✓ SSL security
- ✓ Transport over any available network ports
- ✓ Local scripted inputs
- ✓ Centralized management

[Windows](#)

[Linux](#)

[Solaris](#)

[Mac OS](#)

[FreeBSD](#)

[AIX](#)

[HP-UX](#)

Click the Linux button, and a long list of package options will appear. Click on the 64-bit package that ends in “.deb”.

Splunk Universal Forwarder For Linux

X

OS version

2.6+ kernel Linux distributions (64-bit)

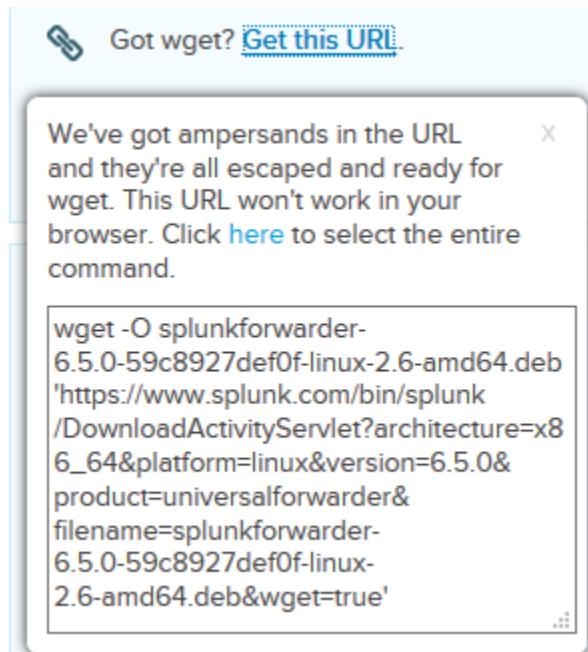
[Release Notes](#)

[!\[\]\(9ca535c607e33c999aadc30d92690dcb_img.jpg\)splunkforwarder-6.5.0-59c8927def0f-linux-2.6-x86_64.rpm](#)

[!\[\]\(587c5633e747434aeb0f9d28b44dbb97_img.jpg\)splunkforwarder-6.5.0-59c8927def0f-linux-2.6-amd64.deb](#)

[!\[\]\(5184c69e333fd0f864924789ded87029_img.jpg\)splunkforwarder-6.5.0-59c8927def0f-Linux-x86_64.tgz](#)

After accepting the EULA that pops up, your browser will move to a splash page thanking you for downloading the Universal Forwarder, just like with the Splunk Enterprise server install. You have two options for how getting this .deb package onto the ips VM. You can download the .deb package via your web browser and use SCP to copy it to the ips VM, or you can reveal the wget command to download the universal forwarder, and run wget on the ips VM itself.



Once you have acquired the universal forwarder .deb package and have it on the ips VM, you will need to log in to the ips VM as root if you haven't already, and use the dpkg -i command to install the forwarder. In my case, I used SSH key-based authentication to log in as root, ran wget to download the forwarder package, then ran

```
dpkg -i splunkforwarder-6.5.0-59c8927def0f-linux-2.6-amd64.deb
```

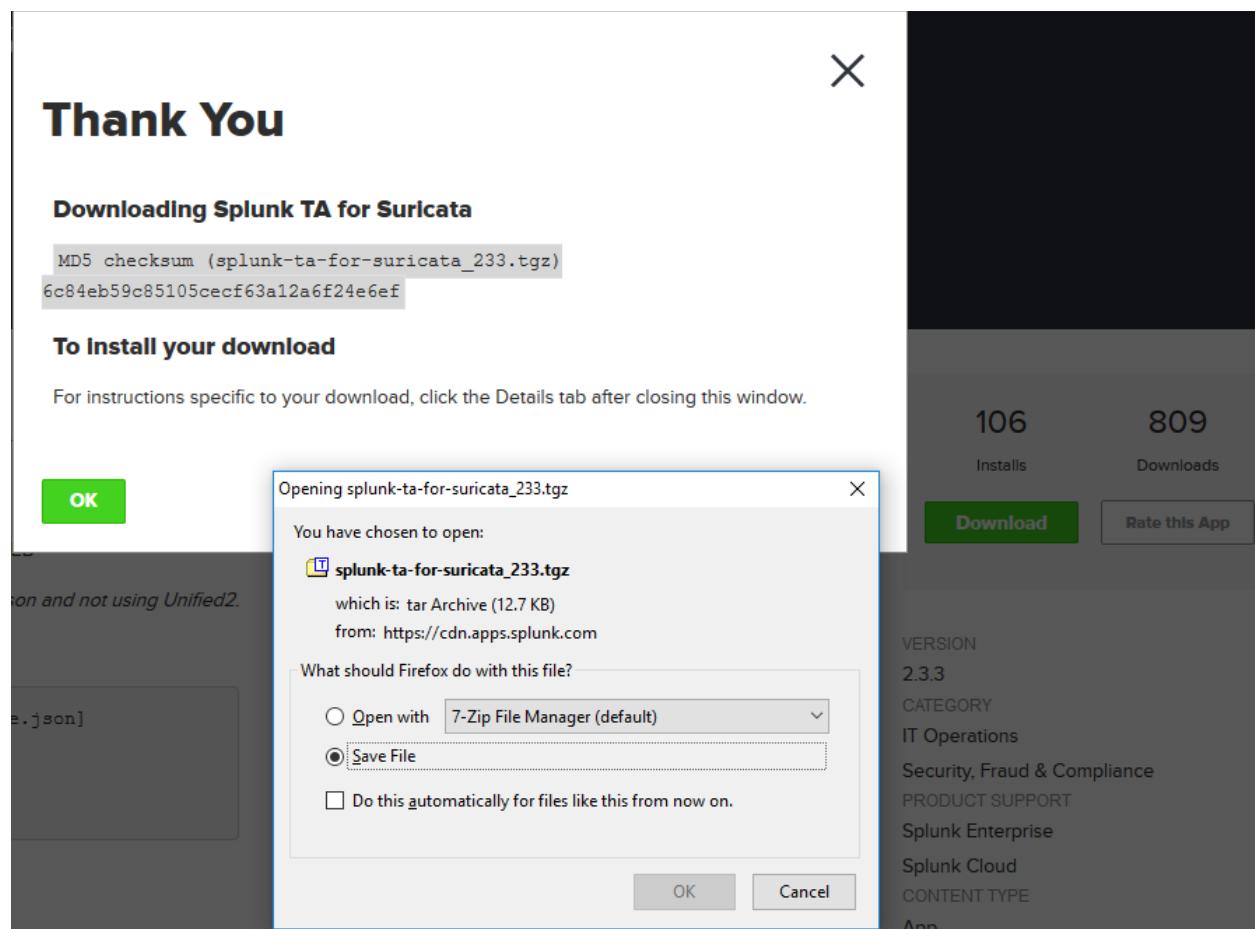
```
splunkforwarder-6.5.0-59c8927d 100%[=====] 18.71M 10.6MB/s in 1.8s
2016-11-08 16:47:37 (10.6 MB/s) - 'splunkforwarder-6.5.0-59c8927def0f-linux-2.6-amd64.deb' saved [19623428/19623428]
root@ips:~# dpkg -i splunkforwarder-6.5.0-59c8927def0f-linux-2.6-amd64.deb
```

By default, the forwarder gets installed to /opt/splunkforwarder. Before we get to starting the forwarder, depending on which IPS software you chose to install and run, you will need to acquire and install either the Splunk TA for Suricata, or the Hurricane Labs Add-On for Unified2 (if you're running Snort).

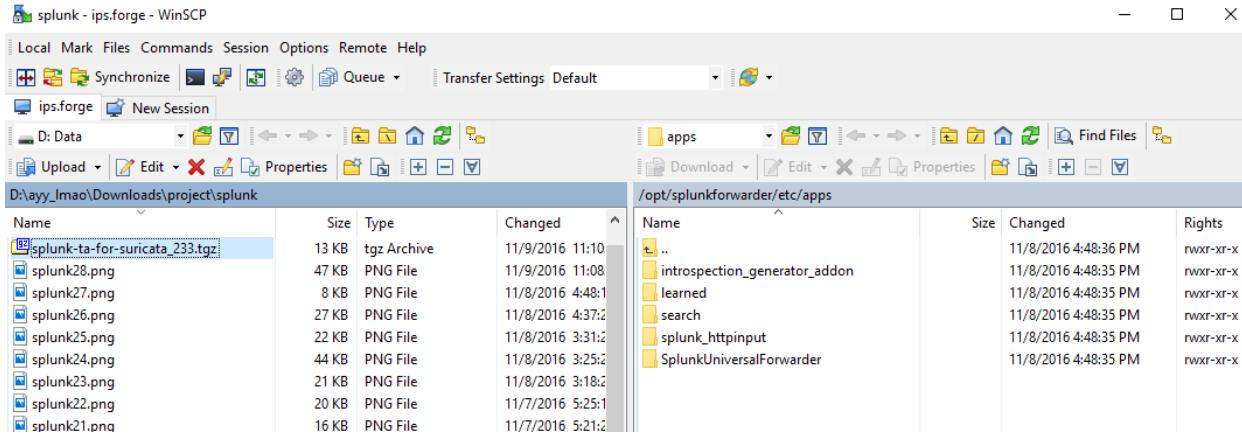
Splunk TA for Suricata

Splunk features what are known as technical applications or add-ons (also known as “TAs”) for Universal Forwarders. These TAs allow the universal forwarder to be able to parse a variety of different log types easily.

Splunk's various TAs and add-ons are available at splunkbase.splunk.com. Use your favorite browser and log in with your splunk.com credentials. The application we want “Splunk TA for Suricata” is currently located at splunkbase.splunk.com/app/2760/, and the current version as of this writing is 2.3.3. Click the download button accept the license terms, and save the application to your workstation.



You'll have to use SCP and copy the application to your ips VM manually. If you haven't already, I recommend setting up key-based authentication for your virtual machines, and enabling key-based authentication for winSCP if you are using a Windows workstation, or hypervisor host. If you are on windows, and using winSCP, copy the “splunk-ta-for-suricata_233.tgz” to the /opt/splunkforwarder/etc/apps directory.



If you are on Linux/OSX, open your favorite terminal app, cd into the directory you downloaded the tgz package to (normally this is ~/Downloads, but may differ for you) and run

```
scp splunk-ta-for-suricata_233.tgz root@172.16.1.4:/opt/splunkforwarder/etc/apps
```

If key-based authentication as the root user has been configured, the scp command should be able to utilize your SSH key to copy the file as the root user.

After you've copied the tgz file to /opt/splunkforwarder/etc/apps, run

```
tar -xzvf splunk-ta-for-suricata_233.tgz
```

This will decompress the app and effectively “install” it to the “TA-Suricata” directory.

```
root@ips:/opt/splunkforwarder/etc/apps# ls -al
total 48
drwxr-xr-x  8 splunk splunk  4096 Nov 10 13:01 .
drwxr-xr-x 13 splunk splunk  4096 Nov  8 16:48 ..
drwxr-xr-x  4 splunk splunk  4096 Nov  8 16:48 introspection_generator_addon
drwxr-xr-x  4 splunk splunk  4096 Nov  8 16:48 learned
drwxr-xr-x  4 splunk splunk  4096 Nov  8 16:48 search
drwxr-xr-x  3 splunk splunk  4096 Nov  8 16:48 splunk_httpinput
-rw-r--r--  1 root   root   12983 Nov  9 11:10 splunk-ta-for-suricata_233.tgz
drwxr-xr-x  4 splunk splunk  4096 Nov  8 16:48 SplunkUniversalForwarder
drwxr-xr-x  7 root   root   4096 Nov  7 09:50 TA-Suricata
```

Cd into /opt/splunkforwarder/etc/apps/TA-Suricata/default, and using your favorite text editor, open the inputs.conf file for editing. Autosuricata reconfigures suricata.yaml to drop the eve.json file into /var/log/suricata, as the inputs.conf file expects, and the default sourcetype should be fine. The only things you should have to modify is the “host = splunk-nat-sec” field, and the index field.. I would change the host field to “host = ips-vm” or something equally descriptive. As for the index file, on our system, the suricata index does NOT exist. Change the index from suricata to main. Jump to the [Starting The Forwarder + Persistence](#) section for your next steps.

```
[monitor:///var/log/suricata/eve.json]
host = ips-vm
sourcetype = suricata
index = suricata
```

Note: A splunk administrator has asked me to note that while in our case, since we don't have a ton of different types of data that we're throwing into our splunk instance that using the default index of "main" is fine. However, in a non-lab environment (say an enterprise network) where you will likely be dealing with IDS/IPS events from multiple sensors, a separate index for your Suricata sensors makes much more sense.

Hurricane Labs Add-On for Unified2

Splunk features what are known as technical applications or add-ons (also known as "TAs") for Universal Forwarders. These TAs allow the universal forwarder to be able to parse a variety of different log types easily. The Hurricane Labs Add-On for Unified2 is a TA, not unlike the Splunk TA for Suricata.

Open your favorite web browser and navigate to splunkbase.splunk.com. Log in using your splunk.com website username and password, then navigate to splunkbase.splunk.com/app/1858/. The current version for this app is 1.0.5. Click the download button accept the license terms, and save the application to your workstation.

Thank You

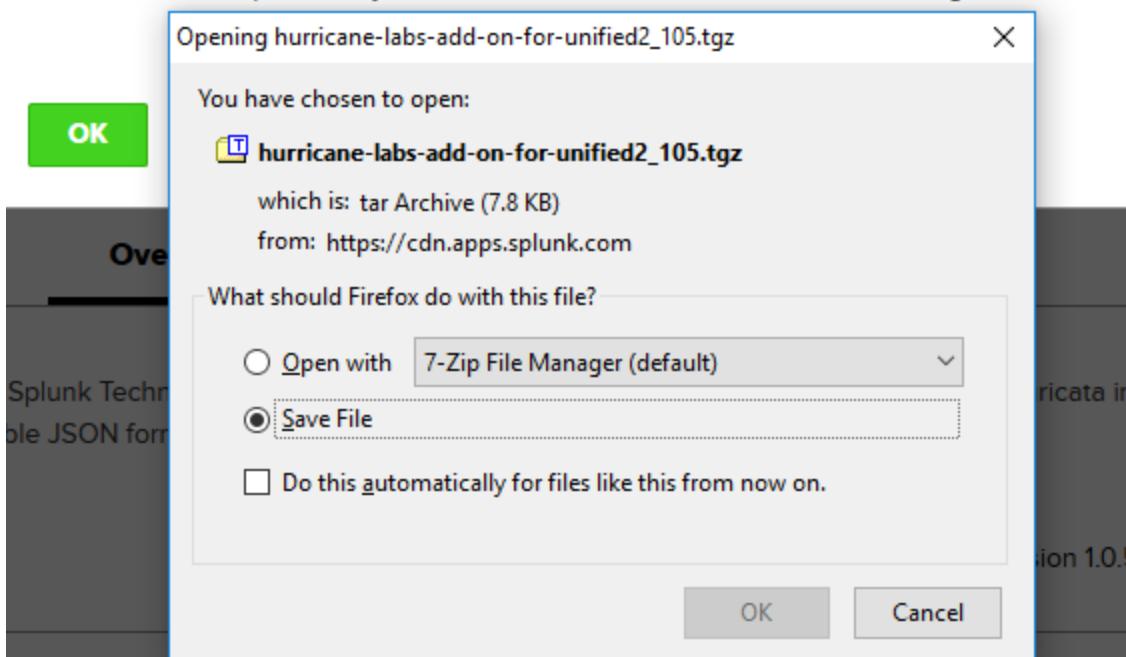
Downloading Hurricane Labs Add-On for Unified2

MD5 checksum (hurricane-labs-add-on-for-unified2_105.tgz)

836498ee32d72ebe7a99142585f263c3

To Install your download

For instructions specific to your download, click the Details tab after closing this window.



You'll have to use SCP and copy the application to your ips VM manually. If you haven't already, I recommend setting up key-based authentication for your virtual machines, and enabling key-based authentication for winSCP if you are using a Windows workstation, or hypervisor host. If you are on windows, and using winSCP, copy the "hurricane-labs-add-on-for-unified2_105.tgz" to the /opt/splunkforwarder/etc/apps directory.

If you are on Linux/OSX, open your favorite terminal app, cd into the directory you downloaded the tgz package to (normally this is ~/Downloads, but may differ for you) and run

```
scp hurricane-labs-add-on-for-unified2_105.tgz root@172.16.1.4:/opt/splunkforwarder/etc/apps
```

If key-based authentication as the root user has been configured, the scp command should be able to utilize your SSH key to copy the file as the root user.

After you've copied the tgz file to /opt/splunkforwarder/etc/apps, connect to the ips VM and run

```
tar -xzvf hurricane-labs-add-on-for-unified2_105.tgz
```

This will decompress the app and effectively “install” it to the “TA-unified2” directory.

```
root@ips:/opt/splunkforwarder/etc/apps# ls -al
total 40
drwxr-xr-x  8 splunk splunk 4096 Nov 21 11:29 .
drwxr-xr-x 13 splunk splunk 4096 Nov 10 13:50 ..
-rw-r--r--  1 root   root  7936 Nov 18 14:37 hurricane-labs-add-on-for-unified2_105.tgz
drwxr-xr-x  4 splunk splunk 4096 Nov  8 16:48 introspection_generator_addon
drwxr-xr-x  5 splunk splunk 4096 Nov 10 13:50 learned
drwxr-xr-x  4 splunk splunk 4096 Nov  8 16:48 search
drwxr-xr-x  3 splunk splunk 4096 Nov  8 16:48 splunk_httpinput
drwxr-xr-x  4 splunk splunk 4096 Nov  8 16:48 SplunkUniversalForwarder
drwxr-xr-x  7 501 staff  4096 Oct  9 2014 TA-unified2
```

Cd into /opt/splunkforwarder/etc/apps/TA-unified2/default, and using your favorite text editor, open the unified2.conf file for editing. There are 3 lines that need to be edited, sid_msg_map, gen_msg_map, and classifications. By default, these lines point to the sid-msg.map, gen-msg.map and classification.config files, and assumes they are installed in /etc/snort. You will need to change the config file so that these lines all point to /opt/snort/etc instead (e.g. /opt/snort/etc/sid-msg.map, /opt/snort/etc/gen-msg.map, and /opt/snort/etc/classification.config).

```
[output]
pretty = false
pcap = true
[unified2]
checkpoint_file = /var/log/snort/alert_json.checkpoint
input u2 = /var/log/snort/snort.u2
sid_msg_map = /opt/snort/etc/sid-msg.map
gen_msg_map = /opt/snort/etc/gen-msg.map
classifications = /opt/snort/etc/classification.config
```

Note: You may notice the lines “pretty = false”, and “pcap = true” in the unified2.conf file. If you change pretty to “pretty = true”, this enables more verbose output from our sensor to our Splunk instance

After modifying these lines, edit the input.conf file. Change the line set to “disabled = 1”, to “disabled = 0”

```
[script://./bin/alert_json.sh]
disabled = 0
interval = 30
sourcetype = snort_json
```

Before we move on, you'll need to install python for the TA-unified2 app to work properly. As root, run the command

```
apt-get -y install python
```

By default, This TA dumps Snort logs to the “snort_json” sourcetype, under the “main” index. This will become important in a moment when we need to query the IDS logs for data to verify everything is working. Jump to the [Starting The Forwarder + Persistence](#) section for your next steps.

Starting The Forwarder + Persistence

Now, we need to start the forwarder to enable sending our log data to the siem VM. to do this, we're going to run the following commands

```
/opt/splunkforwarder/bin/splunk start --accept-license
/opt/splunkforwarder/bin/splunk stop
/opt/splunkforwarder/bin/splunk add forward-server 172.16.1.3:9997
/opt/splunkforwarder/bin/splunk enable boot-start
init 6
```

These commands will set up the universal forwarder by accepting the Splunk EULA, stop the server, tell the forwarder to send its data to 172.16.1.3 to TCP port 9997, enable the universal forwarder on system boot, then immediately reboot the system. After the system reboots, log back in and run

```
ps -ef | grep splunk
```

```
root      1227      1  0 13:56 ?          00:00:00 splunkd -p 8089 start
root      1239    1227  0 13:56 ?          00:00:00 [splunkd pid=1227] splunkd -p 8089 start [process-runner]
```

If your output looks similar to the illustration below, then splunk was able to start successfully. If not, run “service splunk status” to try and determine the issue. Additionally, splunk logs to /opt/splunkforwarder/var/log/splunk/*.log, analyze the log files for clues that may indicate why splunk is misbehaving. Please note that right now, if you were to check splunkd.log, you might notice the following log entries:

```
11-10-2016 14:12:19.080 -0500 WARN TcpOutputFd - Connect to 172.16.1.3:9997 failed.
Connection refused
```

11-10-2016 14:12:19.081 -0500 ERROR TcpOutputFd - Connection to host=172.16.1.3:9997 failed

This is normal, we're going to fix this. Log in to your splunk instance on the siem VM (<https://172.16.1.3:8000>), and navigate to Settings > Forwarding and Recieving

The screenshot shows the Splunk Settings interface. At the top, there is a navigation bar with tabs: Administrator, Messages (highlighted with a red box containing the number 1), Settings, Activity, Help, and Find. Below the navigation bar, there are several sections:

- Add Data** (represented by a green circle icon with a cylinder and downward arrow):
 - Searches, reports, and alerts
 - Data models
 - Event types
 - Tags
 - Fields
 - Lookups
 - User interface
 - Alert actions
 - Advanced search
 - All configurations
- Monitoring Console** (represented by a green circle icon with a gear and person):
- DATA** (represented by a grey cylinder icon):
 - Data inputs
 - Forwarding and receiving** (highlighted with a red box)
 - Indexes
 - Report acceleration summaries
 - Virtual indexes
 - Source types
- DISTRIBUTED ENVIRONMENT**:
 - Indexer clustering
 - Forwarder management
 - Distributed search
- SYSTEM** (represented by a wrench and screwdriver icon):
 - Server settings
 - Server controls
 - Instrumentation
 - Licensing
- USERS AND AUTHENTICATION** (represented by a person icon):
 - Access controls

On the next page, under the section titled “Configure receiving”, click the “Add new” link.

Forwarding and receiving

Forward data

Set up forwarding between two or more Splunk instances.

	Actions
Forwarding defaults	
Configure forwarding	Add new

Receive data

Configure this instance to receive data forwarded from other instances.

	Actions
Configure receiving	Add new

On the next page, enter “9997” in the input box labeled “Listen on this port”, then click the green “Save” button.

[Configure receiving](#)

Set up this Splunk instance to receive data from forwarder(s).

Listen on this port *

For example, 9997 will receive data on TCP port 9997.

[Cancel](#) [Save](#)

The next page confirms that a new TCP listener on port 9997 has been enabled and is waiting for data from our universal forwarder.

Showing 1-1 of 1 item

Listen on this port	Status	Actions
9997	Enabled Disable	Delete

Testing Splunk and the Universal Forwarder

The quickest way to test an IDS is to generate what I like to call “The Full Broadside” Battery. Recall that metasploitable 2 is incredibly vulnerable. You may also recall that I have referred to our kali VM as our obnoxiously loud attacker. You’re about to find out why in a minute.

Photo # DN-ST-85-05379 USS Iowa fires a full broadside, July 1984



It's kinda like that.

Generating The Test Battery

Log in to the console of your kali VM, and open the terminal application. Run the commands

```
msfdb init  
service postgresql start
```

These commands are responsible for initializing and setting up the postgres database backend that metasploit relies on for storing information during standard operation. These operations take a moment or two, depending on the speed of the drive the VM is stored on and the amount of resources available to the kali VM. After the commands complete, run the command

```
armitage
```

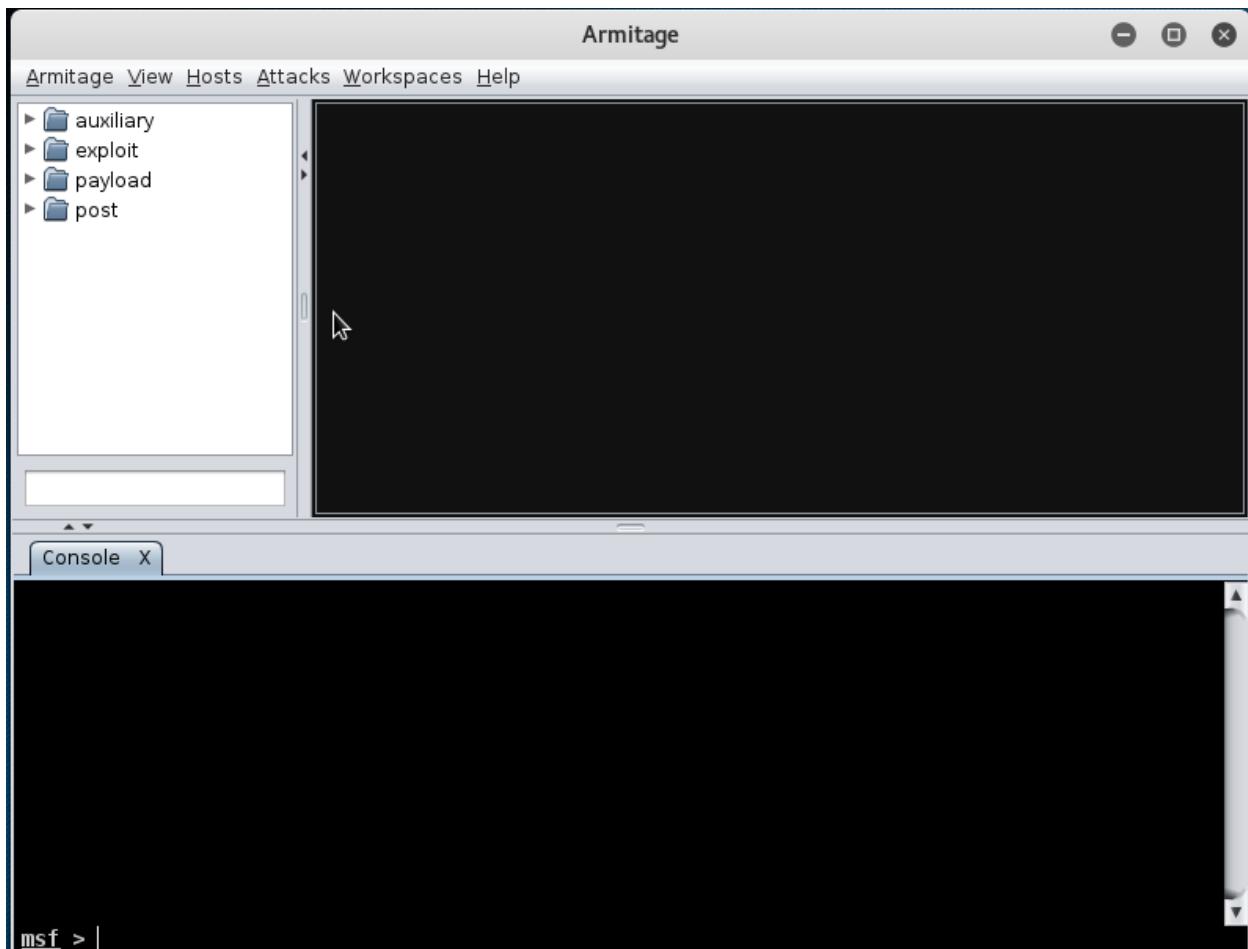
This starts the armitage GUI for the metasploit framework. You'll be greeted by a dialogue box entitled "Connect..." the default setting of 127.0.0.1 on port 55553 with the default username and password should be fine; simply click the connect button.



While armitage is starting, it may ask you if you'd like it to start metasploit/msfrpcd.

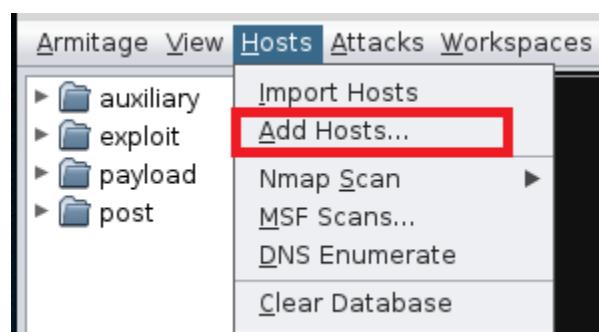


Say yes, and allow the armitage GUI to load.

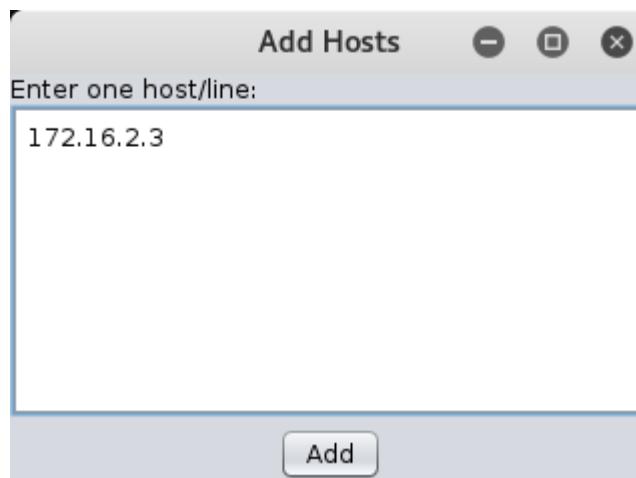


Armitage is best described as a wrapper for the metasploit framework. Metasploit in and of itself is a tremendously useful tool for offensive security professionals; mainly for centralizing a massive database of exploits and payloads, but also for ease of use. Armitage improves the ease of use of the metasploit framework by making certain functions much simpler to perform, while making it possible for die-hard metasploit cli enthusiasts to keep using the CLI framework to their hearts' content.

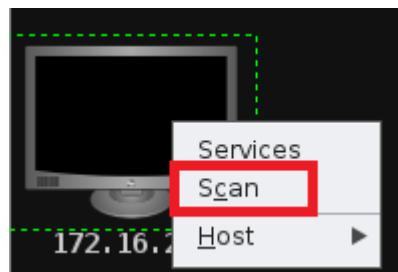
On the menu bar along the top of the Armitage window, click on “Hosts” button, then “Add Hosts...”



An input box will pop up, asking you to enter IP addresses of targets you want to add. input 172.16.2.3 on a single line, then click “Add”



A computer will pop up in the main window. If you right click on it, there is a small selection of options you have available right now. Click on the “Scan” option that appears.



Scan runs several probes on the system to identify open ports and running services on the system. However, there is an issue with some of the probes that are ran, where it won't properly pass the IP address of the target host from armitage to the metasploit framework correctly. You may see this error pop up in the “scan” tab

```
msf auxiliary(ftp_version) > run -j
[-] Auxiliary failed: Msf::OptionValidateError The following options failed to validate:
RHOSTS.
```

You will need to run the following commands in the scan tab of the armitage window for each service/banner grab that armitage attempts to do

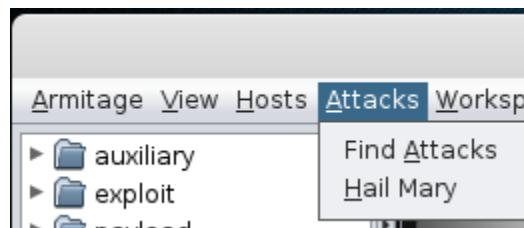
```
set RHOSTS 172.16.2.3
run
```

```
msf auxiliary(ftp_version) > set RHOSTS 172.16.2.3
RHOSTS => 172.16.2.3
msf auxiliary(ftp_version) > run
[*] 172.16.2.3:21 - FTP Banner: '220 (vsFTPD 2.3.4)\x0d\x0a'
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

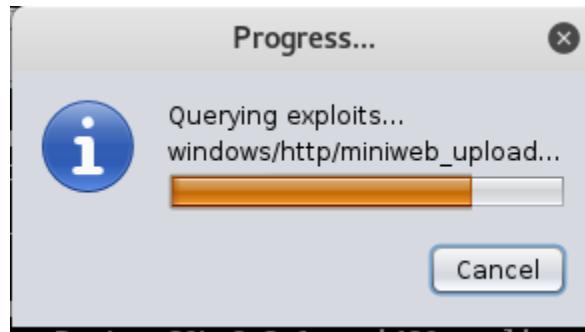
Note: Some pentesters may immediately say “setg” is supposed to set RHOSTS globally, but this will not resolve the issue in my experience.

Armitage runs 7 of metasploit’s auxiliary banner grabs/service scans - FTP(21/tcp), SSH(22/tcp), Telnet(23/tcp), SMTP(25/tcp), HTTP(80/tcp), SMB(445/tcp), java rmi(1099/tcp), mysql(3306/tcp), postgres(5432/tcp)

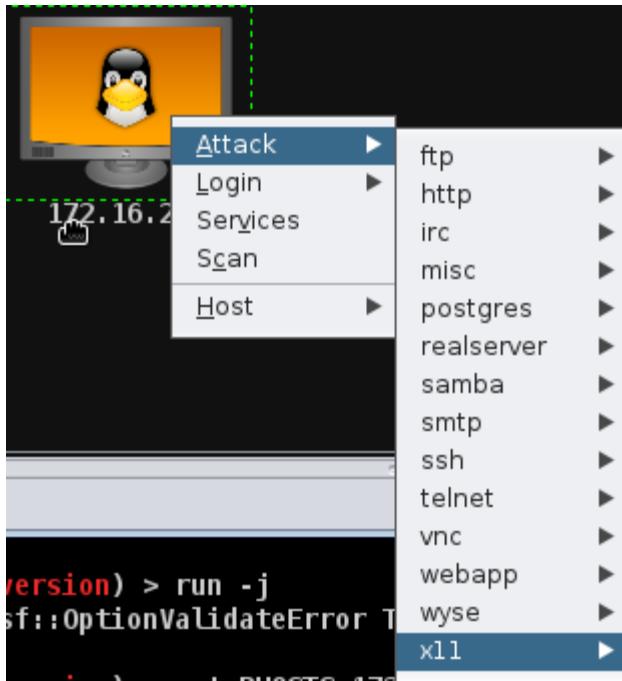
While its annoying to have to reset RHOSTS with each service scan, scanning all of the services is important for the next phase of our test. In the Armitage menu, click “Attacks”, then click “Find Attacks”.



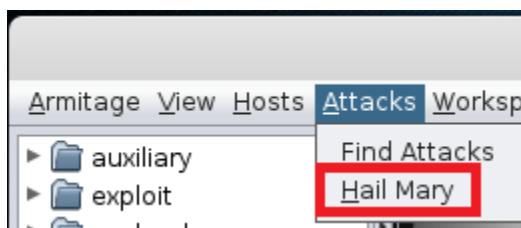
Thanks to the port scans and service detection we did, this allows Armitage to make a bunch of attack/exploit recommendations.



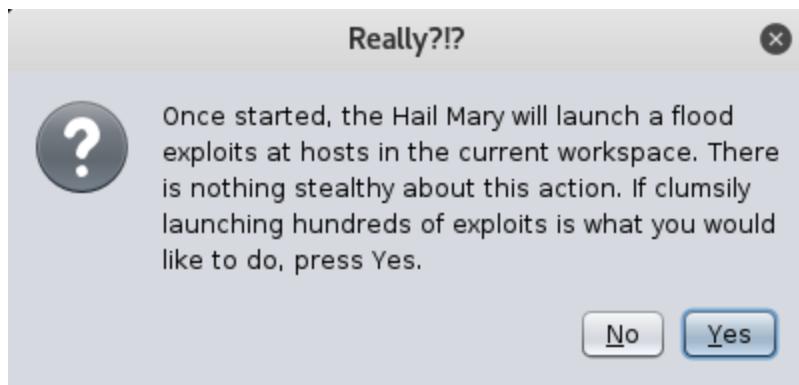
This takes a moment or two to complete, but when its done, it’ll notify you. If you right click on the computer (that now has a “Tux” penguin in the center, denoting it as a Linux system), you’ll have an attack menu available. These are *possible* attacks you can use against this system. The exploits mapped are NOT guaranteed to work. Don’t ever rely on the “Find Attacks” sub-menu to be accurate or provide you with 100% correct exploits to a given target host. Ever.



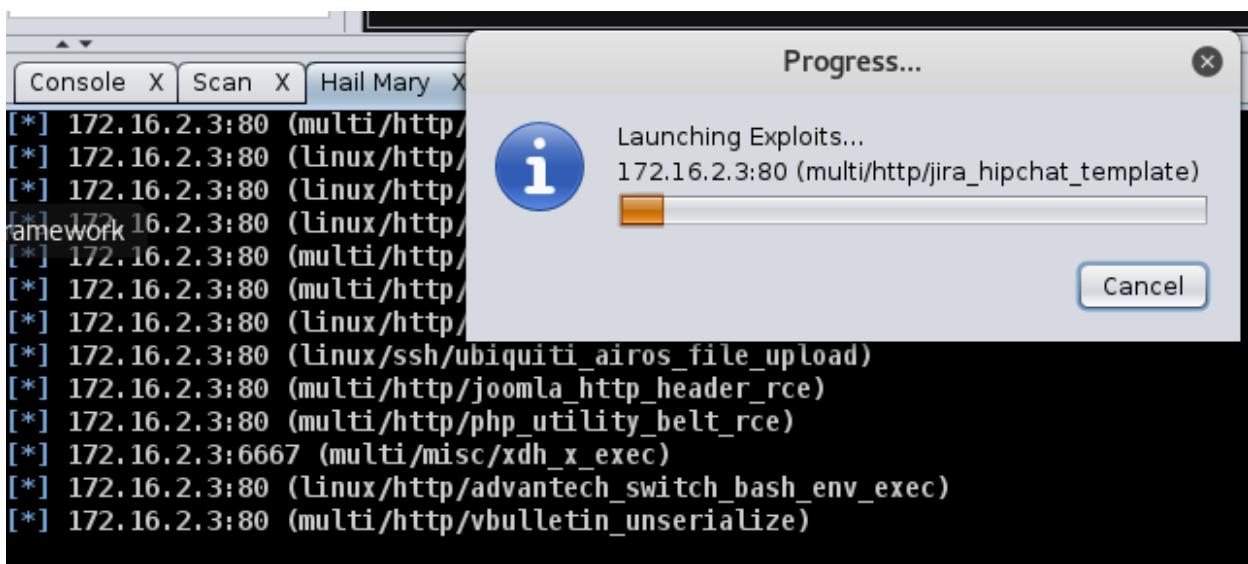
Now, we could try running each one of these exploits individually, taking our sweet time, or, considering that the goal of this exercise is to generate as much noise as possible, fire as many of them as we can at once. There is a feature in the metasploit framework called db_autopwn. In Armitage, this feature is called "Hail Mary". This feature can be found under the Attacks menu, underneath the "Find Attacks" option.



When you run the Hail Mary, you are throwing everything you can at this host with absolutely no regard for stealth or tradecraft; Armitage even warns you when you try to run it, and asks if you are absolutely sure you want to do this.



In our case, This is what we're looking for; to generate as much noise as we can, with minimum effort, so say yes and let 'er rip. This will take some time, and you will see a progress bar go by as Armitage keeps “throwing” all the exploits it can map to our metasploitable host.



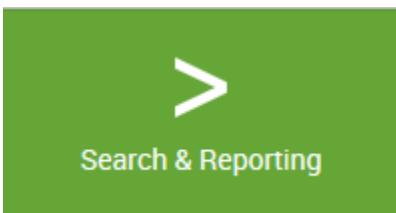
The aftermath is pretty bloody. You'll end up with half a dozen shells on the metasploitable host. If you right click on our target system, you'll see several prompts for "shell #" and/or "meterpreter #" for each successful exploit that resulted in remote code execution on the host.



At this point, this should have been more than enough noise to cause the IDS to flag some alerts. Log in to the splunk search head/index at <https://172.16.1.3:8000>. At this point, depending on whether you have Snort or Suricata installed, you will be jumping to “[Verifying Results with Snort](#)” or “[Verifying Results with Suricata](#)”.

Verifying Results with Snort

Once you are logged in on the home page, click the “search & reporting” app on the left side of the browser window.



On the next page, you will see a search bar in which you can enter a query for information. Input the following query

```
index="main" sourcetype=snort_json
```

Do not click the spyglass just yet to begin the search, because now we're going to adjust the timeframe. To the right of the input box for search, there is a small drop down labeled “All time” (by default). You can click the drop-down and choose a time range for splunk to search for certain events. In my case, the “Hail Mary” attack I had launched was less than 60 minutes ago, so I set the time range for “Last 60 minutes”. Adjust the time/date range to suit when you actually launched your “Hail Mary” attack on the metasploitable 2 VM.

The screenshot shows the Splunk search interface with the time range selector at the top. The 'All time' dropdown is open, revealing several time range options: 'Real-time', '30 second window', '1 minute window', '5 minute window', '30 minute window', '1 hour window', 'All time (real-time)', 'Relative', 'Today', 'Week to date', 'Business week to date', 'Month to date', 'Year to date', 'Yesterday', 'Previous week', 'Previous business week', 'Previous month', and 'Previous year'. The 'Last 60 minutes' option is highlighted with a red box. Below the dropdown, there is a sidebar with expandable sections: 'Relative', 'Real-time', 'Date Range', 'Date & Time Range', and 'Advanced'.

My query looked like this when I was finished:

```
sourcetype="snort_json" | table signature.msg | dedup signature.msg
```

Last 60 minutes



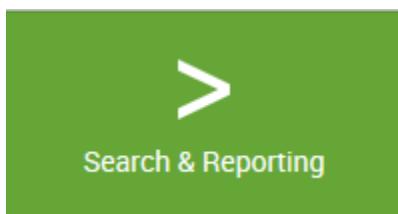
Allow Splunk some time to find the data and display it. What we're doing is looking in the "main" index, under the "snort_json" sourcetype being fed from our IPS sensor. Then we're looking specifically for the contents of the "signature.msg" field. Then we're saying show me only the data in the signature.msg field. Then we're taking the signature.msg field, and removing any duplicate entries. The entirety of this query only applies to data collected within the last 60 minutes.

signature.msg
POLICY-OTHER PHP uri tag injection attempt
POLICY-OTHER Adobe ColdFusion admin API access attempt
SERVER-WEBAPP D-Link DCS-900 Series Network Camera arbitrary file upload attempt
APP-DETECT failed FTP login attempt
MALWARE-OTHER Horde javascript.php href backdoor
SERVER-WEBAPP WebCalendar index.php form_single_user_login parameter command injection
SERVER-WEBAPP Symantec Web Gateway PHP remote code injection attempt
SERVER-WEBAPP Symantec Web Gateway pbcontrol.php filename parameter command injection attempt
SERVER-WEBAPP Cisco Prime Data Center Network Manager processImageSave.jsp directory traversal attempt
SQL use of concat function with select - likely SQL injection
SERVER-WEBAPP Zabbix httpmon.php SQL injection attempt
SERVER-WEBAPP WebTester install2.php arbitrary command execution attempt
MALWARE-CNC Win.Trojan.Dexter variant outbound connection
MALWARE-CNC Win.Trojan.Dexter CasinoLoader SQL injection
SERVER-WEBAPP Dell KACE Appliance KSudoClient privilege escalation attempt
SERVER-WEBAPP Dell KACE Appliance kbot_upload.php authentication bypass attempt
SERVER-WEBAPP Dell KACE Appliance kbot_upload.php directory traversal attempt
SQL union select - possible sql injection attempt - GET parameter
ftp_pp: FTP parameter length overflow
OS-OTHER Bash environment variable injection attempt

The output, while it might not have captured every single attack, and may not be 100% accurate, confirms that Snort is configured properly, to send its data to Splunk via the Hurricane Labs Unified 2 Add-on, and Universal Forwarder. Tuning your IDS and modifying the ruleset is beyond the scope of this guide.

Verifying Results with Suricata

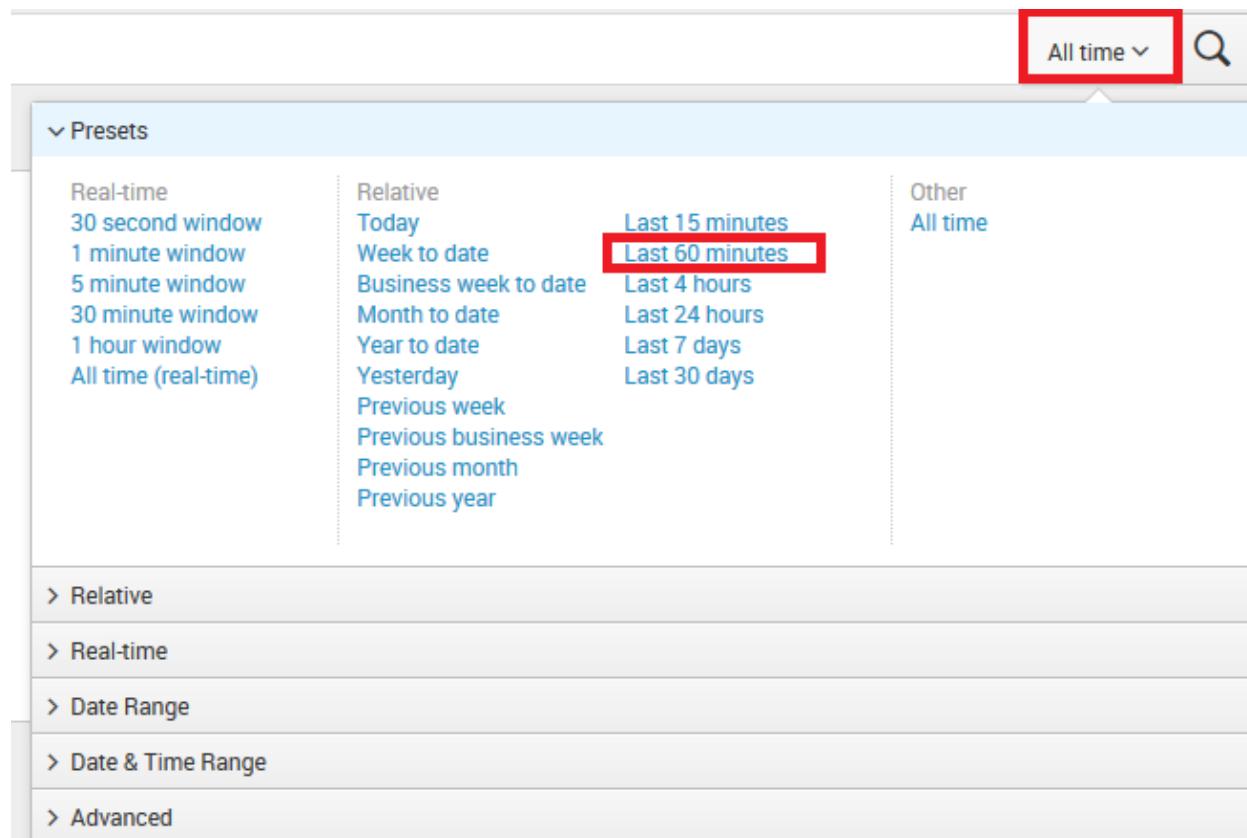
Once you are logged in on the home page, click the “search & reporting” app on the left side of the browser window.



On the next page, you will see a search bar in which you can enter a query for information. Input the following query

```
index="main" * event_type=alert | table alert.signature | dedup alert.signature
```

Do not click the spyglass just yet to begin the search, because now we're going to adjust the timeframe. To the right of the input box for search, there is a small drop down labeled "All time" (by default). You can click the drop-down and choose a time range for Splunk to search for certain events. In my case, the "Hail Mary" attack I had launched was less than 60 minutes ago, so I set the time range for "Last 60 minutes". Adjust the time/date range to suit when you actually launched your "Hail Mary" attack on the Metasploitable 2 VM.



My query looked like this when I was finished:

```
index="main" * event_type=alert | table alert.signature | dedup alert.signature
```

Last 60 minutes

Allow Splunk some time to find the data and display it. What we're doing is looking in the "main" index for any data. Then we're looking specifically for any records that have the event_type field set to "alert" (denoting that it's a Suricata IDS alert). Then we're saying show me only the data in the alert.signature field. Then we're taking the alert.signature field, and removing any duplicate entries. The entirety of this query only applies to data collected within the last 60 minutes.

alert.signature ◊
ET CHAT IRC PRIVMSG command
ET CHAT IRC NICK command
ET CHAT IRC USER command
ET FTP ProFTPD Backdoor Inbound Backdoor Open Request (ACIDBITCHEZ)
ET EXPLOIT Possible Pure-FTPd CVE-2014-6271 attempt
ET CHAT IRC JOIN command
SURICATA SMTP data command rejected
ET WEB_SERVER Possible CVE-2014-6271 Attempt
SURICATA TLS invalid record version
SURICATA TLS invalid record/traffic
ET POLICY Http Client Body contains pwd= in cleartext
ET POLICY Http Client Body contains passwd= in cleartext
ET POLICY Outgoing Basic Auth Base64 HTTP Password detected unencrypted
SURICATA HTTP Host header invalid
ET POLICY Http Client Body contains pass= in cleartext
ET EXPLOIT Possible CVE-2014-3704 Drupal SQLi attempt URLENCODE 2
ET WEB_SERVER Possible CVE-2014-6271 Attempt in Headers

The output, while it might not have captured every single attack, and may not be 100% accurate, confirms that Suricata is configured properly, to send its data to Splunk via the Suricata TA, and Universal Forwarder. Tuning your IDS and modifying the ruleset is beyond the scope of this guide.

In Your Own Image

Congratulations. With the creation of your VM lab, the world of information security or information technology study and research is well within your grasp. Where you take your lab environment from here is entirely up to you, and depends on what purpose you want your lab to serve. The purpose of this lab design I taught you how to build is for it to be multi-purpose; capable of filling multiple roles for IT and information security professionals looking for a secure, self-contained environment in order to practice their trade.

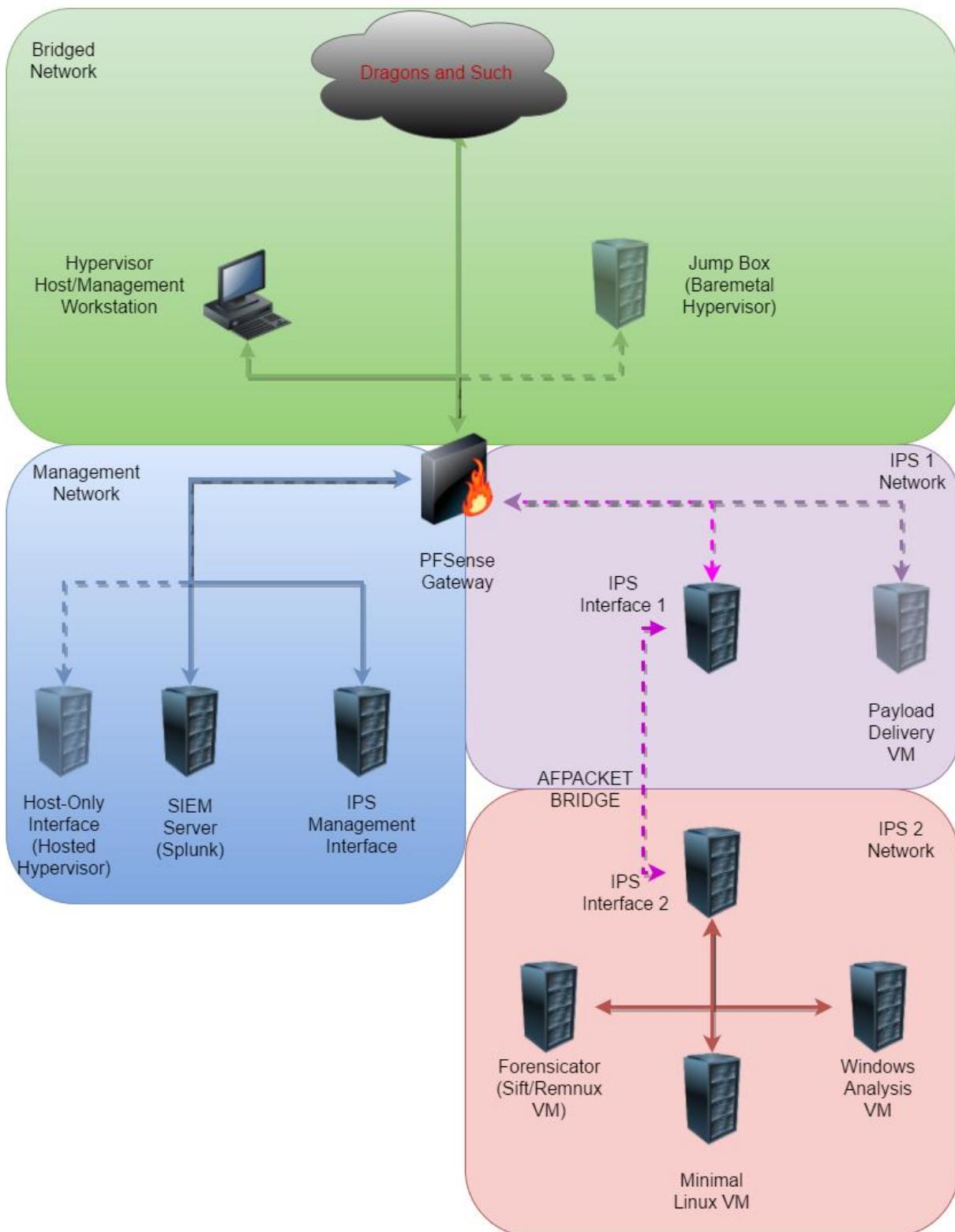
What if you don't like Splunk, and would like to use an ELK (Elasticsearch, Logstash, Kibana) for logging/SIEM instead? What if you don't want to run your services on Ubuntu, but instead

would rather use Redhat or another Linux distro? What if you don't want to use pfSense, but would rather use OPNSense, Vyatta, or Untangle as your firewall distribution instead? These choices are yours, and yours alone to make. I chose Ubuntu as our Linux distribution of choice for building the lab baseline, because the LTS releases of Ubuntu Server (currently at 16.04) have a long support cycle of five years. I chose pfSense because it has a long development history and a rich featureset. If you would rather use other distributions in building your lab network to better serve your needs or preferences, that is totally up to you.

The goal of this exercise is to teach you how to create, install, and configure VMs and virtual networks across a variety of common and popular hypervisors used all over the world today. Once you have achieved that knowledge, there is no limit (aside from hardware, of course) to how you can configure your VM lab; you can use whatever operating systems you want and make the lab as complex or as minimal as you like. This chapter is dedicated to giving you a few ideas towards modifying your lab to better suit your needs.

Visions of What Might Be

It's all well and good to tell you that the possibilities are endless, but it's another thing altogether to actually show you and give you ideas. Below are a couple of network diagrams that you could use as a springboard to develop/design your own lab network.

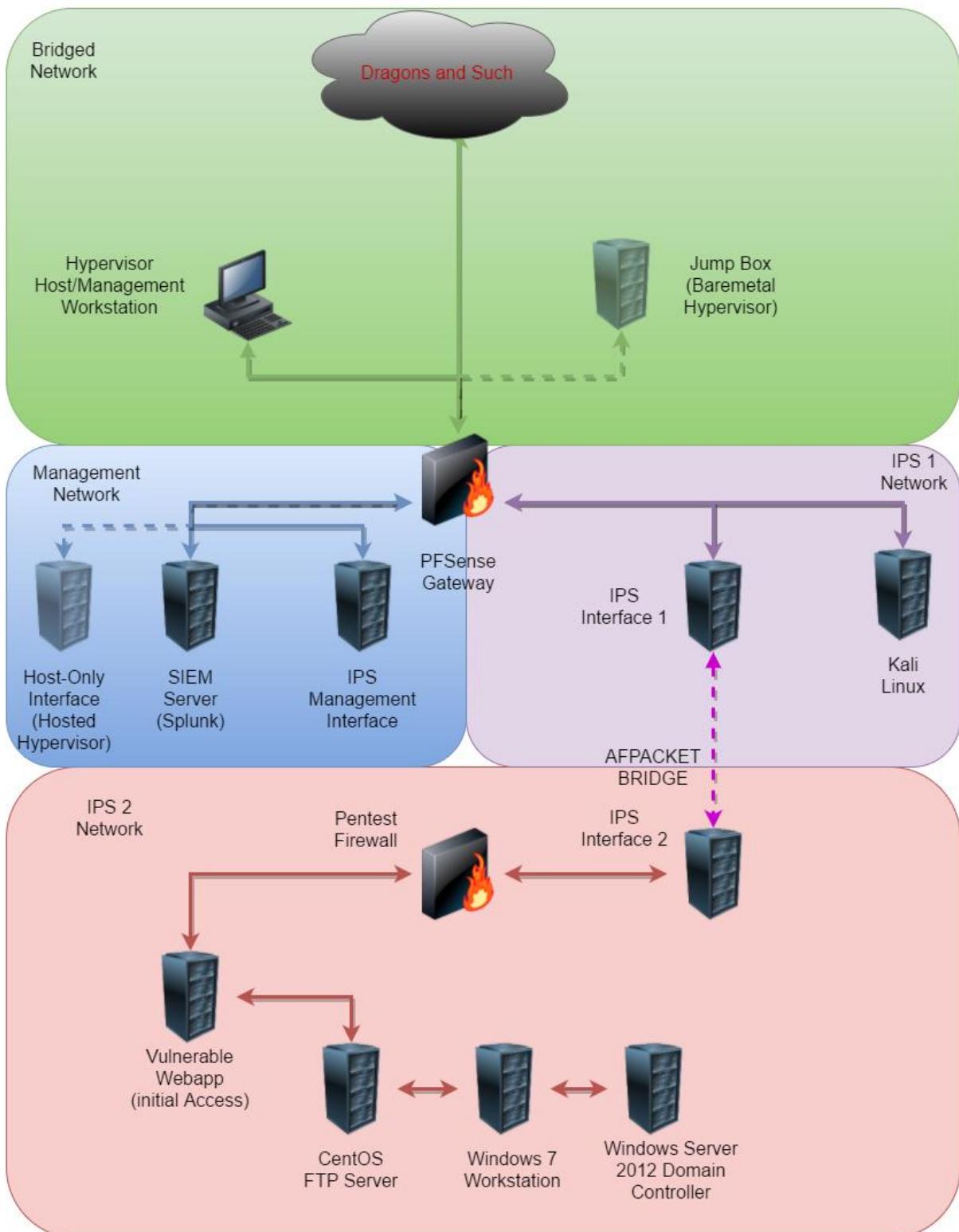


Fortunately for you, If you wanted to build a malware analysis lab out of the baseline configuration we set up, there isn't much you have to change. Obviously, you can remove the metasploitable 2 and kali linux VM entirely, since they won't help. In my diagram above on the "IPS 1" network, I have a "Payload Delivery VM" that I specify as being an optional component (transparent). The idea behind this extra VM is that if you have a malware sample you downloaded from a physical system or otherwise need to transfer to the analysis network, you can use this VM as a diode, or halfway point for delivering it to the malware network. Barring that, the VM could be used to download samples via wget, etc. This VM would ideally be either a Linux or BSD VM that is as minimal as possible -- very similar to how a jump box or bastion host would be set up.

The "IPS 2" network is where most of the action will be taking place, and where your malware analysis VMs should live. The VM labeled "Forensicator" is a specialty VM consisting of packages from a pair of special Linux distributions provided by the SANS organization. One of these distros is "SIFT", and is loaded with digital forensics and incident response tools, while the other distro is "Remnux" and is loaded with malware analysis and reverse engineering tools. To go along with our Linux analysis VM, "IPS 2" would also host a Windows analysis VM for dynamic analysis (sometimes known as "sandboxing"), or running specialty malware analysis tools that are simply Windows only. Finally, the IPS 2 network also hosts a minimal Linux VM. On Linux, there are several tools you can use to trace system activity as it is going on. Obviously, the more applications and services installed on a system, the more system activity you will have to filter out. Having a VM available with a minimal number of services installed allows you to monitor system activity with little to no overhead to worry about filtering out. This could come in handy for trying to perform dynamic analysis of Linux malware, to determine what changes to the system are being made when you execute a given malicious payload. Alternatively, you could potentially have this system perform double duty as your malware payload delivery VM, and do away with the payload delivery VM in the "IPS 1" network altogether.

Other possibilities for your analysis lab would be to possibly allocate more resources to your IPS VM and install a second network monitoring package on the system, such as bro IDS, or perhaps run a full packet capture tool such as Moloch or even just tcpdump to gather all of the network traffic coming from the ips 2 network to try and capture command and control traffic, beaconing, or other network artifacts.

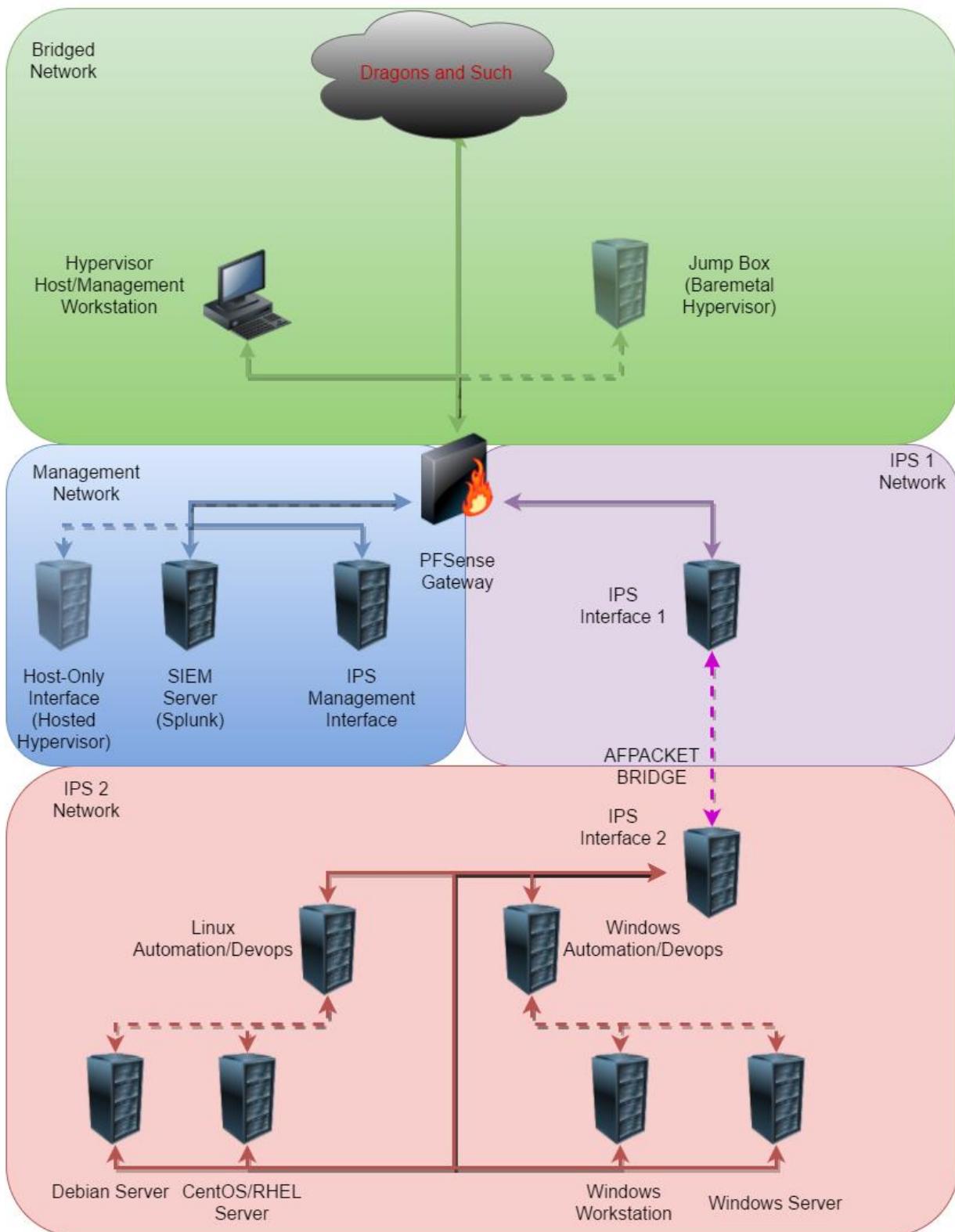
Depending on the number of VMs deployed or resources reallocated, the full lab might require probably around 18-22GB of RAM, somewhere around 650GB of disk, and a quad core CPU (or better) to fully support. Again, this depends on how you allocate resources and how many additional VMs you create. You may be able to get away with fewer resources, provided you have some of the VMs turned off when they are not in use (e.g. turn off the Windows Analysis VM when looking at Linux malware, and vice-versa).



They told me red team was sexy, and I bought into it. So now I have to build a pentesting lab to learn to be leet. Depending on skill level, technically, you could download the metasploitable 2 exploitation guide(<https://community.rapid7.com/docs/DOC-1875>), and get started immediately. Our baseline VM lab includes Kali Linux and Metasploitable 2, which is practically everything you need to get started. Even though metasploitable 2 is considerably old (and metasploitable 3 is now available), Metasploitable 2 is still a perfectly acceptable starting point to learning about exploitation and post-exploitation operations once you have successfully gained control of a system. Not only is this a perfectly good starting point to learn the basics of exploitation, you also have an IPS configured to generate alerts on the attacks you perform against the Metasploitable 2 VM. As a red teamer, it is important to understand what the blue team sees, or what possible alerts may be generated by your actions in order gain better tradecraft and improve your skills as a penetration tester. After all, offense informs defense, but the opposite is also true. Consider implementing more network security tools and technology in order to measure the impact of your attack and operations as you progress in skill.

The lab design I concocted above however, is considerably more complicated than the base lab network we worked on together. The network diagram above consists of another pfSense VM on the IPS 2 network acting as a firewall and DHCP server for the internal IPS 2 network. The firewall would only allow ports 80 and 443/tcp in and outbound. The only system that would be accessible from the Kali Linux VM on the IPS 1 network would be a system running a vulnerable web application that the attacker would have to figure out how to compromise in order to drop a webshell. From there, the attacker would need to discover how to pivot from this initial system to a Linux FTP server. From there discover how to pivot to Windows workstation that the administrator uses, and finally from there, discover how to pivot to the domain controller (usually a major objective in most pentesting engagements due to domain administrator accounts having access to essentially everything in the network). The idea behind this pentesting lab would be to discover which machines (if any) could talk to the internet, which machines the attacker would need to pivot off of, and learning various methods of acquiring elevated privileges in a target network until complete control is achieved. Sniffing network traffic for plaintext credentials, cracking weak passwords, passing the hash, beaconing implants, pivoting, etc.

To fully implement this pentesting lab, you would need somewhere in the neighborhood of around 17-22GB of RAM, and probably around 600+GB of disk space to store the VM files and snapshots, plus at least a quad core CPU. You might be able to get away with less RAM, but it would be a very tight fit. Bear in mind that this lab network is merely a suggestion. You could choose to implement it in full, part, or go do your own thing entirely; not unlike the malware analysis lab, the possibilities are endless.



To represent the IT folks and sysadmins in training, I also made a network diagram on how the lab network might look if it were reconfigured to support experimentation with IT Automation,

Monitoring and/or Devops tools. For instance, this might be an environment where you'd want to try out spiceworks, nagios, icinga, making your own software repository (e.g. APT mirror, or WSUS, etc.) as well as devops technologies such as containers (e.g. lxc, docker, etc.) and other automation tools (e.g. puppet, chef, ansible, etc.).

Not unlike the pentesting lab, the resources required to run this lab are likely to be extensive. Depending on RAM and disk space allocations, you would likely require around 20-24GB of RAM, and around 750GB to fully support this lab environment, plus the CPU requirements (again, minimum of a quad core CPU). You could probably get away with less resources, so long as you're not running any of the systems particularly hard, or if perhaps you had some VMs powered off while you were experimenting with others.

Summary

In this chapter, we're going to take the time to review all the ground we've covered.

What Have We Learned Today?

Let's review what we've accomplished here so far:

- General hardware requirements have been provided as a starting point for building your own VM lab
- A network diagram illustrating the end result of our work has been provided to grant a better understanding of the goal we worked towards
- Instructions have been provided on how to create and configure these VMs, and virtual networking on five different hypervisors:
 - Oracle VirtualBox
 - VMWare Workstation
 - VMWare Fusion
 - VMWare vSphere Hypervisor (ESXi)
 - Microsoft Client Hyper-V
- The VMs we have created and configured are as follows:
 - A pfSense VM with a set of core network services configured/provided (NTP, DHCP, DNS) and strict firewall rules for three separate network zones (Outbound/Bridged, IPS/test, Management/Secure)
 - An Ubuntu VM running either Snort or Suricata in inline mode via afpacket, serving as a network bridge between one segment of our IPS/test network, and the other half of the IPS/test network, as well as a Splunk universal forwarder with either the Unified2 or Splunk TA for pushing the IDS logs this system generates to our splunk instance

- An Ubuntu VM running a Splunk index/search head where we can query our IDS logs, and possibly expand our logging system to host additional logs
 - A kali linux VM we used to verify the IDS bridge works properly, and that the IDS properly sends IDS alerts to our Splunk instance
 - A metasploitable2 VM we used to serve as a test system in our IPS/test network, as well as to verify that the bridge our IPS VM provides is working properly
- The instructions for each hypervisor provide as much isolation between the hypervisor host and the VMs as possible through disabling excess features (e.g. guest extensions, copy and paste, drag and drop, etc.) and removing excess virtual hardware
- The hypervisor virtual networking, as well as the network configuration of the ips VM itself, should be able to support bridging the IPS 1 and IPS 2 network segments via the AF_PACKET bridging functionality that Snort or Suricata can provide
- Additional instructions have been provided to secure Windows workstations, and Windows hosted hypervisor systems against possible compromise from lab VMs
- Instructions have been provided to enable remote management (SCP, SSH, key-based authentication, adding static network routes for hosted hypervisors, etc.) for Windows, Linux, OSX, and BSD systems
- Instructions have been provided on how to enable access to VMs directly from a management workstation to VMs hosted on a baremetal hypervisor
- Instructions have been provided on how to enable access to VMs hosted on a baremetal hypervisor via a jump box (bastion host) system
- A simple script to automate system updates on Linux VMs has been provided, as well as instructions on how to implement it
- Instructions and scripts to automate the installation of Snort or Suricata have been provided
- Instructions have been provided on how to install a splunk indexer and search head on the siem VM, acquire dev licensing for a personal lab (optional), and finally, install a universal forwarder and necessary TAs to support parsing and forwarding ids logs for Snort or Suricata as necessary.
- Instructions have been provided on how to use Armitage and its “Hail Mary” function as a test battery to verify that the IDS is functioning and that data is being logged to the Splunk instance.
- All VMs should have a baseline snapshot to revert each VM to a known-good working state
- You have been given a couple of ideas in the form of general network diagrams on how the baseline lab network could be reconfigured to support red team, blue team, or IT monitoring for the express purpose of learning

Epilogue: We Need You (Now More than Ever)

If you’re still reading at this point, then gods help you, I have no idea why. When I first started writing this book (project AVATAR), I didn’t intend for it to be a primer on virtual networking or

crash-course on virtualization, but here we are, approximately 5 months later, over 400 pages, 80,000 words, and well over 500 screen captures and illustrations. I can't say that I have any regrets. I did what I could and shared what I know.

I did it because truth be told, I always wanted to be able to say I was a published author (even if it's self-published), and for the fact that we need more new IT and security professionals. Especially with the proliferation of Computer Network Operations and Defense, and the likelihood that this trend will continue into the future, we need you now more than ever. New blood needs guidance, and frankly I'm tired of the blank looks when I tell newbies to "Go build a lab!" and them not having any idea what I'm talking about. Now, there is a somewhat comprehensive starting point that I can jerk a thumb at and say, "Get to work. Come back to me if you have questions."

Can it be made better? Probably. There are several hypervisors I didn't create guides for, simply because I have never used them or never heard of them. Can it be more automated and devopsy? No idea; I have Grampa Simpson syndrome where devops is new and scary to me and I don't trust it yet. After all, these automated build tools, containers, and crap have a fatal flaw: if the container is broken, or the build process fails catastrophically, often times, there's no effective error-catching where the build process will detect that a step has failed and attempt to retry so many times before giving up. No, sometimes these tools just plow forward without a care in the world until you're left with a broken mess in an unknown state. Then the developers of said image or container simply shrug, say 'works on my machine', and go about their business. This is why I didn't go into devops and VM automation in this book. It is my fervent belief that as an IT or security professional, you need to know how to crawl before you walk, and walk before you run. This entails learning how to do things the hard way before being coddled by automation tools doing all of that stuff for you.

I did this because at some point, you're going to be the ones having to fight the good fight and continue to solve IT and Security problems long after I'm in the retirement home, killing ayys in XCOM 5 while enjoying my tapioca pudding and pill combo. So cheers to you, and here's hoping that I continue in my book writing career.

Tony "da_667" Robinson