Course Lab Introduction

We plan to use a virtual machine and docker containers for our labs and to achieve a clean and predictable working environment where all of us will be able to work without issues of replication and confusion that can stem from using custom Virtual Machines. In the next two sections, we have instruction on how to setup the virtual machines both locally and on Amazon AWS. For the AWS installation, please register and use an Amazon "Free Tier" account so that you can avoid incurring costs of using the virtual machines.

Ethical Considerations

An ethical hacker's primary purpose is to view security from the adversary's perspective in an effort to find vulnerabilities that could be exploited by bad actors. This provides defensive teams the opportunity to mitigate by devising a patch before a real attack can occur. This objective is served by executing **simulated cyberattacks in a controlled environment.**

Some ethical hacking guidelines (optional reading):

Ethical Hacker Reading (here)

Ethical Hacking terminology (here)

Ethical Hacking Tools & Techniques (here)

Please be responsible with the use of what you will learn in this class! With great power comes great responsibility.

Install SEED VM on VirtualBox

Account Information of this VM

- User name: `seed`
- Password: `dees`

Preparation

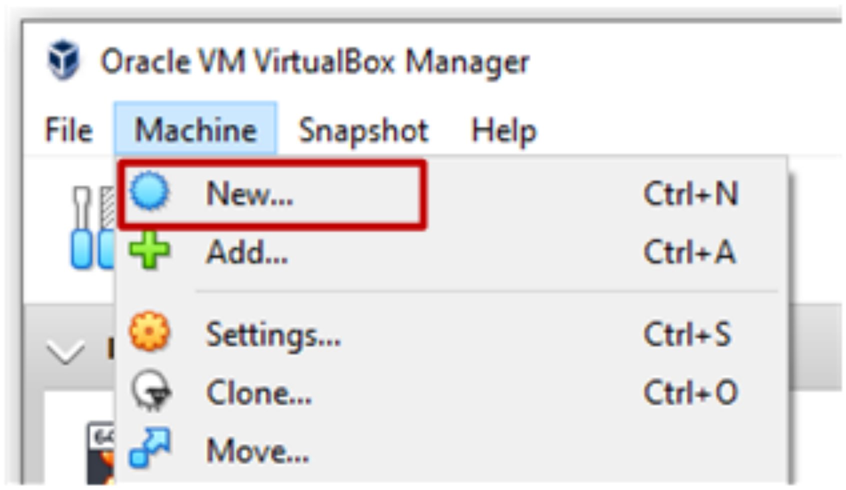
Before installing the SEED VM, please do the following:

- Install the free VirtualBox software first. The VM has been tested on Version `6.1.16`.
- **Use a pre-built SEED VM.** We provide a pre-built SEED Ubuntu 20.04 VirtualBox image (`SEED-Ubuntu20.04.zip`, size: 4.0 GB), which can be downloaded from the following links (chose **ONE**):

- Google Drive
- DigitalOcean
- MD5 value: f3d2227c92219265679400064a0a1287

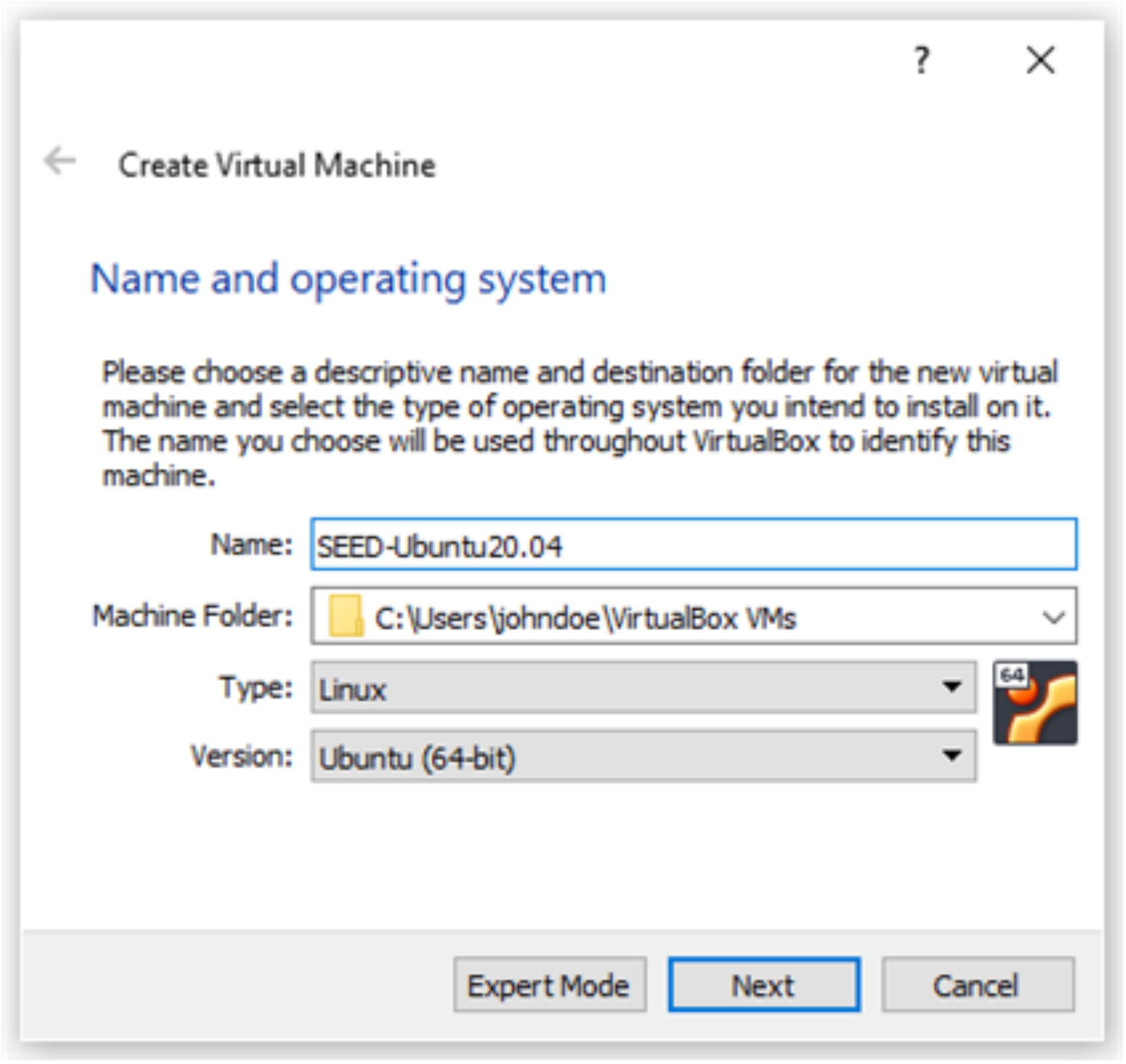
Step 1: Create a New VM in VirtualBox

We need to use `New` to create a new virtual machine.



Step 2: Provide a Name and Select the OS Type and Version

Our prebuilt Ubuntu 20.04 VM is 64-bit, so pick Ubuntu (64-bit).


Create Virtual Machine

Name and operating system

Please choose a descriptive name and destination folder for the new virtual machine and select the type of operating system you intend to install on it. The name you choose will be used throughout VirtualBox to identify this machine.

Name: SEED-Ubuntu20.04

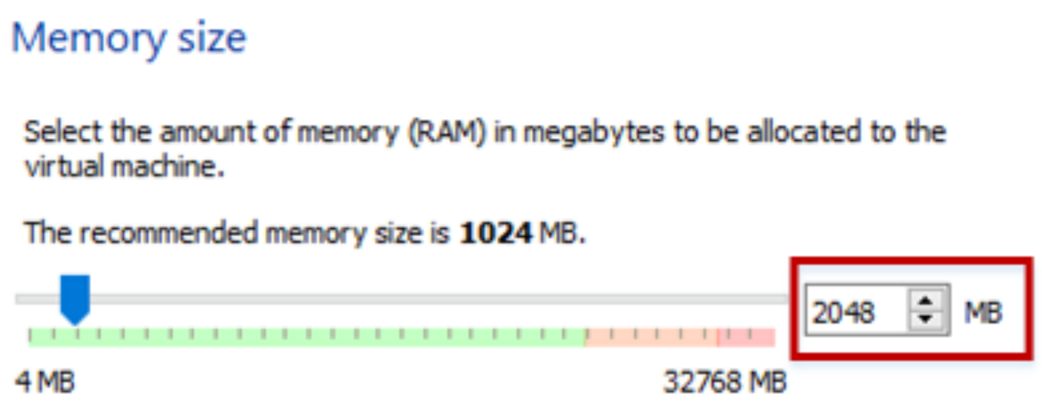
Machine Folder: C:\Users\johndoe\VirtualBox VMs

Type: Linux

Version: Ubuntu (64-bit)

Step 3: Set the Memory Size

We need to allocate dedicated memory for the VM. 1024 MB should be sufficient, but we recommend 2GB. If your computer has more RAM, you can increase accordingly. The more memory you give to the VM, the better the performance you will get.



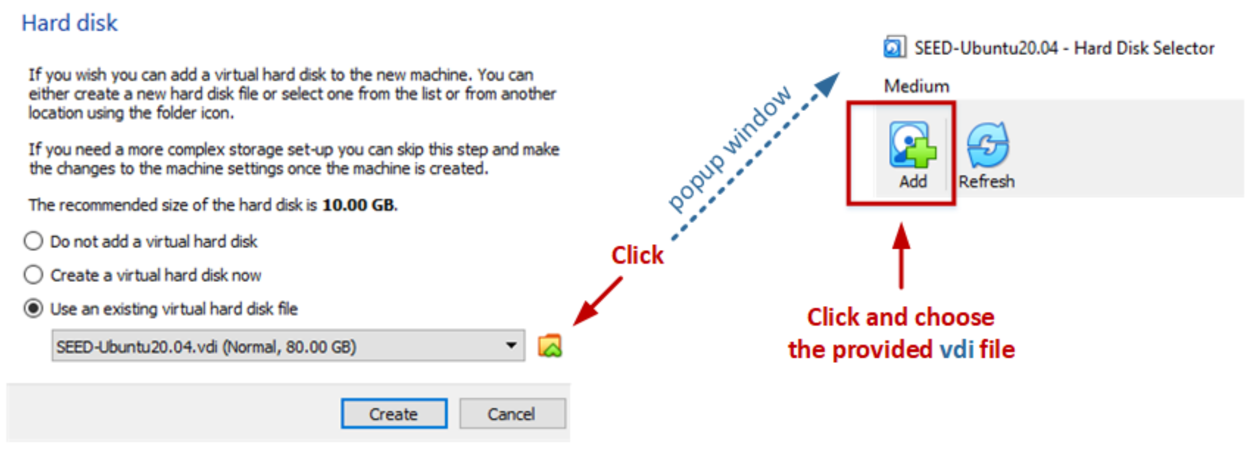
Memory size

Select the amount of memory (RAM) in megabytes to be allocated to the virtual machine.

The recommended memory size is **1024** MB.

Step 4: Select the Pre-built VM File Provided by Us

Click the folder image. On the popup window, use the `Add` button to select the `.vdi` file downloaded from the SEED website.



Hard disk

If you wish you can add a virtual hard disk to the new machine. You can either create a new hard disk file or select one from the list or from another location using the folder icon.

If you need a more complex storage set-up you can skip this step and make the changes to the machine settings once the machine is created.

The recommended size of the hard disk is **10.00 GB**.

○ Do not add a virtual hard disk
○ Create a virtual hard disk now
● Use an existing virtual hard disk file

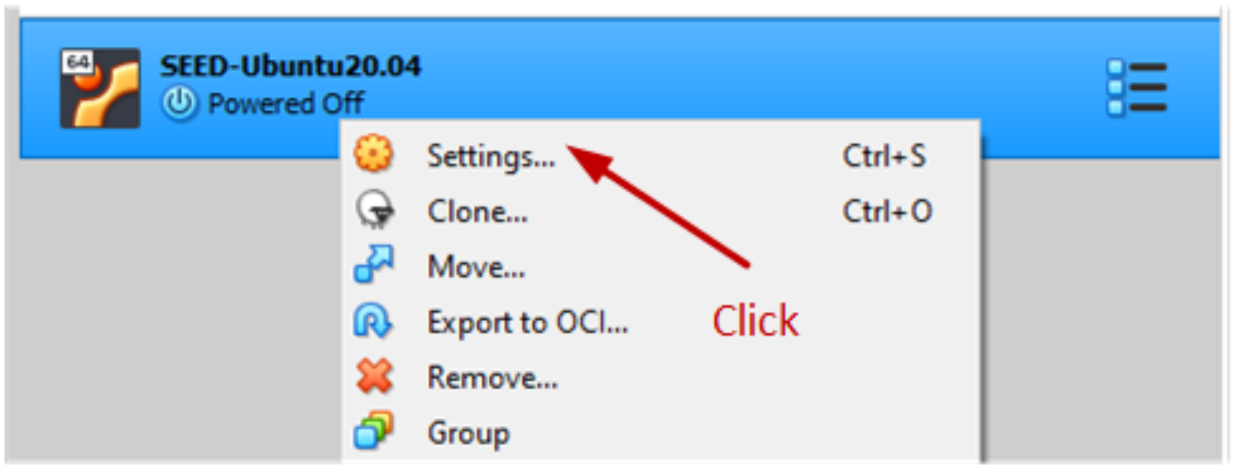
SEED-Ubuntu20.04.vdi (Normal, 80.00 GB)

Create
Cancel
Click
popup window
SEED-Ubuntu20.04 - Hard Disk Selector
Medium
Add
Refresh
Click and choose the provided vdi file

Note: If you get an error message saying that the UUID already exists, this is because the UUID in the selected `vdi` file is the same as the one used by an existing VM. You can either remove the other VM or change the UUID in the `vdi` file.

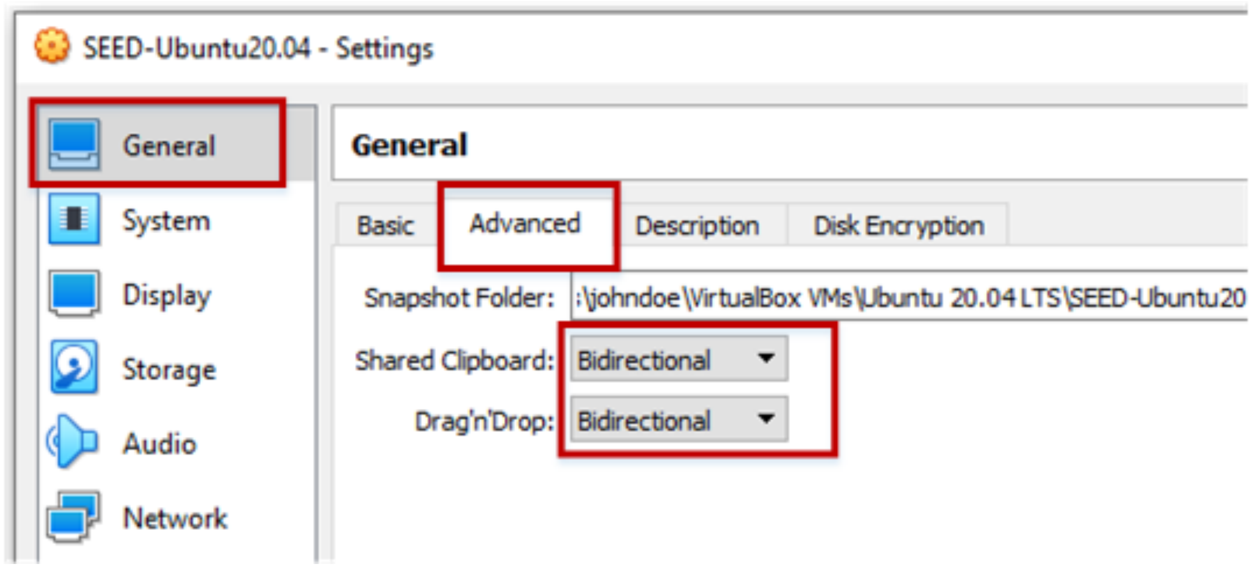
Step 5: Configure the VM

After the previous step, your VM will be created, and you will see it on VirtualBox's VM panel. We need to do some further configuration. Right-click the M, click the `Settings` option, and we will see the Settings window.



Step 5.a: Enable Copy and Paste

Go to the `General` category, and select the `Advanced` tab. Select `Bidirectional` for both items. The first item allows users to copy and paste between the VM and the host computer The second item allows users to transfer files between the VM and the host computer using Drag'n Drop (this feature is not always reliable).

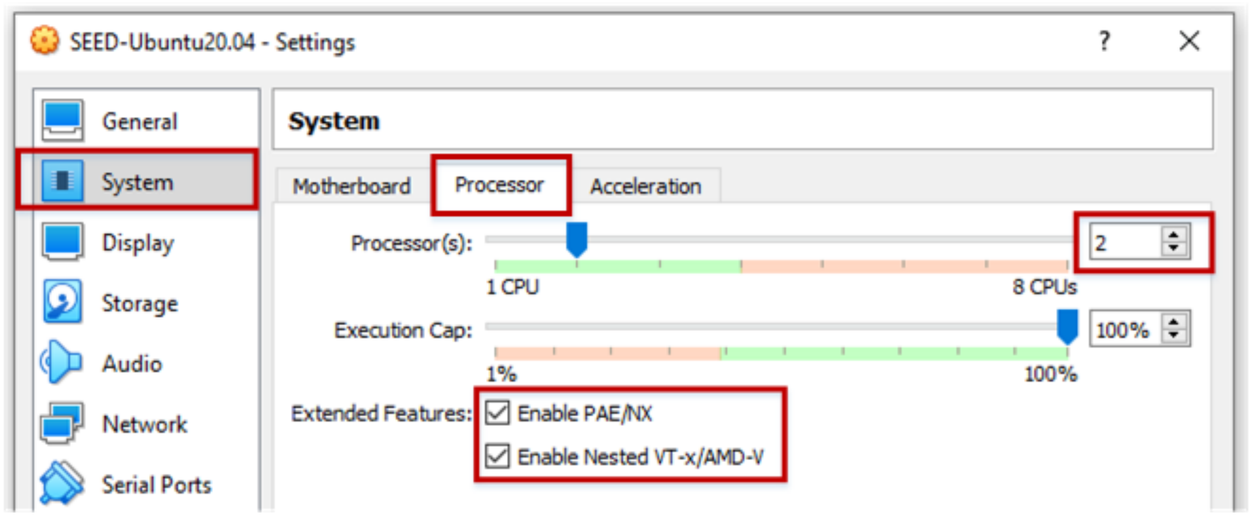


Snapshot Folder: :\johndoe\VirtualBox VMs\Ubuntu 20.04 LTS\SEED-Ubuntu20
Shared Clipboard: Bidirectional
Drag'n'Drop: Bidirectional

The copy-and-paste feature is very useful. If you can't do copy and paste, chances are that you forgot to do this step. You can always do it later by selecting the `Devices` menu item, and you will see the `Shared Clipboard` submenu.

Step 5.b: CPUs

Go to the `System` category, and select the `Processor` tab. Assign number of CPUs to this VM if you prefer. Although may be sufficient, if the performance seems to be an issue, increase the number.


SEED-Ubuntu20.04 - Settings

General
System
Display
Storage
Audio
Network
Serial Ports

System

Motherboard | Processor | Acceleration

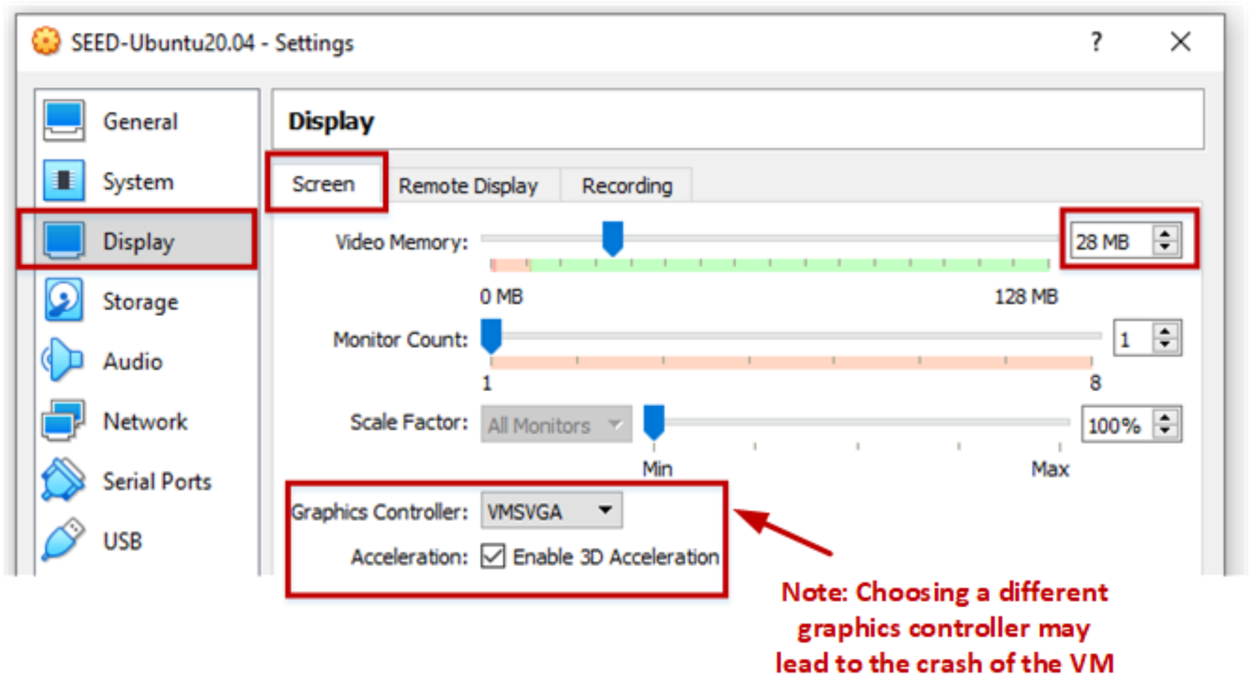
Processor(s): 2 (1 CPU – 8 CPUs)

Execution Cap: 100% (1% – 100%)

Extended Features: ☑ Enable PAE/NX ☑ Enable Nested VT-x/AMD-V

Step 5.c: Display

Go to the `Display` category, and select the `Screen` tab. If the display does not seem to work properly, try to increase the amount of video memory. In our testing, `28 MB` seems to be sufficient.


SEED-Ubuntu20.04 - Settings

General
System
Display
Storage
Audio
Network
Serial Ports
USB

Display

Screen | Remote Display | Recording

Video Memory: 28 MB (0 MB – 128 MB)

Monitor Count: 1 (1 – 8)

Scale Factor: All Monitors 100% (Min – Max)

Graphics Controller: VMSVGA

Acceleration: ☑ Enable 3D Acceleration

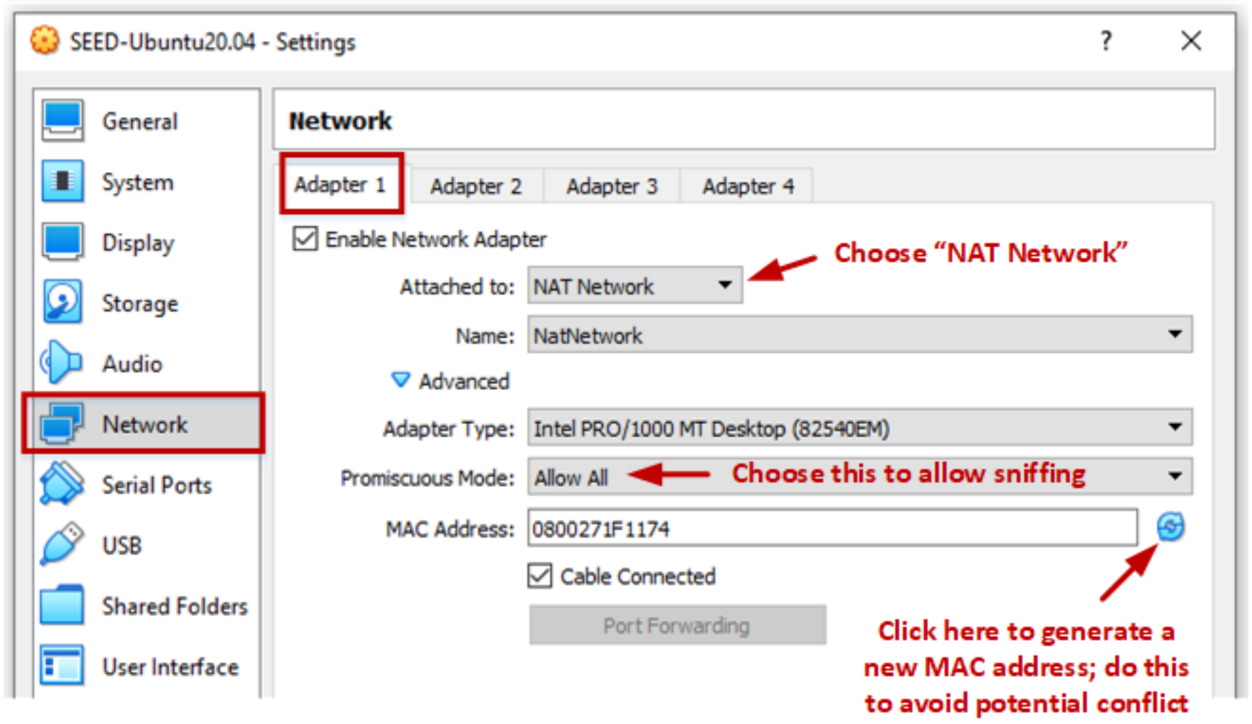
Note: Choosing a different graphics controller may lead to the crash of the VM

Note 1: Make sure to select `VMSVGA`, as choosing other graphic controllers may lead to the crash of the VM.

Note 2: If your computer's screen resolution is too high, the VM may not be able to match the high resolution. As results, your VM will be very small on your screen. To make it bigger, adjust the `Scale Factor` in this setting.

Step 5.d: Network

Go to the `Network` category, and select the `Adapter 1` tab. We will choose the `NAT Network` adaptor. Click the `Advanced` drop-down menu to further configure the network adaptor. If you don't see such an adaptor, see the note below.

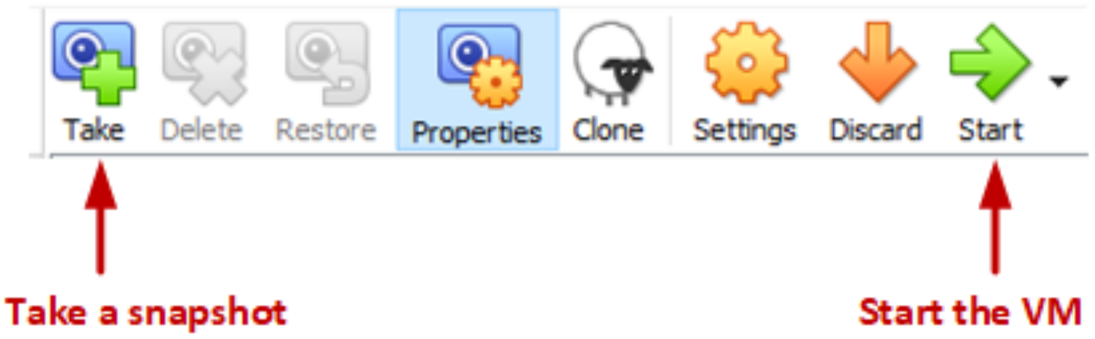


☑ Enable Network Adapter
Attached to: NAT Network
Name: NatNetwork
Advanced
Adapter Type: Intel PRO/1000 MT Desktop (82540EM)
Promiscuous Mode: Allow All
MAC Address: 0800271F1174
☑ Cable Connected

Note: If you don't see the `NAT Network` adaptor, you need to create one. Go to the `File` menu, click `Preferences....` You will see a popup window. Go to the `Network` tab, and you can add a new `Nat Network` adaptor there.

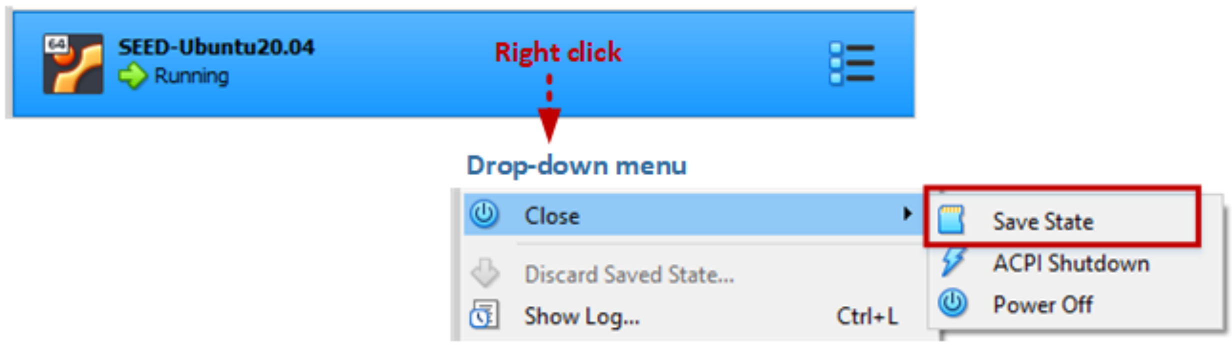
Appendix A: Start the VM and Take Snapshot

We can now start the VM. You can also use the `Take` button to take a snapshot of your VM. This way, if something goes wrong, you can roll back the state of your VM using the saved snapshots.



Appendix B: Stop the VM

There are many ways to stop the VM. The best way is to use the `Save State`. This is different from shutting down the VM. It saves the current VM state, so next time when you restart the VM, the state will be recovered. Moreover, the speed is also faster than booting up a VM.

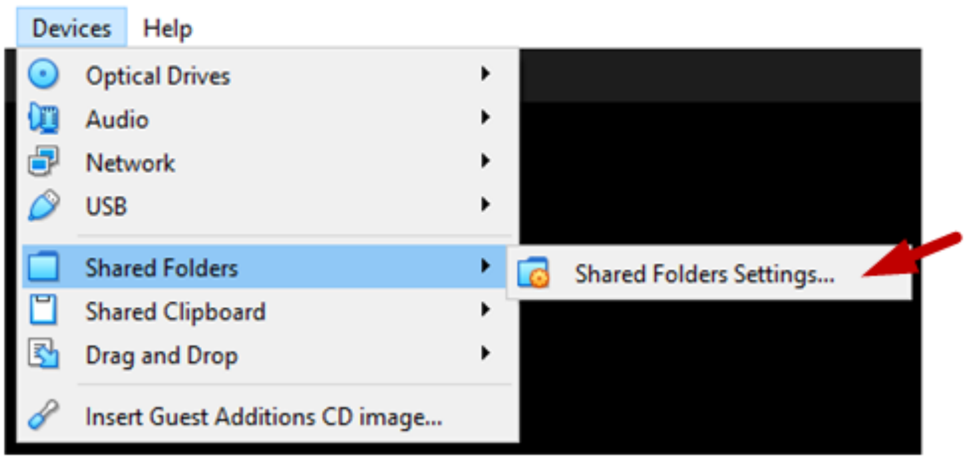


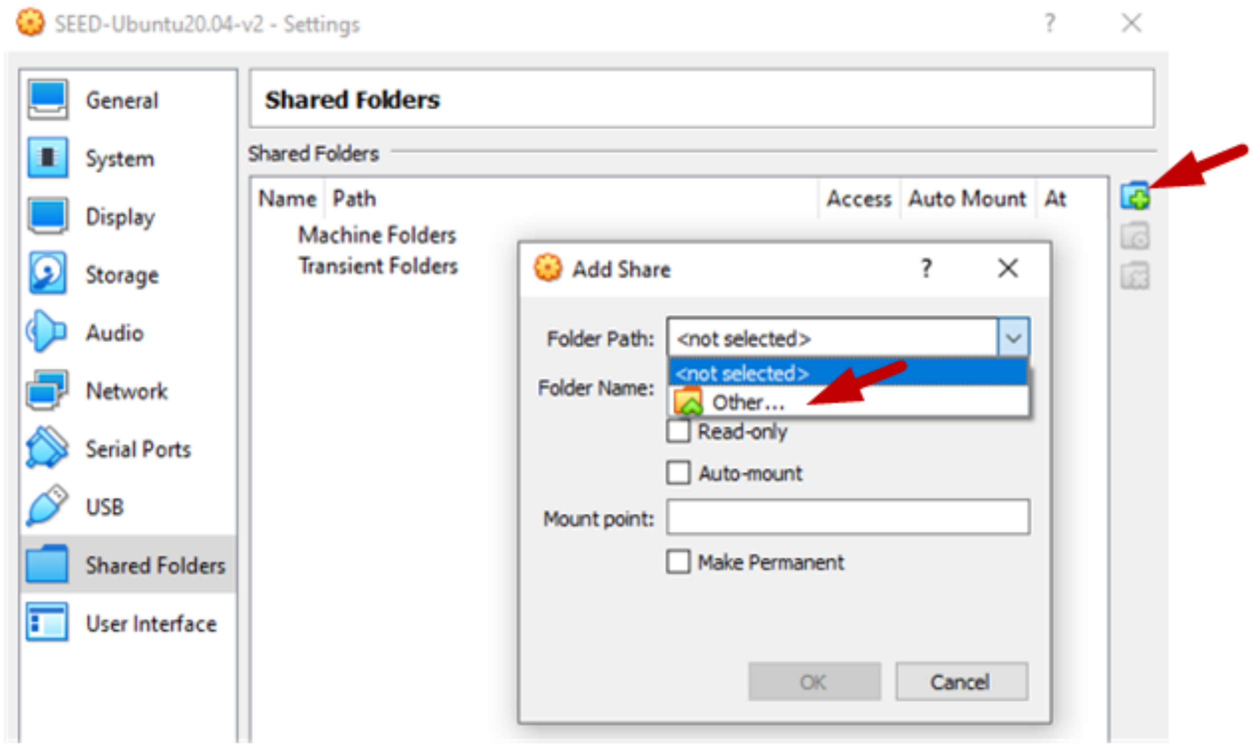
Appendix C: Creating a Shared Folder

Sometimes, we need to copy files between the host machine and the VM. If you are using the VM from the cloud, you can see our cloud VM manual for instructions. Or, you can just use a cloud storage service, such as Dropbox and Google Drive to share files between your VM and host machine.

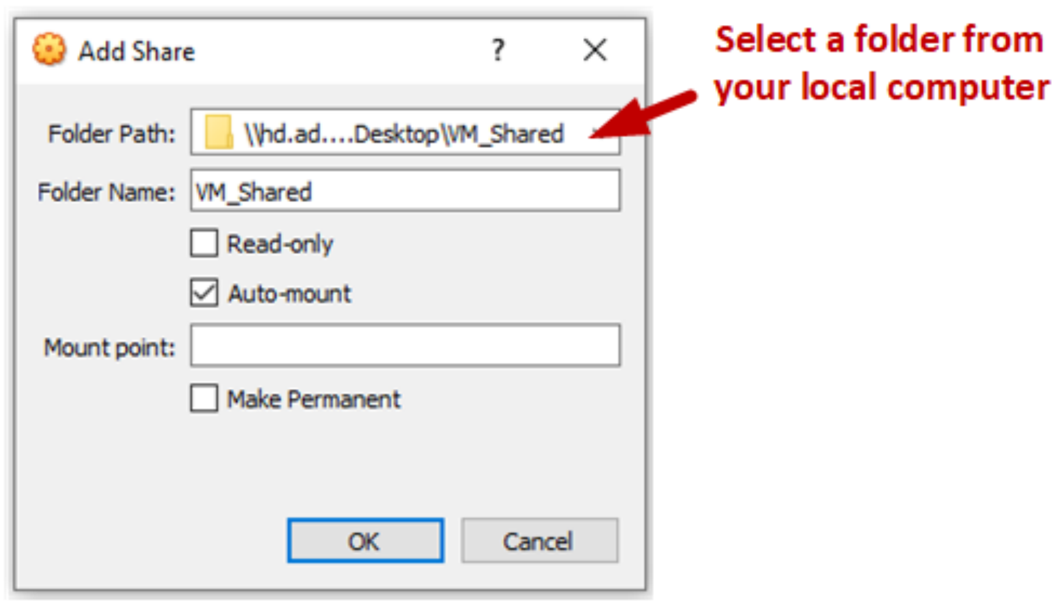
If you run the VM on your local computer, you can create a shared folder between your computer and the VM.

Step A. First you need to create a folder on your local computer (or using an existing folder). We will let the VirtualBox know that this folder should be shared with the VM. Go to the following menus:





Once you see a `Add Share` popup window, select the folder that you want to share, click OK, and you will see that the folder is now made available for sharing.



Folder Path: \\hd.ad....Desktop\VM_Shared

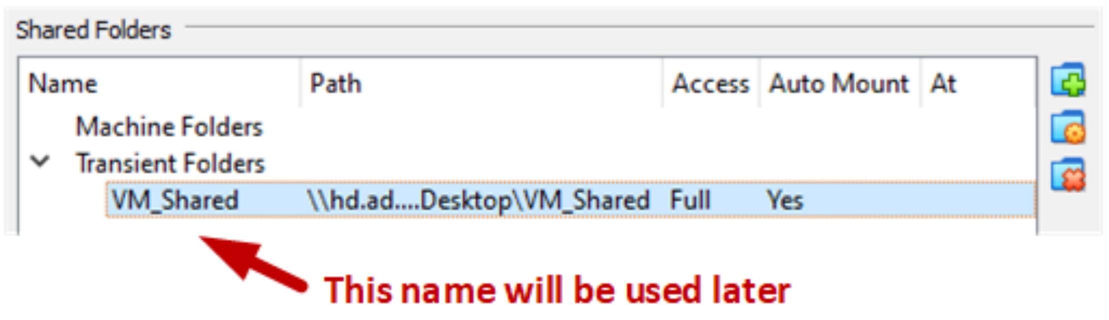
Folder Name: VM_Shared

☐ Read-only

☑ Auto-mount

Mount point:

☐ Make Permanent



Name	Path	Access	Auto Mount	At
VM_Shared	\\hd.ad....Desktop\VM_Shared	Full	Yes	

Step B. Inside the VM, we need to mount the shared folder somewhere. Let's mount it to the home directory as a folder `Share`. We will create a folder called `Share` in the home directory, and then mount the shared folder `VM_Shared` to this `Share` folder using the following command. After that, you can access the shared folder from `~/Share`.

```
$ mkdir -p ~/Share
$ sudo mount -t vboxsf VM_Shared ~/Share
```

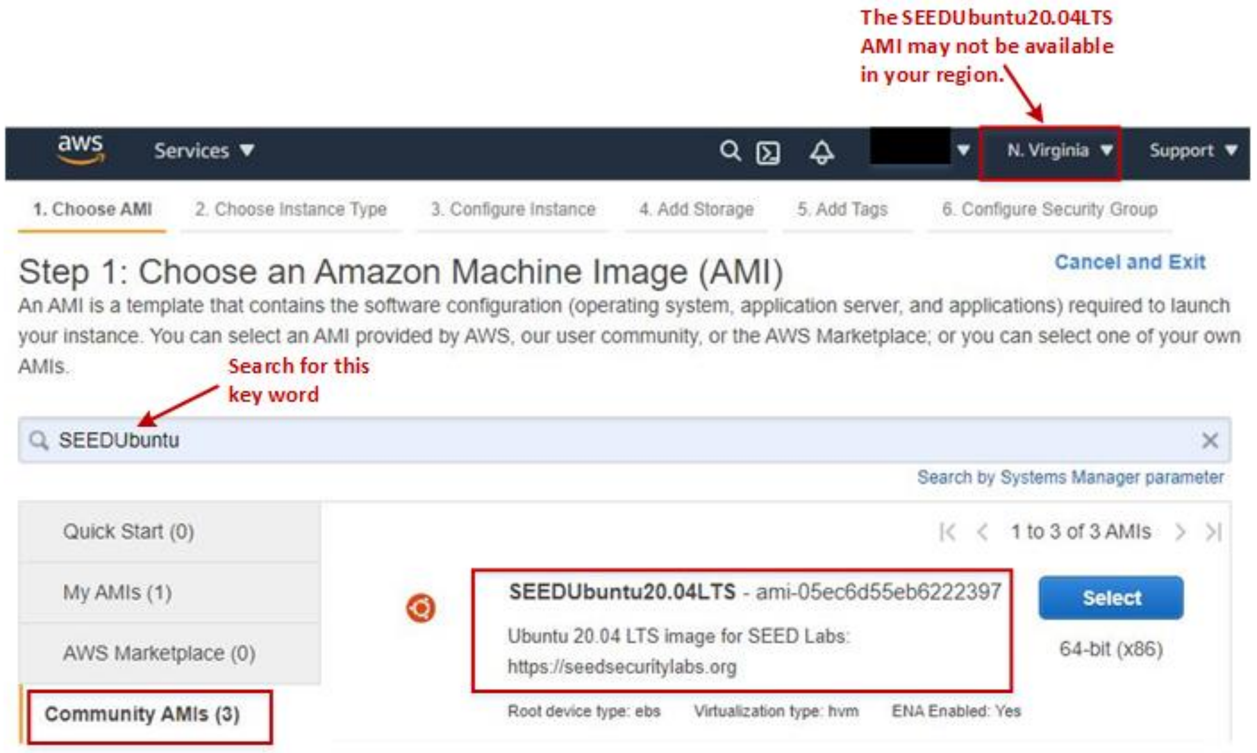
Creating SEED VM on Amazon Web Services (AWS)

(Not needed for initial Labs, costs might be incurred if you use your personal account!)

Step 1: Choose an Amazon Machine Image

We will build a Ubuntu Server 20.04 64-bit (x86) image. There are two options.

- **Use a SEEDUbuntu20.04 AMI:** We have created a community AMI, and made it available in several regions. This image already installed all the software packages for the SEED labs, and is already configured. If you use this image, your job is much simplified.



Step 1: Choose an Amazon Machine Image (AMI)

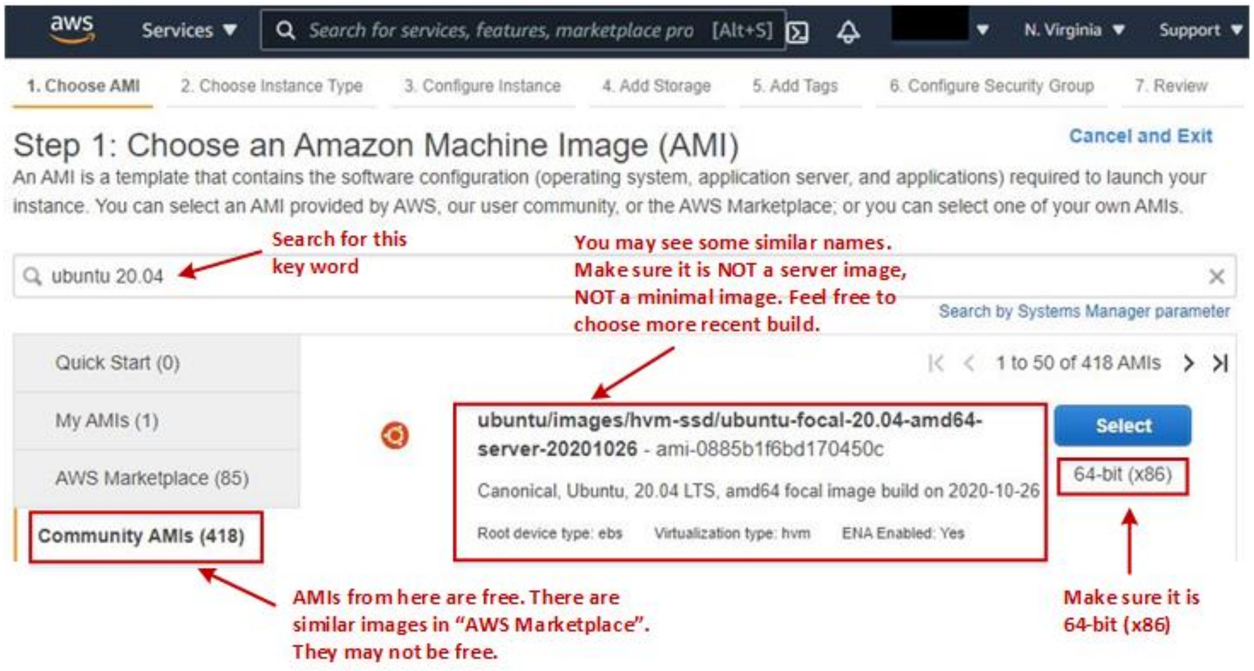
An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace; or you can select one of your own AMIs.

SEEDUbuntu20.04LTS - ami-05ec6d55eb6222397

Ubuntu 20.04 LTS image for SEED Labs: https://seedsecuritylabs.org

Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

- **Use an Ubuntu 20.04 AMI:** If the SEEDUbuntu20.04 AMI is not available in your region, or you just want to build a SEED image yourselves, you can select a generic Ubuntu 20.04 image from community AMIs.


Step 1: Choose an Amazon Machine Image (AMI)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace; or you can select one of your own AMIs.

Search for this key word

You may see some similar names. Make sure it is NOT a server image, NOT a minimal image. Feel free to choose more recent build.

ubuntu/images/hvm-ssd/ubuntu-focal-20.04-amd64-server-20201026 - ami-0885b1f6bd170450c

Canonical, Ubuntu, 20.04 LTS, amd64 focal image build on 2020-10-26

Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

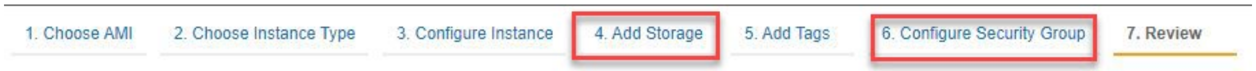
AMIs from here are free. There are similar images in "AWS Marketplace". They may not be free.

Make sure it is 64-bit (x86)

Step 2: Choose an Instance Type

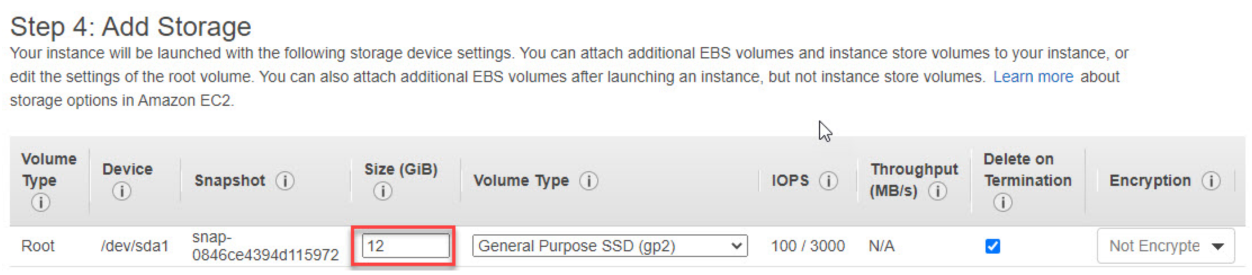
We need to choose an instance type for our virtual machine. Different type has different cost. The minimal configuration for SEED VM is 1 vCPU, 2GB of memory, and 10 GB of disk space. The `t3-small` type is recommended (it has 2 vCPU and 2GB of memory). If you are concerned about the cost, you can start with a small configuration, and change it easily later (using the `Actions` --> `Instance setting` menu).

After selecting the type, you can click `Review and Launch`, and you will directly jump to the `Review` Step. We need to go back a little bit to Steps 4 and 6 to add storage and configure the firewalls.



Step 4: Add Storage

By default, AWS sets the storage size to 8GB. This is barely enough for SEED labs. We need to increase it to 12GB minimal. You should be noted that the size does affect the cost, although not by much.



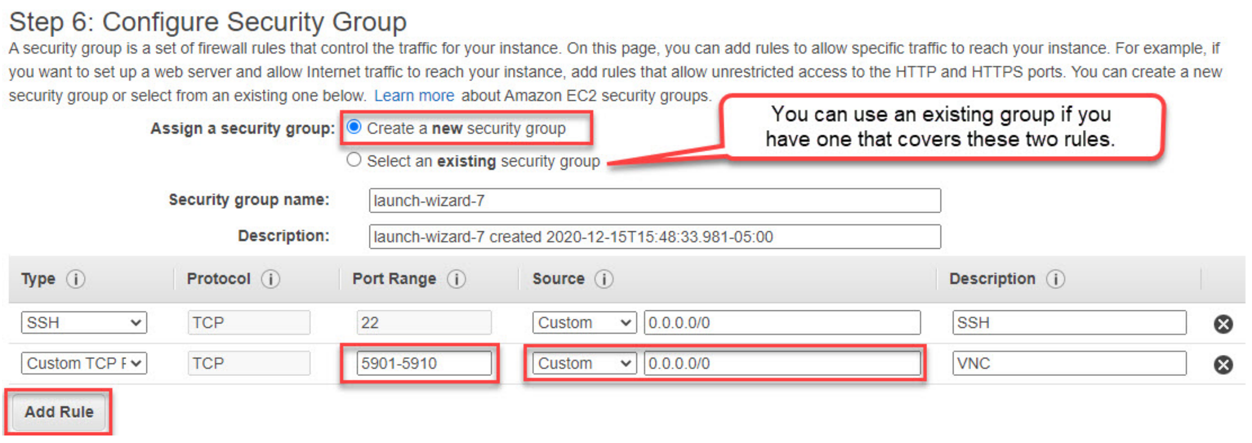
Step 4: Add Storage

Your instance will be launched with the following storage device settings. You can attach additional EBS volumes and instance store volumes to your instance, or edit the settings of the root volume. You can also attach additional EBS volumes after launching an instance, but not instance store volumes. Learn more about storage options in Amazon EC2.

Volume Type	Device	Snapshot	Size (GiB)	Volume Type	IOPS	Throughput (MB/s)	Delete on Termination	Encryption
Root	/dev/sda1	snap-0846ce4394d115972	12	General Purpose SSD (gp2)	100 / 3000	N/A	✓	Not Encrypte

Step 6: Configure Security Group (Firewall)

We need to add firewall rules to allow two types of access, SSH and VNC. By default, AWS already added the rule to allow SSH, so we just need to add one rule to allow VNC. VNC server listens to port 5900 + N, where N is the display number. For display `:1`, the port number is `5901`. To allow VNC server to have multiple displays, we specify a port range in the rule. After this step, the VM instance is created, and we are ready to launch this instance.



Step 6: Configure Security Group

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. Learn more about Amazon EC2 security groups.

Assign a security group: [✓ Create a **new** security group] Select an **existing** security group

You can use an existing group if you have one that covers these two rules.

Security group name: launch-wizard-7

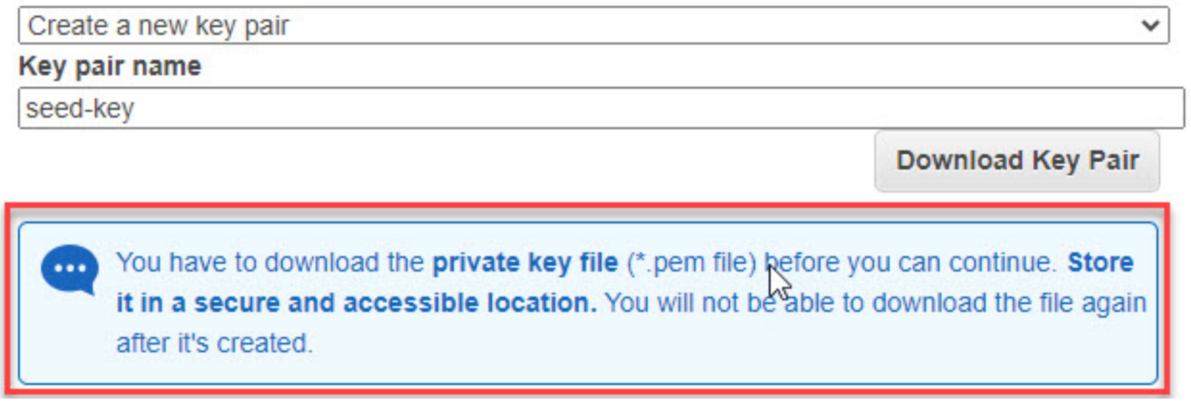
Description: launch-wizard-7 created 2020-12-15T15:48:33.981-05:00

Type	Protocol	Port Range	Source		Description
SSH	TCP	22	Custom	0.0.0.0/0	SSH
Custom TCP F	TCP	5901-5910	Custom	0.0.0.0/0	VNC

Step A.1: Create/Select Key Pair

To SSH into the VM instance, we need to create a key pair or use an existing one that you created before. AWS will save the public key part of this pair to the `.ssh/authorized_keys` file inside the accounts created in the VM. This allows you to log into those accounts using the private key part of the pair.

If you choose to create a new key pair, make sure you download the key file, and save it in a secure place (see the notice in the figure). You need the key when you use a third-party SSH client to log into the server.

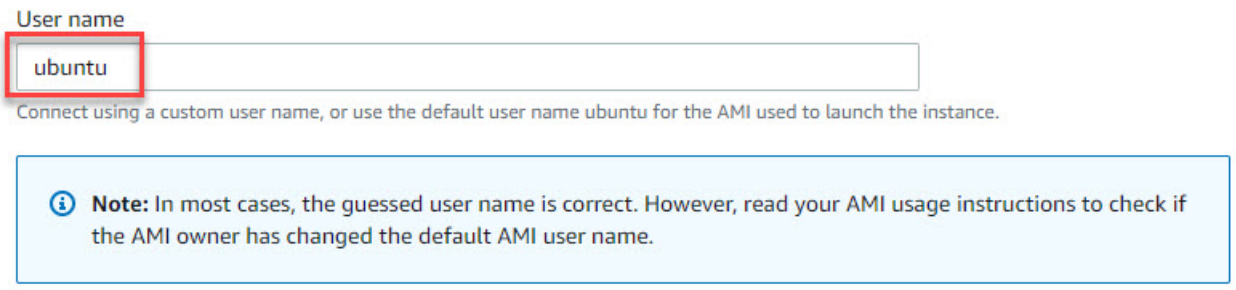

Create a new key pair
Key pair name
seed-key

You have to download the **private key file** (*.pem file) before you can continue. **Store it in a secure and accessible location.** You will not be able to download the file again after it's created.

Step A.2: SSH into VM

There are two typical ways to SSH into an AWS VM. Before doing that, we need to make sure that the VM instance is running. You can start it from the AWS console if it is not running.

From a web browser: After selecting an VM instance on the AWS Console, you can click the `Connect` button. A new window will come up. You need to type a user name. In a typical Ubuntu 20.04 AMI, `root` and `ubuntu` are the accounts created, and their `.ssh/authorized_keys` have already been set up using the keys you provided in the previous step.


User name
ubuntu
Connect using a custom user name, or use the default user name ubuntu for the AMI used to launch the instance.

Note: In most cases, the guessed user name is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI user name.

From a Terminal: If you prefer to SSH into the VM from a third-party SSH client, you can do that with the key that you have downloaded in the previous step. There are many SSH client programs, and their usages are different. The following example is for Linux:

```
ssh -i <private key file> ubuntu@<external IP address of instance>
# example: ssh -i seed-key.pem ubuntu@12.34.56.78
```

Step 2 (Option A): Access the VM Using VNC

For most labs, being able to SSH into the VM should be sufficient. Some labs do need to access GUI applications on the VM, such as Firefox and Wireshark. If your network bandwidth is not too bad, being able to get a graphical desktop of the remote VM is always more desirable than getting a text terminal via SSH. We will use VNC (Virtual Network Computing) to get the remote desktop.

- **On the cloud VM:** We need to make sure that we are in the `seed` account. If you are still in the default account, do the following, and you will be in the `seed` account:
- `sudo su seed`

  Our installation script has already installed the TigerVNC server program on the VM. You need to start the server.

  ```
  vncserver -localhost no
  ```

  By default, TigerVNC server only listens to localhost/127.0.0.1. The purpose of the `-localhost no` option means accepting access from the outside. When we first start the `vncserver`, we will be asked to provide a password. Make sure this password is strong enough. Moreover, VNC communication itself is not encrypted, so you should not send anything personal. If you do want to secure it, you can run an SSH tunnel or VPN tunnel to protect the VNC communication.

- **On your computer:** You need to have a VNC viewer installed on your computer, such as TigerVNC, and RealVNC. If you prefer other VNC viewers, it is fine. Most of them are compatible with one another. Some users have reported that TigerVNC have issues on macOS, but RealVNC has no problem.

  Start your VNC viewer program, and type the IP address of the VM, along with the port number, such as `35.236.203.131:5901`. Most cloud VMs have two IP addresses; make sure you use the external IP address, not the internal one. You will be prompted for password, which is the one you typed when you first run the VNC server. If everything is done correctly, you will see the desktop of your remote VM.

- If you VNC does not work, check your firewall to make sure TCP traffics to the port `5901` on the VM is allowed. Also check whether your VNC server is running properly. Here are some useful commands to help you manage the VNC server on the VM:
- `vncserver -list       # List the VNC server sessions`
- `vncserver -kill :1    # Kill the session for :1 display`

Step 2 (Option B): Access the VM Using SSH

To run VNC, you need to have reasonable bandwidth. If your VNC performance is bad, you should switch to SSH. You can get by with many of the SEED labs using just terminals. There are many ways to SSH into the cloud VM:

- Most cloud platforms provide a default browser-based SSH client. Google cloud's SSH client even allows you to upload and download files, which is very convenient.
- You can also find many third-party SSH clients. Some clouds may have disabled the password authentication in SSH, so you have to use public keys for the authentication. You need to generate public/private key pairs on your SSH client machine, and save the public key into the `/home/seed/.ssh/authorized_keys` file on the server machine. You can easily find instructions from online resources, so we will not provide one here.

Notes on Cost

Without using a "Free Tier" account or unless you have a special deal with cloud company, you will be charged for using the cloud VM. Please keep an eye on your bill, because sometimes, there are costs that you may not be aware of, such as bandwidth cost, storage cost, etc. Understanding where your expense is can help you reduce it. Moreover, to avoid wasting money, remember to suspend your VMs if you are not working on them. Although a suspended VM still incurs storage cost (usually very small), it does not incur any computing costs. You can easily resume them when you are ready to continue your work.