EC2-Elastic Compute Cloud

* Amazon EC2 is a web service that provides resizable compute capacity in the cloud. It reduces the time required to obtain and boot new server instances to minutes, allowing scaling (both up and down).

EC2 Options

* On Demand – allow you to pay a fixed rate by the hour with no commitment.
* Users that want low cost and flexibility of amazon EC2 **without any upfront 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
* Reserved – provides a capacity reservation and offers a significant discount on the hourly charge for an instance (1 or 3 year contract)
* Application with steady state or predictable usage
* Application that require reserved capacity
* Users able to make upfront payments to reduce their total computing costs even further
* Spot- enables you to bid whatever price you want for instance capacity. To be used when your application has flexible start and end times.
* Applications that have flexible start and end times.
* Applications that are feasible only at very low compute prices
* Pharmaceutical companies use them usually

**Note**: If Spot instances are terminated by Amazon EC2 service, you will not be charged for a partial hour(this happens when spot prices goes higher than your bid price).

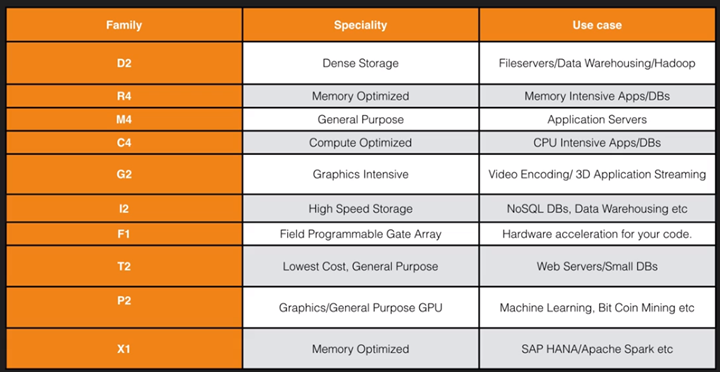
However if you terminate the instance by yourself you are charged for partial hour as complete hour.

* Dedicated hosts – Physical EC2 server dedicated for your use. Dedicated Hosts can help reduce costs by allowing you to use your existing server-bound software licenses.
* Useful for regulatory requirements that may not support multi-tenant virtualization. Eg- govt bodies, banks etc
* Good for licensing which does not support multi-tenancy or cloud deployments

Features of EC2

* Amazon EC2 provides the following features:
* Virtual computing environments, known as **instances**
* Preconfigured templates for your instances, known as**AmazonMachineImages(AMIs)**, that package the bits you need for your server (including the operating system and additional software)
* Various configurations of CPU, memory, storage, and networking capacity for your instances, known as **instance types**
* A related service for managing numbers of instances running for a purpose called **Auto Scaling.**
* 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 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)**

EC2 Instance Types



\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*to be done from docs\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

EBS-Elastic Block Storage

* Amazon EBS allows creating storage volumes and attaching them to EC2 instances. After attaching we can create file system, format, partition, run a DB or anything we would do with a block storage.
* EBS volumes are placed in an AZ and are automatically replicated there (in the same AZ ,so failure of a complete AZ can be serious in terms of EBS)

EBS Types

* Amazon EBS provides the following volume types, which differ in performance characteristics and price, so that you can tailor your storage performance and cost to the needs of your applications. The volumes types fall into two categories:
* SSD-backed volumes optimized for transactional workloads involving frequent read/write operations with small I/O size, where the dominant performance attribute is IOPS
* HDD-backed volumes optimized for large streaming workloads where throughput (measured in MiB/s) is a better performance measure than IOPS
* Solid State Drives:
* General Purpose SSD- GP2 (2 is the generation)

1. Default type for creating EBS volume from console
2. Balances both price and performance
3. System Boot Volumes
4. 1GB-16TB
5. 3 IOPS per GB. Max IOPS 10,000 per GB

* Provisioned IOPS SSD- IO1

1. High performance SSD for critical low latency
2. I/O intensive apps likeLarge NoSql or SQL DBs
3. Bootable
4. 4GB – 16 TB
5. Max IOPS 32000

* Hard Disk Drives:
* Throughput Optimized HDD - ST1

1. Low cost HDD
2. Big Data, Data warehousing, Log Processing
3. For storing data in sequential manner
4. Cannot be a Boot Volume
5. 500 GB – 16TB
6. 500 IOPs / GB

* Cold HDD – SC1

1. Lowest Cost HDD designed for less frequently accessed workloads
2. Cannot be a boot volume
3. 500GB – 16TB
4. IOPS – 150/GB

* Previous Generation Drives
* EBS Magnetic
* 40-200 IOPS/GB
* Can be a boot volume
* Just a cold HDD that is bootable
* NOTES:
* Burst Bucket Behaviour - For GP2 I/O credits are stored and when available can be used at a rate of 3000 IOPS/GB for 30 minutes bursts. Burst bucket behavior is different for different volume types.
* Linux AMIs require GPT partition tables and GRUB 2 for boot volumes 2 TiB (2048 GiB) or larger. Many Linux AMIs today use the MBR partitioning scheme, which only supports up to 2047 GiB boot volumes. If your instance does not boot with a boot volume that is 2 TiB or larger, the AMI you are using may be limited to a 2047 GiB boot volume size. Non-boot volumes do not have this limitation on Linux instances.
* AMI - Amazon Machine Images is a template that contains a software configuration( eg : OS+ app server+ other apps )
* EC2 Root Device Volume – It contains the image used to boot the instance.
* Initially all AMIs were backed by Amazon EC2 instance store. This means that a root volume is created from a volume template (the template is like a snapshot of a storage containing the AMI) stored in S3.
* After EBS was introduced – AMIs were backed by EBS volumes called root volumes. This means that root device for an instance is launched from the AMI is an EBS volume created from an EBS snapshot.
* We can choose AMIs backed my EBS or AMIs backed my instance store.
* Instance Store Backed Instances – any instance on launch has atleast one volume(root volume) attached to it(additional storage volumes may be attached).On launch the AMI is copied from instance store into this root volume (when EBS was not there ephemeral storage was used, its property was that it was volatile). Any data on the instance persists as long instance is running, but is deleted when the instance is terminated (instance store backed instances have no STOP state, only RUNNING and TERMINATED).
* Amazon EBS backed instances - Instances that use Amazon EBS for the root device automatically have an Amazon EBS volume attached. When you launch an Amazon EBS-backed instance, we create an Amazon EBS volume for each Amazon EBS snapshot referenced by the AMI you use. You can optionally use other Amazon EBS volumes or instance store volumes, depending on the instance type. An Amazon EBS-backed instance can be stopped and later restarted without affecting data stored in the attached volumes. There are various instance– and volume-related tasks you can do when an Amazon EBS-backed instance is in a stopped state. For example, you can modify the properties of the instance, you can change the size of your instance or update the kernel it is using, or you can attach your root volume to a different running instance for debugging or any other purpose.
* If an Amazon EBS-backed instance fails, you can restore your session by following one of these methods:

1. Stop and then start again (try this method first).
2. Automatically snapshot all relevant volumes and create a new AMI.
3. Attach the volume to the new instance by following these steps:

* Create a snapshot of the root volume.
* Register a new AMI using the snapshot.
* Launch a new instance from the new AMI.
* Detach the remaining Amazon EBS volumes from the old instance.
* Reattach the Amazon EBS volumes to the new instance.
* COMPARISON OF INSTANCE STORE & EBS STORE AMI: <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ComponentsAMIs.html#storage-for-the-root-device>

Instance store-backed is faster than EBS for the simple fact that it is not persistent

You can’t stop your instance in order to pay less for example, however if you do you will simply lost everything there — So make sure to have backups before doing that.You can’t upgrade your instance or scale vertically (i.e. changing instance type), so you will have to create an AMI and from there launch a bigger instance

* An EC2 instance can be Stopped, Started or Terminated
* Stopping an instance performs a normal shutdown. That means any changes done to your OS configs, softwares, stored data is safely written to root volume and OS is loaded from that volume only. This root volume was not a separate EBS until EBS was launched so there was no state as stop since the OS was loaded from S3 instance store directly into RAM and all configs done to OS were in RAM. In this state (only available for EBS backed AMIs) all EBS volumes remain unattached in shutdown state.
* Terminating an instance also performs a shutdown, the root device volume is deleted BY DEFAULT, it can be set by deleteOnTermination attribute setting.
* **To migrate an instance to another Availability Zone**

1. Create an AMI from the instance. The procedure depends on the operating system and the type of root device volume for the instance. For more information, see the documentation that corresponds to your operating system and root device volume:
   * [Creating an Amazon EBS-Backed Linux AMI](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/creating-an-ami-ebs.html)
   * [Creating an Instance Store-Backed Linux AMI](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/creating-an-ami-instance-store.html)
   * [Creating an Amazon EBS-Backed Windows AMI](http://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/Creating_EBSbacked_WinAMI.html)
2. [EC2-VPC] If you need to preserve the private IPv4 address of the instance, you must delete the subnet in the current Availability Zone and then create a subnet in the new Availability Zone with the same IPv4 address range as the original subnet. Note that you must terminate all instances in a subnet before you can delete it. Therefore, you should create AMIs from all the instances in your subnet so that you can move all instances in the current subnet to the new subnet.
3. Launch an instance from the AMI that you just created, specifying the new Availability Zone or subnet. You can use the same instance type as the original instance, or select a new instance type. For more information, see [Launching Instances in an Availability Zone](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-regions-availability-zones.html#using-regions-availability-zones-launching).
4. If the original instance has an associated Elastic IP address, associate it with the new instance. For more information, see [Disassociating an Elastic IP Address and Reassociating with a Different Instance](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/elastic-ip-addresses-eip.html#using-instance-addressing-eips-associating-different).
5. If the original instance is a Reserved Instance, change the Availability Zone for your reservation. (If you also changed the instance type, you can also change the instance type for your reservation.) For more information, see [Submitting Modification Requests](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ri-modifying.html#ri-modification-process).
6. (Optional) Terminate the original instance. For more information, see [Terminating an Instance](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/terminating-instances.html#terminating-instances-console).

Security Groups

* A security group acts as a virtual firewall that controls inbound and outbound traffic for one or more instances. When you launch an instance, you associate one or more security groups (min 1 max 5) with the instance. You add rules to each security group that allow traffic to or from its associated instances
* Security Group Rules –

1. By default security groups allow all outbound traffic and all inbound traffic from the same SG itself.
2. Security Group rules are always permissive. You cannot create rules that deny access in security groups.
3. The rules are stateful – if you send a request from your instance, the response traffic for that request can flow in regardless of the inbound rules of the instance. Similarly responses to inbound requests can flow out regardless of outbound rules.
4. One instance can be applied multiple SGs.
5. In each rules we specify the following:

* Protocol – the protocol to allow - tcp, udp, icmp
* Port Range – the port range to allow.You can specify a simple port number eg:22 or port range eg: 7000-8000
* Source or Destination – IP range/IP of the source/destination
* For individual ipv4 addr use /32 prefix length. Eg:203.0.113.1/32
* For ipv6 addr use /128
* For a range of ipv4 change the prefix length.eg: 203.0.113.0/24
* Another Security Group – this allows instances associated access to instances associated. Note that this does not add rules from one security group to another.
* Description (Optional) – rule description
* **PCI-DSS compliance** - Amazon EC2 supports the processing, storage, and transmission of credit card data by a merchant or service provider, and has been validated as being compliant with Payment Card Industry (PCI) Data Security Standard (DSS).
* **Tag**- tagging your amazon resources can be really helpful for grouping or filtering of resources. Tags are attrubutes in key value pair form, attached to a resource defining metadata of resource.Maximum number of tags on a resource can be 50.Tags are case sensitive.DO NOT USE aws: PREFIX FOR TAG NAME BECAUSE THEY ARE RESERVED FOR AWS USE.

**Eg**: name,description,owner,environment,stack,maintainer,application

* Elastic IP – Before we understand elastic IP we need to understand a few concepts:
* Private IP - A private IPv4 address is an IP address that's not reachable over the Internet. You can use private IPv4 addresses for communication between instances in the same network. A primary private ip is assigned (using DHCP) when the instance is launched, and it is inside the range of the subnet. Additional (secondary) private IPs can also be assigned. Each EC2 instance is attached to by default on networking interface, that interface is actually assigned the private IP.
* Internal DNS Hostname - When you launch an instance, we allocate a primary private IPv4 address for the instance. Each instance is also given an internal DNS hostname that resolves to the primary private IPv4 address; for example, ip-10-251-50-12.ec2.internal. You can use the internal DNS hostname for communication between instances in the same network, but we can't resolve the DNS hostname outside the network that the instance is in.
* Public IP - A public IP address is an IPv4 address that's reachable from the Internet. You can use public addresses for communication between your instances and the Internet. You can choose to assign a public IP to an instance associated to a VPC while launching. You cannot manually associate or disassociate public IP.
* External DNS Hostnames – each instance that receives a public IP also is assigned an external DNS hostname. The external DNS hostname is resolved to public IP if hit by outside the network (VPC) and resolves to private IP if hit by inside the network (VPC). The public IP is mapped to private IP by NAT.
* Amazon DNS server – Amazon provides a DNS server that resolves instance dns hostnames to ipv4 addresses. The DNS server is located at your network ID + 2 address
* Elastic IP – if you need a public IP that can persist and can be associated to and from instances use an Elastic IP. An Elastic IP address is a static IPv4 address designed for dynamic cloud computing. An Elastic IP address is associated with your AWS account. With 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.
* EC2 Auto Scaling – It helps you ensure that desired number of instances is available handle the load of your application. You can create collections of EC2 instances called Auto Scaling Groups. You can set min max and desired count in ASGs. Also you can provide configuration that will be applied on the launched instances.
* Auto Scaling Components :

1. AS Groups – Auto Scaling Groups are collection of EC2 instances that are similar and are treated as a logical grouping for the purpose of instance scaling and management. We can specify the criteria on which the group scales up or down. To maintain the desired count the ASG service performs periodic health checks on instances in group. We can specify the period as well as configure the health check.
2. Launch Configuration – it is asset of settings (including everything that you set while launching) that ASG uses to launch new instances. Launch configs cannot be updated, new ones have to be created.
3. Scaling Options – there are several ways to scale your ASG:

* Maintain current instance levels – specifying min or exact number of instances to maintain
* Manual Scaling- updating min,max or desired in ASG manually.
* Scale Bases on schedule
* Scale based on Demand – defining scaling policies for scale in and scale out. Multiple Scaling Policies can also be set. If multiple policies meet scale in condition or scale out condition at same time, ECS ASG chooses to scale on basis of policy having maximum amount of impact eg: one policy scales up by one, the other by two, if both are hit simultaneously the latter one would be activated.