**ELASTIC COMPUTE [EC2]**

Amazon Elastic Compute Cloud (Amazon EC2) provides scalable computing capacity in the Amazon Web Services (AWS) cloud. Using Amazon EC2 eliminates your need to invest in hardware up front, so you can develop and deploy applications faster.

**Features of EC2**

* Renting Virtual machines [EC2]
* Preconfigured templates for your instances, known as *Amazon Machine Images (AMIs)*, that package the bits you need for your server (including the operating system and additional software)
* Persistent storage volumes for your data using Amazon Elastic Block Store known as *Amazon EBS volumes [EBS]*
* Distributing load across machines [ELB]
* Scaling the services using an auto-scalling group [ASG]

**AWS REGIONS**

* A region is a cluster of data centers

**CLI COMMANDS**

#aws ec2 describe-regions describe the Regions that are enabled for your account

#aws ec2 describe-regions --all-regions any Regions that are disabled for your account

**AWS AVAILABILITY ZONES [AZ]**

* Each region has many AZ [usually 3, min:3 max:6]
* Each AZ is one or more discrete data centers with reduntant power, n/w and connectivity.
* They are seprate from each other, so that thery are isolated from disasters.
* They are connected with high bandwidth, ultra latency n/w**.**

**CLI COMMANDS**

#aws ec2 describe-availability-zones --region region-name AZ & Local Zones in specified Region

#aws ec2 describe-availability-zones --all-availability-zones

**Local Zones**

A Local Zone is an extension of an AWS Region in geographic proximity to your users. When you launch an instance, you can select a subnet in a Local Zone. Local Zones have their own connections to the internet and support AWS Direct Connect, so resources created in a Local Zone can serve local users with very low-latency communications.

The Availability Zones and Local Zones are listed under Service Health, Availability Zone Status.in EC2 DASBOARD

**AMI**

* An *Amazon Machine Image (AMI)* is a template that contains a software configuration (for example, an operating system, an application server, and applications)
* AMI built for specific region only
* All AMIs are categorized as either *backed by Amazon EBS*, which means that the root device for an instance launched from the AMI is **an Amazon EBS volume**, or *backed by* ***instance store***, which means that the root device for an instance launched from the AMI is an instance store volume created from a template stored in Amazon S3.

**Custom AMI**

* Pre-installed packages needed
* Fatser boot time [no need for ec2 user data at boot time]
* Machine comes configured with monitoring / enterprice software
* Security concerns: control over the machines in the network
* Active Directory Integration out of box
* Installing your app ahead of time [for faster deploys when auto scaling]
* Using some1’s AMI that is optimized for running an app DB etc..

**AMI Storage**

* Your AMI take space and they live in s3
* S3 is a durable cheap and resilient storage where most of your backups will live [but you see in the s3 console]
* By default your AMI are private locked for your account / region.
* You can also make your AMI s public and share them with other AWS A/C or sell them on the market place.

**Cross Account AMI copy**

* You can share an AMI with another AWS account.
* Sharing an AMI does not affect the ownership of the AMI.
* If you copy an AMI that has been shared with your account, you are the owner of the target AMI account
* To copy an AMI that was shared with you from another account, the owner of the source AMI grant read permission for the storage that backs the AMI, either the associated EBS snapshot [for an Amazon EBS backed AMI] or an s3 bucket [for an instance store backed AMI]
* Limits:
* You can’t copy an encrypted AMI that was shared with you from another account. Instead if the underlying snapshot and encryption key were shared with you, you can copy the snapshot while encrypting it with a key of your own. You own the copied snapshot and can register it asa new AMI.
* You can’t copy an AMI with an associated billing product code that was shared with from another account. This includes Windows AMI and AMI from market place. To copy a shared AMI with a billing product code launch an EC2 instance in your account using the shared AMI and then create an AMI from the instance.

**EC2 Hibernate**

* Stop: the data on disk [EBS] is kept intact in the next start.
* Terminate: any EBS Volumes [root] also set up to be destroyed is lost.
* On Start the following happens:
* First start: the OS boot & the EC2 User Data script is run
* Following starts: the OS boot up
* Then your application starts, caches, get warmed up and than it take time!
* Hibernate:

The in-Memory [RAM] state is preserved.

The Instance boot is mush faster ! [The OS is not stopped / restarted]

The RAM state is written to a file in the root EBS volume.

The root EBS volume must be encrypted.

Instance RAM size must be less than 150GB

Not supported for bare metal instances.

Available on-demand & reserved instances

An instances cannot be hibernated more than 60 days

Use cases:

Long running processing

Saving RAM state

Services that take time to initialize

**Connecting to your Linux instance by Linux SSH**

#ssh -i /path/my-key-pair.pem my-instance-user-name@my-instance-public-dns-name

ssh –I <ec2.pem> [ec2-user@192.15.100.144](mailto:ec2-user@192.15.100.144)

**Connecting to your Linux instance by Windows putty**

* Connecting to your Linux instance by Windows SSH [WIN 10>]
* Download pem file & go to properties security and make owner as your name and remove other users by enable inheritance.
* Open Powershell or cmd >>>>$ssh [ec2-user@100.12.12.11](mailto:ec2-user@100.12.12.11) –I <pem file>

**Connecting to your Linux instance by EC2 instance connect**

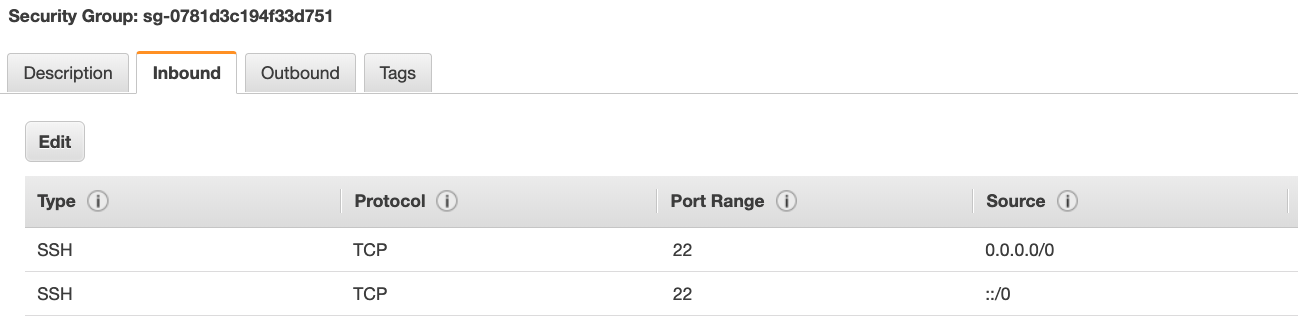
[It supports only ubumtu & amazon linux ami]

**SSH Troubleshooting**

#### **SSH Troubleshooting**

**1) There's a connection timeout**

This is a security group issue. Any timeout (not just for SSH) is related to security groups or a firewall. Ensure your security group looks like this and correctly assigned to your EC2 instance.



**2) There's still a connection timeout issue**

If your security group is properly configured as above, and you still have connection timeout issues, then that means a corporate firewall or a personal firewall is blocking the connection. **Please use EC2 Instance Connect as described in the next lecture.**

**3) SSH does not work on Windows**

* If it says: ssh command not found, that means you have to use Putty
* Follow again the video. If things don't work, please use EC2 Instance Connect as described in the next lecture

**4) There's a connection refused**

This means the instance is reachable, but no SSH utility is running on the instance

* Try to restart the instance
* If it doesn't work, terminate the instance and create a new one. Make sure you're using **Amazon Linux 2**

**5)  Permission denied (publickey,gssapi-keyex,gssapi-with-mic)**

This means either two things:

* You are using the wrong security key or not using a security key. Please look at your EC2 instance configuration to make sure you have assigned the correct key to it.
* You are using the wrong user. Make sure you have started an **Amazon Linux 2 EC2 instance**, and make sure you're using the user **ec2-user.**This is something you specify when doing **ec2-user@**<public-ip> (ex: ec2-user@35.180.242.162) in your SSH command or your Putty configuration

**6) Nothing is working - "aaaahhhhhh"**

Don't panic. Use **EC2 Instance Connect** from the next lecture. Make sure you started an **Amazon Linux 2** and you will be able to follow along with the tutorial :)

**Security Group**

* It controls how inboud and outboubd traffic is allowed into or out of our EC2 machines.
* It act as firewall on EC2 instances
* They regulate:
  + Access to ports
  + Authorized IP ranges [ipv4 –ipv6]
  + Control Inbound & outbound traffic
* Can be attached to multiple ec2 instances
* Locked down to a region /VPC combination [need to create new security grp for new region or new vpc]
* Can be attached to multiple ec2 instances
* If your application is not accessible (timeout error), then its security grp issue.
* If your application gives a “connection refused” error then it’s application error or its not launched.

**EC2 User Data [bash script #!/bin/bash]**

* It is possible to bootsrap our instances using an EC2 User data script.
* Bootstraping means launching commands when a machine starts.
* That script is only run once at the instance first start.
* It used to automate boot task such as:
  + Installaing updates
  + Installing softwares
  + Downloading common files from internet.

**EC2 Instance Types**

* R: applications that needs a lot of RAM : in memeory cache
* C: applications that needs good CPU: compute / database
* M: applications that are balanced [think “medium”]: general / web app
* I: applications that need global local I/O [instance storage] : database
* G: applicationsthat need a GPU : video rendering / machine learning
* T2/ T3: burstable instances [upto a capacity]
* T2/ T2: unlimited: unlimited burst

**EC2 Instance Launch Types**

* OnDemand Instance: Short workload, Predictable price

Pay for what u use [Billing per second, after first minute]

* Reserved: Min 1 year
* Reserved Instances: Long workloads
* Pay upfront for what you use with long term commitment
* Recommended for steady state usage applications [thing database]
* Upto 75 percent discount
* Convertible Reserved Instances: Long workloads with flexible instances
* Can change the EC2 insatnce type
* Upto 54% discount
* Scheuled Reserved Insatnces: example every thurday 3 AM to 5PM
* Spot Instances: short workload, for cheap, can lose instances [less reliable]
* Can get 90% discount compared to on-demand
* Instances that you can “lose” at any point of time if your max price is less than the current spot price.
* Most cost efficient but workloads resilient to failure
  + Batch jobs
  + Data Analysis
  + Image Processing
* Dedicated Instances: Book an entire physical server, control instance placement
* Physical dedicated EC2 server for your use.
* Full control of EC2 Instance Placement
* Visibility into the underlying sockets / physical cores of the hardware
* Allocated for your account for a 3 years period reservation.
* More expensive.
* Useful for software that have complicated lincencing model.

**Elastic Network Interface [ENI]**

* Logical component in a VPC that represents a virtual network card.
* Each ENI can have the following attributes:
* Primary private ipv4 one or more secoandary ipv4
* One Elastic IPv4 per private ipv4
* One or more security group.
* A MAC address
* You can create ENI independentently and attach them on the fly [move them] on EC2 instances for failover.
* Bound to specific Availability Zone