**Q: What is DevOps?**

A: DevOps is a methodology.

Or: Implementing automation at each and every stage.

**Q: Why organization needs DevOps specialist?**

A: - Fast delivery.

- Higher quality.

- Less Capex + Opex.

- Reduced outage.

**SDLC (Software development lifecycle):**

**Development team: Operations team:**

Developer Deploy

Build Maintenance

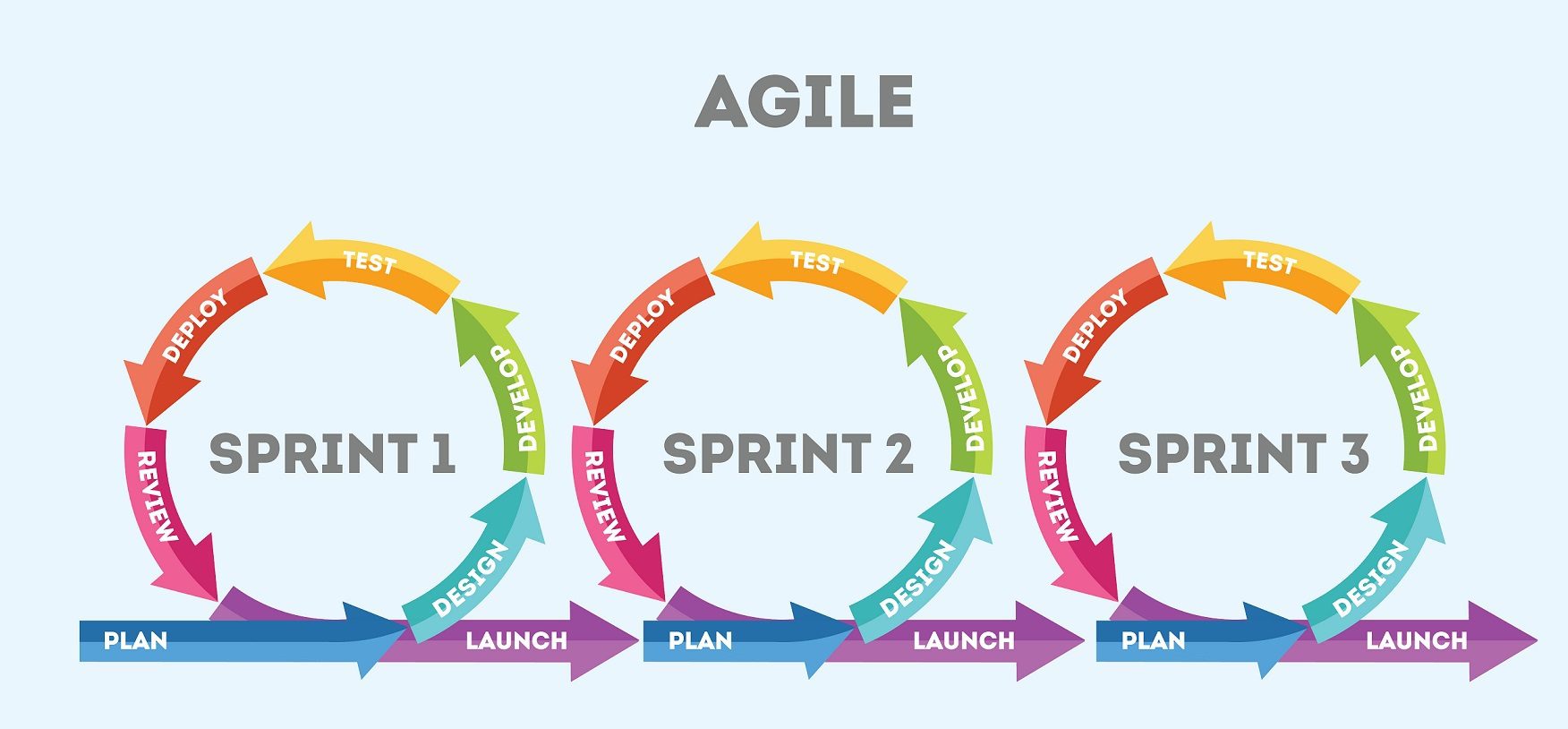
Test Monitoring

QA

**Methodologies:**

1. Waterfall Methodology. (Runs step by step).
2. Agile Methodology:

(Build short, Build often), (Work done by in parts), (So that if any changes required then it can be done in next step), Silos (isolation) issues was coming in this methodology.



1. DevOps Methodology (Successor of Agile methodology).

**DevOps Methodology:**

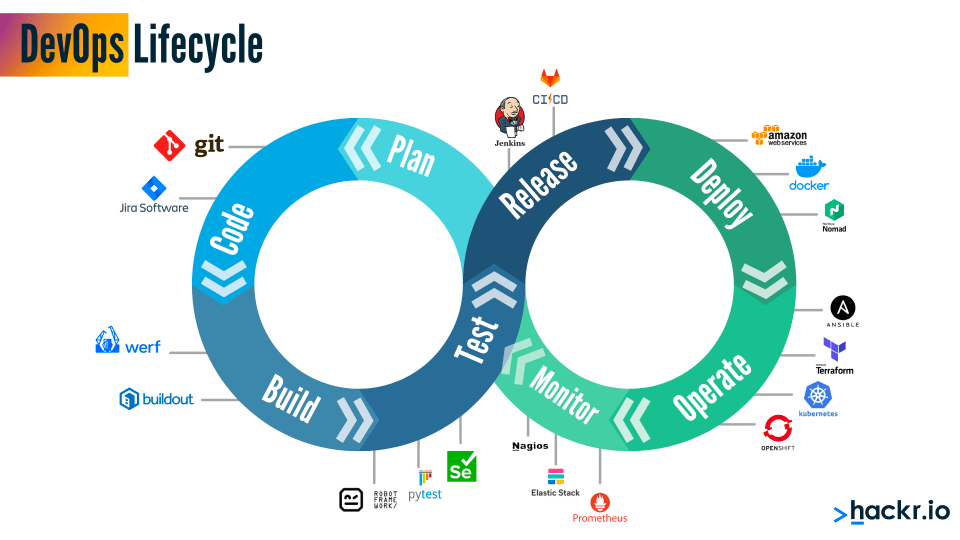
1. The term DevOps is a combination of two words, (Development & Operation).
2. DevOps is a methodology that allows a single team to manage the entire application development lifecycle.
3. The objective of the devops is to shorten the system development life cycle (reduce the time period).
4. DevOps is a software development approach through which superior quality software can be developed quickly and with more reliability.

|  |
| --- |
| DEPLOY  OPERATE  MONITOR |

**SDLC (Software development lifecycle) DevOps:**

|  |
| --- |
| PLAN  CODE  BUILD  TEST |

|  |
| --- |
| INTEGRATION |



**Amazon Web services (AWS)**

(The AWS Cloud spans 102 Availability Zones within 32 geographic regions around the world)

1. What is cloud and how it works.

Cloud is a technology or technique which gives you ability to share your resources through the internet. Internet connection is must.

OR

Cloud is a service which gives you resources, which you can use from a remote location and you can pay according to your usages.

1. IAAS, PAAS & SAAS in AWS.
2. How to create AWS Free tier account.
3. Types of EC2 Instances.

* General purpose
* Compute optimized
* Memory optimized
* Accelerated computing
* Storage optimized
* HPC (High performance computing) optimized

1. VPC (Virtual private cloud).

* VPC Creation
* VPC-NAT Gateway
* VPC Peering
* NACL (Network access control list)
* VPC Endpoint

1. AWS Storage.

* Simple storage service (S3)
* Elastic file system (EFS)
* Elastic block storage (EBS)
* S3 Glacier
* AWS Snowball
* Amazon EC2 instance storage
* AWS storage gateway
* Amazon FSx

1. EBS (Elastic block store).

* Solid state drive (SSD) volumes

General purpose SSD Volumes - (GP2 & GP3).

Provisioned IOPS SSD volumes – (iO2 & iO1)

* Hard disk drive (HDD) volumes

Throughput optimized HDD Volumes – (st1)

Cold HDD Volumes – (sc1)

* Previous generation volumes

Magnetic

1. AMI (Amazon machine image).
2. AWS Autoscaling.
3. AWS Elastic load balancer (ELB).
4. AWS Application load balancer (ALB).
5. AWS Network load balancer (NLB).
6. AWS IAM (Identity and Access management).
7. Database.
8. Dynamo DB.
9. AWS Route 53.
10. AWS CloudFront.
11. AWS SQS (Simple queue service).
12. AWS SNS (Simple notification service).
13. AWS Lambda.

**Basic of Linux:**

1. Linux made by the Linus Torvald in 1991.
2. Linux is a Kernel. {OS = Kernel + Software}.
3. Linux is not a Unix derivative. It was written from scratch.
4. A Linux distribution = is the Linux kernel and a collection of software, that together create OS.
5. Linux OS = Linux kernel + GNU.
6. Linux is the combination of GNU software and Linux kernel.
7. Linux is based on CLI (command line interface) while windows is based on GUI (Graphical user interface). Linux too has GUI but the CLI is very popular for Linux.

**Linux features (Advantages):**

1. Open source
2. Secure
3. Simplified updates for all installed software
4. Light weight
5. Multiuser – Multitask
6. Multiple distribution – Red Hat, Debian, Fedora, Ubuntu, KALI Linux.

**File system hierarchy:**

1. **/** = Forward slash – Top level root directory.
2. **/home** = It is the home directory for root user.
3. **/root** = It is home directory for root user.
4. **/boot** = It contains bootable files for Linux.
5. **/etc** = It contains all configuration files.
6. **/usr** = By default software are installed in this directory.
7. **/bin** = It contains commands used by all user. Root user can too use these commands.
8. **/sbin** = It contains commands used by only root user.
9. **/opt** = Optional application software packages.
10. **/dev** = Essential device files, this includes terminal devices, USB or any device attached to the system.

**Linux Commands and it’s use cases:**

1. **How to create a file.**
2. cat
3. touch
4. vi/vim
5. nano
6. **cat =**
7. Create file: To create a single file.
8. Concatenate file: To add more than one file into a single file.
9. Copy files: To copy the content of file x into file y.
10. To exit: Use ctrl+d.
11. If you want to add something in your existing file, you can add with the help of this command. But you cannot edit anything in that file, you can only add.
12. tac: To see the content, from bottom to top.
13. **Touch=**
14. Create an empty file.
15. Create multiple empty files.
16. Change all timestamp of a file.
17. You can update access time or modify time individually too, of the file.
18. touch -a & touch -m (commands for above line).

**Time stamp: (cmd: stat ‘filename’)**

1. Access time: Last time when a file was accessed.
2. Modify time: Last time when a file was modified.
3. Change time: Last time when file metadata was changed.

**3. vi/vim (editor):**

1. A programmer text editor.

2. It can be used to edit all kinds of plain text, it is especially useful for editing programs, mainly used for Unix programs.

Commands –

**:w** – to save

**:wq** or **:x** – To save & quit.

**:q** – Quit only (it will not save your current work).

**:q!** – Force quit (No save).

i = Press i to insert (to edit or add something in the file).

Esc = Press esc to come back from the editing or adding zone.

1. **Nano:**

To create a file, and edit a file too.

Use **ctrl+x = to exit**, and **Y = to save** and press enter.

**Some more basic commands:**

**sudo su = super user do switch user**

**ls** = To see the list of directory and files.

**ls -l or ll** = To see the list of directory and files, with details (ll includes hidden files).

**ls -a** = To see the list of all directories and files (included hidden one).

**history** = to see the history of the commands, which you have already run.

**Ctrl+l** or clear = To clear the page.

**mkdir** = To create a directory. Use -p to create a directory inside a directory.

**cd** = To change the directory.

**cd ..** = To come out from the directory (backspace).

**cd ../../..** = 3 time back, in a single command.

**pwd** = Print working directory.

**.**filename/**.**directoryname = If you want to create a hidden file or hidden directory, just use **.** (dot) before the file name.

**cp** = To copy a file data into another file. But the data of another file will be removed, once used this command.

Example – cp file1 file2

**cp -r** = To copy a directory and its contents.

**mv** = To cut and paste. And to rename a file or directory.

Example – mv file1 dir1 **or** mv dir1 dir2 **or** mv file1 myfile

**rm =** To remove (delete) a file.

**rm -r** = To remove directory.

**rm -rf** = To remove non empty files and directory. (forcefully).

**which** = To check the location of installed Package (software) or whether it is installed in your system or not.

**tree** = To check the details of the directory and files in one page. (apt-get install tree).

**less** = To see the content of first entire page. To exit press q.

**head** = To see the content of 10 starting lines.

**tail** = To see the content of 10 last lines.

**more** = To see the entire content, page by page.

**sudo apt install “package name”** = To install a package (software) in our system.

**Hostname** = To see the private IP address for your machine.

**Cat /etc/os-release** = To see the basic details of your machine.

**Ifconfig** = To see the IP addresses and ethernet port of your machine.

**Apt install nginx** = To install nginx in your machine. (apt – advanced package tool).

**Apt remove nginx** = To remove nginx from your machine.

**Apt update nginx** = To update, installed nginx in your machine.

**Service nginx start** = To start the nginx (if it is showing inactive).

**Service nginx status** = To check the status of your nginx. It is active (running) or not.

Cd /var/www/html = It is the location, where we can see and make index.html file.

**Whoami** = To see who you are.

**Echo** = If your machine has been using by others users too, then this command will help you to pass the message to them. (To highlight the message).

We can create the file too by using the echo command.

Eg – echo “welcome” >filename.

Echo “namaste” >>filename

Echo >filename

**Grep** = If you want to see the specific content in your file (like – name, contact no or whatever). It works same like, ctrl+f works in windows.

**Sort** = To change the data alphabetically into the file. (A-B-C).

**useradd** = To create the user. Like in windows we create user by the control panel, so that the user cannot have the admin rights. Eg- useradd ajay

Cat /etc/passwd – it is the location where we can see the users.

Note – when we create a user, then the group will automatically be created by the same name.

**groupadd** = to create the group.

Eg- cat /etc/group – location to see the existing groups. We can also see the users which is already added in that group.

Note – But when we create the group, the user will not be created by the same name.

**gpasswd -a** = to add the user into group. & use **-M** to add multiple users into the group. Use (,) to separate the users.

Eg- gpasswd -a username groupname

cat /etc/group – to check, if the user has been added in the group or not.

Note – if you want to add any user in the group, then you must create the user first.

**ln** = To create a hard link (backup) of your file. And the hard link automatically syncs with your original file. Even if you delete your original file, the data will not be affected here. As it works like a backup file. Weather you update your original file or backed up (hard link) file. The data will be updated in both files at the same time.

**ln -s** = To create a soft link of your file. It is also sync with your original file, but when you delete your original file then you will not see the content of your file here, as it works like a shortcut.

**Tar** = To group/pack multiple files into one file. Tar is an achiever.

Eg – **tar -cvf** filename.tar filename. **tar -xvf** filename.tar – to extract the tar file.

**gzip** = To compress the file or directory.

Eg – gzip filename

**gunzip** = To upzip the zip file/directory.

**Access modes/Permissions**

**The three basic permission modes are:**

**Read (r) (4)** = Allows users to view (display) the content of a file or list the contents of a directory.

**Write (w) (2)** = Allows users to modify the content of a file or create/delete files within a directory.

**Execute (x) (1)** = Allows users to execute a file (if it's a script or executable) or enter into a directory.

These permissions are assigned to three categories of users:

**Owner/user** = The user who owns the file or directory.

**Group** = A group of users who share certain permissions.

**Others** = Any user who is not the owner and not part of the group.

**Method 1:**

= Each permission is assigned a numeric value: **read (4)**, **write (2)**, and **execute (1)**.

These values can be added to represent the permission modes.

For example, "rwx" becomes 7 (4 + 2 + 1), "rw-" becomes 6 (4 + 2), and "r--" becomes 4 (4).

**(r = 4, w=2, x=1)**

**Commands:**

**chmod = To change the access mode of a file.**

**chown = To change the owner of a file or directory.**

**chgrp = To change the group of a file or directory.**

* **rw- r-- r-- 1 root root 18 Aug 15 13:45** filename

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Represent: file or directory or link** | **owner/root user** | **group** | **other users** | **symbolic link** | **who is owner** | **group name** | **file size (bytes)** | **date** | **time** | **filename** |
| -/ d / l | rwx | rwx | rwx | 1 | root | root | 18 | Aug-15 | 13:45 | filename |

**Method: 2 = (=, +, –)**

**U = user/owner**

**G = Group**

**O = Others**

**u=rwx, g=rw-, o=r—**

**AWS Solution architect:**

Que: What is cloud and how it works.

Ans: Cloud is a technology or technique which gives you ability to share your resources through the internet. Internet connection is must.

OR

Cloud is a service which gives you resources, which you can use from a remote location and you can pay according to your usages.

OR

Cloud computing is the on-demand delivery of, compute power, database, storage, applications, and other IT resources, through a cloud service platform via the internet with pay as you model.

Cloud service platform:

1. AWS – Amazon web services.
2. GCP – Google cloud platform.
3. Azure – Microsoft Azure.
4. Alibaba.
5. Oracle cloud.
6. IBM cloud.
7. VMware.

**Characteristics of cloud:**

1. On demand self-service.
2. Broad network access.
3. Scalability.
4. Resource pooling.
5. Measured services.

**Service model/services in cloud:**

1. IAAS – Infrastructure as a service.
2. PAAS – Platform as a service.
3. SAAS – Software as a service.

**AWS History:**

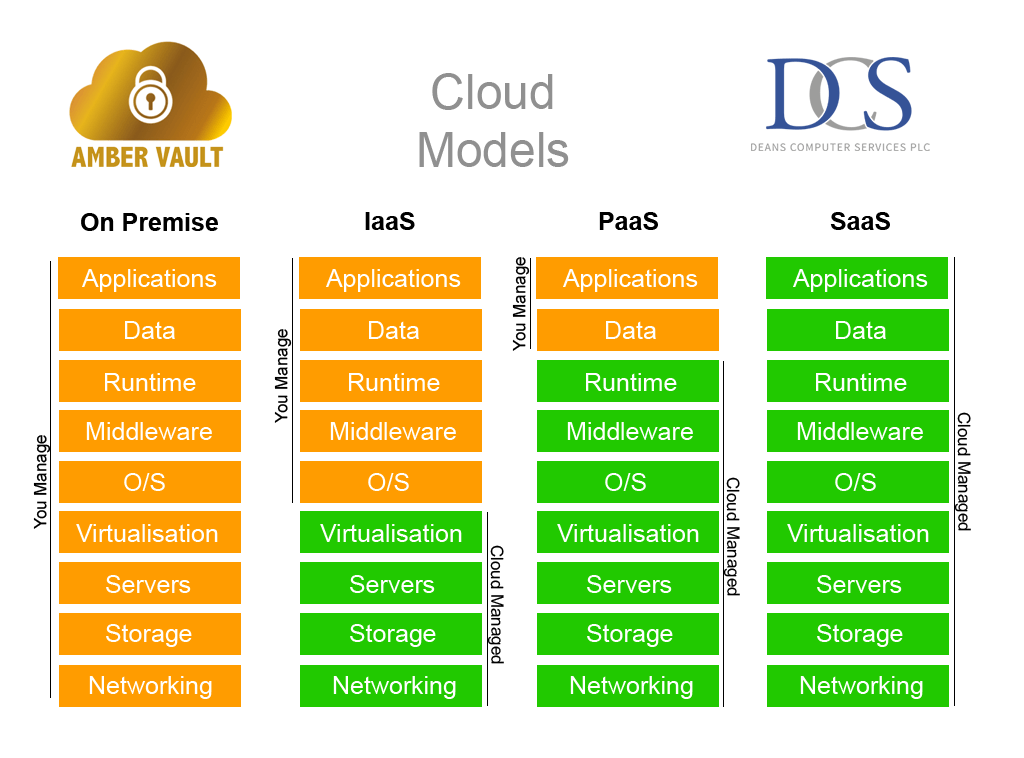
2006 – AWS Launched.

2010 – Moved to Amazon.com

2013 – Certification launched.

**Certifications:**

1. AWS solution architect.
2. AWS developer (DevOps). (Developer + Operation).
3. AWS SysOps (System admins + Operation).



The concepts of IaaS, PaaS, and SaaS:

1. Infrastructure as a Service (IaaS): IaaS is a cloud computing service model that provides virtualized computing resources over the internet. In an IaaS model, users can rent IT infrastructure components on a pay-as-you-go basis, avoiding the need to invest in and maintain physical hardware.

Key characteristics of IaaS include: -

Virtualization: IaaS providers create virtual instances of computing resources such as virtual machines, storage, and networking.

Scalability: Users can easily scale up or down based on their resource needs, often in real-time.

Self-Service: Users have control over the provisioning, management, and configuration of their virtualized resources.

Resource Management: IaaS allows users to control and manage various aspects of their infrastructure, such as configuring firewalls, load balancers, and storage solutions.

Examples: Amazon Web Services (AWS) Elastic Compute Cloud (EC2), Microsoft Azure Virtual Machines, Google Compute Engine.

1. Platform as a Service (PaaS): PaaS is a cloud computing service model that provides a platform and environment for developers to build, deploy, and manage applications without needing to worry about the underlying infrastructure. PaaS abstracts away much of the complexities of infrastructure management, allowing developers to focus on coding and application development.

Key characteristics of PaaS include:

Development Frameworks: PaaS provides development frameworks, runtime environments, and tools to streamline the development process.

Automatic Scaling: PaaS platforms often offer automated scaling options based on application demand.

Database Management: PaaS services may include managed database solutions, reducing the need for manual database administration.

Deployment Automation: PaaS platforms provide tools for continuous integration, continuous deployment (CI/CD), and DevOps practices.

Examples: Heroku, Google App Engine, Microsoft Azure App Service.

1. Software as a Service (SaaS): SaaS is a cloud computing service model where software applications are delivered over the internet on a subscription basis. In this model, users can access and use software applications without needing to install or maintain them locally. SaaS applications are hosted and maintained by the service provider.

Key characteristics of SaaS include:

Accessibility: SaaS applications can be accessed from anywhere with an internet connection and a compatible device.

Automatic Updates: Providers handle updates and patches, ensuring users are always using the latest version of the software.

Multi-Tenancy: A single instance of the software can serve multiple customers, sharing resources securely.

Subscription Model: Users typically pay a recurring fee, which often includes support and maintenance.

Examples: Salesforce, Google Workspace (formerly G Suite), Microsoft 365, Dropbox.

In summary, IaaS provides virtualized infrastructure resources, PaaS offers a platform for application development and deployment, and SaaS delivers fully functional software applications over the internet. These cloud service models allow businesses and developers to focus on their core activities while leveraging the benefits of cloud computing, such as scalability, cost-efficiency, and flexibility.

**Virtualization:**

We need a hypervisor to virtualise the infrastructure.

Hypervisor – A hypervisor is software that creates and runs virtual machines.

OR

A hypervisor is a software that you can use to run multiple virtual machines on a single physical machine.

Every virtual machine has its own operating system and applications. The hypervisor allocates the underlying physical computing resources such as CPU and memory to individual virtual machines as required.

The hypervisor treats resources—like CPU, memory, and storage—as a pool that can be easily reallocated between existing guests or to new virtual machines.

**Hypervisor:**

Microsoft: Hyper-V

AWS: AWS Nitro system (Xen).

VMware: Vsphere (ESXi).

**AWS - Elastic compute cloud (EC2)**

1. Amazon EC2 provides scalable computing capacity in the AWS Cloud.
2. You can use Amazon EC2 to launch as many or as few virtual servers as you need.
3. Amazon EC2 enables you to scale up or scale down the instance.
4. Amazon EC2 is having 2 storage options – EBS (Elastic block store) and instance store.
5. Pre-configured templates are available, known as amazon machine image (AMI).
6. By default, when you create an EC2 account with amazon, your account is limited to a maximum of 20 instances per EC2 region with two default high I/O instances.

**Types of EC2 Instances:**

* General purpose
* Compute optimized
* Memory optimized
* Accelerated computing
* Storage optimized
* HPC (High performance computing) optimized

[Compute – Amazon EC2 Instance Types – AWS.html](Compute%20–%20Amazon%20EC2%20Instance%20Types%20–%20AWS.html)

**EC2 Purchasing options**

Amazon EC2 offers several purchasing options to cater to different usage scenarios and budget considerations. These purchasing options allow you to optimize costs and performance based on your specific needs. As of my last update in September 2021, here are the main EC2 instance purchasing options:

1. **On-Demand Instances:**

On-Demand instances allow you to pay for compute capacity by the hour or by the second without any long-term commitments. This option is ideal for short-term workloads, unpredictable workloads, or when you need instances quickly without upfront costs.

**2. Reserved Instances (RIs):**

Reserved Instances provide a discount (compared to On-Demand prices) in exchange for a commitment to use a specific instance type in a particular Availability Zone for a 1-year or 3-year term. There are three types of Reserved Instances:

**- Standard RIs:** Provide a significant upfront cost saving compared to On-Demand instances.

**- Convertible RIs:** Offer flexibility to change the instance type later if your needs evolve, while still benefiting from cost savings.

**- Scheduled RIs:** Allow you to reserve instances for specific time periods, which is useful for predictable workloads.

**3.** **Spot Instances:**

Spot Instances allow you to bid for unused EC2 capacity in the Spot market. These instances can provide significant cost savings, sometimes up to 90% off On-Demand prices. However, they can be terminated if the capacity is needed by someone willing to pay more than your bid.

**4. Dedicated Hosts:**

Dedicated Hosts are physical servers with EC2 instance capacity dedicated to your use. This option is useful for compliance requirements, software licensing, or when you need control over the underlying physical infrastructure.

**5. Dedicated Instances:**

Dedicated Instances are instances that run on hardware dedicated to a single customer. While not necessarily a separate purchasing option, they ensure isolation from instances of other customers on the same physical hardware.

**6. Capacity Reservations:**

Capacity Reservations allow you to reserve capacity for On-Demand instances in a specific Availability Zone. This option helps ensure you have the capacity you need even during peak times.

**How to attach extra volume in your existing server?**

1. Go into your AWS account – Ec2 – Elastic block store – Volumes.
2. Create a Volume – Select the volume type – Give the volume size as per your requirement – Availability zone of the Volume and your instance must be the same.
3. Give the name tag to the volume, and create it.
4. Once volume creation completed, Select the volume, and go into the Action and attach it to your instance, where you wanted to attach this additional volume.
5. Now login your instance. Run the **lsblk** command to list the available block devices and their mount points. It will be showing in xvdf.
6. Now run the command **df -h** to check if the added volume is showing here or not. As of now you won’t see it here.
7. Create a filesystem on the volume – Run **sudo mkfs -t ext4 /dev/xvdf**
8. Create a mount point – **sudo mkdir /mnt/myvolume**
9. Mount the volume – **sudo mount /dev/xvdf /mnt/myvolume**
10. Run **df -h** to check.
11. Verify and Test: **cd /mnt/myvolume** & **ls -l**
12. You can create files, read, and write data to this directory to ensure that the mounted volume is working as expected.

**How to retrive meta data of your server/instance?**

Run command: curl <http://169.254.169.254/latest/meta-data>

**Virtual Private Cloud (VPC)**

VPC is a Virtual Private Network. It allows you to create isolated and logically segmented virtual networks within a cloud environment.

* It is logically isolated from other virtual networks in the AWS Cloud.
* Max 5 VPC can be created in one region and 200 subnets in one VPC. =1000
* VPC is Region Specific.
* Subnet is Availability zone specific.
* We can allocate max 5 Elastic IPs.
* A VPC is confined to an AWS Region and does not extend between Regions.
* Once the VPC is created, you cannot change it’s CIDR block range.
* If you need a different CIDR size then create a new VPC.

**Components of VPC:**

* CIDR and IP Address subnets.
* Implied router & Routing table.
* Internet gateway.
* Security groups.
* NACL (Networks access control list).
* Virtual private gateway.
* Peering connections.
* Elastic IP.

**Private IP Range of CIDR:**

10.0.0.0

172.0.0.0

192.0.0.0

**Steps to creating a VPC:**

1. Create a VPC
2. Create the Subnet
3. Create Internet gateway
4. Create Route table

* When you create a VPC, you must specify an IPv4 CIDR Block for the VPC. The allowed block size is between /16 to /28.
* The fist four and last IP addresses of subnet cannot be assigned.
* For Example – **10.0.0.0/24**

Total host = 256

Reserved host = 5

Available host = 256-5 = 251

**Reserved host:**

* + - 1. – Networks address
      2. – Reserved by AWS for the VPC router
      3. – The IP address of DNS server (Reserved by AWS)
      4. – Reserved for future use

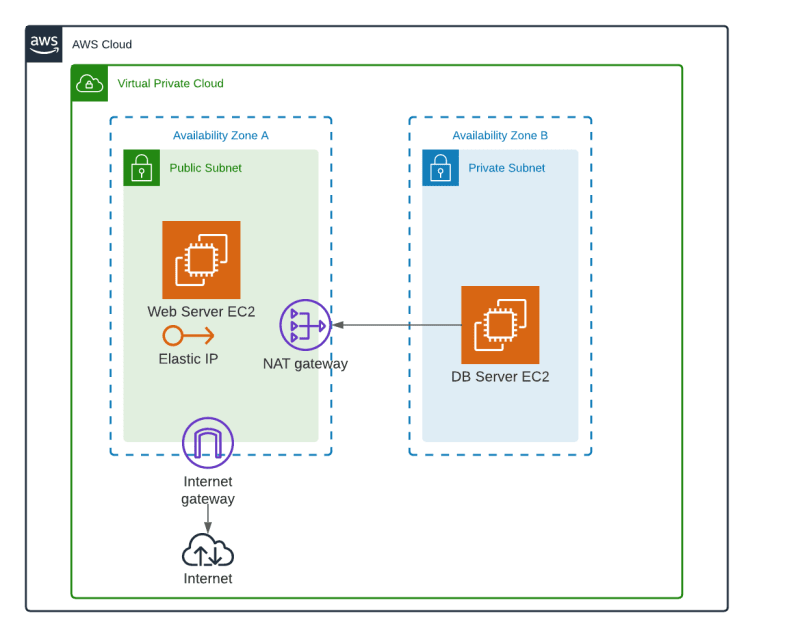
.

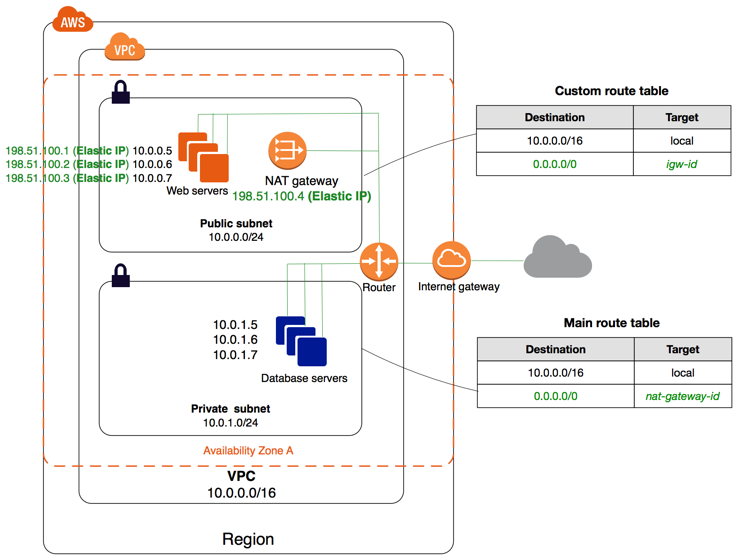
.

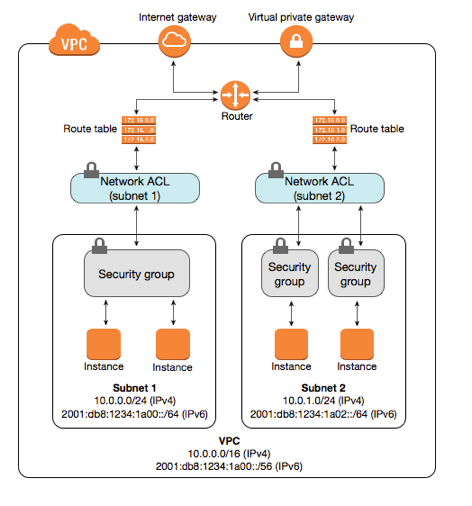
* + - 1. – Broadcast address

**ssh -i** keyname.pem **ubuntu@**ip\_address

**chmod 600**







**VPC Endpoints**

A VPC endpoint enables you to privately connect your VPC to supported AWS services.

Instances in your VPC do not require public IP address to communicate with resources in the service.

1. Create a VPC.
2. Create two subnets (Public & Private)
3. Create internet gateway and attach it to your VPC.
4. Create two route tables, One for Public subnet and one for private subnet.
5. Create two EC2 server – one in public subnet and another one in private subnet.
6. Now go into – VPC Services – Endpoints
7. Create an Endpoint – Select AWS Services – select S3, type – gateway.
8. Select your VPC – select your private subnet – create endpoint.

Login to your private instance through public instance.

To get your AWS access key ID and secret access key – Open (Login) the AWS – click on your name (**Account name**) on the right top side. Click on **security credentials** – go down and click on **create access key,** check the I understand box and proceed. And download the .csv file. You will see your Access key ID and secret access key in this file.

**Commands:**

aws configure

Fill your access key ID & secret access key

aws s3 mb s3://Bucketname (mb = make bucket) (This command is used to create a s3 ……………………………………………………………………………. bucket in your AWS account)

aws s3 ls (To see the list of existing buckets)

aws s3 ls s3://Bucketname (To see the content, what is inside your bucket)

aws s3 rb s3://Bucketname (To remove (delete) the bucket), (rb = remove bucket).

Bucket must be empty, only after you can remove (delete) it.

If you are not able to use the command aws configure:

1. Download the AWS CLI bundle installer, by below command:

wget <https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip>

once download done run the below command:

unzip awscli-exe-linux-x86\_64.zip

sudo ./aws/install

Now try to run the aws configure command and fill the credentials.

**AWS Storage**

Block storage: Block storage is suitable for transactional databases, random read/write Loads and structured database storage.

Block storage divide the data into the blocks, to be stored in evenly sized blocks for instance, a file can we split into evenly sized blocks before it is stored.

Data blocks stored in block storage would not contain meta data (data created, data modified, content type etc).

Block storage only keeps the address (index) where the data blocks are stored. It does not care what is in that block, just how to retrieve it when required.

Example of block storage – EBS

Data can be retrieved only through instance, where the EBS is connected.

Object storage: Object storage stores the file as a whole and does not divide them.

In object storage an object is – The file/data itself – Its meta data – object global unique ID.

The object globally unique ID is a unique identifier for the object (can be the object name itself) and it must be unique such that it can be retrieved this regarding where its physical storage location is.

Object storage cannot be mounted as a drive.

Example of object storage solution, Amazon S3, Dropbox.

Data can be retrieved from anywhere via http or https.

Simple storage services (S3):

* S3 is a storage for the internet. It has a simple webservice interface for simple storing and retrieving of any amount of data, anytime from anywhere from the internet.
* S3 is an object-based storage.
* You can not install operating system on S3.
* Data is stored in bucket.
* Max capacity of a Bucket is 5TB.
* You can create folders in your bucket.
* You cannot create nested Buckets.
* Bucket ownership is non-transferable.
* S3 Bucket is region specific.
* You can have up to 100 Buckets per account.

S3 Bucket Naming rules:

* S3 bucket name are globally unique.
* Bucket names cannot be changed.
* Bucket name must be at least 3 and no more than 63 characters.
* Bucket names are the part of URL, used to access a bucket.
* Bucket names can contain lowercase letter, numbers, and hyphen. Cannot use uppercase letter.
* Bucket name should not be an IP address.
* By default, Buckets and its objects are private. So only owner can access this bucket.

S3 Bucket Versioning:

* Bucket versioning is a S3 Bucket sub-resource, used to protect data/object, against accidental deletion or overwrites.
* Once you enable versioning on a bucket, it cannot be disabled, however it can be suspended.
* When enabled, bucket versioning will protect existing and new objects, and maintain their versions as they are updated.
* When versioning is enabled, and you try to delete an object, a delete marker is placed on the object.
* You can still view the object and the delete marker.
* If you want to reconsider the deleted object, you can delete the “delete marker” and the object will be available again.
* Versioning applies to all the objects in the bucket.

Storage classes of Amazon S3:

**S3 Standard:** This is the default storage class, offering high durability, availability, and low latency. It is suitable for frequently accessed data and is designed for high-performance applications.

**S3 Intelligent-Tiering:** This storage class automatically moves objects between two access tiers: frequent and infrequent access, based on changing access patterns. It is designed to optimize costs for data with unknown or changing access patterns.

**S3 One Zone-IA (Infrequent Access):** Data stored in this class is stored in a single availability zone and is ideal for infrequently accessed data that can be recreated if lost.

**S3 Glacier:** This class is designed for archiving data that is rarely accessed. Data retrieval times can be longer compared to other classes, but it offers very cost-effective storage.

**S3 Glacier Deep Archive:** This storage class offers the lowest cost storage for archiving data, but with even longer retrieval times. It is suitable for data that is rarely, if ever, accessed.

**S3 Outposts:** This class is used with AWS Outposts, allowing you to extend Amazon S3 storage to on-premises environments.

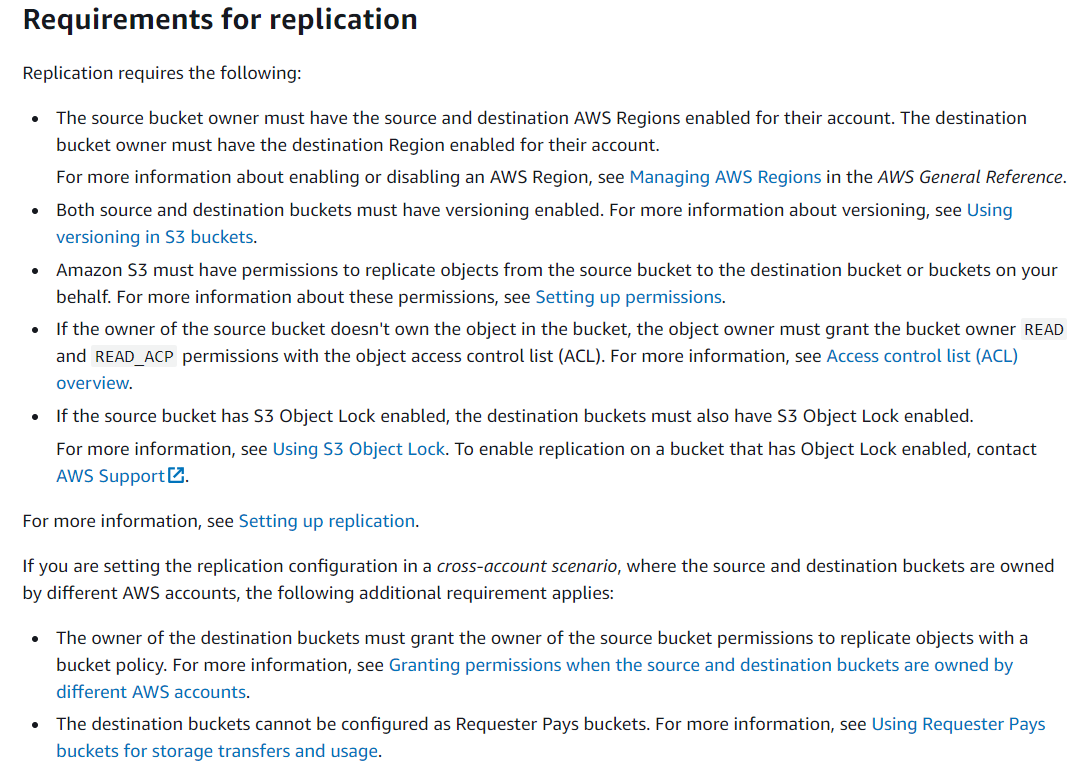
**S3 Replication:** While not a traditional storage class, S3 provides options for replicating data to other S3 buckets, which can be in the same or different regions. This can help improve data durability and availability.

**S3 Object Lock:** This feature enforces retention policies on objects to ensure they are not deleted or modified for a specified period. It helps with compliance and data protection.

**Lab to practice:**

* S3 Bucket creation.
* Enable versioning on S3 Bucket.
* Cross region replication (CRR).
* Lifecycle management

<https://docs.aws.amazon.com/AmazonS3/latest/userguide/replication.html>



EFS (Elastic File System) LAB:

* It is a shared storage.
* Create two EC2 instance in different availability zone (diff subnet).
* Allow NFS in the security group.
* Go into storage – EFS – Create one EFS (Elastic file system).
* Choose the same security group, which you have made for your EC2 instances.
* Login your EC2 instance.
* Run command Sudo apt install nfs-common
* mkdir efs
* Copy the mount address (command) from EFS details, from attach option.
* Make sure the EFS security group must be same for both EC2 and EFS, otherwise you will not able to attach it.
* To check the security group in EFS – open your EFS – go into network option and check there.

EBS (Elastic block store): (NAS) (network attached storage).

It is persistent – Means data will be deleted only after termination.

It is network attached storage.

An EBS volume can attach to a single EC2 instance only, at a time.

Both EBS volume and EC2 instance must be in the same AZ.

**EBS Volume Types:**

* **Solid state drive (SSD) volumes** [Bootable]

General purpose SSD Volumes - (GP2 & GP3).

Provisioned IOPS SSD volumes – (iO2 & iO1)

* **Hard disk drive (HDD) volumes** [non-Bootable]

Throughput optimized HDD Volumes – (st1)

Cold HDD Volumes – (sc1)

* **Previous generation volumes** [Bootable]

Magnetic

Instance store backed: Direct Attached Storage