TCSS 562: Software Engineering for Cloud Computing
Fall 2021
http://faculty.washington.edu/wlloyd/courses/tcss562

School of Engineering and Technology
University of Washington – Tacoma

Tutorial 3 – Best Practices for Working with Virtual Machines on Amazon EC2

Disclaimer: Subject to updates as corrections are found
Version 0.12
Scoring: 20 pts maximum
** ***This tutorial can optionally be completed in two-person teams.*** **

The purpose of this tutorial is to introduce the Amazon Elastic Compute Cloud (EC2) service, an Infrastructure-as-a-Service platform that provides virtual machines (i.e. VMs) on demand, and discuss best practices for cost saving when creating/running VMs on Amazon EC2.

This tutorial will cover the use of VMs known as spot instances which are available at a reduced price when there is excess capacity across particular cloud regions. We will also discuss best practices for data storage when working with EC2 VMs including the use of Elastic Block Store (EBS) persistent disk volumes, as well as local ephemeral disks. We will discuss ways to create, suspend, and resurrect VMs based on when they are actually used to save costs. To complete this tutorial, the use of Amazon Cloud Credits is required as creating EC2 spot instances is not free.

Instance Types

Amazon EC2 instances, also known as virtual machines or VMs have a letter designating the family, and a number identifying the generation. Presently there are 6 generations (1-6). Most families do not have VM types spanning all 6 generations of instances (only m).

CPU Instances combine a family letter and generation number followed by a period and a size (small, large, xlarge, 2xlarge, 4xlarge, 8xlarge, etc.) The sizes correspond to the same configuration, but an increasing quantity of available resources (e.g. CPU cores, memory, network capacity, storage capacity). The most common instance families include:

<u>Commonly used EC2 instance families:</u>

c	Compute Optimized Instances	Typically fast CPUs, but less memory
m	General Purpose Instances	More memory than C-family, slower CPUs
r	High Memory Instances	More memory than C or M
t	Burstable instances (cpu-time limited)	Lower-cost instances with CPU quotas

<u>Less common families include:</u>

f	FPGA Instances	Instances with an on-board programmable FPGA
h/hs	Storage-optimized instances	Instances with enhanced disk capacity (HDDs)
i	Storage-optimized instances	Instances with enhanced disk capacity (SSDs)
p/g	GPU Instances	Instances with on-board GPU(s)
x	Extra-high memory	Instances with extreme memory

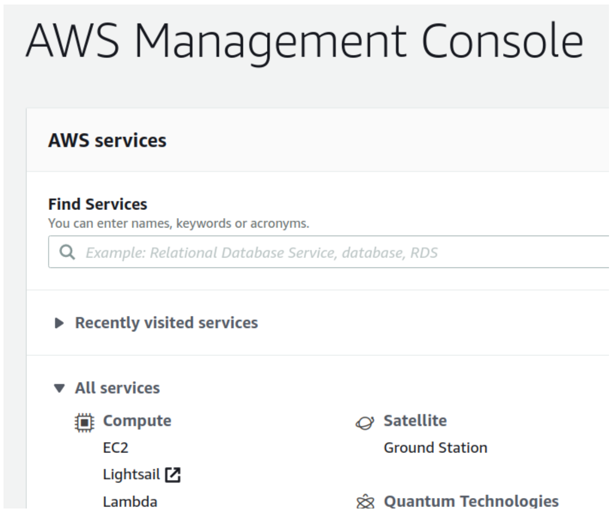
Note on how the CPUs supporting EC2 VMs are evolving:
Presently 6th generation instances are based on the proprietary Amazon Graviton2 processor which is an ARM processor which stands for Advanced RISC Machine. RISC stands for "Reduced Instruction Set Computing". This CPU architecture is different than that used by classic Intel Complex Instruction Set Computing (CISC) processors known as x86. **The key detail is that software for Intel (x86) may not run on ARM directly without recompilation or emulation.** Compilers (e.g. C/C++, etc.) create machine specific code for the target processor. Linux applications generally are quite portable from x86 to ARM through recompilation. Windows 10 is not supported on ARM processors, but applications that haven't been compiled may only run through 32-bit emulation.

The website **https://www.ec2instances.info/** has an excellent tool to query/search the various types of VM instances available from EC2. On this page, check out the "Columns" drop-down list which is used to customize meta-data returned about instance types. It is possible to examine the **Physical Processor** and **Clock Speed(GHz)** of instances. These are key parameters that impact a VM's performance - more than an instance's Name or API Name. The website also summarizes Linux/Windows on-demand and reserved pricing. On demand pricing is the standard price offered for ad-hoc usage. Reserved processing requires a long-term usage contract where they buyer commits to a minimum run-time. Generally, the customer is billed 24/7 for reserved instances even if they are not active. Minimum contract commitment is generally 1-year. Recently third parties have formed a new marketplace to resell unused reserved instance time.

It is recommended to complete the tutorial using a web browser from the same operating system as your Putty or SSH client (e.g. Ubuntu terminal). This will make it easy to copy-and-paste between the browser and terminal.

Starting Out

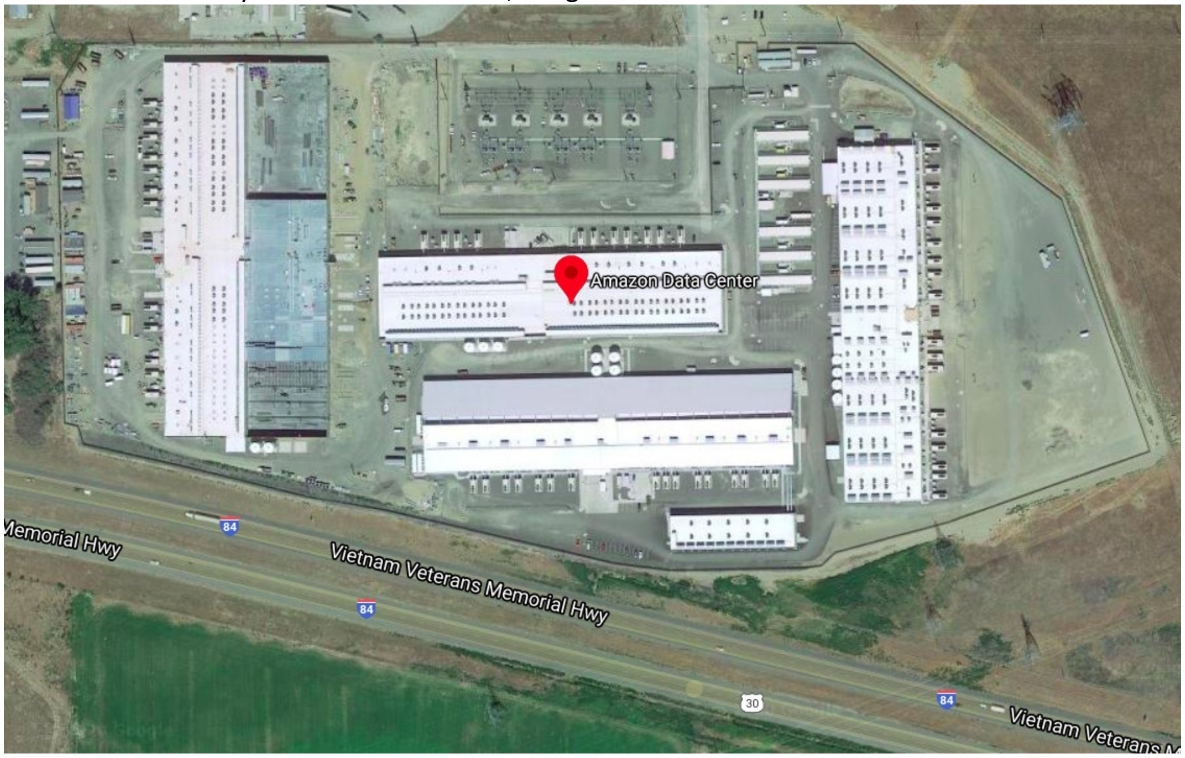
Log into the AWS Management console, and select the EC2 service:



Regions and Availability Zones

The United States has four distinct AWS regions. Each region consists of at least three distinct availability zones. Think of an availability zone as a separate physical data center. Think of a physically separate facility that requires travel (e.g. by car) to reach. Availability Zones are regionally local, but it may take from ~ 10 to 60 minutes or more to travel between the separate facilities. Availability Zones have separate power sources, and are ideally designed to minimize the likelihood that all zones within a Region can fail at the same time. Data centers may also be located in areas with low cost and highly reliable electricity.

Satellite view of availability zone near Boardman, Oregon:



US Regions:

Name	Location	Notes
us-east-1	Northern Virginia	The first AWS region
us-east-2	Ohio	Currently the least expensive
us-west-1	Northern California	Most expensive US region
us-west-2	Oregon	Boardman and Umatilla Oregon

It is recommended to use the **us-east-2 Ohio** region for cloud resources throughout TCSS 562. Currently (Fall 2020) the Ohio region has some of the lowest prices for on-demand and spot VM instances. Consolidating resources in one region reduces duplication of data and also the likelihood of accidentally creating resources (e.g. VMs) in another region and forgetting to terminate them resulting in accidental charges. While network latency, the time it takes for network traffic to route out and back to us-east-2 (Ohio) from Washington state will be slightly longer, costs for using resources will be lower. One way to mitigate this latency is by launching VMs to serve as service clients in the availability zone.

The AWS region can be changed using the dropdown list in the upper right-hand corner of the GUI. Select "**Ohio**".

Spot Instances

Spot instances provide savings vs. on-demand VM instances. Amazon sells VMs at reduced "spot" market prices to sell unused cloud capacity. Pricing is generally only 1/3 to 1/4 of full price. The caveat is that when demand spikes for a particular VM type in a specific Region and Availability Zone, instances will be automatically terminated when the market price (e.g. going rate) exceeds the customer's bid price (called a spot price). There is a 2-minute warning provided to users when this occurs which can be checked for.
(see article: https://aws.amazon.com/blogs/aws/new-ec2-spot-instance-termination-notices/)

In nearly all cases, **spot instances** should be used for course work/research.

Cost Saving Measure #1

ALWAYS USE SPOT INSTANCES FOR COURSE/RESEARCH RELATED PROJECTS

VM Disk Storage

Every VM needs a disk. The disk volume where the operating system is installed is called the ROOT volume. There are two primary types of disk volumes on EC2: **Elastic Block Store (EBS)** and **Instance Store**. EBS disk volumes are hosted by remote servers that are attached to Amazon EC2 virtual machines using a very high speed network connection. EBS volumes are replicated to provide data redundancy and to improve Input/Output (I/O) performance. Instance-store volumes are hosted using local disks connected directly to the server hosting your VM. This was the predominant type of disk on EC2 with 1st and 2nd generation instances. Starting with 3rd generation instances, EBS volumes became more prolific/common. Starting with 4th generation instances, ***only*** EBS volumes are allowed to serve as the ROOT volume on a VM. This has important cost implications. With an instance store volume, there is no additional storage cost for the VM. Because this feature is now only available with legacy instances (3rd generation and before) it is no longer a best practice / cost savings measures.

With EBS volumes, in addition to standard VM costs, every VM incurs a 24/7 storage charge for the EBS disk volume. This results in a hidden-cost of cloud computing. For every VM you pay twice. Once for the VM, and again for the disk. The extra cost is "justified" because EBS volumes are "persistent" in that they can optionally be RETAINED/KEPT to preserve all data when a VM is shutdown intentionally or accidentally. The additional cost is 10c/GB/month. The standard EBS ROOT volume size is currently 8GB for Ubuntu Linux. This results an additional charge of 80 cents per month. With larger ROOT volumes, the additional cost can become substantial. For example a 800 GB ROOT volume will add an additional ~11 cents per hour to the cost for the VM. This can be more than the actual VM cost !!

WARNING ABOUT SAVING EBS VOLUMES

A common accident that results in unexpected cloud computing charges, is to create a VM, and mark the VM to have a persistent EBS volume that does not get deleted on termination. When the volume is large (e.g. 800 GB), this results in a charge of $80/month. Think of an EBS VOLUME as a <u>LIVE PIECE of HARDWARE</u>. When you have one in your account it can be attached immediately to *any VM* with no preparation/initialization. This "LIVE" behavior justifies the high price. Some server is actively preserving your data in a LIVE-STATE so that it is available in a split-second for connection to a new VM. The key take home message to save cloud computing costs is:

Cost Saving Measure #2:

**NEVER LEAVE AN EBS VOLUME IN YOUR ACCOUNT
THAT IS NOT ATTACHED TO A RUNNING VM**

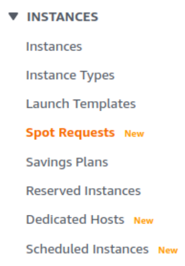
Starting with 5th generation instances, EC2 offered "d" versions. The "d" indicates a local ephemeral (which means temporary) disk is available to the VM. These ephemeral local disks are not backed up when the VM terminates. All data is lost unless the user writes code/scripts to back up the data. A huge advantage of the "d" instances though is that this local storage is very inexpensive. For spot instances in the Ohio region, in some cases it is FREE! The spot prices for c5.large and c5d.large instances was the same in October 2020. A local ephemeral storage is good for applications that need access to large amounts of local, fast disk space.

Here is a feature comparison of instance store vs. EBS volumes:

Instance Store Volumes	Elastic Block Store (EBS) Volumes
- Disk hosted local to VM (on same physical server) - Provides cheap temporary data storage to VM - Non-persistent volume (no backup) - No automatic replication of data - Supports both VMs run in paravirtual mode (pv) as well as with full virtualization (hvm) mode. - 3 types of disks: HDD (old), SSD, and NVMe SSD - I/O operations per second (IOPS) based on HW capabilities and sharing	- Data is persistent and saved when VM terminates - 99.999% availability - Automatic replication within availability zone - Snapshot support - Can modify volume size as needed (vertical scaling) - SSD or HDD - Auto recovery - common types: gp2 and io1 - scalable IOPS with i01 type (I/O operations per second) - default ROOT volume size is based on the OS snapshot used – typically 8 GB for Linux - when used for ROOT volume, virtualization type is full virtualization (hvm) mode

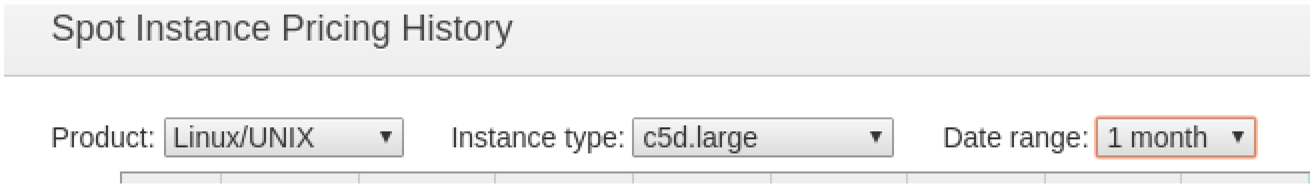
1. **Launch an EBS-backed Amazon EC2 Ubuntu 20.04 instance**

From the EC2 Dashboard.
Click on the "Spot Requests" option on the left-hand menu:



Before launching a auction-based“spot instance”, check the going rate for VMs in your region. Select the “Pricing History” button:

Filter the graph as follows (note GUI has changed):


Spot Instance Pricing History

Product: Linux/UNIX Instance type: c5d.large Date range: 1 month

Write down the least expensive “availability zone” within the Ohio region.
The options include: us-east-2a, us-east-2b, and us-east-2c. The price shown is the price per hour to rent a c5d.large VM.

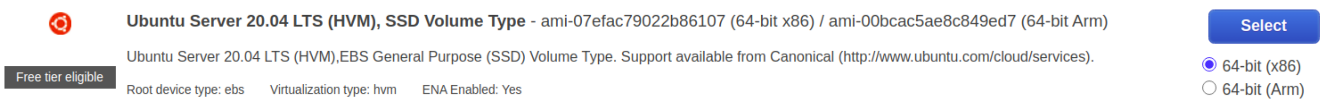
Next, compare the pricing with “c5.large” VMs. The c5.large VMs do not offer a local instance store nVME SSD disk.

You can further check the capabilities of the c5d.large VM using this website:
https://www.ec2instances.info/?filter=c5d

Adjust or remove the filter to inspect other types of VMs.

Next, let’s launch a c5d.large spot instance.
Escape out of the price graph.
On the left-hand-side click “Instances”.
A spot instance can be launched using the normal instance launch wizard.
Press the “Launch Instances” button:

Select Ubuntu 20.04 LTS (HVM) as the Amazon Machine Image.
Amazon Machine Images are snapshots of pre-installed machines. Choosing an AMI will cause its snapshot to be replicated onto your live EBS volume:


Ubuntu Server 20.04 LTS (HVM), SSD Volume Type - ami-07efac79022b86107 (64-bit x86) / ami-00bcac5ae8c849ed7 (64-bit Arm)

Ubuntu Server 20.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services).

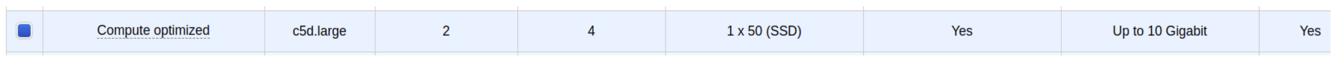
Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

Free tier eligible

[✓ 64-bit (x86)] 64-bit (Arm)

Next, select the Instance Type.
To find “c5d.large”, choose the “Compute optimized” family in the Filter by dropdown list.

Next, using the web browser's search feature (CTRL-F), search for "c5d.large".
Select this type:



■	Compute optimized	c5d.large	2	4	1 x 50 (SSD)	Yes	Up to 10 Gigabit	Yes

Next, configure the instance details.
For purchasing option, select "Request Spot Instance".
This will display current prices in each Availability Zone.

Specify a "Maximum price" of "10 cents".

DO NOT SELECT persistent request.
When a spot request is a "persistent request", then whenever the VM is terminated, either voluntary, or accidentally, the VM will always automatically RECREATE itself!
This leads to the next cost savings measure:

Cost Saving Measure #3:

BE CAREFUL USING PERSISTENT REQUESTS FOR SPOT INSTANCES

When using this feature VMs that you intentionally terminate, will automatically be recreated within 1-2 minutes resulting in ongoing charges !!! Imageine, you shut your computer down for the weekend, only to learn that the VM came back to life, and charged you for multiple days of inactivity !! When using Persistent Spot Requests, it is necessary to physically **DELETE THE SPOT REQUEST** in addition to terminating the instance. Failure to do so, will result in ADDITIONAL CHARGES ! BE WARNED !

Next, we will discuss other key settings:

Network:
This is how you select the Virtual Private Cloud (VPC) that the VM is created on. Every VM has a VPC. Most of the time we use the default VPC. Creating a custom VPC allows for specialized network configurations.

Subnet:
This is how you select a specific availability zone. This is important if spot instances have different prices in different zones, and you need to select the least expensive availability zone:

Cost Savings Measure #4:

SET THE SPOT INSTANCE SUBNET TO MATCH
THE LEAST EXPENSIVE AVAILABILITY ZONE

Auto-assign Public IP:
Subnets have a default setting whether a VM should receive a public IP address.
The property is called "Auto-assign public IPv4 address".

To check this, you can open a second AWS management console window, and select the "VPC" service, and choose the "Subnets" option on the left-hand menu. Match the Subnet ID with the subnets in the VM launch wizard to check if they will automatically assign a public IP to your VM.
Note: the default setting is YES- a public IP will be associated.

To be safe, it is always possible to simply select "Enable" to force a public IP address to be associated with this VM.

<u>Placement group:</u>
Placement groups allows for strategic placement of multiple VMs across physical servers. This feature only pertains when you're launching more than 1 VM. This is important for distributed computing tasks where 5 or more VMs are needed.

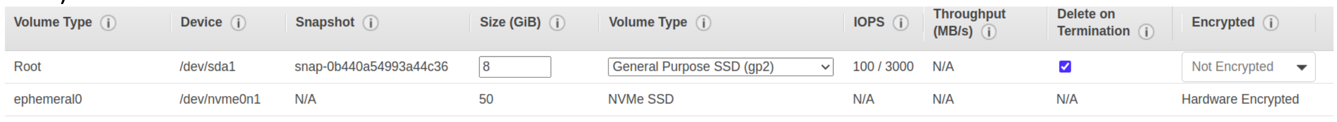
For distributed computing (multiple VMs) you can select "Add instance to placement group" and "Add to new placement group".

Now select the "Placement group strategy". The table below summarizes the options, and their corresponding implications for distributed computing:

Placement group strategy	**Implications for distributed computing**
cluster	Groups VMs as close together on the same HW as possible. Useful when low network latency between VMs is important.
Spread	Spreads VMs across physically separate racks/HW. Limited to 7 VMs per availability zone. Important when running many instances of the same kind of work at the same time. This is ideal for MapReduce clusters.
Partition	Launchs VMs into distinct partitions, where each partition consists of servers very close together having low latency. This is helpful if an application has distinct components that need to be scaled and hosted separately where individual nodes of an application tier need low network latency.

Next, configure Storage options.

By default, c5d.large will configure the following ROOT (EBS) with 8GB, and display any ephemeral (temporary local) disks:



Volume Type	Device	Snapshot	Size (GiB)	Volume Type	IOPS	Throughput (MB/s)	Delete on Termination	Encrypted
Root	/dev/sda1	snap-0b440a54993a44c36	8	General Purpose SSD (gp2)	100 / 3000	N/A	✓	Not Encrypted
ephemeral0	/dev/nvme0n1	N/A	50	NVMe SSD	N/A	N/A	N/A	Hardware Encrypted

Ephemeral literally means "lasting for a very short time". These are disks whose data is not persisted. In computing, ephemeral refers to temporary, non-persisted resources (i.e. with no backup).

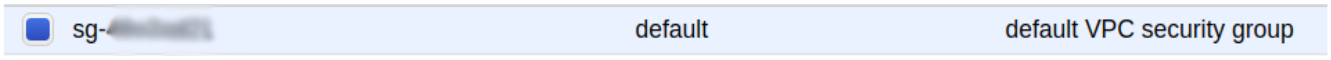
We can identify an EBS volume based on the Volume Type. Available types for EBS include general purpose (gp2), provisioned SSD (io1), provisioned SSD (io2), and magnetic (which is fact no longer "standard" by the way).

***** HERE we need to make changes to the "Root" Volume. Increase the size from 8 GB to 15 GB. *****
This additional disk space, will be needed later on in the tutorial on page 19.

Note it may not be possible to reduce the volume below 8GB. This is because the originating snapshot (e.g. snap-0b440a54993a44c36) expects at least 8 GB.

Next, "Add Tags".
Tags allow key/value assignments to help identify VMs.
These key/value pairs can be queried using the EC2 API to programmatically identify specific VMs that have a designated purpose for your application.
We will not use this feature right now.

Next, "Configure Security Group":
Select the radio button: "Select an existing security group".
And select the default security group from the list of choices.



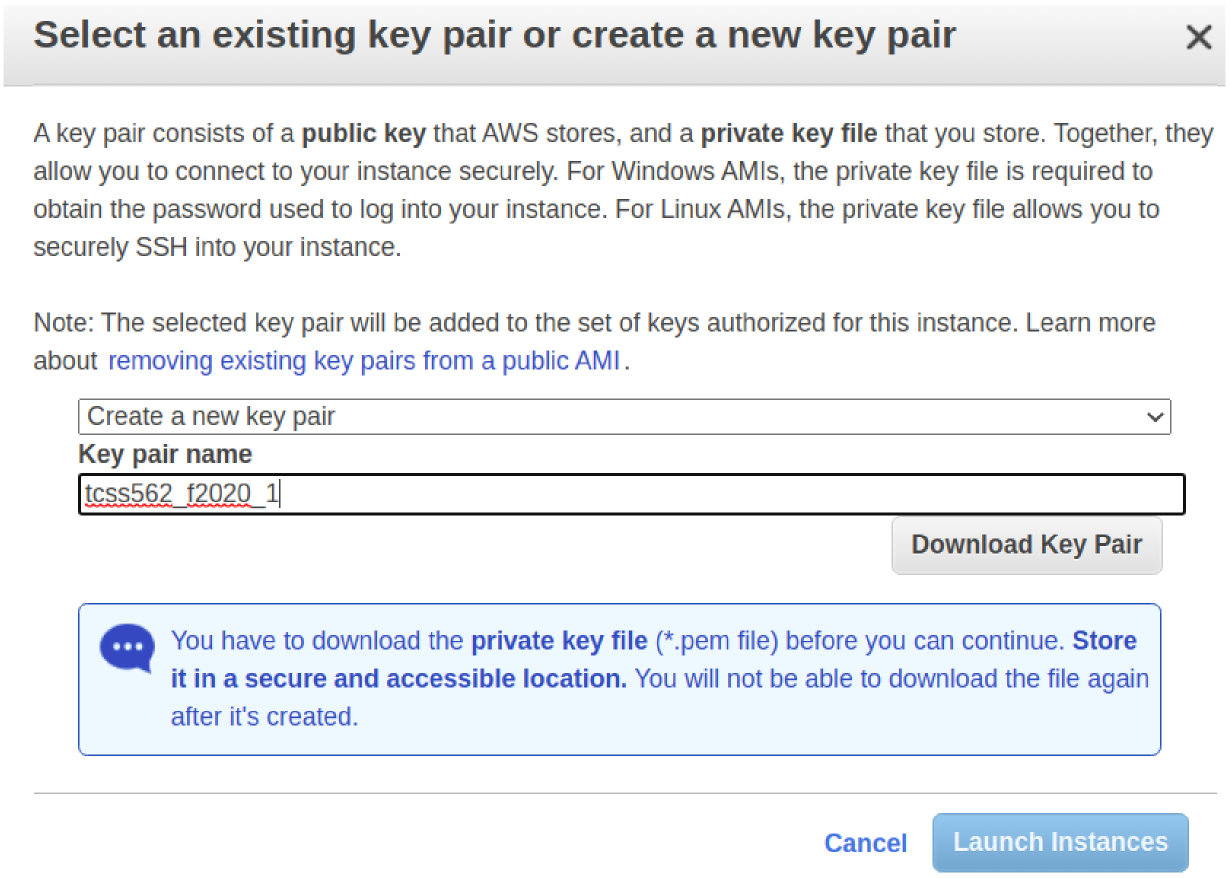
sg-[redacted]	default	default VPC security group

Selecting the default security group, allows you to add common firewall access rules to enable rapid access to newly created VMs. When selecting the security group, firewall rules are displayed. They can not be changed here. They can be modified later using the Security Group GUI.

Next, "Review and Launch".
This step previews all settings for acceptance prior to creating the VM.
If everything looks okay, press the blue "Launch" button:

If this is the first time you've created a VM, you'll need to create a new key pair which is used to establish a secure shell (ssh) connection to your new VM.

In the dialog box, select "Create a new key pair", and assign a name.
Then click the "Download Key Pair" button.
Store the keypair in a safe location, preferably a location that is backed-up.
Note the absolute path (full directory name) where the key is stored.



Select an existing key pair or create a new key pair

A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.

Note: The selected key pair will be added to the set of keys authorized for this instance. Learn more about removing existing key pairs from a public AMI .

Create a new key pair

Key pair name

tcss562_f2020_1

Download Key Pair

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.

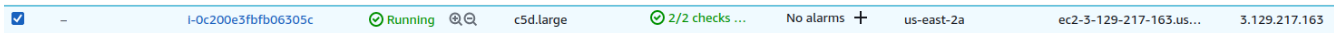
Cancel   Launch Instances

Finally, launch the instance by pressing the blue button.
Escape out of the launch wizard.

2. Log into your Amazon EC2 Spot Instance

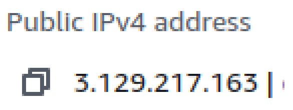
Select "Instances" on the left-hand side of the screen.

Locate your newly launched VM. It should be the only one!
Select your VM on the left:



✓	–	i-0c200e3fbfb06305c	Running		c5d.large	2/2 checks ...	No alarms +	us-east-2a	ec2-3-129-217-163.us...	3.129.217.163

Then, look for "Public IPv4 address" and select the copy-icon just to the LEFT of the IP address:



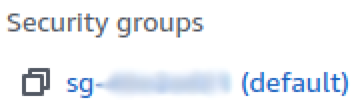
Public IPv4 address

3.129.217.163

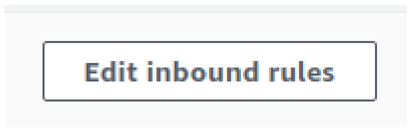
This copies the IP address to the clipboard.
Before connecting to the VM, configure the security group to allow SSH access.

Click on the “Security” tab of the instance.

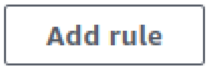
Click on the security group ID:



On the list of Inbound Rules, press the button:

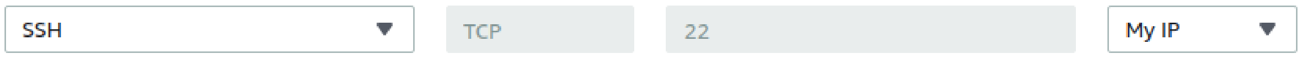


At the bottom of the list, press the button:



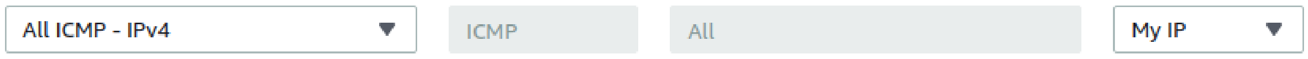
Configure a new rule as follows:

Type: “SSH”, Source: “My IP”



SSH | TCP | 22 | My IP

Another Security Group option is to add the “All ICMP – IPv4” permission. This enables you to “ping” your VM.



All ICMP - IPv4 | ICMP | All | My IP

For every new VM that is launched, when you associate the default security group, these security group settings will be applied. Overtime you can accumulate rules for popular places where you use your laptop. Prior to covid-19, this included places like the local Starbucks, and the UW-Tacoma campus.
As an alternative to selecting “MyIP” you can specify a wider subnetwork address range. This way if your compute receives different IP addresses within the same CIDR (address) block, you don’t have to update the security group as often. (e.g. every day) CIDR block stands for classless-inter-domain routing. A CIDR block is a group of IP addresses the form a “sub-network”. Within your house, all of the devices sharing your WiFi connection usually receive private IP addresses sharing the same CIDR-block, where all addresses are private (not public IPs).

Find out your IP address.
In your web browser, open Google search, and type in “What is my IP?”.

Your IP address should appear.
Note the first 3 numbers.
Let's add SSH permission for your CIDR network block.

If your IP is for example **120.118.53.**108, then your 24-bit CIDR block which would include all 255 addresses on the local subnet will be 120.118.53.0/24.

As an alternative to MyIP, in the security group, add your entire CIDR block, e.g. 120.118.53.0/24

The motivation for adding your CIDR block is that if you commonly use a buliding's WiFi network you may receive various addresses in the same block from day-to-day. If you add ranges vs. individual IPs you'll hopefully not need to update the security as often.

Once complete, save changes:

Return to the list of Instances, by clicking the "Instances" option on the left-hand menu.

Now navigate to a Linux terminal.
If using Windows without a Linux environment, a 3rd-party program like PuTTY is required.
To learn more, see this tutorial:
https://www.ssh.com/ssh/putty/windows/install

Otherwise, open a Linux terminal, and navigate to the subdirectory using "cd" where you have stored your key using the absolute path. For example: "/home/username/awskeys/"

First try pinging your VM

```
ping 3.129.217.163
```

Use CTRL-C to stop the ping:

```
PING 3.129.217.163 (3.129.217.163) 56(84) bytes of data.
64 bytes from 3.129.217.163: icmp_seq=1 ttl=33 time=70.3 ms
64 bytes from 3.129.217.163: icmp_seq=2 ttl=33 time=70.7 ms
64 bytes from 3.129.217.163: icmp_seq=3 ttl=33 time=72.3 ms
64 bytes from 3.129.217.163: icmp_seq=4 ttl=33 time=70.5 ms
```

Pinging provides a rough estimate of the network latency between your computer, and the VM. Here the ping sends a 64-byte packet to the cloud VM. The cloud responds and the client measures the round-trip response time.

Ping-time to the VM is a good way to test your network's ability to access cloud resources. A VM in Oregon should have a lower ping time than Ohio, Virginia, or a VM in a different continent.

Now launch an ssh session as follows:

<replace with your VM's IP>

```
ssh -i mykey.pem ubuntu@3.129.217.163
```

or if not in the same directory as the key:

<replace with your VM's IP>

```
ssh -i /home/username/awskeys/mykey.pem ubuntu@3.129.217.163
```

You may receive the following error:

```
The authenticity of host '3.129.217.163 (3.129.217.163)' can't be established.
ECDSA key fingerprint is SHA256:dhfCG+k/Zz7p13d39cAIiGfTCKW0zTHwtLTdoJQspp4.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '3.129.217.163' (ECDSA) to the list of known hosts.
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@         WARNING: UNPROTECTED PRIVATE KEY FILE!          @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
Permissions 0664 for 'tcss562_f2020_1.pem' are too open.
It is required that your private key files are NOT accessible by others.
This private key will be ignored.
Load key "tcss562_f2020_1.pem": bad permissions
Permission denied (publickey).
```

This indicates that the ssh key permissions are too open.
Adjust with the following command:

```
chmod 0600 mykey.pem
```

or

```
chmod 0600 /home/username/awskeys/mykey.pm
```

After adjusting the key permissions, log into the VM again.

Inspect various aspects of your newly launched VM with the following commands:

Operation	Command
Inspect CPU	`lscpu`
Inspect memory	`free -h`
Inspect the disks	`df -Th`
Inspect the OS kernel	`uname -a`
Inspect the OS version	`cat /etc/os-release`

3. Preparing ephemeral disk volumes

By default, ephemeral disk volumes for c5d instances are not pre-initialized on launch.

The “lsblk” command describes block devices on Linux.
Check your VM’s block devices:

```
$ lsblk
NAME        MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
loop0         7:0    0 96.6M  1 loop /snap/core/9804
loop1         7:1    0 55.3M  1 loop /snap/core18/1885
loop2         7:2    0 70.6M  1 loop /snap/lxd/16922
loop3         7:3    0 28.1M  1 loop /snap/amazon-ssm-agent/2012
nvme1n1     259:0    0 46.6G  0 disk
nvme0n1     259:1    0    8G  0 disk
└─nvme0n1p1 259:2    0    8G  0 part /
```

In Linux, disks are assigned block device names. VMs have an odd convention where on EC2 block names for disks share the same letters, but have different numbers. For the c5d.large, the EBS volume block device name is therefore “nvme0n1”. This is the Linux physical device name. Disks are organized into partitions, where each partition can be formatted (i.e. organized) using a different file system. The file system provides a catalog allowing the operating system to quickly store and retrieve files. The most common for Ubuntu is currently “ext4”.

Filesystem types are seen using the “df” command with the “T” option.
Filesystems use inodes, which are file records, to store and retrieve individual files. If a disk runs out of inodes it will no longer be able to store additional files even if there is space remaining on the disk. Inodes can be displayed with the “df” command using the “i” option.

Inspect your two devices with the “sudo fdisk -l” command:

```
sudo fdisk -l /dev/nvme0n1

sudo fdisk -l /dev/nvme1n1
```

Make a note one which device is the EBS disk? And which is the instance store volume.

Using the device name for the ephemeral instance store volume, execute the following command sequence.

Note these commands could be used in a script to auto-initialize the VM at boot time:

IMPORTANT: BEFORE USING THESE COMMANDS IT IS ESSENTIAL TO VERIFY THAT YOU ARE PERFORMING THEM ON THE NVME INSTANCE STORAGE DISK. SWAPPING THEM WILL DELETE ALL DATA ON YOUR ROOT DISK AND WILL RENDER YOUR VM INOPERABLE

Verify the proper disk.
Check the “Disk model”

```
$ sudo fdisk -l /dev/nvme0n1
Disk /dev/nvme0n1: 8 GiB, 8589934592 bytes, 16777216 sectors
Disk model: Amazon Elastic Block Store
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
```

```
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0xa2f52878

Device         Boot Start      End  Sectors Size Id Type
/dev/nvme0n1p1 *     2048 16777182 16775135   8G 83 Linux

$ sudo fdisk -l /dev/nvme1n1
Disk /dev/nvme1n1: 46.58 GiB, 50000000000 bytes, 97656250 sectors
Disk model: Amazon EC2 NVMe Instance Storage
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

Once verifying that you have the correct disk:

```
#assign a label to the disk
sudo parted -s /dev/nvme1n1 mklabel GPT

#create the disk partition, use all the space
sudo parted -s /dev/nvme1n1 mkpart primary ext4 2048s 100%

#format the new partition with the ext4 file system
sudo mkfs.ext4 -q /dev/nvme1n1p1

#mount the newly formatted filesystem
sudo mount /dev/nvme1n1p1 /mnt
```

Check that the new file system has been created and mounted with the df -hT command.

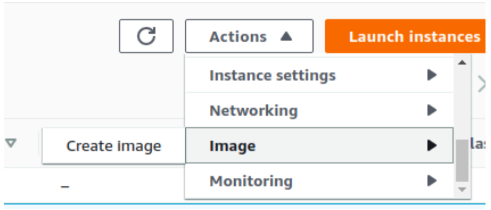
At this stage, the EBS volume is mounted at "/", and the local NVMe SSD disk is mounted at "/mnt". We can profile performance of both types of cloud disks using the same VM.

4. Make a Snapshot of a live ROOT filesystem to backup data

As you install new software on the VM and perform custom configuration, it becomes desirable to SAVE the modified ROOT file system containing the Ubuntu operating system and any new software packages and configuration installed.

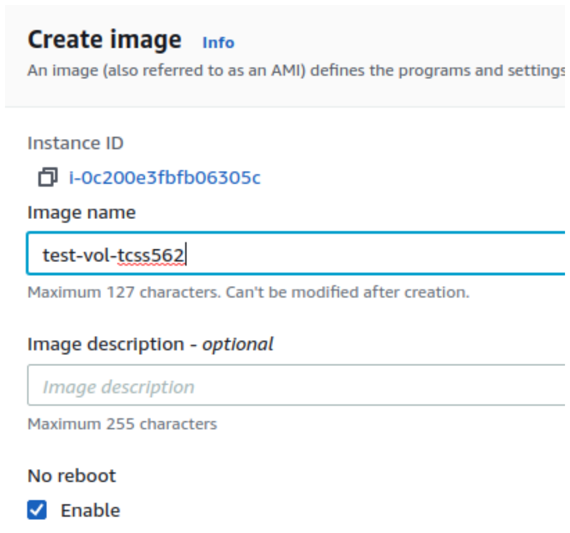
To do this, return to the EC2 management console.
Select "Instances" on the left-side menu, and find and select your instance.

In the upper right hand corner select the "Actions" drop-down menu:



Scroll down and select “Image” and “Create Image”.

This will launch a wizard to configure saving a new Amazon Machine Image (AMI) that will store all changes made to the ROOT filesystem. To prevent the VM from shutting down and rebooting during the snapshot process, you can select “No reboot” Enable. In some cases it may be preferable to shutdown the VM when making a backup to ensure that server applications have stopped writing to the disk, a condition which could result in the snapshot having corrupted partially written data. If you know your VM is dormant/quiet it is generally safe to not reboot.


Create image Info

An image (also referred to as an AMI) defines the programs and settings

Instance ID

i-0c200e3fbfb06305c

Image name

test-vol-tcss562

Maximum 127 characters. Can't be modified after creation.

Image description - *optional*

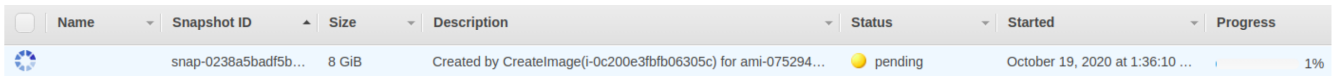
Image description

Maximum 255 characters

No reboot

☑ Enable

While the snapshot is being created, check its status by selecting “Snapshots” on the left-hand menus:



	Name	Snapshot ID	Size	Description	Status	Started	Progress
		snap-0238a5badf5b...	8 GiB	Created by CreateImage(i-0c200e3fbfb06305c) for ami-075294...	● pending	October 19, 2020 at 1:36:10 ...	1%

Cost Savings Measure #5:

TO SAVE/PERSIST DATA, USE EBS SNAPSHOTS AND THEN DELETE EBS VOLUMES FOR TERMINATED EC2 INSTANCES.

EBS snapshot storage is only 5 cents/GB/month. For subequent snapshots of the same volume, you are only charged for the deltas (differences) in storage size. In additional snapshots are stored using compression, and charges only for the actual data stored. For example, if you’re disk volume is 16GB, but only 8GB is filled with data, then you’re only charged for the 8GB.

There are some hidden things going on here.

AWS does not disclose the actual storage used per snapshot in a way that is easy to access. The total monthly snapshot storage charges shown in the monthly bill under “My Billing Dashboard” provides an aggregate for all snapshots. The GUI and CLI do not expose detailed snapshot storage consumption information on a per-snapshot basis.

THIS SUGGESTS THAT THE BILLLING POLICY MAY NOT MATCH ACTUAL STORAGE USED BEHIND THE SCENES. AS THE STORAGE ACTUALLY USED IS OBSCURE, THIS PROVIDES A HIDDEN MECHANISM TO ALLOW AWS TO ALTER STORAGE PRICING WITHOUT NOTIFYING USERS DIRECTLY.

For example, AWS may use a new compression algorithm to reduce physical volume storage, but not pass savings on to end users. Information regarding compression type and efficiency is not exposed in a way that is easily visible to EC2 users.

Determining actual snapshot storage size appears to require programming and use of the "Direct EBS API".

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-accessing-snapshot.html

This leads to another cost savings measure:

Cost Savings Measure #6:

**UNUSED SNAPSHOTS AND UNUSED EBS VOLUMES
SHOULD BE PROMPTLY DELETED !!**

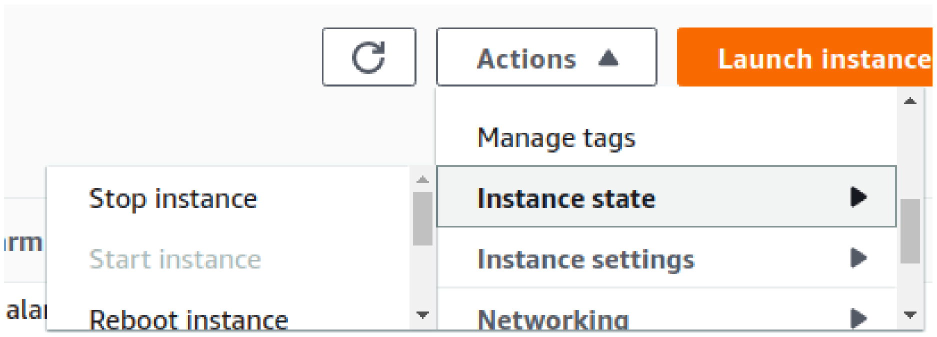
They will incur charges at high rates over time if left unused in the account draining cloud credits leading to incurring credit card charges. Deleting a snapshot is a two-step process. First, the AMI must be deregistered. Second, the snapshot must be deleted. These are separate steps in the GUI. Deregistering the snapshot does not prevent storage charges.

5. Pausing an instance to save cost

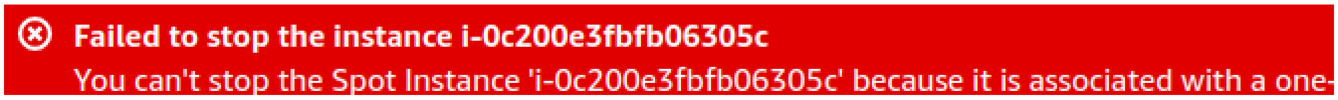
When working on a project, it may be desirable to PAUSE the virtual machine to save/retain its state to prevent having to recreate it. This is acceptable if changing working locations, for example going from school to home. This can also be acceptable if stopping work for the day, and looking to resume from the same point tomorrow.

While paused, a VM incurs only EBS storage charges at a rate of 10c/GB/month.

Try pausing your running VM.
In the ec2 console, select your instance, and from the "Actions" menu, select "Instance state" and "Stop instance".



Unfortunately, unless you specified the “persistent” spot request earlier, the instance can’t be stopped. It must be terminated, and recreated.

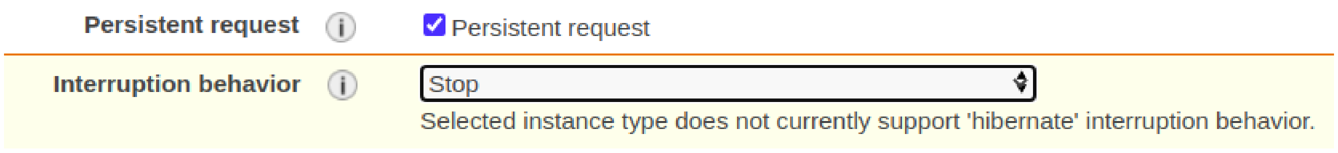

Failed to stop the instance i-0c200e3fbfb06305c
You can't stop the Spot Instance 'i-0c200e3fbfb06305c' because it is associated with a one-

Because you’ve created an snapshot (AMI) from step 4, try recreating the instance using your AMI as persistent spot request. Only the left-hand side menu select “AMIs” and select your recently created AMI.

Press the blue launch button:

Follow the wizard steps as before, except this time it is not necessary to select the operating system.

A spot request can be made persistent when creating the VM.
Select “Persistent request”, and specify the “Interruption behavior” as “STOP”.


Persistent request ✓ Persistent request
Interruption behavior Stop
Selected instance type does not currently support 'hibernate' interruption behavior.

Once then instance is create and running, you can now try the “Stop instance” feature to pause the VM. This will allow you to Stop and Start the VM at-will pausing the hourly VM cost. While stopped the EBS charges remain: 10c/GB/month for EBS volumes.

The caveat is that when you “Terminate instance” you also must **DELETE THE SPOT REQUEST**. Failure to do so will result in the VM resurrecting itself indefinitely in your account resulting in ongoing unwanted charges.

Cost Savings Measure #7:

USE PERSISTENT SPOT REQUESTS AND THE “STOP” FEATURE
TO PAUSE VMS DURING SHORT BREAKS

Stopping a VM will persist the data using the EBS volume, preventing ongoing charges for the instance.

Again is it vital to **DELETE THE SPOT REQUEST** when entirely done with the instance.

6. Compare EBS and Instance Store Disk Performance using Bonnie++

The last step of this tutorial is to compare disk performance using Bonniee++.
It is easy to install bonnie++.
From your VM’s command line simply type:

```
$sudo apt install bonnie++
```

We've mounted your instance store volume under "/mnt". The instance store volume represents space on the local hardware that hosts the VM. For the c5d.large instance type, this is a local NVMe SSD drive.

Your root partition is an EBS volume, mounted under "/".
By default, the EBS volume is created as a GP2 – General Purpose 2 SSD EBS volume.
This volume is granted a baseline of 100 IOPS (I/O operations per second), and is burstable to 3,000 using a credit-based allocation approach.

For a detailed description of how Amazon manages IOPS for EBS volumes see this article:
http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSVolumeTypes.html

Amazon allows a certain number of IOPS relative to the overall disk size. This is approximated based on the typical number of IOPS for a hard disk based on different usage scenarios.

7. Run the Bonnie++ disk benchmark utility to test EBS and Instance Store disk performance

To run our disk benchmark, we will use ~ 7,520 MB of free disk space.

From the command line:

Create a instance store temp directory and set the permissions:

```
$sudo mkdir /mnt/tmpi
$sudo chmod a+rwx /mnt/tmpi
```

Now run bonnie on the EBS volume:

The /tmp directory is on the "/" root partition.
Using /tmp as the bonnie directory will test the EBS volume performance.

Bonnie++ wants to create files that are twice the size of the RAM of the VM.
Here you should have increased the root volume size to 15 GB or else the EBS volume won't have enough space on a c5d.large. Available disk space can be checked with: "df -h"

```
$sudo chmod a+rwx /tmp
$bonnie++ -d /tmp -s 7520M -n 0 -m TEST-EBS -f -b -q  > bonnie.csv
```

This will capture the output to a text file.

Next, run bonnie on the Instance store volume:

The /mnt/tmpi directory is on the "/mnt" partition, which is the auto-mounted instance store volume which is hosted local to the VM.
Using /mnt/tmpi tests the instance store volume performance.

We usethe same file size on the instance store volume for comparison purposes.

```
$bonnie++ -d /mnt/tmpi -s 7520M -n 0 -m TEST-IS -f -b -q  >> bonnie.csv
```

Bonnie provides formatting utilities which take the CSV output and convert the output to either text (txt) or html.

These utilities are:
bon_csv2html
bon_csv2text

To use the utilities to generate formatted output, simply redirect your bonnie.csv output file into the utility:

```
$bon_csv2txt < bonnie.csv
```

AND

```
$bon_csv2html < bonnie.csv
```

Now, using the clipboard, copy the contents of the bon_csv2html output, and save this as a local .html file on your laptop. Try opening the file in a web browser by navigating from "file://"

Tutorial Questions:

Submit written answers as a PDF file on canvas. If submitting with a partner, include both names at the top of the PDF as they appear in CANVAS.

Only one person in the team should submit the assignment!

Include the HTML table output (or CSV if HTML is unavailable) from all Bonnie tests at the bottom of the PDF. Failure to include Bonnie output will result in a 0 for Bonnie questions.

Bonnie Questions

1. (2 points) What EC2 instance type did you run Bonnie++ on?

2. (2 points) What was the sequential output block read throughput (in K/sec) for the EBS and Instance Store volumes?

3. (2 points) How much CPU capacity was required for sequential output block reads for the EBS and Instance Store volumes?

4. (2 points) What was the sequential output block rewrites and sequential input block read throughput for the EBS and Instance Store volumes?

5. (2 points) How much CPU capacity was required to perform sequential output block rewrites and sequential input block reads for the EBS and Instance Store volumes?

6. (2 points) What was the random seek disk performance for the EBS and Instance Store volumes?

Cost Savings Questions

7. (2 points) If using a spot instance with a persistent spot request, what must be done when terminating the instance to prevent accidental charges?

8. (2 points) After a course/research project ends, what must be done to delete EBS snapshots used as AMIs with ROOT filesystems?

9. (2 points) When finished with a VM, how much money can be saved using an EBS snapshot to backup the data for a 10 GB EBS volume versus retaining the original EBS volume?

10. (2 points) When creating spot instances, what approach can be used to minimize costs within an AWS Region?

11. (2 points) What are persistent spot requests good for when working with spot instances? What cost savings feature do they enable?

12. (2 points) What best practice should be used when creating ec2 instances for course projects and research to save cloud computing credits and money?

8. Bonus Activity: Benchmark a "Provisioned IOPS" EBS Volume

(Optional / Non-graded) Provisioned IOPS EBS volumes forgo the credit based approach to provide consistent, guaranteed performance of EBS volumes within +/- 10%. The catch, this performance costs more, and is not offered as a FREE tier resource. The free tier offers up to 30GBs of general purpose EBS storage a month.

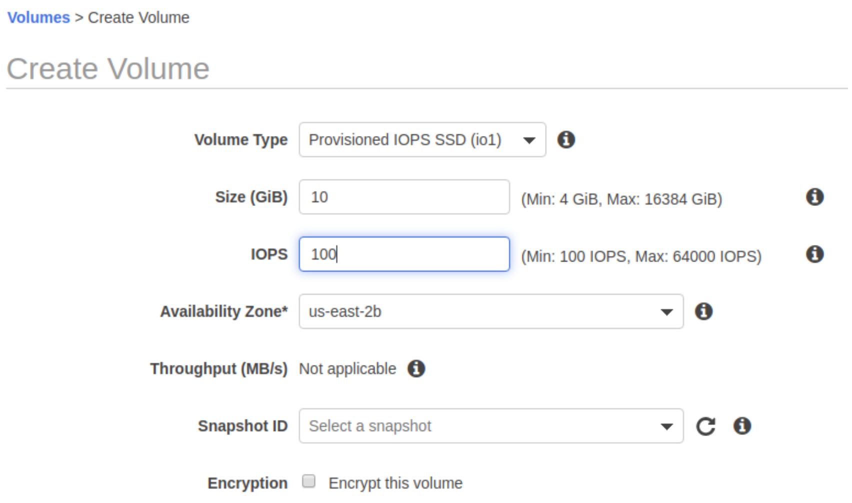
Let's create a new IOPS EBS volume, and attach it to our currently running spot instance:

Go to EC2 | Elastic Block Store | Volumes

And then click "Create Volume".

Specify as follows:

Volume Type: Provisioned IOPS SSD
Size (Gib): 10
IOPS: 100
Availability Zone: us-east-1e
The availability zone must match where your EC2 spot instances is running…


Volumes > Create Volume

Create Volume

Volume Type: Provisioned IOPS SSD (io1)

Size (GiB): 10 (Min: 4 GiB, Max: 16384 GiB)

IOPS: 100 (Min: 100 IOPS, Max: 64000 IOPS)

Availability Zone*: us-east-2b

Throughput (MB/s): Not applicable

Snapshot ID: Select a snapshot

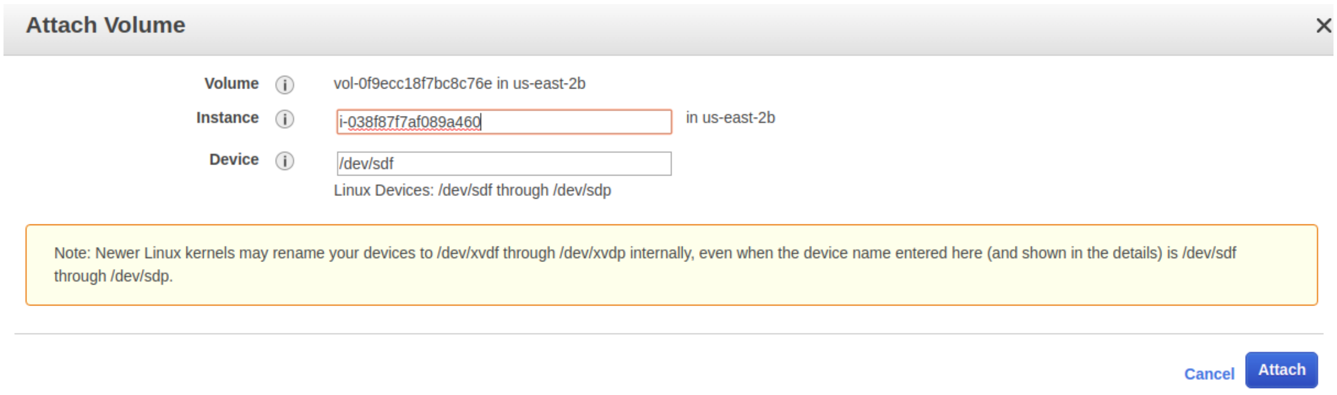
Encryption: Encrypt this volume

<<< <u>WARNING</u> - BE SURE TO DELETE THIS VOLUME AFTER COMPLETING TESTS. IT IS EXPENSIVE TO KEEP IN YOUR ACCOUNT !!! >>>

According to Amazon, an IOPS EBS volume with these settings will cost 12.5 cents per GB compared to 10 cents per GB for a general-purpose (GP2) EBS volume. Additionally Amazon charges for the guaranteed IOPS. The charge for 1-month is $.065 x 100 IOPS or $6.50/month. The total monthly cost of a 10GB volume with a 100 IOPS guarantee is $7.75/month.

**** There would be no practical reason to ever create this EBS volume since the guaranteed IOPS is no better than what GP2 provides. ****

Once the volume has been created (try clicking refresh a few times), select "Action"
and then "Attach Volume"


Attach Volume

Volume: vol-0f9ecc18f7bc8c76e in us-east-2b

Instance: i-038f87f7af089a460 in us-east-2b

Device: /dev/sdf

Linux Devices: /dev/sdf through /dev/sdp

Note: Newer Linux kernels may rename your devices to /dev/xvdf through /dev/xvdp internally, even when the device name entered here (and shown in the details) is /dev/sdf through /dev/sdp.

Please note the disclaimer in yellow.

As of this writing the disclaimer is out of date for 5th gen SSD enabled resources.

When attaching to your c5d instance the device name will be mapped to /dev/nvme2n1.

Next you'll need to partition, format, and mount the disk:

```
$sudo parted /dev/nvme2n1 mklabel msdos
$sudo parted -a optimal /dev/nvme2n1 mkpart primary ext4 0% 10.0GB
$sudo mkfs.ext4 /dev/nvme2n1p1
$ sudo mkdir /mnt2
$ sudo mount /dev/nvme2n1p1 /mnt2 -t ext4
```

Then check that the disk is available at /mnt2:

```
$ df -h
```

Create a local tmp directory and grant world read write execute permission:

```
$sudo mkdir /mnt2/tmp_ebsp
$sudo chmod a+rwx /mnt2/tmp_ebsp
```

Now, let's test this volume and append our results to the bonnie.csv output file:

```
$bonnie++ -d /mnt2/tmp_ebsp -s 7520M -n 0 -m TEST-EBSP -f -b -q >> bonnie.csv
```

Note, the test may be slower because of the very low IOPS of the EBS volume.

Once the test completes, capture your bon_csv2html output, and copy and paste the HTML to a html file on your laptop, save it, and view in a browser.

Or, alternatively, use bon_csv2text and view using the command line.
How does the provisioned IOPS EBS volume perform?

9. **Cleanup**

At the end of the tutorial:

Be sure to:

#1 - **TERMINATE** all EC2 instances

#2 - **DELETE** all EBS volumes

Failing to clean up could result in loss of AWS credits and/or AWS charges to a credit card.

10. SAVING STORAGE COSTS WITH S3

The Simple Storage Service (S3) is a key-value object storage facility on AWS capable of storing any type of data. If you are working to deploy an application that requires large datasets, it is recommended to create virtual machines with ephemeral disks. VMs with ephemeral disks often have a lower-case d such as c5d.large as used in this tutorial. There is also m5d, r5d, and z1d VM types among others. The idea is that any data stored on the ephemeral disk will be lost if the VM is stopped or shutdown. The simple storage service offers a low cost option for storing this data. S3 is a good alternative to EBS volumes and snapshots. S3 storage is just 2.3 cents/GB/month in contrast to 5 cents/GB/month for EBS Snapshots and 10 cents/GB/month for EBS volumes. Using the AWS CLI, a BASH script can be written to create a tar zip archive file which can then be pushed to an S3 bucket to store any data from your ephemeral disk(s).

KEY POINT: be cognizant when working with large volumes of data. Creating a large 500 or 1000GB EBS volume can be VERY EXPENSIVE. Instead use a c5d/m5d instance with ephemeral disks, and then use S3 and tar/gzip to shuttle data to and from the ephemeral drive(s) at less than ¼ the monthly cost.