Ec2

22 May 2023 11:27

What is AWS EC2?

Amazon Elastic Compute Cloud, EC2 is a web service from Amazon that provides **re-sizable** compute services in the cloud.

Features of Amazon EC2

- Instance Families are collections of instance types designed to meet a common goal. To make it easier for you to select the best option for your applications, Amazon EC2 instance types are grouped together into families based on target application profiles.
- A vCPU is a virtual Central Processing Unit (CPU). A multicore processor has two or more vCPUs.
- Virtual computing environments, known as instances
- 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). Instance Types comprise various combinations of CPU, memory, storage, and networking capacity and give you the flexibility to choose the appropriate mix of resources for your applications. Each instance type has one or more size options that address different workload sizes. For the best experience, you should launch on instance types that are the best fit for your applications.
- Various configurations of CPU, memory, storage, and networking capacity for your instances, known as instance types
- Secure login information for your instances using key pairs (AWS stores the public key, and you store the private key in a secure place)
- Storage volumes for temporary data that's deleted when you stop, hibernate, or terminate your instance, known as instance store volumes
- Persistent storage volumes for your data using Amazon Elastic Block Store (Amazon EBS), known as Amazon EBS volumes
- Multiple physical locations for your resources, such as instances and Amazon EBS volumes, known as Regions and Availability Zones
- A firewall that enables you to specify the protocols, ports, and source IP ranges that can reach your instances using security groups
- Static IPv4 addresses for dynamic cloud computing, known as Elastic IP addresses
- Metadata, known as tags, that you can create and assign to your Amazon EC2 resources
- Virtual networks you can create that are logically isolated from the rest of the AWS Cloud, and that you can optionally connect to your own network, known as virtual private clouds (VPCs)

New

- Virtual Machines: EC2 enables users to create and manage virtual machines, known as instances, in the AWS cloud. These instances can be easily scaled up or down to meet changing requirements.
- Instance Types: EC2 offers a wide range of instance types, each optimized for specific use cases. Instance types vary in terms of computing power, memory, storage, and networking capabilities, allowing users to select the most appropriate configuration for their workloads.
- Operating Systems: EC2 supports a variety of operating systems, including various Linux distributions, Windows Server, and other operating systems such as FreeBSD. Users can choose the desired operating system when launching an EC2 instance.
- Scalability: One of the key benefits of EC2 is its scalability. Users can easily increase or decrease the number of instances based on demand, ensuring that applications can handle fluctuations in traffic and workload.
- Elastic IP Addresses: EC2 provides elastic IP addresses that can be associated with instances. These IP addresses are static and can be remapped to different instances if needed, allowing users to maintain a consistent IP address for their applications.

- Storage Options: EC2 instances can be equipped with various storage options, including Amazon Elastic Block Store (EBS) for block-level storage, Amazon S3 for object storage, and instance store volumes for temporary storage. Users can choose the appropriate storage type based on their performance and durability requirements.
- Security: EC2 offers several security features, including virtual private clouds (VPCs) for network isolation, security groups for controlling inbound and outbound traffic, and access management through AWS Identity and Access Management (IAM). Users can also enable encryption for data at rest and in transit.
- Integration with Other AWS Services: EC2 seamlessly integrates with other AWS services, such as Amazon RDS for managed relational databases, Amazon S3 for object storage, and Amazon VPC for networking. This allows users to build comprehensive and scalable architectures using multiple AWS services.
- Pricing: EC2 pricing is based on various factors, including instance type, region, usage duration, and additional services utilized. Users can choose between on-demand instances for flexible and short-term workloads or reserved instances for longer-term commitments, which offer significant cost savings.
- Auto Scaling: EC2 Auto Scaling allows users to automatically adjust the number of instances based on predefined scaling policies. This feature helps maintain application availability, optimize resource utilization, and handle traffic spikes efficiently.

Real Time Benefits of Ec2 : -

- There is no need to hire an IT team which can handle them.
- Also, having a fault in the system is unavoidable, therefore we have to bear the cost of getting it fixed, and if you don't want to compromise on your up-time you have to keep your systems redundant to other servers, which might become more expensive.
- Your own purchased assets will depreciate over the period of time, however, as a matter of fact, the cost of an instance have dropped more than 50% over a 3-year period, while improving processor type and speed. So eventually, moving to Cloud is all more suggested.
- For scaling up we have to add more servers, and if your application is new and you experience sudden traffic, scaling up that quickly might become a problem.

AWS EC2 Instance Types Comparison

	General- Purpose	Compute-Optimized	Memory-Optimized	Accelerated-Computing	Storage-Optimized
• Example applications	• Web server, code repository • Enterprise-level apps, mobile, and game dev environments	• Data analytics, video processing • Processor with high computing power, scientific modelling, complex calculations	• In-memory databases, Apache Spark • Processing large datasets, big data analytics	• Video rendering, GPU-assisted machine learning • GPU, FPGA (graphic cards)	• Distributed databases, data analytics • Data caching, low latency, high speed, data analytics, data warehousing
• Special features	• Burstable performance (some instances)	• SSD storage (some instances)	• SSD storage (most instances; some instances have no persistent storage)	• Special GPUs (different types available for different instances and use cases)	• High-throughput disks (SSD as well as conventional options)

Ec2 Pricing options:

- **On-demand**, which provides instant access to as much compute, memory and storage as you need. However, on-demand pricing is generally the most expensive.

Features of On Demand : -

- Billing by hour or second is applicable to users who do not need any upfront payment or long-term commitment.

- For applications with Applications with short-term, spiky, or unpredictable workloads that cannot be interrupted.
- For applications being developed or tested on Amazon EC2 for the first time.

- **Spot instances**. With spot instances, you can bid for EC2 resources that otherwise go unused. Spot instances are kind of like eBay for EC2. They can save you a lot of money, but you have to work with whatever is available in the spot category at a given moment. You don't get to access resources on demand.

Features of Spot Instances : -

- Up to 90% off the On-Demand price.
- For applications that have flexible start and end times.
- For applications that are stateless or fault-tolerant.

- **Reserved instances**, which give you access to a dedicated but finite amount of resources. Reserved instances don't provide the scalability of on-demand pricing, but they are less expensive. They work well in cases where the demand on your application is consistent and predictable, and will not exceed the amount of capacity you reserve through a reserved instance.

Features of Reserved Instances : -

- Up to 72% off the On-Demand price.
- Customers that can commit to using EC2 over a 1- or 3-year term to reduce their total computing costs
- For applications with steady state usage.
- For applications that may require reserved capacity.

- **Dedicated hosts**. With a dedicated host, you get exclusive access to a physical server in the AWS cloud. Dedicated host pricing is generally expensive, but if used in the right way, a dedicated host can save you money. For example, because you get access to a physical server, you can install a commercial operating system on it using a license that you already own.

Features of Dedicated hosts : -

- Physical EC2 server dedicated for your use, up to 70% off the On-Demand price.
- For customers who need to use existing server-bound software licenses, including Windows Server, SQL Server, SUSE Server etc.
- For customers who need to meet compliance requirements.

Examples

- **Mnemonic: t is for tiny or turbo**
- **Mnemonic: m is for main choice or happy medium.**
- **Mnemonic: c is for compute (at least that one's easy!)**
- **Mnemonic: r is for RAM.**
- **Mnemonic: x is for xtreme, as in "xtreme RAM" seems to be generally accepted, but we think this is a bit weak.**
- **Mnemonic: h is for HDD.**
- **Mnemonic: i is for IOPS.**
- **Mnemonic: d is for dense.**

How to choose the right type of EC2 instance

- Considering the many instance types offered by AWS, it is difficult for someone to select the right EC2 instance for a particular application. Here are the steps to make that journey easier:

- Step 1 – Evaluate your application needs

- First, you need to evaluate your application workload. Will it need more memory or more CPU? Does it need high input/output operations per second? Identify the needs in terms of the following parameters
  - CPU
  - Memory
  - Storage
  - Networking
- After you have identified the needs in terms of the above attributes, the next step is to select the right instance family matching your needs.

- Step 2 – Select the instance type family based on application needs

- Now you need to select the instance type family suitable to your specific needs. As a general rule:
  - Select "**M**" family for general purposes, including web servers and databases
  - Select "**C**" family for compute-intensive workloads such as **HPC, gaming, etc**.
  - Select "**R**" family for memory-intensive applications like enterprise databases, big data analytics, etc.
  - Select "**T**" family for burstable workloads that are **spikey** in nature e.g., small databases, VDIs, etc.

- Step 3 – Experiment with different instances sizes in the same instance family

- After selecting an instance-type family, you should start with a basic instance in the same family and experiment with it. If it underperforms, upgrade to a better instance in the same family. Repeat it until you reach your desired instance size.

Ec2 Instance

EC2 Instance: AMI (OS) +
Instance Size (CPU + RAM) +
Storage +
Security Groups (Firewall attached to the EC2 instance) +
EC2 User Data (Script launched at the start of an instance)

EC2 (IaaS) mainly consists of the capability of:
- Renting virtual machines (EC2 instances)
- Storing data on virtual drives (EBS volumes)
- Distributing load across machines (ELB)
- Scaling the services using an auto-scaling group (ASG)

1.1 EC2 Sizing & Configuration Options
- How much compute power & cores (CPU)
- How much random-access memory (RAM)
- How much storage space: **Network-attached (EBS & EFS)**, **Hardware (EC2 Instance Store)**
- Network card: Speed of the card, Public IP address
- Firewall rules: security group
- Bootstrap script (configure at first launch): EC2 User Data

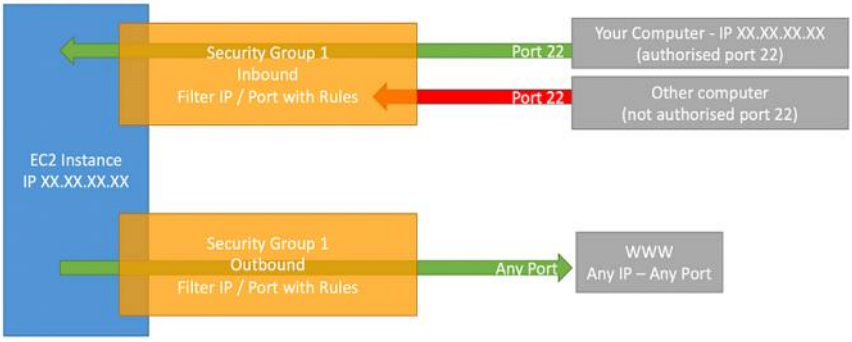
1.2 EC2 User Data
- It is possible to bootstrap our instances using an **EC2 User data** script.
- Bootstrapping means launching commands when a machine starts.
- That script is **only run once** at the instance **first start**.
- EC2 **user data** is used to **automate boot tasks** such as *installing updates, installing software, downloading common files from the internet, anything you can think of*
- The EC2 **User Data Script runs with the root user** — any command you have will have the `sudo` right.

2. Security Groups & Classic Ports

Security groups are fundamental to network security in AWS. They control **how traffic is allowed into or out of our EC2**

Instances. They are acting as a “firewall” on EC2 instances. Specifically, they regulate

- Access to Ports
- Authorised IP ranges — IPv4 and IPv6
- Control of inbound network (from other to the instance)
- Control of outbound network (from the instance to other)



Security groups

- only contain **allow** rules. Rules can reference **by IP or by security groups.**
- Can be attached to multiple EC2 instances
- Locked down to a region / VPC (Virtual Private Cloud) combination
- Does live “outside” the EC2 — if traffic is blocked, the EC2 instance won’t see it
- **It’s good to maintain one separate security group for SSH access**
- If your application is not accessible (time out) -> security group issue
- If your application gives a “connection refused“ error -> application error or it’s not launched
- All inbound traffic is **blocked** by default. All outbound traffic is **authorized** by default
- EC2 instance role: link to IAM roles

2.1 Classic Ports to know

- **22 = SSH (Secure Shell)** — log into a **Linux** instance
- 21 = FTP (File Transfer Protocol) — upload files into a file share
- 22 = SFTP (Secure File Transfer Protocol) — upload files using SSH
- 80 = HTTP — access unsecured websites
- 443 = HTTPS — access secured websites
- **3389 = RDP** (Remote Desktop Protocol) — log into a **Windows** instance

Use Cases of EC2

- Building and deploying apps to the cloud.
- Trying out a new operating system (OS), including beta releases.
- Spinning up a new environment to make it simpler and quicker for developers to run dev-test scenarios.
- Backing up your existing OS.
- Accessing virus-infected data or running an old application by installing an older OS.
- Running software or apps on operating systems that they weren't originally intended for.

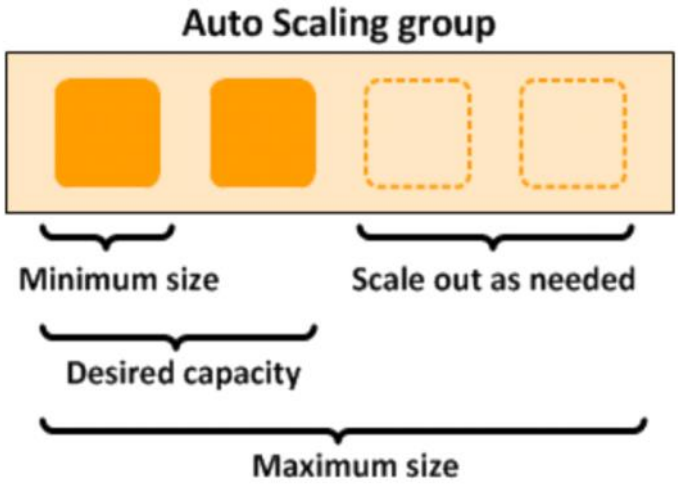
Auto Scaling in AWS

In Tradition IT World, there are limited number of servers to handle the application load. when the ***number of requests increases load on the server also increases***, which cause failures and latency in system.

Amazon web service provide **Amazon EC2 auto scaling services to overcome this failure**. Auto Scaling ensures that Amazon EC2 instances are sufficient to run our application.

What is Amazon EC2 Auto Scaling ? — *helps you to ensure that we have the correct number of ec2 instances available to handle the load of you application.*

1. ***Collection of EC2 instances*** *called Auto Scaling Groups.*
2. *Specify the* ***minimum number of instances*** *in each auto scaling groups and auto scaling groups ensures that* ***never goes below the size.***
3. *Specify the* ***maximum number of instance*** *in each auto scaling groups and auto scaling groups ensures that* ***never goes above the size.***
4. *Specify the* ***desired capacity*** *either when create or after any time — ensures that group has this many instances.*
5. *Specify the* ***Scaling Policies- Amazon EC2 Auto scaling can launch or terminate instances as demand on our application increases or decreases.***

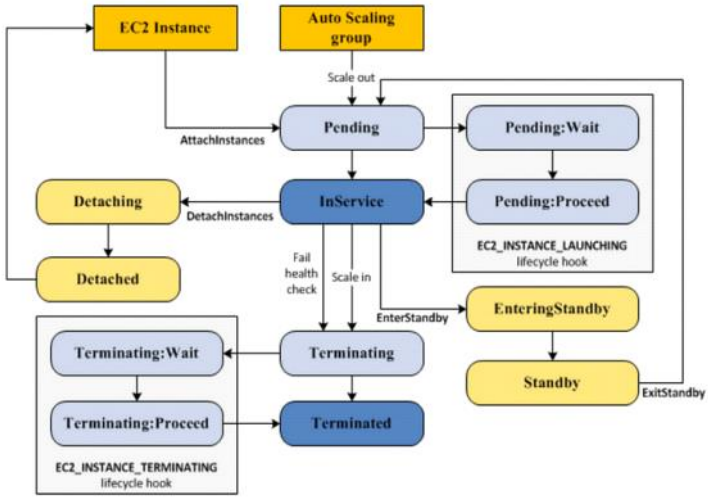


Auto Scaling Components:

1. **Groups** — Each EC2 instances are organized into groups so that they can be treated as logical units for the purpose of scaling and management.(When you create a group we can specify minimum, maximum, desired number of EC2 instances).
2. **Configuration Templates-** Our Group uses launch configuration or launch template for its ec2 instances,we can specify information such as AMI ID, instance type, security groups , block device mapping…
3. **Scaling Options —** Amazon EC2 Auto scaling provides several ways to scale our auto scaling groups. we can configure scale based on occurrence of specified conditions (dynamic scaling) or on a schedule.

Auto Scaling Life Cycle :- EC2 Instance in Auto Scaling Group has a **path or life cycle (that differ from other ec2 instances)**

1. Life cycle **starts when Auto Scaling group launches an instance** and puts into service.
2. Life cycle ends when **Auto Scaling group terminate the instance, or Auto Scaling group takes the instance out of service and terminates it.**



life cycle of auto scaling group ec2 instance

Let see detail about each state whats happening in life cycle.

1. **Scale Out: —** Directs the Auto Scaling group to **launch EC2 instance and attach then to group.**

How to attach ec2 instance to group ?

1. Manually increase the size of group.
2. create a scaling policy to automatically increase the size of group based on increase in demand.
3. set up scaling by schedule to increase the size of group.

When a scale out event occurs, the auto scaling group launches the required number (one or more) of ec2 instances, using it launch configuration.

These instances start in **PENDING STATE** , if we added life cycle hook we can perform custom action, when each instance is fully configured and passes EC2 health check ,it is attached to Auto Scaling group and it enters the **In Service State.**(The instance is counted against the desired capacity of the auto scaling group).

2. **Instance In Service :-** Instance remains In service state until one of following state occurs.

a. Scale In event occurs and Amazon Auto Scaling chooses to terminate the instance in order to reduce the size of the Auto Scaling group.

b. put the instance into a **Stand By state.**

c. Detach the instance from Auto scaling group.

d. Instance fails a required number of health checks, so it is removed from the auto scaling group.

3. **Scale In: —** Directs the Auto Scaling group to **detach EC2 instance from the group and terminate them.**

How to detach ec2 instance to group ?

1. Manually decrease the size of group.
2. create a scaling policy to automatically decrease the size of group based on decrease in demand.
3. set up scaling by schedule to decrease the size of group.

It is important that we create a corresponding scale in event for each scale out event that you create.

when a scale in event occurs, the Auto Scaling group detaches one or more instances. The Auto Scaling group uses its termination policy to determine which instances to terminate.Instance that are in process of detaching from auto scaling group and shutting down enter the **TERMINATING STATE** and cant put back into the service.if we added life cycle hook we can perform custom action, Finally instances are completely terminated and enter the **TERMINATED STATE.**

4. **Attach an Instance —**we can attach a running EC2 instance that meets certain criteria to our Auto Scaling group. After the instance is attached, it is managed as part of the Auto Scaling group.

5. **Detach an Instance —** we can detach an instance from your Auto Scaling group. After the instance is detached, you can manage it separately from the Auto Scaling group or attach it to a different Auto Scaling group.

6. **Life cycle Hooks —** we can add life cycle hook to our auto scaling group so that we can perform custom actions when instances launch or terminate.

a. **Scale Out** : (it launches one or more instances) -instances start in the **PENDING STATE** if we added “autoscaling:EC2 _INSTANCE_LAUNCHING” hook to our auto scaling group, instance move from **PENDING WAIT STATE,** after completion of life cycle action, the instances enter the **PENDING: PROCEED state.**When the instances are fully configured, they are attached to the Auto Scaling group and they enter the Inservice state.

PENDING -> {PENDING WAIT & PENDING PROCEED STATE} lifecycle action added -> INSERVICE STATE.

b. **Scale In** : (it detaches one or more instances) -instances are detached in **TERMINATING STATE** , if we added “autoscaling:EC2 _INSTANCE_TERMINATING” hook to our auto scaling group, instance move from **TERMINATING WAIT STATE,** after completion of life cycle action, the instances enter the **TERMINATING: PROCEED state.**When the instances are fully terminated, they are detached from Auto Scaling group and they enter the **Terminated** state.

TERMINATING -> {TERMINATING WAIT & TERMINATING PROCEED STATE }life cycle action added -> TERMINATED .

7. **Enter and Exit Stand By :-** put any instance that is in an **INSERVICE** state into a **STANDBY** state. This enables you to remove the instance from service, troubleshoot or make changes to it, and then **put it back into service.**

Instances in a **STANDBY** state continue to be managed by the Auto Scaling group. However, they are **not an active part of your application until you put them back into service.**

Auto Scaling Limits:

Default Limits

- Launch configurations per Region: 200
- Auto Scaling groups per Region: 200

Auto Scaling Group Limits

- Scaling policies per Auto Scaling group: 50
- Scheduled actions per Auto Scaling group: 125
- Lifecycle hooks per Auto Scaling group: 50
- SNS topics per Auto Scaling group: 10
- Classic Load Balancers per Auto Scaling group: 50
- Target groups per Auto Scaling group: 50

Scaling Policy Limits

- Step adjustments per scaling policy: 20

Ec2

Auto Scaling

ECS

Load Balancers

WAF - > CloudFront

S3.

Elastic IP addresses

An *Elastic IP address* is a static IPv4 address designed for dynamic cloud computing. An Elastic IP address is allocated to your AWS account, and is yours until you release it. By using an Elastic IP address, you can mask the failure of an instance or software by rapidly remapping the address to another instance in your account. Alternatively, you can specify the Elastic IP address in a DNS record for your domain, so that your domain points to your instance. For more information, see the documentation for your domain registrar, or Set up dynamic DNS on your Amazon Linux instance.
An Elastic IP address is a public IPv4 address, which is reachable from the internet. If your instance does not have a public IPv4 address, you can associate an Elastic IP address with your instance to enable communication with the internet. For example, this allows you to connect to your instance from your local computer.

Elastic IP address basics

The following are the basic characteristics of an Elastic IP address:

- An Elastic IP address is static; it does not change over time.
- An Elastic IP address is for use in a specific Region only, and cannot be moved to a different Region.
- An Elastic IP address comes from Amazon's pool of IPv4 addresses, or from a custom IPv4 address pool that you have brought to your AWS account.
- To use an Elastic IP address, you first allocate one to your account, and then associate it with your instance or a network interface.
- When you associate an Elastic IP address with an instance, it is also associated with the instance's primary network interface. When you associate an Elastic IP address with a network interface that is attached to an instance, it is also associated with the instance.
- When you associate an Elastic IP address with an instance or its primary network interface, the instance's public IPv4 address (if it had one) is released back into Amazon's pool of public IPv4 addresses. You cannot reuse a public IPv4 address, and you cannot convert a public IPv4 address to an Elastic IP address. For more information, see Public IPv4 addresses.
- You can disassociate an Elastic IP address from a resource, and then associate it with a different resource. To avoid unexpected behavior, ensure that all active connections to the resource named in the existing association are closed before you make the change. After you have associated your Elastic IP address to a different resource, you can reopen your connections to the newly associated resource.
- A disassociated Elastic IP address remains allocated to your account until you explicitly release it. We impose a small hourly charge for Elastic IP addresses that are not associated with a running instance.
- When you associate an Elastic IP address with an instance that previously had a public IPv4 address, the public DNS host name of the instance changes to match the Elastic IP address.
- We resolve a public DNS host name to the public IPv4 address or the Elastic IP address of the instance outside the network of the instance, and to the private IPv4 address of the instance from within the network of the instance.
- When you allocate an Elastic IP address from an IP address pool that you have brought to your AWS account, it does not count toward your Elastic IP address limits. For more information, see Elastic IP address limit.
- When you allocate the Elastic IP addresses, you can associate the Elastic IP addresses with a network border group. This is the location from which we advertise the CIDR block. Setting the network border group limits the CIDR block to this group. If you do not specify the network border group, we set the border group containing all of the Availability Zones in the Region (for example, us-west-2).
- An Elastic IP address is for use in a specific network border group only.

From <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/elastic-ip-addresses-eip.html>