EC2:

Amazon EC2 (Elastic Compute Cloud) is a web service that provides resizable compute capacity in cloud. It reduces the time required to obtain and boot new server instances to minutes, allowing to quickly scale capacity both upwards and downwards based on changing computing requirements.

It has changed the economics of computing by allowing to pay for what is being used. It provides the tools needed to build a failure resistant application.

**Types of Instances based on Pricing**

* On Demand - allows to pay by the hour or second(Linux is by second and Windows is by hour)
* Reserved - Reservation for 1 or 3 years, certain or entire amount upfront, but large discount compared to on demand price.
* Spot - enables to bid a price for instance capacity, if application has flexible timings, this can lead to significant savings.
* Dedicated host - These are physical ec2 server dedicated for use. They allow to bring existing server-bound software licenses over to aws and thus save costs.

**On Demand Instances**

* Perfect for users who want low cost and flexibility of AWS EC2 without any long term commitment or contract
* Applications with Unpredictable workloads that cannot be interrupted
* Development and testing

**Reserved Instances**

* Applications with steady state or predictable usage that require reserved capacity
* Users willing to make upfront payment to reduce computing cost even further
  + Standard RIs(up to 75% off on demand) - if entire payment is made up front and contract is for 3 years
  + Convertible RIs(up to 54% off on demand) - capability to change attribute of instance from say compute to memory intensive provided the exchange is of equal or greater value
  + Scheduled RIs - available to launch within a scheduled time window. It allows to obtain compute capacity within a certain recurring schedule. For example if a company has large sales during Fridays, then it will go for RIs scheduled on every Friday.

**Spot Instances**

* Applications with flexible start and end times which are feasible only at a very low cost. e.g. : Genomics and Pharma companies use this to perform research by running resource intensive apps on say a Sunday at 4 am when price is very low.
* Users who suddenly need additional compute capacity
* Spot instances are terminated by AWS if spot price for that capacity increases. However Amazon does not charge for partial usage of hour in that case. However, if instance is terminated by customer, then whole hour is charged for.

**Dedicated host**

* Used for regulatory requirement or to save licensing costs which do not allow multi-tenant virtualization
* Can be purchased on demand
* Can be purchased as a reservation which saves about 70% compared to on demand

#### Types of instances based on hardware

| **Family** | **Speciality** | **Usecase** |
| --- | --- | --- |
| F1 | Field gate programmable array | Genomic research, Video processing, Financial analytics, big data |
| I3 | High Speed Storage | No SQL DB and Data Warehousing |
| G3 | Graphics Intensive | Video Encoding |
| H1 | High Disk throughput | Distributed file systems like HDFS and Map Reduce based workloads |
| T2 | Lowest Cost General Purpose | Web Server, Small DBs |
| D2 | Dense Storage | File Servers, Data Warehouse, Hadoop |
| R5 | Memory Optimized | Memory Intensive Apps, DB |
| M5 | General Purpose | Application Servers |
| C5 | Compute Optimized | CPU Intensive Apps/DBs |
| P3 | General Purpose , Graphics Intensive | Bit coin, Machine Learning |
| X1 | Memory Optimized | SAP HANA, Apache Spark |
| Z1D | High compute capacity and high memory footprint | Electronic Design automation(EDA) and  certain relational database workloads with high licensing costs |
| A1 | Arm based workloads | Scale-out workloads such as web servers |
| U-6tb1 | Bare Metal | Bare Metal Capabilities eliminating virtualization overhead |

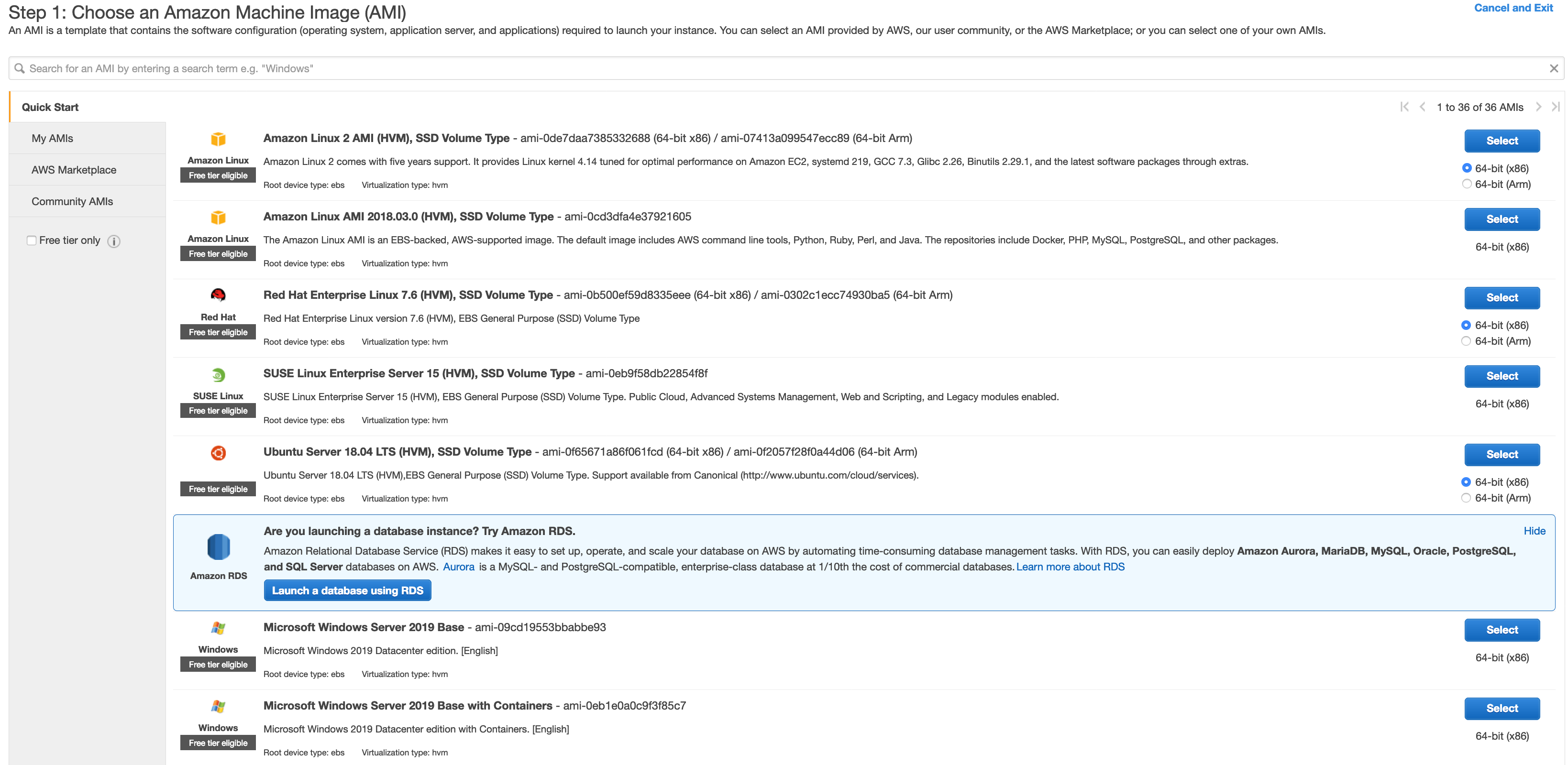
Remember by **FIGHT DR MCPXZ AU**

**Launch Instance:**

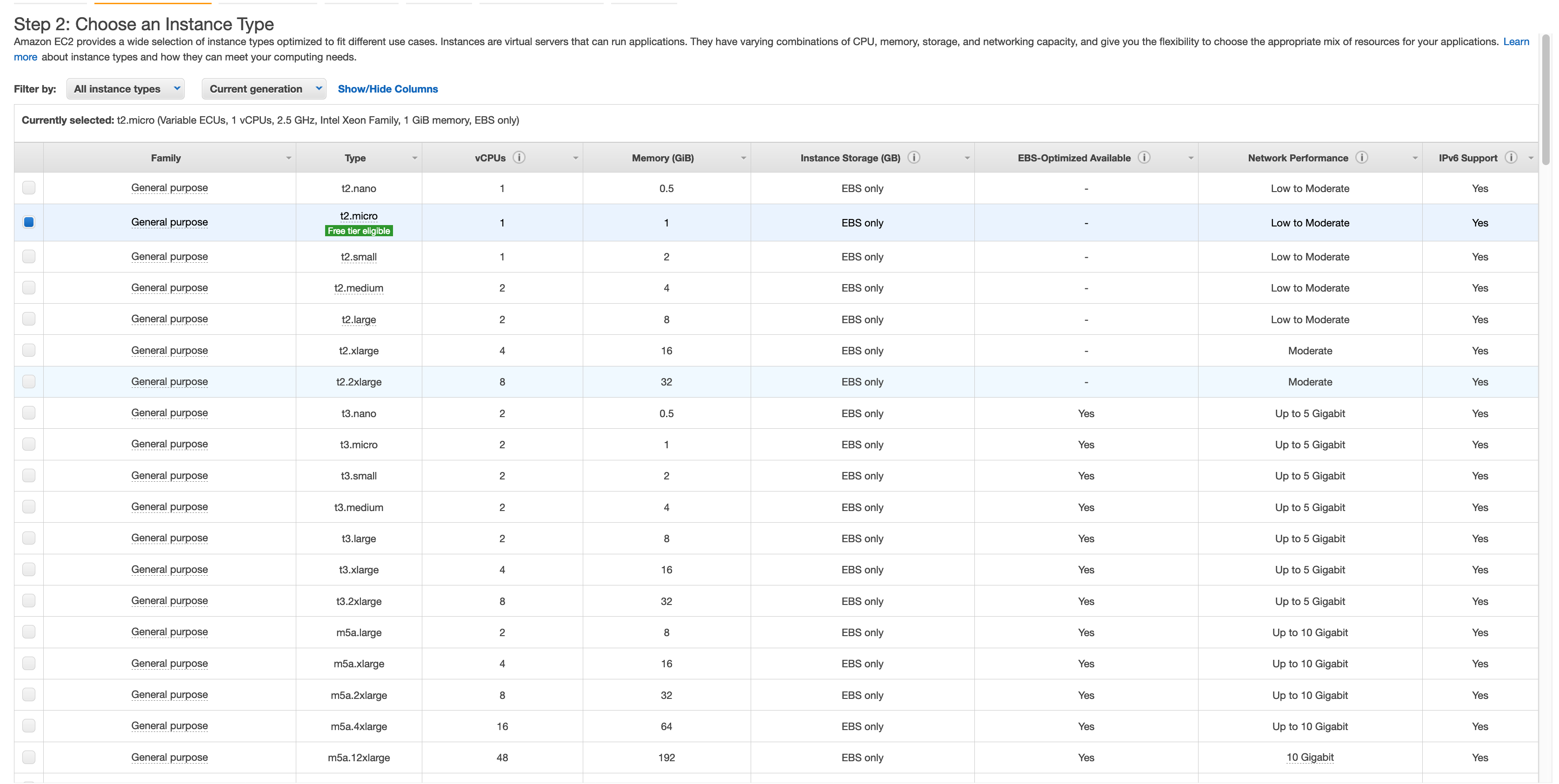
AMI (Amazon Machine Image) – snapshots of virtual machines that can be boot up.

The virtualization can be of 2 types – HVM and PV (para virtual)

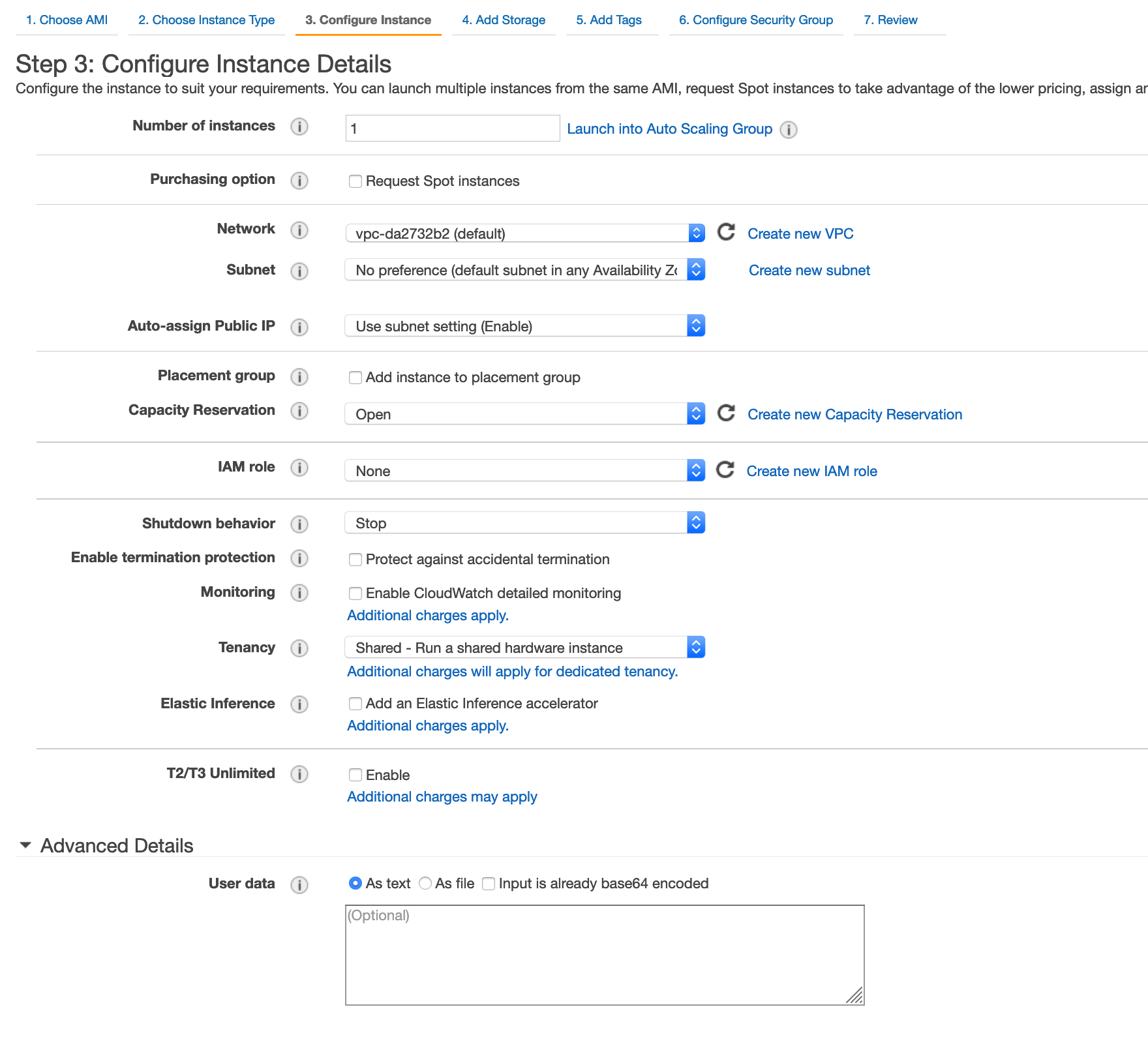
Different AMIs:



Different instances:

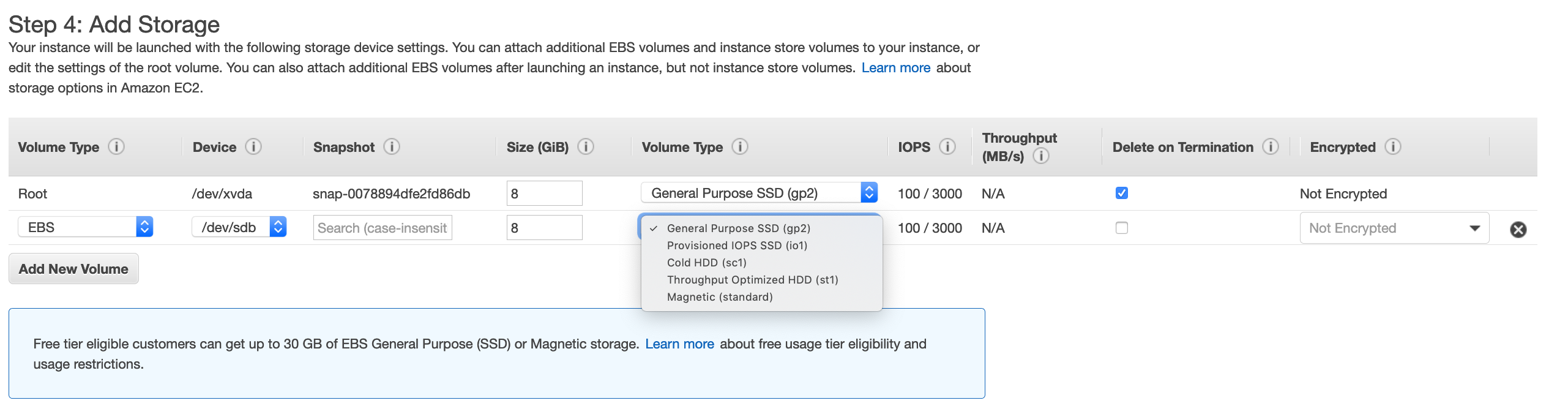


Configure Instance:



* To request spot instances – Purchasing option
* To use dedicated instances – Tenancy
* Subnet – 1 subnet cannot span multiple availability zones, it can only be in one availability zone
* Shutdown behavior – what to do if os shuts down - stop/kill instance
* Termination protection – to prevent accidental termination (turned off by default)

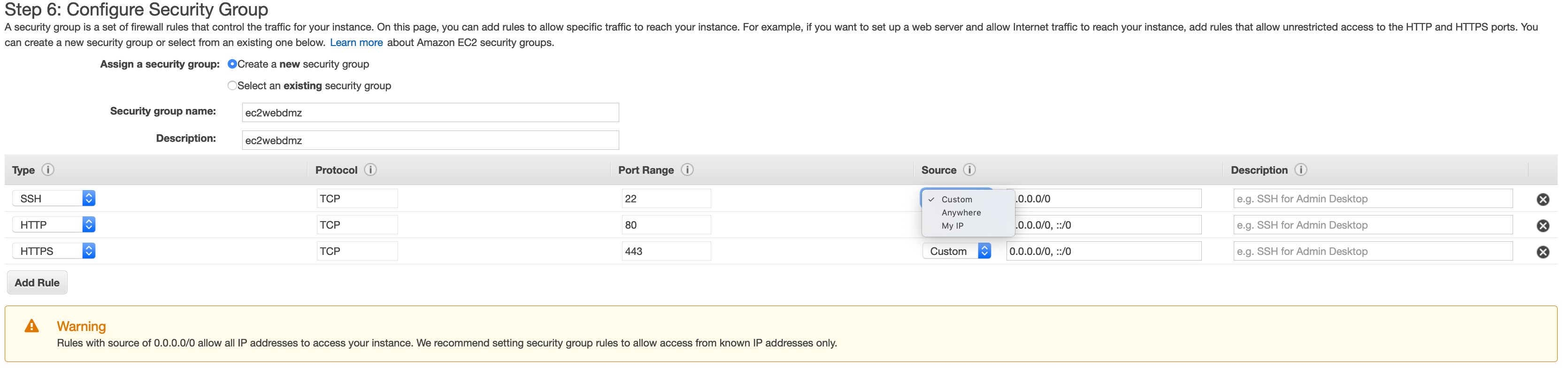
Add Storage:



* Delete on termination – virtual disk will be deleted on termination (default action for EBS volumes)
* SC1 and st1 only available for additional volume
* GP2, io1, magnetic can be root bootable volume

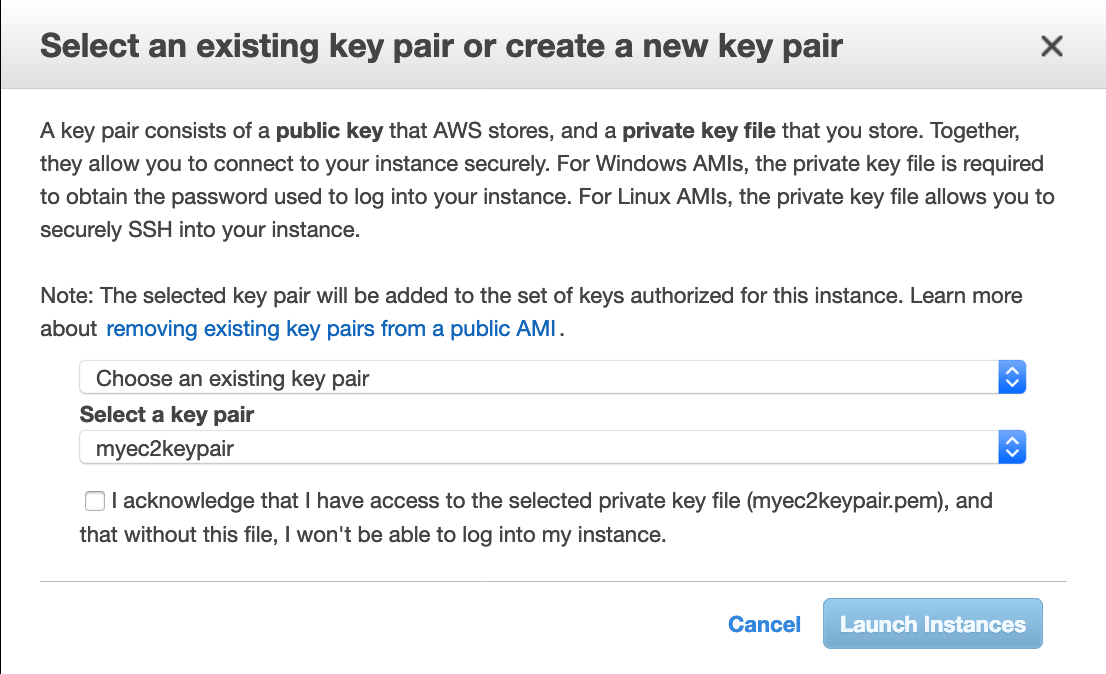
Tags can be added to track ec2 instance

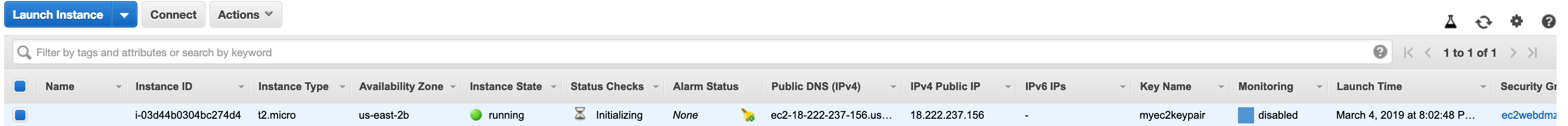
Security group:



* This is a virtual firewall. Here we allow access.
* My IP allows access from a fixed ip.

While launching instance, we can create a key pair which can be used to SSH to instance which is a .pem file.





SSH into an EC2:

**1)Lock down key file.**

CHMOD 400 myec2keypair.pem

**2)SSH into instance using public ip and keypair.**

ssh ec2-user@18.222.237.156 -i myec2keypair.pem

\_\_| \_\_|\_ )

\_| ( / Amazon Linux 2 AMI

\_\_\_|\\_\_\_|\_\_\_|

https://aws.amazon.com/amazon-linux-2/

1 package(s) needed for security, out of 3 available

Run "sudo yum update" to apply all updates.

**3)elevate and update**

sudo su

[root@ip-172-31-26-212 ec2-user]# yum update -y

**4)Install apache**

yum install httpd

**5)Put a html page**

cd /var/www/html

nano index.html

Put html contents in file and hit ctrl+x, then enter

**6)Start apache service**

service httpd start

**7)Go to web browser and type** <http://18.222.237.156>

**8)Start apache HTTP service every time instance is rebooted**

chkconfig httpd on

Note: Forwarding request to 'systemctl enable httpd.service'.

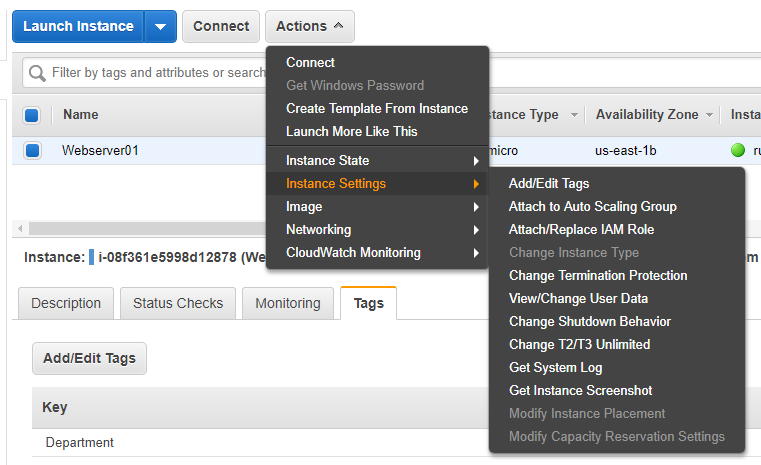
Created symlink from /etc/systemd/system/multi-user.target.wants/httpd.service to /usr/lib/systemd/system/httpd.service.

If an instance has termination protection on, we have to disable termination protection before we can terminate the instance.

SSH using windows:

Puttygen : Using puttygen, load the pem file and save the private key as PPK.

In putty: host = public ip, user = [ec2-user@18.222.237.156](mailto:ec2-user@18.222.237.156), connection = ssh > Auth > Load PPK file.



**Status Checks:**

There are 2 types of status checks:

* System status checks (it is reachable) – in case of failure, terminate instance and relaunch, maybe an issue with infrastructure.
* Instance status checks (traffic can go to OS) – in case of failure, reboot your instance

**Terminate Instance:**

Actions>Instance-state>Terminate

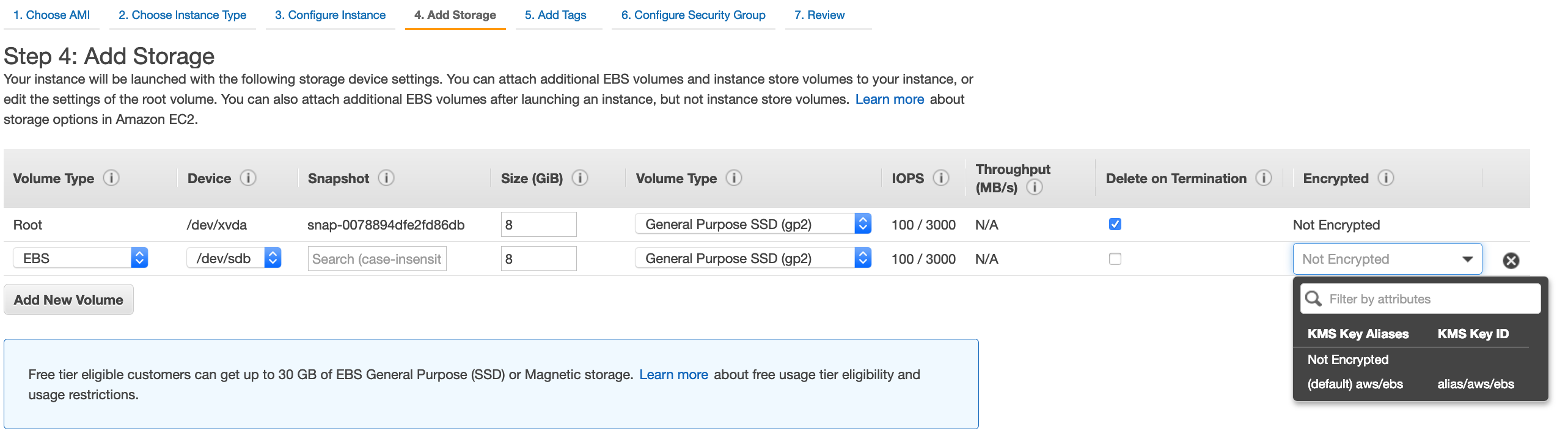
**Reserved Instance:**

Instances>Reserved Instances: All upfront saves you the most.

We cannot encrypt the root device volume for the default AMIs provided by amazon.

We can create a copy of the AMI and encrypt the root device volume of that AMI.

We can encrypt additional volumes.



Exam Tips:

* Termination protection turned off by default
* On an EBS instance, by default root EBS volumes will be deleted on instance termination
* Root EBS volumes cannot be encrypted. However we can encrypt amis while creating them or for windows amis, we can run bitlocker inside.

**Security Groups:**

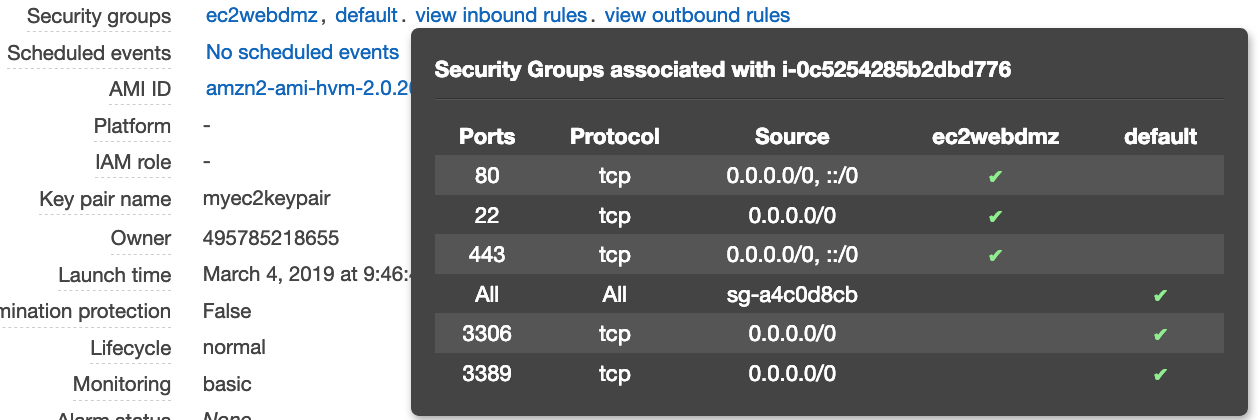
They are virtual firewalls that control traffic to EC2 instance. One EC2 instance can be behind multiple security groups. If a security group associated to an EC2 instance defines SSH rules that allows login only from a particular ip, then no one can ssh to ec2 instance apart from the owner of that ip.

Security groups have both outbound and inbound rules. Inbound rules control all requests to EC2 server. Outbound rules control any response from the EC2 server.

* Any changes to rules in security groups are immediately effective.
* All inbound traffic is blocked and outbound traffic is allowed by default. We have to create rules to allow them in. We cannot explicitly deny any traffic since everything apart from what we have allowed is blocked by default. Hence only allow rules and no deny rules.
* All rules in security groups are stateful which means that any inbound rule will have the corresponding outbound rule enabled by default. No outbound rules need to be added specifically.
* One ec2 instance can be behind multiple security groups and one security group can have multiple ec2s. To change security groups assigned to an EC2, go to

Actions>networking>change security groups and check or uncheck security groups.

* We cannot block ip addresses using security groups, we need to use Network access control lists.

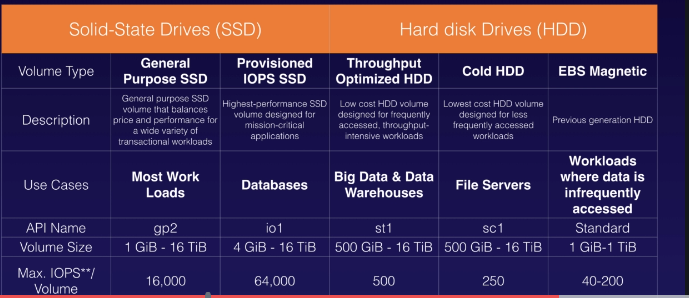


**Exam Tips:**

* All inbound traffic is blocked by default.
* All security group rules are **STATEFUL**, any inbound defined rule will have a corresponding outbound rule enabled by default.
* All changes to security groups are effective immediately.
* Any number of ec2 instances in a security groups and one ec2 instance can have multiple security groups
* Specific IP addresses cannot be blocked using security groups.
* No deny rules, only allow rules.

**EBS (Elastic block storage)**

This is a virtual disk just like EC2 is a virtual machine. It provides persistent block storage volumes for use with EC2. Each volume is replicated within availability zone to protect from failure. It allows to create storage volumes and then add to EC2 instance. Once attached we can create a file system, run a database etc. They are placed in a specific availability zone and are automatically replicated to protect from failure.



We can add volumes to ec2 while launching. After instance and volumes are created, we can modify volume type and size through actions>modify volumes under Elastic Block Store > volumes. Thus we do not have to shut down the instance but we can change the type and size of its storage.

**Moving an instance and attached volumes to different availability zone:**

* Create a snapshot of the root volume. Can be found under snapshots.
* Create an image (AMI) out of it.
* Launch an instance out of the AMI. Choose appropriate subnet maybe us east 1f. Both instance and volume will be in 1f now.

Snapshot -> AMI -> Launch instance into different availability zones.

We can also copy the AMI into different regions and then launch an instance out of the ami in availability zones of that region.

Deregistering the AMI will clean up the AMI.

**Exam Tips:**

* Volumes exist on EBS but snapshots (point in time copies) of volumes exist in S3.
* Snapshots are incremental, if we take a snapshot of a volume and after sometimes again take a snapshot of the same volume, then the delta will be copied over to S3.
* It is recommended to take snapshots of root devices while the instance is stopped, otherwise the snap may become inconsistent.
* EBS volume size and type can be changed on the fly.
* The EBS will always be in same availability zone as the ec2 instance.
* The EBS volume will be deleted if ec2 is terminated.
* EC2 instances can be migrated to different availability zones and regions by first creating a snapshot 🡪 create AMI out of snapshot 🡪 Launch instance from AMI