SDLC for the project.

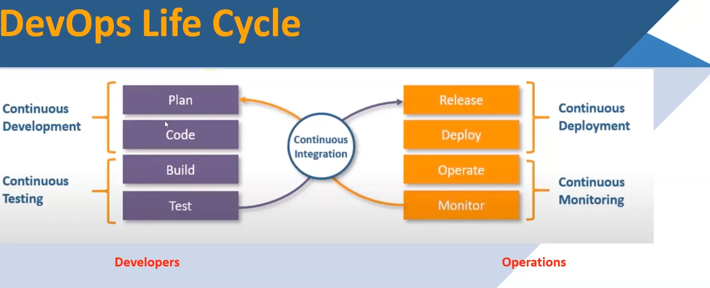
* Planning -> Defines project goals ,scope, resoureces, costs and risks.
* Requirement analysis -> Involves gathering, analyzing ,and documenting functional and non functional requirements.
* Design -> Translates requirements into a blueprint for the application, covering architecture, components and user interface.
* Implement -> Actual coding and development take place based on design specifications.
* Test -> Conducts various test such as unit testing, integration testing, system testing and user accept testing(UAT).
* Deploy -> Releases the software to users after through testing.
* Maintain -> Ensure the software continuous to operate as a required. Address bug fixes etc.

SDLC Models

1. Waterfall Model :-> A linear and sequential approach, where each phase must be complete before the next begins.
2. Agile Models -> an iterative approach that emphasizes flexibility, collaboration and frequent releases.

DevOps -> DevOps started in 2007 by Patrics debois

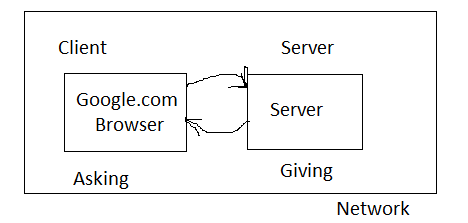
DevOps lifecycle



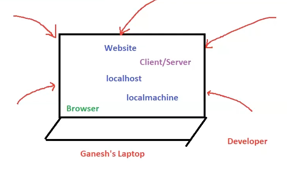
Client Server Architecture

* Client = which required a resources
* Server= which response to the resources

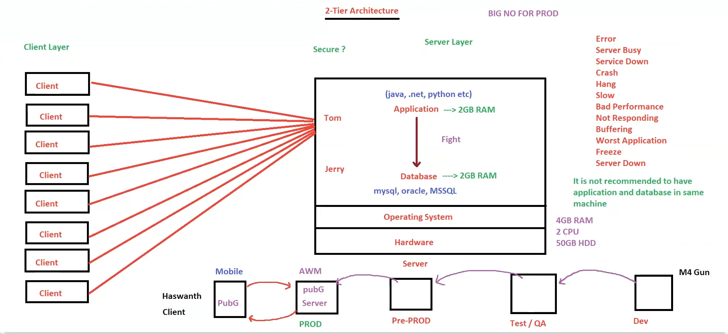
note – Resource = information



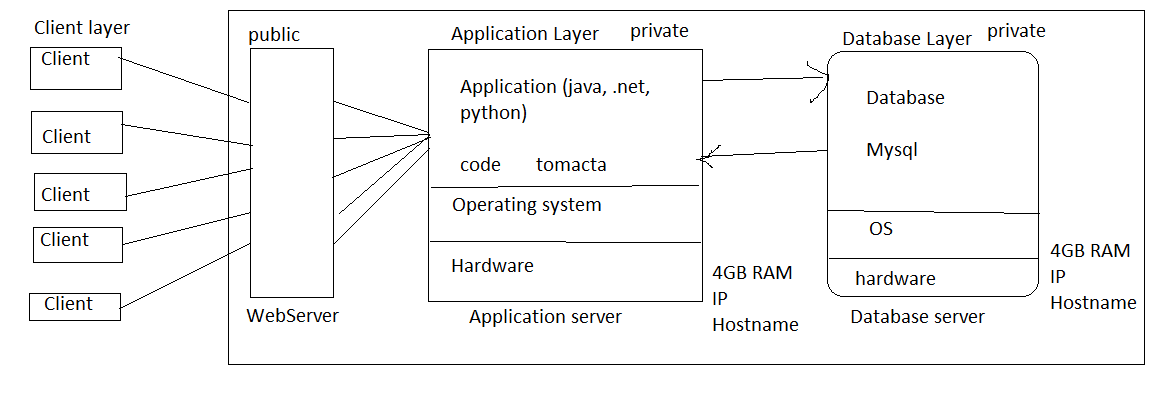
* 1 – Tier Architecture



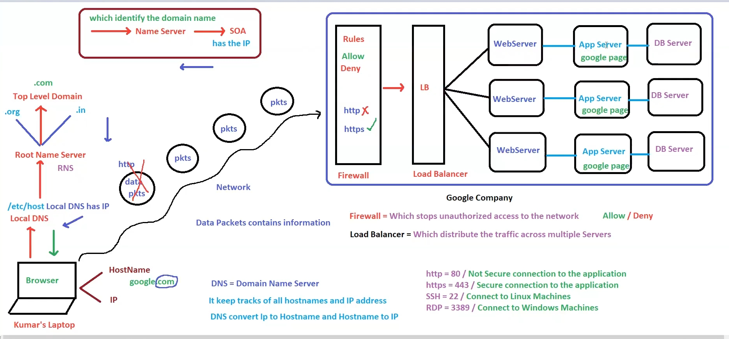
* 2 Tier Architecture

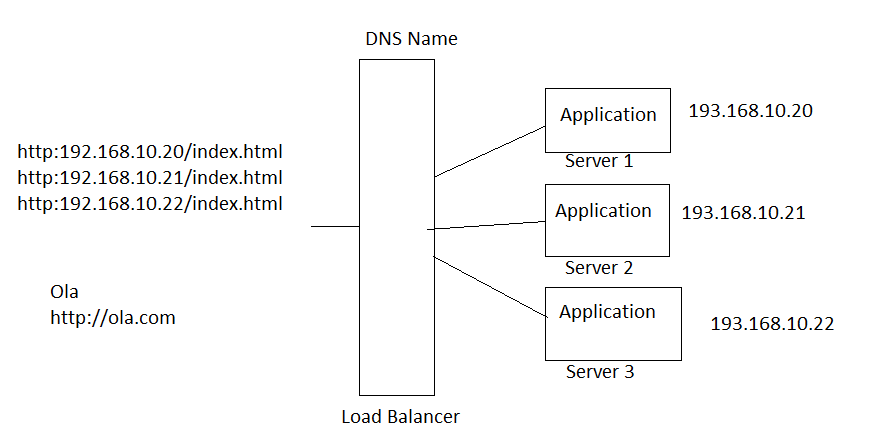


3 Tier Architecture



* Application Server = The server which has application hosted is called application server.
* Database Server = The server which has database installed, the server where the database is stored is called a database server.
* Device communicate with each other in the network with IP and Hostname.
* IP (Internet Protocol) = IP is unique identifier for a device in the network
* Host name = name of the server.
* Webserver = Takes the request and redirect to the application server.
* Application server should be always in private network, don’t expose to Internet

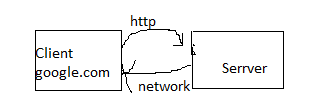




* Round robin method
* Load Balancer = which distribute the traffic across multiple servers

Protocols

* HTTP – Hyper Text Transfer Protocol.
* HTTP default port number 80.
* HTTP transfer the data to and fro from browser to server.



* For customer it should be always default port number (80 or 443).
* htpp://google.com - > protocol://domainname:portNumber
* <http://192.168.10.20:80> –wrong

<http://192.168.10.20:8080> – correct

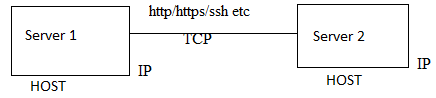
<http://192.168.10.20> – wrong

* You can customize the port number at application level but for customer, it should be always default port number.
* HTTP Status code

404 – page not found , 500 – Internal server error , 503 – service unavailable

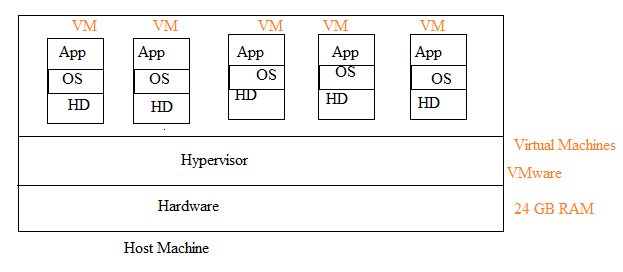
200 – page found, success

* HTTP – is not secure
* HTTPS is secure https = 443 certificates SSL/TLS/HTTPS – Encrypted.
* HTTP Over TCP/IP (Transmission control protocol)
* TCP establish the connection between 2 hosts.
* TCP is like a bridge.



DataCenters

Data Centers which are handled by us is called on-premises .



* P2V Migration (Physical to virtual)
* V2C Migration(Virtual to Cloud)

AWS

* AWS has global infrastructure AWS is providing infrastructure as a service , cloud is present in the remote location remote location contains datacenters datacenter contains infrastructure
* Infrastructure contains storage, database, servers, network, VM etc .
* AWS has approximate 36 place his infrastructure
* we need Internet to connect to the cloud
* Amazon Web Services access through Amazon management console
* AWS is a group of services AWS is a cloud provider who provide infrastructure as a service.

Cloud computing

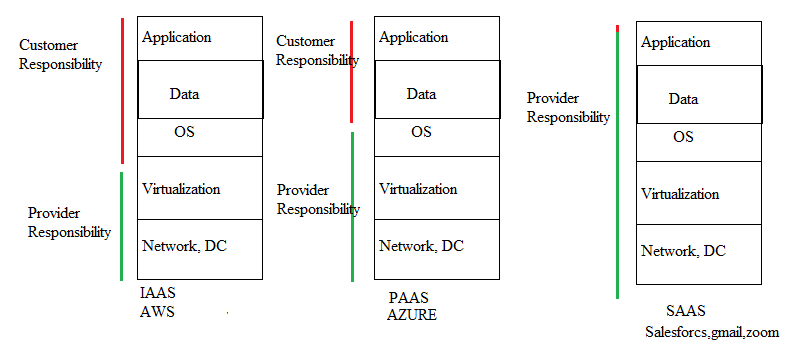
* Instead of doing computing on on-premises/ local machine, now you are doing computing in the remote location(cloud) that is called cloud computing.

Deployment model in cloud (Type of clouds)

* public cloud : - the providers services which are accessed by everyone like AWS, Azure, GCP etc.
* private cloud : - the providers services which are access within the organization like Oracle, IBM
* hybrid cloud : - the combination of public and private cloud.

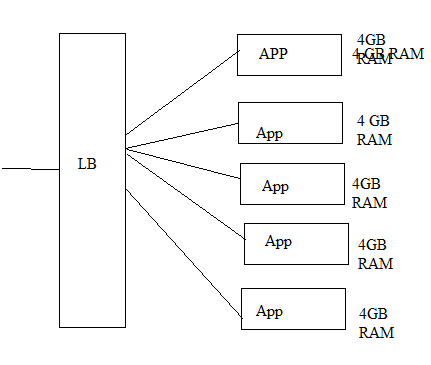
Service model

* Infrastructure as a service (IAAS)
* Platform as a service [PAAS]
* Software as a service [SAAS]
* AWS does not have any access inside your VM .
* ElasticBeanStalk = Easy and quick deployment of application in AWS.
* AWS work on share responsibility model.
* AWS is a group of service we can access AWS service through Amazon management console
* VM = instances
* EC2 = Elastic compute cloud.
* EC2 is AWS service where we can create virtual machines
* EC2 is aws service where we can launch AC2 instances.



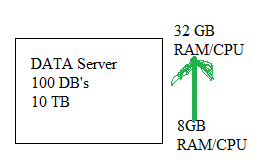
Elasticity

* Increasing and decreasing the number of servers/instances based on the load is called electricity
* Elasticity is a short term
* Elasticity can be achieved in AWS using auto scaling
* Auto scaling = scale out[increasing ,adding] and scale in[decreasing ,removing]
* Elasticity is also called as horizontal scaling
* Use the same capacity of the server is auto scaling



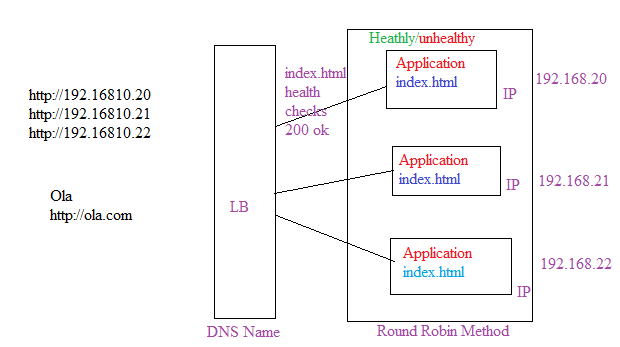
Scalability

* Increase the capacity of servers is called scalability
* Scalability = scale up and scale down
* Scalability is also called as vertical scaling.
* Scalability is long term.
* Scalability can be achieved in AWS by changing the instance type.
* Instance type = memory +CPU



High Availability

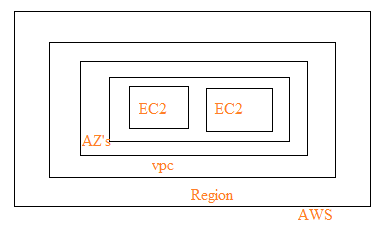
* The period of time the service is available to the customer is called high availability
* The period of time the service is not available to the customer is called downtime.
* Load balancer will do the health checks for application not a server
* Load balancer is doing the failover.
* Redundancy = duplicate/ having the same application on different servers
* Monitoring = load balancer will check if application is reachable or not using health checks
* Failover = if one server goes down, other server will take the request send by load balancer
* 0 downtime = auto scaling = fault tolerance



Regions and Availability Zones.

* Region = Its a geographical area, example AWS- Region = Mumbai
* Availability zone = Simply a Datacenter[AZ]
* AWS has global infrastructure
* Region is a place where AWS has its infrastructure
* A region has multiple data centers, A region has multiple DataCenters
* A Region has multiple AZ’s.
* Server = Instances Server/Instance are placed in AZ’s
* AZ’s are sync with each other[network], not a data
* Best practice is to distribute the instances across multiple AZ’s
* 1a or 1b or 1c = group of DataCentres
* 1 AZ’s is a group of DataCentres
* Instances across AZ’s can share the data if required as AZ’s are inter-connected with each other.
* Load balancer can distribute the traffic to multiple EC2 instance across AZ’s
* LB is a specific to region not a AZ’s
* Mumbai = ap-south-1 |||| AZ’s ap-south-1a, ap-south-1b, ap-south-1c
* Region and AZ’s are managed by AWS
* AZ’s can communicate with each other by default
* AZ’s network are inter-connected
* Region don’t communicate with each other by default, if required yes.
* EC2 instance is a specific to Region and AZ

VPC (Virtual private cloud)



* Every Region contains a default VPC
* 2 VPC’s will not communicate with each other by default , if required yes.

EC2 (Elastic compute cloud)

* In EC2 service, we can launch EC2 instances
* Servers = Instance / EC2 instance (VM’s)
* AWS service can be either regional or global
* EC 2 is a regional service.
* Load balancer = which distribute the traffic to multiple servers.
* Elastic Load Balancer(ELB) = ELB distribute the traffic to multiple EC2 instance across AZ’s.
* ELB is completely managed by AWS (HA,AS, scalability, performance etc
* ELB is a service from AWS not a server.
* You can’t login into ELB, you can access ELB with DNS name .
* ELB doesn’t have any AZ’s it is created at regional level.
* EC2 to launch easy , configure ,deployment, maintain.
* BeanStack - just upload the application and give some configuration.
* Elastic BeanStack = easy and quick deployment of application in AWS ,in general ,PAAS -> you don’t have any control on the servers
* Backbone of BeanStack EC2 instance.
* AWS BeanStack you have full control on EC2 instance launch by BeanStack
* BeanStack handle EC2 instance behalf of us.
* LightSail = if you want to setup and create a virtual lightSaill instance which already have everything installed and ready( WordPress, GitLab, Node jS, Joomla ) NO HA, NO AS, NO scalability.

Lambda

* Lambda is a Serverless
* You can run the code without server
* Lambda is used for automations
* Create function Lambda function created in java,python,.net, ruby etc.
* Lambda is invoked based on the trigger/ Event.
* All Event stored in the Event Bridge

S3

* In AWS all services will start with simple and end with service
* SNS - simple notification service.
* SES - simple Email service
* S3 - simple storage service
* S3 is unlimited storage by AWS, S3 is used to store the files, S3 can store any kind of files.
* With S3 we can upload, download, store the data and access your files.
* You cannot execute any files in S3. you cannot install, run execute any files in S3.
* S3 is serverless, AWS handle ha, performance, scalability etc for S3
* Bucket is a container objects
* Object is a file
* Name of file /object is a key
* S3 is a regional, bucket are regional.
* S3 supports static website hosting -> create a bucket and upload all the HTML files and enable static website hosting. no need to worry about HA, performance, Salability etc because S3 handles it.
* S3 is a object base storage

Laptop - S3

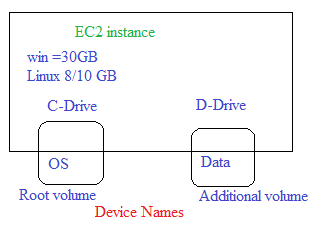
Folder - Bucket

File - Objects

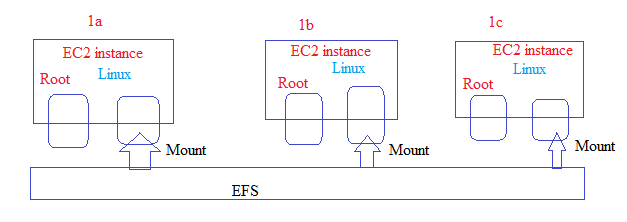
pushpa .MP4 - key

EBS- (Elastic Block Storage)

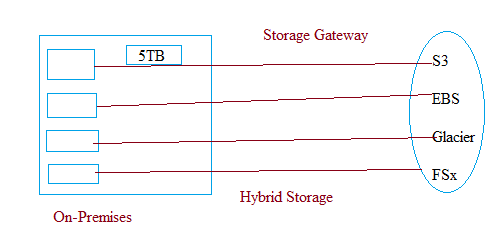
* Hard Disk = volume = EBS volume
* EBS is a block based storage
* Volumes can be attached and detached. you can attach multiple volumes to the EC2 instance.
* EC2 instance has default volume, that volume is called ROOT volume.
* The root value always contain OS[windows, Linux]
* EC2 supports only server side OS not client side OS.
* If you have OS on the volume, the volume is called root volume
* EC2 instance can have only 1 root volume. EC2 instance can have multiple additional volumes.
* Maximum size of the EBS volumes is 16TB.
* Volume should be pre-provisioned like 50GB, 100GB …Max 16 TB
* You cannot attach volume to a multiple EC2 instance at the same time\*\*\*
* Volume size can be increased on FLY [no need to stop the EC2 instance, no downtime]
* Volume size can’t be decrease[delete the volume and re-create is based on requirement]
* Q - It is possible to detach the root volume while EC2 is running -> No stop the EC2 instance first and then detach the root volume.
* Q- Is it possible to detach additional volume while EC2 is running - >Yes it is not recommended to detach while running. Stop first.
* We can not attach 1a volume to 1b EC2 instance(diff Az)
* We can attach 1a volume to 1a EC2 instance(same AZ)
* You can’t share the volumes among EC2 instances
* For win/Linux -/dev/sda1 /dev/sdb, e f etc
* Root volume is always mounted/attached as /dev/sda1
* EC2 instance has AZ,volumes also has AZ
* EC2 instance and volume should be in the same AZ
* EBS is regional .



EFS = Elastic File System



* EFS is completely managed by AWS, EFS is only for Linux EC2 instances
* FSX is for Windows EC2 instances
* EFS works with NFS protocol
* EFS is a file based storage
* EFS is unlimited storage
* EFS does not require any pre-provisioning[ it will automatically increase and decrease the base on the data put in EFS]
* EFS can be mounted to multiple EC2 instances at the same time across AZ’s.
* EFS can be replicated to other regions
* Is it possible to attach a single volume to multiple EC2 instance at the same time? No



Snow family

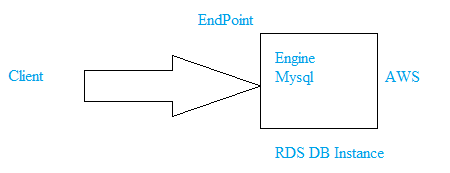
* SnowCone - 8TB
* SnowEdge - 100 TB
* SnowMobile PB’s
* snow family is used to transfer huge data from on-prem to AWS and vice versa
* SnowFamily is a physical data transfer using devices

Database service

RDS- Relational database service.

* RDS is completely managed by AWS.
* RDS is a service where we can setup, configure and maintain RDBMS databases.
* RDS is a database service not a database.
* RDS DB instance, RDS support only RDBMS database only.
* RDS support 7 engines

1. MYSQL
2. Oracle
3. MSSQL
4. PostGresql
5. MariaDB
6. Aurora
7. IBM DB2



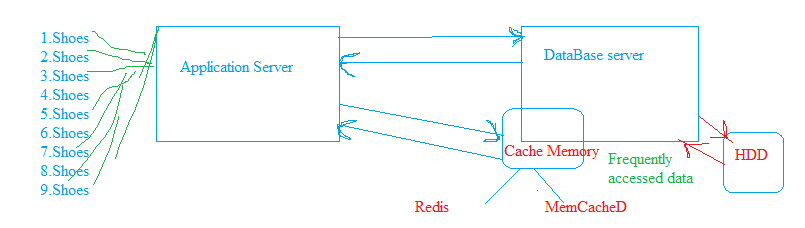
DynamoDB - NOSQL database service[Non-relational]

* Database is used to store the data.
* DataWarehouse it is used to store huge data.
* RedShift = DataWarehouse in AWS

DMS - Data Migration Service

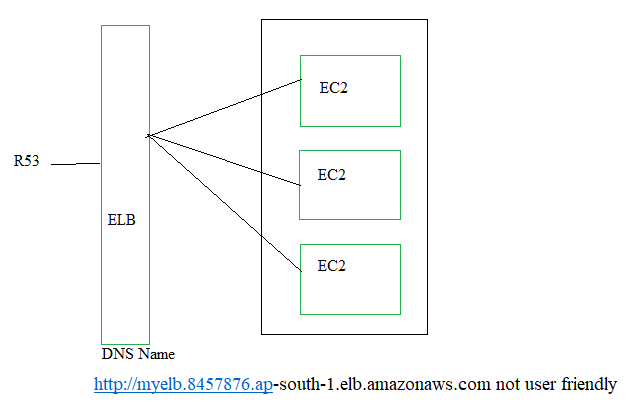
ElasticCache = In memory database caching service

Low latency, High Performance



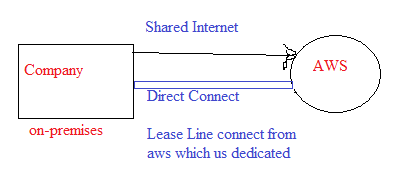
Route53

* R53 is DNS service from AWS, DNS port number is 53
* R53 contains records
* In R53 we do mapping
* R53 is a global



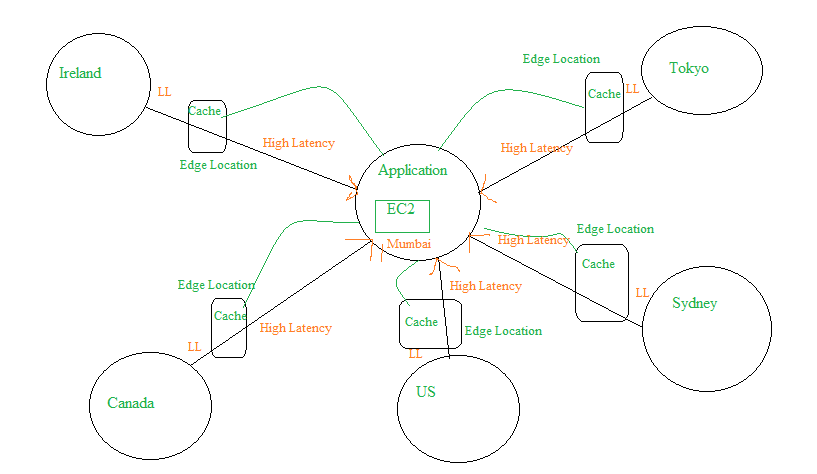
VPC - virtual private cloud

* It is like a virtual datacenter on cloud
* Every region has a default VPC
* VPC is a regional



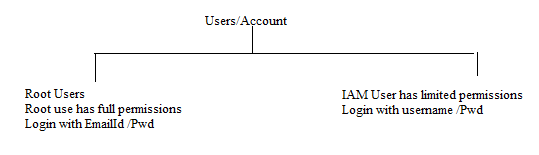
CloudFront

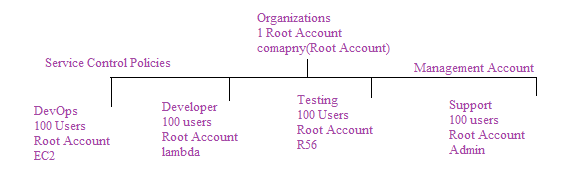
* CDN = Content Delivery Network
* Without CF - High latency -> user-boom.com- R53-ELB-EC2
* With CF - low latency
* User-boom.com-R53-CF(Edge locations) – ELD-EC2
* TTL = time to live example 12 hours
* Invalid cache = delete the cache, it will be catched based on TTL value
* EL’s are manage by AWS
* Edge location = application catching, static and dynamic data
* Create distribution = where is your organization? EC
* North America, Europe, Asia
* Use all edge locations
* Create distribution



1 IAM - Indentity and Access Management

* You can control the entire AWS using IAM by providing proper permissions to the I am users
* Service quotas - show AWS resource limit
* SNS – simple notification service, SES - simple email service, SQS - simple queue service
* Trusted advisor
* AWS inspector
* Every AWS account has AWS account number /AWS ID 12 digit.
* Organizations = To manage multiple root accounts for better management





CloudWatch

* CloudWatch is used to monitor all AWS resources[EC2, ELB, S3, RDS etc]
* In CloudWatch we create alarms to monitor AWS resources, cloudwatch monitors the performance
* Basic monitoring = you will get the data points every 5 minutes ,free ,default
* Details monitoring = you will get the data points every one minute, billable

CloudTrail = monitor entire AWS account

* Record, monitor, track, audit, logs etc = investigation purpose

Config = monitors the changes in AWS resources

Secret manager = It is used to store secret [credentials keys etc]

AWS backup= centrally manage and automate backup

WAF = web application firewall

protect your web application from common web exploits

AWS shield = managed DDoS protection secure

AWS support

* Basic support = free
* Developer support
* Business support
* Enterprise support

Security

Encryption can be done in two way

* Encryption In-Transit = while data is travelling
* HTTPS
* Certificates
* ACM = Amazon certificate manager
* Encryption at Rest = while data is resting ,
* keys
* Encryption keys
* KMS = key management service

AWS Services Deep Drive

IAM

* IMI is used for security purpose, with IAM, you can control entire AWS resources centrally by giving proper permission.
* Don’t share your e-mail or password to others
* You can share the ROOT account by creating IAM user .
* IAM is a global. IAM is a free.
* With IAM user, we can access AWS console and services.
* It is not recommended to use a root account for a daily activity or work instead IAM user.
* MFA = multi factor authenticate
* MFA is a high recommended for root and IAM account
* We need to set up MFA for every individual IAM user
* Open AWS page -> login with e-mail and password -> MFA code -> login to AWS console.

Two way to access

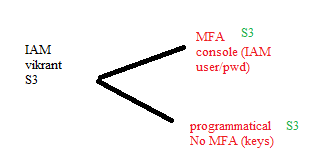
1. Console access: AWS console [GUI]

[Email / password -> root] or [username / password –

>IAM ]

1. Programmatical Access[CLI, SDK’s , Developer tools]

* We need to install AWS CLI software in Linux and windows[ CMD] for java, .net, Python etc we need to install AWS SDK’s
* Authentication can be done on programmatically access using [Access key and secret key]
* “aws configure” is used to configure AWS CLI on windows CMD and on Linux
* keys are user specific, individual IAM user have their own keys
* It is not recommended to share the keys to anyone
* Create the keys based on the requirement, don’t create it unnecessarily.
* KEYS also have the same permission like console
* Every IAM user can have max 2 sets of keys
* Best practice is to rotate the password and keys for every certain period of time.



* Keys(secret) is visible only for 1 time while creating
* Once keys are lost it is lost. you cannot get the same keys back.
* But we can regenerate you will get the new keys, you cannot get the old keys back .
* Don’t create and use the keys off root account.

IAM Group

* IAM group = collection of IAM users.
* Group under group are not possible/ nested group are not possible.
* It is possible to attach multiple policies to the IAM user and IAM group maximum 10.
* You can add and remove policies to IAM user and group anytime.
* If you attach any IAM users to the IAM group, IAM user individual permission will not be lost, group level permission will be inherited
* An IAM users can be attached to multiple IAM groups at the same time
* For new IAM user, by default there are no policy attached
* IAM groups are used to assign policies to bunch of IAM users at the same time

User base permission : - EC2 full access, S3 read access -> manage

Resource level permissions:- Granular level / deeper level -> customize permissions = inline.

* Policies contains permissions
* Policies / permissions are written in Jason format
* AWS has a policy editor or policy generator, this will help to generate Json code automatically.

Manage policy = create and manage by AWS [predefined policy]

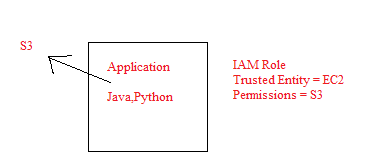
Inline policy = created and managed by customer [customer manage policies]

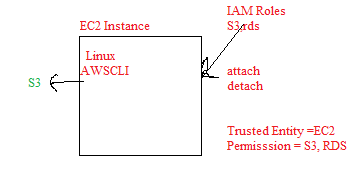
ARN = Amazon Resource Name

* ARM will be used in Json policies
* AWS resource has ARN
* IAM user ARN = arn::IAM:268958:user/vikrant

IAM Roles

* ROLES = Temporary access without credentials
* If you use the IAM roles, you no need to configure keys on the machines.
* Based on the permissions that you have attached to the role, those permission are available from the instance
* One EC2 instance can have only one role attached at a time
* 1 role can be attached to multiple EC2 instances
* Two AWS services will not talk to each other by default, we should use roles.
* IAM roles can be attached to any AWS services





Instance IP

Instance username

Instance password

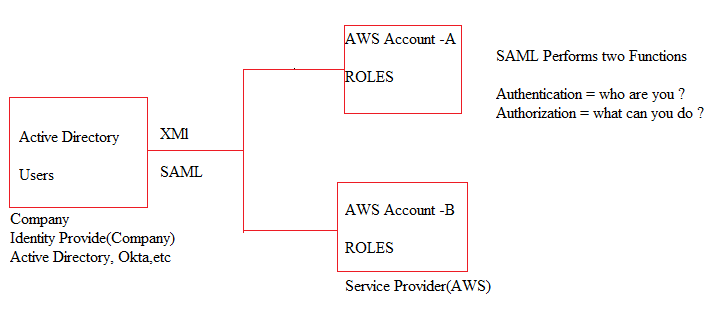
* IAM user is used to access AWS console and service but not log to EC2 instance
* If you don’t configure keys on the Linux EC2 instance, you cannot access AWS service[S3]
* If you configure the keys on the Linux EC2 instance, keys are stored locally on the instance, which is not safe and not secure.
* Example : - If Lambda want to stop the EC2 instance

Create a role and give the permissions

* Trusted entity = Lambda[To whom you are attaching the IAM role]
* Permissions = EC2 [what kinds of permissions role should have]

What is SAML

Security Assertion Markup Language 2.0( SAM L) is a open federation standard that allow an identity provider( idp) so authenticate users and pass identity and security information about them to a service provider [SP] this information is sent in XML document .



1. Login to your normal AWS account [either IAM[admin] or root account] add enable organization. once you enable organizations, this account will become management account.
2. Go to organization, and invite AWS account[member accounts]- member account will get the invitation on email. accept the invitation. once accepted you will become the member account.
3. Now in your organizations, you have a management account and member account ready.
4. You can control them these member account using SCP’s
5. In management account, go to the identity center and create users and keep the credentials in the notepad.
6. Assign the user to member account, click on AWS account in identity centre -> select member account -> assign users or groups -> select user -> create permission set[policy] -> submit.
7. Management account will give you the URL [AWS access portal URL]
8. User will access this URL and login to the member account with the user created in identity centre without IAM users in member account