

-: Amazon Elastic Compute Cloud (EC2):-

What is AWS EC2?

1. EC2 is a web service from Amazon that provides re-sizable compute capacity in the cloud.
2. It is designed for developers to have complete control over web-scaling and computing resources.
3. EC2 instances can be resized and the number of instances scaled up or down as per our requirement.
4. These instances can be launched in one or more geographical locations or regions, and Availability Zones (AZs).
5. Each region comprises of several AZs at distinct locations, connected by low latency networks in the same region.
6. EC2 is a most important service in the whole of the compute Domain.
7. LAMBDA and Elastic Beanstalk are the advanced version of the EC2. EC2 is a raw server.

What is an Instance?

1. An instance is a virtual server for running applications on Amazon's EC2.
2. It can also be understood like a tiny part of a larger computer, a tiny part which has its own Hard drive, network connection, OS etc.
3. But it is actually all virtual. You can have multiple "tiny" computers on a single physical machine, and all these tiny machines are called Instances.

Difference between service and an instance ?

1. EC2 is a service along with other Amazon Web Services like S3 etc.
2. When we use EC2 or any other service, we use it through an instance, e.g. t2.micro instance, in EC2 etc.

What are the EC2 Components ?

In AWS EC2, the users must be aware about the EC2 components, their operating systems support, security measures, pricing structures, etc.

Migration:-

This service allows the users to move existing applications into EC2.

It costs \$80.00 per storage device and \$2.49 per hour for data loading.

This service suits those users having large amount of data to move.

What are the Features of EC2?

Here is a list of some of the prominent features of EC2 –

1. Reliable – Amazon EC2 offers a highly reliable environment where replacement of instances is rapidly possible. Service Level Agreement commitment is 99.9% availability for each Amazon EC2 region.

2. Designed for Amazon Web Services – Amazon EC2 works fine with Amazon services like Amazon S3, Amazon RDS, Amazon DynamoDB, and Amazon SQS. It provides a complete solution for computing, query processing, and storage across a wide range of applications.

3. Secure – Amazon EC2 works in Amazon Virtual Private Cloud to provide a secure and robust network to resources.

4. Flexible Tools – Amazon EC2 provides the tools for developers and system administrators to build failure applications and isolate themselves from common failure situations.

5. Inexpensive – Amazon EC2 wants us to pay only for the resources that we use. It includes multiple purchase plans such as On-Demand Instances, Reserved Instances, Spot Instances, etc. which we can choose as per our requirement.

Why AWS EC2? (or) Why not buy your own stack of servers and work independently?

1. Because, suppose you are a developer, and since you want to work independently you buy some servers, you estimated the correct capacity, and the computing power is enough.
2. Now, you have to look after the updating of security patches every day, you have to troubleshoot any problem which might occur at a back end level in the servers and so on.
3. These are all extra chores that you will be doing or maybe you will hire someone else to do these things for you.
4. But if you buy an EC2 instance, you don't have to worry about any of these things as it will all be managed by Amazon; you just have to focus on your application.
5. That too, at a fraction of a cost that you were incurring earlier! Isn't that interesting?

Let's understand the types of EC2 Computing Instances:

Computing is a very broad term; the nature of your task decides what kind of computing you need.

Therefore, AWS EC2 offers 5 types of instances which are as follows:

1. **General Instances:** For applications that require a balance of performance and cost.
 - E.g email responding systems, where you need a prompt response as well as it should be cost effective, since it doesn't require much processing.
2. **Compute Instances:** For applications that require a lot of processing from the CPU.
 - E.g analysis of data from a stream of data, like Twitter stream
3. **Memory Instances:** For applications that are heavy in nature, therefore, require a lot of RAM.
 - E.g when your system needs a lot of applications running in the background i.e multitasking.
4. **Storage Instances:** For applications that are huge in size or have a data set that occupies a lot of space.
 - E.g When your application is of huge size.
5. **GPU Instances:** For applications that require some heavy graphics rendering.
 - E.g 3D modelling etc.

Now, every instance type has a set of instances which are optimized for different workloads:

1. General Instances are **t2, m3 and m4**
2. Compute Instances are **c3 and c4**
3. Memory Instances are **r3 and x1**
4. Storage Instances are **i2 and d2**
5. GPU Instances are **g2**

Now let's understand the kind of work that each instance is optimized for, in this AWS EC2 Tutorial:

Burstable Performance Instances

- *T2 instances* are burstable instances, meaning the CPU performs at a base line; say 20% of its capability. When your application needs more than 20% of the performance of the CPU, the CPU

enters into a burst mode giving higher performance for a limited amount of time, therefore work happens faster.

- You get these credits when your CPU is idle.
- Each CPU credit gives a burst of 1 minute to the CPU.
- If your CPU credits are not used they are credited to your account and they stay there for 24 hours.
- Based on your credit balance, you can decide whether the t2 instance, should be scaled up or down.
- These bursts happen at a cost, every time a burst happens in a CPU, CPU credits are used.

EBS-optimized Instances

- *C4, M4, and D2 instances*, are EBS optimized by default, EBS means Elastic Block Storage, which is a storage option provided by AWS in which the IOPS* rate is quite high. Therefore, when an EBS volume is attached to an optimized instance, single digit millisecond latencies can be achieved.

*IOPS (Input/Output Operations Per Second, pronounced eye-ops) is a performance measurement used to characterize computer storage devices.

Cluster Networking Instances

- *X1, M4, C4, C3, I2, G2 and D2 instances* support cluster networking. Instances launched into a common placement group are put in a logical group that provides high-bandwidth, low latency between all the instances in the group.
 - A placement group is basically a logical cluster where some select EC2 instances which are a part of that group can utilize up to 10Gbps for single flow and 20Gbps for multi flow traffic in each direction.
 - Instances which are not a part of that group are limited to 5 Gbps speed in multi flow traffic. Cluster Networking is ideal for high performance analytics system.

Dedicated Instances

- They are the instances that run on single-tenant hardware dedicated to a single customer.
- They are perfect for workloads where a corporate policy or industry regulation requires that your instance should be isolated from any other customer's instance, therefore they go for their own separate machines, and their instances are isolated at the hardware level.

Let's understand this through an example. Suppose in our company Edureka, we have the following tasks:

- *Analysis of customer's data*
 - Customer's website activity, etc. should all be monitored in real-time. There will be times when the traffic on the website will be minimum, therefore using a very powerful processor should not be considered, since it will become expensive for the company because it will not be used for every hour of the day. Hence, for this task, we might take **t2 instances** because they give **Burstable CPU performance** i.e when the traffic will be more the CPU performance will be increased accordingly to meet the requirements.
- *Our auto-response emailing system*
 - It should be quick, therefore we would require systems, where the response time is as short as possible. This could be achieved by using **EBS optimized instances**, as they offer high IOPS and hence, low latencies.
- *The search engine on our website*

- It should be able to sort the keywords and return relevant results, therefore we might have 2 servers for this. One is the database and the other server for processing the keywords. Therefore, the communication between these servers should be at the maximum possible rate. To achieve this, we can put them in a placement group and for that we have to use **Cluster Networking Instances**.
- *Some processes in every organization are highly confidential*
 - Because these processes give us an edge over other companies, no matter how secure the servers, maybe, some policies are still made to be sure. Therefore, we might use **Dedicated Instances** for these kind of processes.

We now know about instances, let's learn how to launch these instances?

How to run systems in EC2?

- Login to your AWS account and click on AWS EC2.
- Under create instance, click on launch instance.

Now you have to select an **Amazon Machine Image (AMI)**, AMI is templates of OS and they provide the information needed to launch an instance.

When we want to launch an instance we have to specify which AMI we want to use. It could be Ubuntu, windows server etc.

- The AMIs could be preconfigured or you can configure it on your own according to your requirements.
 - For preconfigured AMIs you have to select it from AWS marketplace.
 - For setting up your own, go to quick-start and select one.
- While configuring you will reach a point where you have to select an **EBS** storage option.

Elastic Block Storage (EBS) is a persistent block level storage volumes which are used with EC2. Here each block acts as a hard drive.

But why do we need EBS with EC2?

Just like your computer needs a hard drive, you need AWS EC2 Tutorial, AWS EC2 needs a storage volume to store the OS that your instance will be specifying. Options for EBS are:

Provisioned IOPS: This category is for workloads which are mission critical, it provides high IOPS rates.

General Purpose: It is for workloads which need a performance and cost balance.

Magnetic: It is for data which is accessed less frequently, and also retrieval time is more.

- After selecting a suitable option in EBS, we give the instance a name and then we create a **security group**.
- A security group acts as a firewall to control inbound and outbound traffic. Each security group has rules according to which the traffic is governed.
- Each instance, can be assigned up to 5 security groups.
- Finally, in the last step the console shows all the settings that you have done, you can verify and launch it.

Now in this AWS EC2 Tutorial, when you launch the instance, it will need some authentication from you; otherwise anyone can access your instance! To address that in this AWS EC2 Tutorial, let's take a plunge into the security aspects of EC2,

Security in AWS EC2

To authenticate users to their instances, AWS employs a **key pair** method.

What is key pair?

Amazon EC2 uses public-key cryptography to encrypt and decrypt login information. Public-key cryptography uses a public key to encrypt a piece of data, such as a password, then the recipient uses the private key to decrypt the data. The public and private keys are known as a *key pair*.

Additional Benefits: Every service from Amazon is designed keeping in mind the customer. They claim to be the earth's most customer-obsessed company. Having said that let's understand some other benefits of EC2.

Auto Scaling

Auto Scaling is a service designed by AWS EC2, which automatically launch or terminate EC2's instances based on user defined policies, schedules and health checks.

Elastic Load Balancing

- *Elastic Load Balancing* (ELB) automatically distributes incoming application traffic across multiple EC2 instances, in multiple Availability Zones.
- Availability zones are basically places where amazon has set up their servers. Since they have customers from the whole globe, they have set up multiple Availability zones to reduce the latency.
- *Elastic IP Addresses* are static IP addresses which are associated with your AWS account, they can be used to mask the failure of an instance by automatically remapping your address to another working instance in your account.

AWS EC2 Pricing

In this AWS EC2 Tutorial, let's start with the free things first!

AWS EC2 free tier allows 750 hrs of t2.micro instance usage per month!

The free tier for EC2 is valid for 1 year from SignUp of your AWS account.

There are basically 3 pricing options in EC2:

- Spot Instances
- On Demand Instances
- Reserved Instances

Spot Instances is a pricing option which enables you to bid on unused EC2 instances. The hourly price for a Spot Instance is set by AWS EC2, and it fluctuates according to the availability of the instances in a specific Availability zone.

- Basically, you will set a price for an instance above which you do not wish to get charged for.
- The price that you set is for per hour basis, therefore the moment the price for that instance becomes greater than what you have set, the instance gets shut down automatically.

On Demand Instances are used when you want to pay for the hour, with no long term commitments and upfront payments. They are useful for applications that may have unpredictable workloads or for test applications that are being deployed for the first time.

Reserved Instances provide you with significant discounts as compared to On Demand Instances. With Reserved Instances you reserve instances for a specific period of time with three payment options:

- No Upfront
- Partial Upfront
- Full Upfront

And two term lengths:

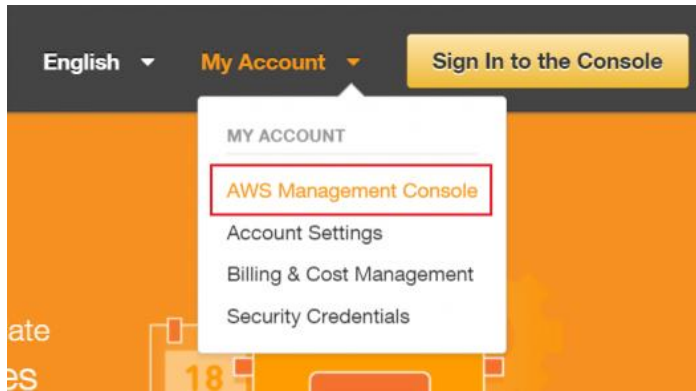
- One Year Term
- Three Year Term

The higher the upfront payment is, the more you save money.

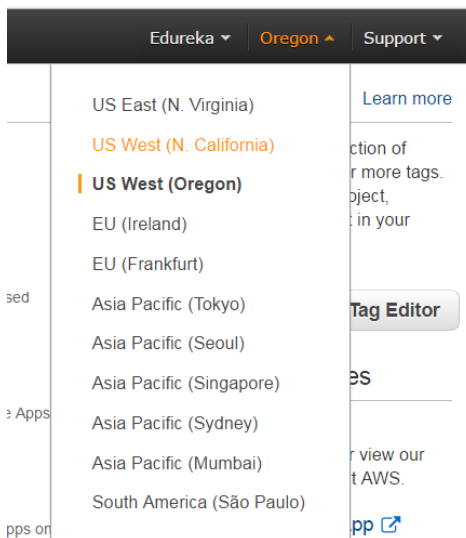
-: AWS EC2 Use Case 1 :-

Next in this AWS EC2 Tutorial, let's understand the whole EC2 instance creation process through a use case in which we'll be creating an Ubuntu instance for a test environment.

- **Login to AWS Management Console.**



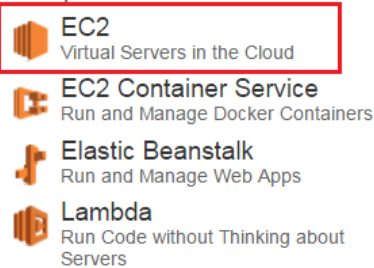
- **Select your preferred Region.** Select a region from the drop down, the selection of the region can be done on the basis of the criteria discussed earlier in the blog.



- **Select EC2 Service** Click EC2 under Compute section. This will take you to EC2 dashboard.

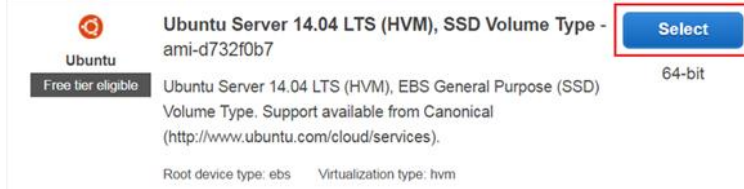
Amazon Web Services

Compute



- Click **Launch Instance.**

- **Select an AMI :** because you require a Linux instance, in the row for the basic 64-bit Ubuntu AMI, click Select.



Ubuntu Server 14.04 LTS (HVM), SSD Volume Type - ami-d732f0b7

Free tier eligible

64-bit

Root device type: ebs Virtualization type: hvm

- **Choose an Instance**

Select t2.micro instance, which is free tier eligible.

Step 2: Choose an Instance Type

	Family	Type	vCPUs	Memory (GiB)	Instance Storage (GB)
<input type="checkbox"/>	General purpose	t2.nano	1	0.5	EBS only
<input checked="" type="checkbox"/>	General purpose	t2.micro Free tier eligible	1	1	EBS only

- **Configure Instance Details.**

Configure all the details and then click on add storage

Step 3: Configure Instance Details

Number of instances [Launch into Auto Scaling Group](#)

Purchasing option ☐ Request Spot instances

Network [Create new VPC](#)

Subnet [Create new subnet](#)
245 IP Addresses available

Auto-assign Public IP

IAM role [Create new IAM role](#)

[Cancel](#) [Previous](#) [Review and Launch](#) [Next: Add Storage](#)

- **Add Storage**

Step 4: Add Storage

Volume Type	Device	Snapshot	Size (GiB)	Volume Type	IOPS
Root	/dev/sda1	snap-47713105	8	General Purpose SSD (GP2)	100 / 3000

[Add New Volume](#) [Cancel](#) [Previous](#) [Review and Launch](#) [Next: Tag Instance](#)

- **Tag an Instance**

Type a name for your AWS EC2 instance in the value box. This name, more correctly known as tag, will appear in the console when the instance launches. It makes it easy to keep track of running machines in a complex environment. Use a name that you can easily recognize and remember.

Step 5: Tag Instance

Key	Value
Name	ec21-linux

[Create Tag](#) (Up to 50 tags maximum) [Cancel](#) [Previous](#) [Review and Launch](#) [Next: Configure Security Group](#)

- **Create a Security Group**

Step 6: Configure Security Group

Assign a security group: ☒ Create a **new** security group

☐ Select an **existing** security group

Security group name:
Description:

Type	Protocol	Port Range
SSH	TCP	22


[Add Rule](#) [Cancel](#) [Previous](#) [Review and Launch](#)

- **Review and Launch an Instance**

Verify the details that you have configured to launch an instance.

Step 7: Review Instance Launch

▼ **AMI Details** [Edit AMI](#)

 **Ubuntu Server 14.04 LTS (HVM), SSD Volume Type - ami-d732f0b7**

Free tier eligible Ubuntu Server 14.04 LTS (HVM), EBS General Purpose (SSD) Volume Type. Support available from Canonical
Root Device Type: ebs Virtualization type: hvm

▼ **Instance Type** [Edit instance type](#)

Instance Type	ECUs	vCPUs	Memory (GiB)	Instance Storage (GB)
t2.micro	Variable	1	1	EBS only

[Cancel](#) [Previous](#) [Launch](#)

- **Create a Key Pair & launch an Instance**

Next in this AWS EC2 Tutorial, select the option 'Create a new key pair' and give a name of a key pair. After that, download it in your system and save it for future use.

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

epdureka_aws

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

- Check the details of a launched instance.

Launch Instance

Connect

Actions

Filter by tags and attributes or search by keyword

1 to 17 of 17

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks
WPS Instance	i-89bf6251	m4.large	us-west-2a	running	2/2 checks
Ec21_linux	i-8d263195	t2.micro	us-west-2a	running	2/2 checks
DoNotTouch	i-43f6c6f	m1.large	us-west-2b	stopped	

Instances: i-8226319a (Ec21_linux), i-8326319b (Ec21_linux)

- Converting Your Private Key Using PuTTYgen

PuTTY does not natively support the private key format (.pem) generated by Amazon EC2. PuTTY has a tool called PuTTYgen, which can convert keys to the required PuTTY format (.ppk). You must convert your private key into this format (.ppk) before attempting to connect to your instance using PuTTY.

- Click Load. By default, PuTTYgen displays only files with the extension .ppk. To locate your .pem file, select the option to display files of all types.

PuTTY Key Generator

File Key Conversions Help

Key

No key.

Actions

Generate a public/private key pair

Generate

Load an existing private key file

Load

Save the generated key

Save public key

Save private key

Parameters

Type of key to generate:

SSH-1 (RSA)

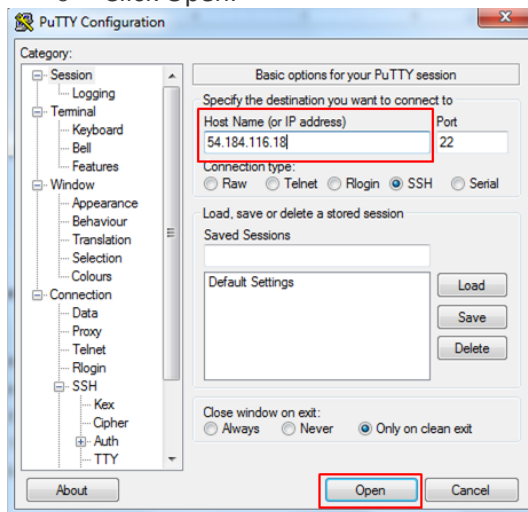
SSH-2 RSA

SSH-2 DSA

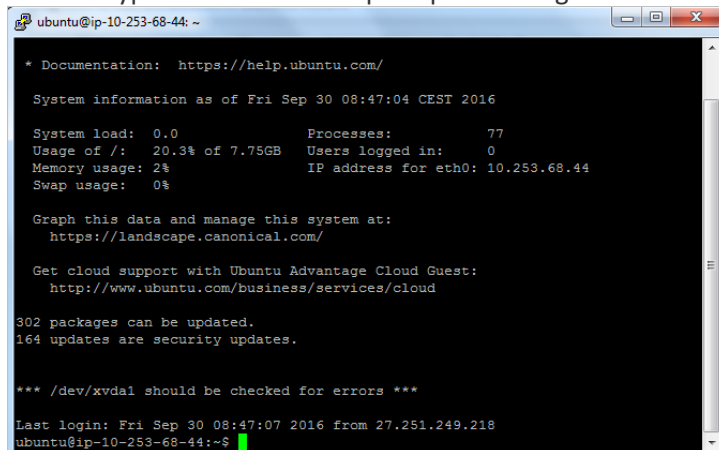
Number of bits in a generated key:

2048

- Select your.pem file for the key pair that you specified when you launch your instance, and then click Open. Click OK to dismiss the confirmation dialog box.
- Click Save private key to save the key in the format that PuTTY can use. PuTTYgen displays a warning about saving the key without a passphrase. Click Yes.
- Specify the same name for the key that you used for the key pair (for example, my-key-pair). PuTTY automatically adds the .ppk file extension.
- **Connect to EC2 instance using SSH and PuTTY**
 - Open PuTTY.exe
 - In the Host Name box, enter Public IP of your instance.
 - In the Category list, expand SSH.
 - Click Auth (don't expand it).
 - In the Private Key file for authentication box, browse to the PPK file that you downloaded and double-click it.
 - Click Open.



- Type in Ubuntu when prompted for login ID.



Congratulations! You have launched an Ubuntu Instance successfully.

Here's a short AWS EC2 tutorial Video that explains Amazon AMI EC2, Demo on AMI creation, Security groups, Key pairs, Elastic IP vs Public IP and a Demo to launch an EC2 Instance etc. This AWS EC2 tutorial is very important for those who want to become AWS Certified Solutions Architect.