Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides resizable compute capacity in the cloud. Amazon EC2 reduces the time required to obtain and boot the new server instances to minutes, allowing you to quickly scale capacity, both up and down, as your computing requirements change.

# EC2 Pricing

Free tier: AWS Free Tier includes 750 hours of Linux and Windows t2.micro instances each month for one year.

More info on free tier: https://aws.amazon.com/free/

On-Demand:With On-Demand instances, you pay for compute capacity by the hour or the second depending on which instances you run. No longer-term commitments or upfront payments are needed.

On-Demand instances are recommended for:

* Users that prefer the low cost and flexibility of Amazon EC2 without any up-front payment or long-term commitment
* Applications with short-term, spiky, or unpredictable workloads that cannot be interrupted
* Applications being developed or tested on Amazon EC2 for the first time

Spot instances**:** Amazon EC2 Spot instances allow you to request spare Amazon EC2 computing capacity for up to 90% off the On-Demand price.

Spot instances are recommended for:

* Applications that have flexible start and end times
* Applications that are only feasible at very low compute prices
* Users with urgent computing needs for large amounts of additional capacity

**Note:** If the Spot instance is terminated by Amazon EC2, you will not be charged for a partial hour of usage. However, if you terminate the instance yourself, you will be charged for any hour in which the instance ran.

Reserved instances**:** Reserved Instances provide you with a significant discount (up to 75%) compared to On-Demand instance pricing.

Reserved Instances are recommended for:

* Applications with steady state usage
* Applications that may require reserved capacity
* Customers that can commit to using EC2 over a 1 or 3 year term to reduce their total computing costs

Savings Plans**:** Savings Plans are a flexible pricing model that offer low prices on EC2 and Fargate usage, in exchange for a commitment to a consistent amount of usage (measured in $/hour) for a 1 or 3 year term.

**Dedicated Hosts:** A Dedicated Host is a physical EC2 server dedicated for your use. Dedicated Hosts can help you reduce costs by allowing you to use your existing server-bound software licenses, including Windows Server, SQL Server, and SUSE Linux Enterprise Server (subject to your license terms), and can also help you meet compliance requirements.

# EC2 Instance Types

## General Purpose:

General purpose instances provide a balance of compute, memory and networking resources, and can be used for a variety of diverse workloads. These instances are ideal for applications that use these resources in equal proportions such as web servers and code repositories.

## Compute Optimized:

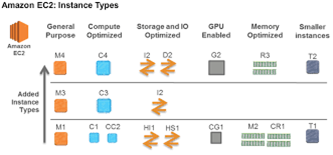
Compute Optimized instances are ideal for compute bound applications that benefit from high performance processors. Instances belonging to this family are well suited for batch processing workloads, media transcoding, high performance web servers, high performance computing (HPC), scientific modeling, dedicated gaming servers and ad server engines, machine learning inference and other compute intensive applications.

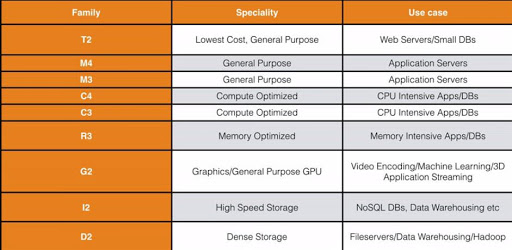
## Memory Optimized:

Memory optimized instances are designed to deliver fast performance for workloads that process large data sets in memory.

## Storage Optimized:

Storage optimized instances are designed for workloads that require high, sequential read and write access to very large data sets on local storage. They are optimized to deliver tens of thousands of low-latency, random I/O operations per second (IOPS) to applications.





# EBS(Elastic Block Storage):

Amazon Elastic Block Store (EBS) is an easy to use, high performance block storage service designed for use with Amazon Elastic Compute Cloud (EC2) for both throughput and transaction intensive workloads at any scale. A broad range of workloads, such as relational and non-relational databases, enterprise applications, containerized applications, big data analytics engines, file systems, and media workflows are widely deployed on Amazon EBS.

## EBS Volume Types:

**General Purpose SSD(GP2):** gp2 is the default EBS volume type for Amazon EC2 instances. These volumes are backed by solid-state drives (SSDs) and are suitable for a broad range of transactional workloads, including dev/test environments, low-latency interactive applications, and boot volumes. gp2 is designed to offer single-digit millisecond latency, deliver a consistent baseline performance of 3 IOPS/GB (minimum 100 IOPS) to a maximum of 16,000 IOPS, and provide up to 250 MB/s of throughput per volume.

**Volume Type**: EBS General Purpose SSD (gp2) \*  
**Short Description**: General Purpose SSD volume that balances price performance for a wide variety of transactional workloads  
**Use Cases**: Boot volumes, low-latency interactive apps, dev & test  
**API Name**: gp2  
**Volume Size**: 1 GB – 16 TB  
**Max IOPS\*\*/Volume**: 16,000  
**Max Throughput\*\*\*/Volume**: 250 MB/s  
**Max IOPS/Instance**: 80,000  
**Max Throughput/Instance**: 2,375 MB/s  
**Price**: $0.10/GB-month

**Provisioned IOPS SSD(IO1):** io1 is backed by solid-state drives (SSDs) and is a high performance EBS storage option designed for critical, I/O intensive database and application workloads, as well as throughput-intensive database and data warehouse workloads, such as HBase, Vertica, and Cassandra. These volumes are ideal for both IOPS-intensive and throughput-intensive workloads that require low latency and have moderate durability requirements or include built-in application redundancy.

io1 is designed to deliver a consistent baseline performance of up to 50 IOPS/GB to a maximum of 64,000 IOPS and provide up to 1,000 MB/s of throughput per volume. To maximize the benefit of io1, we recommend using EBS-optimized EC2 instances. When attached to EBS-optimized EC2 instances, io1 is designed to achieve single-digit millisecond latencies and is designed to deliver the provisioned performance 99.9% of the time.

**Volume Type**: EBS Provisioned IOPS SSD (io1)  
**Short Description**: High performance SSD volume designed for latency-sensitive transactional workloads  
**Use Cases**: I/O-intensive NoSQL & relational databases  
**API Name**: io1  
**Volume Size**: 4 GB – 16 TB  
**Max IOPS\*/Volume**: 64,000  
**Max Throughput\*\*/Volume**: 1,000 MB/s  
**Max IOPS/Instance**: 80,000  
**Max Throughput/Instance**: 2,375 MB/s  
**Price**: $0.125/GB-month + $0.065/provisioned IOPS

**Throughput Optimized HDD(ST1):** st1 is backed by hard disk drives (HDDs) and is ideal for frequently accessed, throughput-intensive workloads with large datasets and large I/O sizes, such as MapReduce, Kafka, log processing, data warehouse, and ETL workloads. These volumes deliver performance, measured in MB/s of throughput, and include the ability to burst up to 250 MB/s per TB, with a baseline throughput of 40 MB/s per TB and a maximum throughput of 500 MB/s per volume.

**Volume Type**: Throughput Optimized HDD (st1)  
**Short Description**: Low cost HDD volume designed for frequently accessed, throughput-intensive workloads  
**Use Cases**: Big data, data warehouses, log processing  
**API Name**: st1  
**Volume Size:** 500 GB – 16 TB  
**Max IOPS\*\*/Volume**: 500  
**Max Throughput\*\*\*/Volume**: 500 MB/s  
**Max IOPS/Instance**: 80,000  
**Max Throughput/Instance**: 2,375 MB/s  
**Price**: $0.045/GB-month

**Cold HDD(SC1):** sc1 is backed by hard disk drives (HDDs) and provides the lowest cost per GB of all EBS volume types. It is ideal for less frequently accessed workloads with large, cold datasets. Similar to st1, sc1 provides a burst model. These volumes can burst up to 80 MB/s per TB, with a baseline throughput of 12 MB/s per TB and a maximum throughput of 250 MB/s per volume. For infrequently accessed data, sc1 provides extremely inexpensive storage.

**Volume Type**: Cold HDD (sc1)  
**Short Description**: Lowest cost HDD volume designed for less frequently accessed workloads  
**Use Cases**: Colder data requiring fewer scans per day  
**API Name**: sc1  
**Volume Size**: 500 GB – 16 TB  
**Max IOPS\*\*/Volume:** 250  
**Max Throughput\*\*\*/Volume**: 250 MB/s  
**Max IOPS/Instance**: 80,000  
**Max Throughput/Instance**: 2,375 MB/s  
**Price:** $0.025/GB-month

**Magnetic (Standard):** Lowest cost per gigabyte of all EBS volume types that is bootable. Magnetic volumes are ideal for workloads where data is accessed infrequently, and applications where the lowest storage cost is important.

## EC2 AMI Types:

* **EBS**
* **Instance Store**

EBS Vs Instance Store**:**

* All AMIs are categorized as either backed by Amazon EBS or backed by instance store
* For EBS Volumes: The root device for an instance launched from the AMI is an Amazon EBS volume created from an Amazon EBS snapshot
* Instance store volumes are sometimes called Ephemeral Storage
* Instance store volumes cannot be stopped. If the underlying host fails, you will lose your data.
* EBS backed instance can be stopped. You will not lose the data on this instance if it is stopped
* You can reboot both, you will not lose your data
* By default, both ROOT volumes will be deleted on termination, however with EBS volumes, you can tell AWS to keep the root device volume.

### Encrypted Root Volumes:

* Snapshots of encrypted volumes are encrypted automatically
* Volumes restored from encrypted snapshots are encrypted automatically
* You can share snapshots, but only if they are unencrypted
* These snapshots can be shared with other AWS accounts
* You can now encrypt root device volumes upon creation of the EC2 instance

Notes:

* Snapshots exists on S3. Snapshots are incremental – means that only the blocks that have changed since your last snapshot are moved to S3.
* By default, root volumes are terminated if the instance is terminated. However, the additional volumes are not terminated by default. We need to terminate them additionally.
* By default, EBS volumes are unencrypted.
* To encrypt the EBS volumes that are unencrypted, create a snapshot of the volume, then copy the snapshot and select the encryption option in the copy snapshot window. Then, create an image from the encrypted snapshot. Launch the EC2 instance from the image that we have created.
* To move an EC2 volume from one AZ to another, take a snapshot of it, create an AMI from the snapshot and then use the AMI to launch the EC2 instance in a new AZ.
* To move an EC2 volume from one region to another, take a snapshot of it, create an AMI from the snapshot and then copy the AMI from one region to the other. Then use the AMI to launch the EC2 instance in a new region
* You can scale EBS volumes up only i.e you cannot decrease the size of the EBS volumes
* Volumes must be in the same AZ as the EC2 instances

**RAID(Redundant Array of Independent Disks):**

* **RAID 0 –** Striped, No Redundancy, Good Performance
* **RAID 1 –** Mirrored, Redundancy
* **RAID 5 –** Good for reads, bad for writes, **AWS doesn’t recommend ever putting RAID 5’s on EBS**
* **RAID 10 –** Striped & Mirrored, good redundancy, Good performance

# Security Groups

* Security groups is a virtual Firewall.
* All inbound traffic is blocked by default
* All outbound traffic is allowed
* Changes to security groups take effect immediately
* You can have any number of EC2 instances within a Security group
* You can have multiple security groups attached to EC2 instances
* Security groups are **STATEFUL**

1. If you create an inbound rule allowing traffic in, that traffic is automatically allowed back out again.

* You cannot block specific IP addresses using Security Groups, instead use Network Access Control Lists.
* You can specify allow rules, but not deny rules.

# Auto Scaling

Amazon EC2 Auto Scaling helps you ensure that you have the correct number of Amazon EC2 instances available to handle the load for your application. You create collections of EC2 instances, called Auto Scaling groups. You can specify the minimum number of instances in each Auto Scaling group, and Amazon EC2 Auto Scaling ensures that your group never goes below this size. You can specify the maximum number of instances in each Auto Scaling group, and Amazon EC2 Auto Scaling ensures that your group never goes above this size.

The EC2 instances in an Auto Scaling group have a path, or lifecycle, that differs from that of other EC2 instances. The lifecycle starts when the Auto Scaling group launches an instance and puts it into service. The lifecycle ends when you terminate the instance, or the Auto Scaling group takes the instance out of service and terminates it.

The following illustration shows the transitions between instance states in the Amazon EC2 Auto Scaling lifecycle.


    The lifecycle of instances within an Auto Scaling group.
   

## Scale Out:

The following scale out events direct the Auto Scaling group to launch EC2 instances and attach them to the group:

* You manually increase the size of the group.
* You create a scaling policy to automatically increase the size of the group based on a specified increase in demand.
* You set up scaling by schedule to increase the size of the group at a specific time

When a scale out event occurs, the Auto Scaling group launches the required number of EC2 instances, using its assigned launch configuration. These instances start in the Pending state. If you add a lifecycle hook to your Auto Scaling group, you can perform a custom action here.

When each instance is fully configured and passes the Amazon EC2 health checks, it is attached to the Auto Scaling group and it enters the InService state. The instance is counted against the desired capacity of the Auto Scaling group.

## Scale In:

The following scale in events direct the Auto Scaling group to detach EC2 instances from the group and terminate them:

* You manually decrease the size of the group
* You create a scaling policy to automatically decrease the size of the group based on a specified decrease in demand.
* You set up scaling by schedule to decrease the size of the group at a specific time.

It is important that you create a corresponding scale in event for each scale out event that you create. This helps ensure that the resources assigned to your application match the demand for those resources as closely as possible.

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. Instances that are in the process of detaching from the Auto Scaling group and shutting down enter the Terminating state, and can't be put back into service. If you add a lifecycle hook to your Auto Scaling group, you can perform a custom action here. Finally, the instances are completely terminated and enter the Terminated state.

## Launch Configurations:

A launch configuration is an instance configuration template that an Auto Scaling group uses to launch EC2 instances. When you create a launch configuration, you specify information for the instances. Include the ID of the Amazon Machine Image (AMI), the instance type, a key pair, one or more security groups, and a block device mapping. If you've launched an EC2 instance before, you specified the same information in order to launch the instance.

You can specify your launch configuration with multiple Auto Scaling groups. However, you can only specify one launch configuration for an Auto Scaling group at a time, and you can't modify a launch configuration after you've created it. To change the launch configuration for an Auto Scaling group, you must create a launch configuration and then update your Auto Scaling group with it.

## Auto Scaling Groups:

An Auto Scaling group contains a collection of Amazon EC2 instances that are treated as a logical grouping for the purposes of automatic scaling and management. An Auto Scaling group also enables you to use Amazon EC2 Auto Scaling features such as health check replacements and scaling policies. Both maintaining the number of instances in an Auto Scaling group and automatic scaling are the core functionality of the Amazon EC2 Auto Scaling service.

The size of an Auto Scaling group depends on the number of instances that you set as the desired capacity. You can adjust its size to meet demand, either manually or by using automatic scaling.

# Elastic Load Balancer

Elastic Load Balancing distributes incoming application or network traffic across multiple targets, such as Amazon EC2 instances, containers, and IP addresses, in multiple Availability Zones. Elastic Load Balancing scales your load balancer as traffic to your application changes over time. It can automatically scale to the vast majority of workloads.

Elastic Load Balancing supports three types of load balancers:

* Application Load Balancers
* Network Load Balancers
* Classic Load Balancers

There is a key difference in how the load balancer types are configured. With Application Load Balancers and Network Load Balancers, you register targets in target groups, and route traffic to the target groups. With Classic Load Balancers, you register instances with the load balancer.

**Application Load Balancer Components**

A load balancer serves as the single point of contact for clients. The load balancer distributes incoming application traffic across multiple targets, such as EC2 instances, in multiple Availability Zones. This increases the availability of your application. You add one or more listeners to your load balancer.

A listener checks for connection requests from clients, using the protocol and port that you configure. The rules that you define for a listener determine how the load balancer routes requests to its registered targets. Each rule consists of a priority, one or more actions, and one or more conditions. When the conditions for a rule are met, then its actions are performed. You must define a default rule for each listener, and you can optionally define additional rules.

Each target group routes requests to one or more registered targets, such as EC2 instances, using the protocol and port number that you specify. You can register a target with multiple target groups. You can configure health checks on a per target group basis. Health checks are performed on all targets registered to a target group that is specified in a listener rule for your load balancer.

Application Load Balancer Overview

An Application Load Balancer functions at the application layer, the seventh layer of the Open Systems Interconnection (OSI) model. After the load balancer receives a request, it evaluates the listener rules in priority order to determine which rule to apply, and then selects a target from the target group for the rule action.

**Network Load Balancer Components**

A load balancer serves as the single point of contact for clients. The load balancer distributes incoming traffic across multiple targets, such as Amazon EC2 instances. This increases the availability of your application. You add one or more listeners to your load balancer.

A listener checks for connection requests from clients, using the protocol and port that you configure, and forwards requests to a target group.

Each target group routes requests to one or more registered targets, such as EC2 instances, using the TCP protocol and the port number that you specify. You can register a target with multiple target groups. You can configure health checks on a per target group basis. Health checks are performed on all targets registered to a target group that is specified in a listener rule for your load balancer.

**Network Load Balancer Overview**

A Network Load Balancer functions at the fourth layer of the Open Systems Interconnection (OSI) model. It can handle millions of requests per second. After the load balancer receives a connection request, it selects a target from the target group for the default rule. It attempts to open a TCP connection to the selected target on the port specified in the listener configuration.

# ENI vs ENA vs EFA

An Elastic Network Interface(ENI) is simply a virtual network card for your EC2 instances. It is used for basic networking. Perhaps you need a separate management network to your production network or a separate logging network and you need to do this at low cost. In this scenario use multiple ENIs for each network

An Enhances Networking(ENA) uses single root I/O virtualization(SR-IOV) to provide high-performance networking capabilities on supported instance types. SR-IOV is a method of device virtualization that provides higher I/O performance and lower CPU utilization when compared to traditional virtualized network interfaces.It is used when you need speeds betweek 10Gbps and 100Gbps. Anywhere you need reliable, high throughput.

An Elastic Fabric Adapter(EFA) is a network device that you can attach to your Amazon EC2 instance to accelerate High Performance Computing(HPC) and machine learning applications. EFA provides lower and more consistent latency and higher throughput than the TCP transport traditionally used in cloud-based HPC systems.

**Placement Groups:**

When we create placement group, we specify on of the following strategies for the group:

**Cluster:** clusters instances into a low latency group in a single Availability Zone. All the instances are placed in same rack. It is great for network as it provides 10Gbps bandwidth between instances. But, if rack fails, all the instances fails at the same time.

**Spread:** spreads instances across underlaying hardware (max 7 instances per group per AZ). Can span across Availability Zones i.e each instance is placed across the Availability Zones on separate Hardware. Used for critical applications, and applications that needs high availability. The limitation is 7 instances per AZ per placement group.

**Partition:** spreads instances across many different partitions( which rely on different set of racks) within an AZ. Scales to 100s of EC2 instances per group(Hadoop, Casandra, Kafka). Can use upto 7 partitions per AZ. The instances in a partition do not share racks with the instances in the other partitions. A partition failure can affect many EC2 but won't affect other partitions. EC2 instances get access to the partition information as metadata.

**Cloud Watch:**

Amazon CloudWatch is a monitoring and observability service built for DevOps engineers, developers, site reliability engineers (SREs), and IT managers. CloudWatch provides you with data and actionable insights to monitor your applications, respond to system-wide performance changes, optimize resource utilization, and get a unified view of operational health. CloudWatch collects monitoring and operational data in the form of logs, metrics, and events, providing you with a unified view of AWS resources, applications, and services that run on AWS and on-premises servers.

Benefits of CloudWatch:

* **Observability on a single platform across applications and infrastructure** - Modern applications such as those running on microservices architectures generate large volumes of data in the form of metrics, logs, and events. Amazon CloudWatch enables you to collect, access, and correlate this data on a single platform from across all your AWS resources, applications, and services that run on AWS and on-premises servers, helping you break down data silos so you can easily gain system-wide visibility and quickly resolve issues.
* **Easiest way to collect metrics in AWS and on-premises** - Monitoring your AWS resources and applications is easy with CloudWatch. It natively integrates with more than 70 AWS services such as Amazon EC2, Amazon DynamoDB, Amazon S3, Amazon ECS, Amazon EKS, and AWS Lambda, and automatically publishes detailed 1-minute metrics and custom metrics with up to 1-second granularity so you can dive deep into your logs for additional context. You can also use CloudWatch in hybrid cloud architectures by using the CloudWatch Agent or API to monitor your on-premises resources.
* **Improve operational performance and resource optimization** - Amazon CloudWatch enables you to set alarms and automate actions based on either predefined thresholds, or on machine learning algorithms that identify anomalous behavior in your metrics. For example, it can start Amazon EC2 Auto Scaling automatically, or stop an instance to reduce billing overages. You can also use CloudWatch Events for serverless to trigger workflows with services like AWS Lambda, Amazon SNS, and AWS CloudFormation.
* **Get operational visibility and insight** - To optimize performance and resource utilization, you need a unified operational view, real-time granular data, and historical reference. CloudWatch provides automatic dashboards, data with 1-second granularity, and up to 15 months of metrics storage and retention. You can also perform metric math on your data to derive operational and utilization insights; for example, you can aggregate usage across an entire fleet of EC2 instances.
* **Derive actionable insights from logs** - CloudWatch enables you to explore, analyze, and visualize your logs so you can troubleshoot operational problems with ease. With CloudWatch Logs Insights, you only pay for the queries you run. It scales with your log volume and query complexity giving you answers in seconds. In addition, you can publish log-based metrics, create alarms, and correlate logs and metrics together in CloudWatch Dashboards for complete operational visibility.

