**LECTURE I**

Regions, Data Centers, Availability Zones (mostly 3 per Data Center)

Regions are geographical locations. Data centers are locations where the infrastructures are built physically. These datacenters are referred to as availability zones.

Infrastructure are Region-scoped, but Identity Access Management IAM is Global service.

Take a look at Amazon Global Infrastructure

1. Take a detailed look at IAM

*IAM allows you to* ***manage users and their level of access to the AWS console****. It is synonymous to creating user accounts on the Windows server and managing the behavior of these users via the group policy management console of the Active Directory.*

*IAM offers the following features:*

* *Centralised control of your AWS account*
* *Shared access to your AWS account*
* *Granular permissions*
* *Identify federation i.e. you can use account in other platforms like AD, Facebook, LinkedIn to log in.*
* *Multifactor Authentication i.e. you log in with a name, password and a special code from a Multifactor application like Authy etc.*
* *It allows you to set up password rotation policies.*
* *Supports PCI DSS compliance i.e. frameworks that are complaint with entering credit card details.*
* *Provide Temporary access for users/devices and services where necessary. E.g. logging into the AWS console using your mobile phone.*
* *It integrates with many AWS services*

***KEY TERMINLOGIES IN IAM***

* ***USERS: These are the end users such as people, employee in an organization etc***
* ***GROUPS: A collection of users. These users inherit the permissions of the group.***
* ***POLICIES: They give permissions to as what a user/group/Role is supposed/allowed to do***
* ***ROLES: These are created and assigned to AWS Resources.***

***N.B Accounts created on IAM are “global” accounts. I.e. they are not locked down to a specific region. BUT their resources are geographically bound.***

***What you have learnt so far?***

* ***IAM is universal. It does not apply to regions at this time.***
* ***The "root account" is simply the account created when first setup your AWS account. It has complete Admin access.***
* ***New Users have NO permissions when first created.***
* ***New Users are assigned Access Key ID & Secret Access Keys when first created.***
* ***Access key ID & Secret Access Key are not the same as a password. You cannot use the Access key ID & Secret Access Key to Login in to the console. You can use this to access AWS via the APIs and Command Line, however.***
* ***You only get to view these once. If you lose them, you have to regenerate them. So, save them in a secure location.***
* ***Always setup Multifactor Authentication on your root account.***
* ***You can create and customise your own password rotation policies.***

1. Create your first EC2 Machine and login

To create your EC2 machine

3 ways to login to your Machine

* MobaXterm (from Windows Machine)
* Instance Connect (direct)
* SSH (from Linux Machine)

1. Create a new account

Create IAM user accounts for all my students

Login and practice

Navigate EC2 in details

*SSH using Putty*

* Download Putty from the internet, then Install and configure
* Right click installed Putty, click Putty Gen
* Click Load
* Pick your .pemKeypair
* Do NOT click Generate. Simply click “Save **Private** Key”

Load Putty

* Hostname or IP address: copy and paste IP address from ***EC2 Management Console on AWS***
* Connection/Data/Auto Login Username: ALWAYS “ec2-user” for IAM users.   
  **Root user not acceptable**
* SSH/Auth/Browse for your saved**PrivateKey**)
* Session/Saved Session
* Open

**Install and configure Apache Web Services on your EC2**

1. ***sudosu***
2. What is the name of the service package you want to configure? ***httpd***
3. Is it already install? ***rpm -aq | grephttpd***
4. If not, install it ***yum install httpd -y***
5. Restart the service so the new configuration you just completed will load: ***$ systemctl restart httpd***
6. So you don’t have to restart the service everytime you boot the system, enable it ***$ systemctl enable httpd***
7. **$ curl localhost:80**
8. ***Browse using Public IP Address*** since there’s no A record for this server and Hostname cannot resolve. But you will not be able to browse because of Firewall.
9. Edit your ***Security Group***to open port 80 (allow HTTP). Now you should be able to browse…
10. ***echo “Hello World$(hostname -f)”>/var/www/html/index.html***

copy your public IP address and paste on your browser. You should be able to browse.

Remember to allow HTTP in your Security Group, else you will not be able to browse.

**OPTIONAL**

EC2 **Userdata** is used for **bootstrapping** (automate boot tasks)

***Shebang is the first EVER 2 characters of (shell) scripts***

Bootstrapping

**#!/bin/bash**

**sudosu**

**yum update -y**

**yum install httpd -y**

**systemctl restart httpd**

**systemctl enable httpd**

**echo "Working from home $(hostname -f)" >/var/www/html/index.html**

Explain Scalability (Vertical (upscale/downscale) and Horizontal (Elastic) and Availability (Active (replications) and inactive (backups &failovers)

Load Balancers

Application, Network, Gateway

**SCALABILITY:** This is the ability of a cloud service to increase or decrease existing resources as needed based on the workload demand. It is an essential feature of the cloud.

Systems have four general areas that scalability can apply to:

* Disk I/O
* Memory
* Network I/O
* CPU

When scaling a system vertically, you add more power to an existing instance. This can mean more memory (RAM), faster storage such as Solid State Drives (SSDs), or more powerful processors (CPUs).

In the event of increasing IT resources on the cloud, this is referred to as UPSCALING and the event of decreasing is termed DOWNSCALING.

**ELASTICITY:** This is the ability of a cloud service to grow or shrink dynamically in response to changing workload demands, like a sudden spike in web traffic.

When scaling your systems horizontally, you generally add more servers to spread the workload across multiple machines.

***.***

AVAILABILITY: This is the percentage of time or period that a data service or system is available. AWS boasts of high availability. Up to 99.999999% of its system are available. To ensure high availability, cloud services are either configured to be active or inactive.

ACTIVE (REPLICATION): This type of availability configuration enables data residing on a cloud instance (primary instance) to be continuously replicated or copied to another cloud instance (standby instance). Organizations replicate data to support high availability, backup, and/or disaster recovery.  This replication can either be synchronously or asynchronously based on the location of the secondary instance. How the data is replicated impacts Recovery Time Objectives (RTOs) and Recovery Point Objectives (RPO).

INACTIVE (BACKUP OR FAILOVER): In this kind of availability configuration, data is copied to a standby (redundant) secondary instance and in the event of the primary instance failure, there is automatic switch to the standby secondary instance.

LOAD BALANCERS:

Load balancers helps to efficiently distribute workload (traffic) across a group of servers in the cloud. (Server Pool).

A [load balancer](https://www.nginx.com/solutions/adc) acts as the “traffic cop” sitting in front of your servers and routing client requests across all servers capable of fulfilling those requests in a manner that maximizes speed and capacity utilization and ensures that no one server is overworked, which could degrade performance. If a single server goes down, the load balancer redirects traffic to the remaining online servers. When a new server is added to the server group, the load balancer automatically starts to send requests to it.

In this manner, a load balancer performs the following functions:

* Distributes client requests or network load efficiently across multiple servers
* Ensures high availability and reliability by sending requests only to servers that are online
* Provides the flexibility to add or subtract servers as demand dictates

**LECTURE 2**

Create 2 Instances in 32different AZs within your Region. Note the labels of your AZs

Create an ALB and make your 2 Instances the target group for your Application Load Balancer.

*Note: Enable only HTTP for this task. We will practice later with the HTTPS protocol*

*A* ***Cloud Architect*** *is responsible for converting the technical requirements of a project into the architecture and design that will guide the final product. Often, Cloud Architects are also responsible for bridging the gaps between complex business problems and solutions in the cloud. Other members of a technology team, including DevOps engineers and developers, work with the Cloud Architect to ensure that the right technology or technologies are being built.*

**Class practice**

Create an Application Load Balancer, ALB, and create Auto Scaling Group to generate 3 instances in 3 separate Availability Zones to serve as TG for the ALB. In creating the Target Group TG, you must use LAUCH TEMPLATE to define and save your “ideal Server” from which your ASG will generate new EC2s in future.

* Test that the ALB is working fine and meeting all expectations with the 3 Instances in the Target Group
* Set necessary parameters to AUTOMATE Scale-in and Scale-out of the Auto Scaling Group and test to confirm that your configuration works.

**ANSWER**

* Create your typical Instance. Browse to test that your instance works
* Create Launch Template to automate future EC2 Instance creation. Test your Launch Template
* Create ASG across 3 AZs
* Create ALB and attach it to the ASG
* Browse the ALB DNS to test that it is connecting alternately to the 3 EC2s across the 3 AZs
* Set Policy for the ALB to trigger the ASG to scale-out or scale-in. If changing policy will not be effective because of NO-TRAFIC, then try reducing the DESIRED instances from 3 to 1 to see the effect.
* Test your configurations

By the way, how does the ASG decide which Instance to REMOVE when scaling in?

*It will look for the subnet with the highest number of Target Groups, and remove the oldest version of Instance.*

**What is Elastic Block Storage, EBS**

They are attached network drives, not Instance store (ephemeral storage)

They are AZ-locked, detachable, increasable,

Types: GP1, IO1 (SSDs), ST1, SC1 (HDDs)

To add EBS vol, SSH into EC2 and invoke your Linux skills (this is SysOps role)

How do we migrate data from our Server (EC2 instance) in one AZ to another Server in another AZ?

To migrate data from server in one AZ to another

Using EBS Snapshot

* Take a snapshot from the source volume
* Create a volume from the snapshot
* Ensure vol is accessible in the destination AZ
* IF you need to, create new instance in destination AZ
* Attach volume to the destination instance

**What’s RDS (Engine)**

In RDS, What’s the difference between Read Replicas and Multi AZs  
(Note: Read replicas are createdin Multi AZs)

RDB Hands On

Download and install **SQLectron** to test your RDB connection

RDB Security can be achieved by

* At Rest Encryption at Master OR Read Replicas with AWS KMS – AES 256 Encryption
* In Flight Encryption using SSL/TLS
* RDS are usually deployed within a private subnet, not a public one
* RDS security works by leveraging Security Groups, just like EC2s
* Use IAM policy to control you can manage the RDS (through the RDS API)

How do we encrypt/unencrypt a DB?

HANDS ON

Features of Aurora

* Backup and Recovery
* Industry compliance
* Push button scaling
* Advanced Monitoring
* Backtrack restore data at any point of time without using backups
* Automatic fail-over
* Isolation and security
* Routine Maintenance
* Automated Patching with Zero

**Lecture 3**

**ROUTE 53**

**mypage.mycloud.com to yourowndomain.com cname**

**ALB to mycloud.com Alias**

Route53 = DNS = 4 records

**A Record:** Resolves hostname/domain name to IPv4

**AAAA***(quadruple A)*Record: Resolves hostname/domain name to IPv6

**CName:**hostname to hostnamee.g maps your subdomain to someone else’s full domain so you can host him

Use CName to point*non-root* record like***teaching*.mycloud.com** to a**abc.com**

**Alias:** hostname/DNS to AWS resourcee.gS3, ALB map to **mycloud.com**

…to point *root record* or *non-root record* to aws resource like **teaching.com** to **EMB endpoint**. Alias is free of charge to use and support native health check.

**mypage.mycloud.com to yourowndomain.com cname**

**ALB to mycloud.com Alias**

Route53 resolves public domain name like teaching.com; but it can also resolve private domain name like teaching.internal or hostnames like dc1, to ip addresses, forward the ip address to the client for onward (direct) connection.

**#!/bin/bash**

**sudosu**

**yum update -y**

**yum install httpd -y**

**systemctl restart httpd**

**systemctl enable httpd**

**echo "Working from home $(hostname -f)" >/var/www/html/index.html**

* Using the above user data, create 3 different servers in 3 different AZs
* Once done, click the regions one by one and write out the Server names and IP address for easy reference
* Browse the server ip addresses or dnsto confirm that the webpages are loading fine
* Now create A records with above IP addresses and test them on your browser
* Create an Application Load Balancer in one of the AZ to link the the EC2 instance in another AZ
* Use the Application Load Balancer to create and test Alias records and CName records

Demonstrate how to register new domain

To test in **Windows** if your domain name is resolving to it’s assigned A record: ***nslookup*** dns name

To test in **LINUX**, it would be ***dig*** dns name

Use the above to test TTL by pointing the A record to a different dns while you wait for the TTL to expir

**LECTURE 4**

**S3: All practical**

**S3: Acronym for simple storage services:** S3 provides developers and IT teams with secure, durable, highly- scalable object storage. i.e it is the safest place to store your files. The data stored is **spread across multiple** devices and facilities. It is an **unlimited** storage and allows a file size from **0B to 5TB**. Files are stored in a **Bucket.**

Amazon S3 is easy to use, with a simple web services interface to store and retrieve any amount of data from anywhere on the web.

Since S3 is a universal namespace, the names used on it must be unique globally. The reason for its uniqueness is because it immediately and automatically creates a web link address to be accessed globally.

Objects (files etc.) stored in S3 contains the following:

Key: the name of the object

Values: the data (sequence of bytes)

Version ID: Used for versioning of files.

Metadata: This is the data about the data you are storing

**N.B: When you upload a file to S3, you will receive a successful HTTP 200 code, signifying that the upload was successful.**

**DATA CONSISTENCY IN S3**

* Read after Write consistency for PUTS of new Objects(i.e.If you write a new file and read it immediately afterwards, you will be able to view that data.)
* Eventual Consistency for overwrite PUTS and DELETES (I.E. If you update AN EXISTING file or delete a file and read it immediately, you may get the older version, or you may not. Basically changes to objects can take a little bit of time to propagate.)

S3 has the following features;

* Tiered Storage Available
* Lifecycle Management
* Versioning
* Encryption
* MFA Delete
* Secure your data using **Access Control Lists** and **Bucket Policies**

**S3 Storage classes:**

**S3 Standard:** This class boasts of 99.99% availability and 99999999999% durability. It is stored redundantly across multiple devices in multiple facilities, and is designed to sustain the loss of 2 facilities concurrently.

**SS-IA (Infrequently Accessed):** This is for data that is accessed less frequently, but requires rapid access when needed. It has lower fee than S3, but you are charged a retrieval fee.

**S3 One Zone - IA:** In a case where you want a lower-cost option for infrequently accessed data, but do not require the multiple Availability Zone data resilience.

**S3 - Intelligent Tiering:** This is designed to optimize costs by automatically moving data to the most cost-effective access tier, without performance impact or operational overhead.

**S3 Glacier:**S3 Glacier is a secure, durable and low-cost storage class for data archiving. You can reliably store any amount of data at costs that are competitive with or cheaper than on-premises solutions. Retrieval times configurable from minutes to hours.

**S3 Glacier Deep Archive:** This is Amazon S3's lowest-cost storage class where a retrieval time of 12 hours is acceptable.

**When using the S3, a user is charged in the following ways;**

* Storage
* Requests
* Storage Management Pricing
* Data Transfer Pricing
* Transfer Acceleration
* Cross Region Replication Pricing

**AWS CLI, SDK, IAM Roles & Policies**

We have covered AWS Architecting in **Console(GUI)**. Take a peep into Development in **Programmatic (CLI)**.

*Google “aws cli installation in…” and follow instructions.*

* **Windows:**

***Local side***

Google and download, and installAWS CLI MSI 64-bit version

To confirm the installation, use the aws--version command at a command prompt.

***To ACCESS/CONNECT your AWS CLI on Local Windows OS TO your EC2***

* + Enter “aws configure”
  + your access key,
  + your secret key,
  + your Region (=us-east-1)
  + Press enter again.
* **Linux (Fedora, RHEL, CenOS):**ubuntu

To install…

*$ sudo***curl "https://awscli.amazonaws.coam/awscli-exe-linux-x86\_64.zip -o awscliv2.zip"**

*$ sudo***unzip awscliv2.zip**

*$ sudo***sudo ./aws/install**

*$ sudo***aws–version**

After successful installation, you can now

Enter “**aws configure**”

* + *your access key*
  + *your secret key*
  + *your Region*
  + *enter again to accept Default Output Format*
* **EC2:** create and attach a IAM Role (never ever enter your personal credential on EC2)

***Now Practice Commands***

* *aws s3 mb s3://james*
* *aws s3 ls*
* *aws s3 ls s3://james*

Other S3 commands you can practice:

* cp - copies a local file or S3 object to another location locally or in S3.

***aws s3****cp filename s3://bucketname*

* ***aws s3 l***s = list
* ***aws s3***mv – move sa local file or S3 object to another location locally or in S3.
* ***aws s3***rb – removes/delete bucket
* ***aws s3***rm – delete s3 OBJECT
* Generate policy in json

**NETWORKING IN CLOUD – VIRTUAL PRIVATE CLOUDS – VPC**

IP ADDRESSING AND SUBNETTING

PROJECT IN VPC

[**https://www.ipaddressguide.com/cidr**](https://www.ipaddressguide.com/cidr)

Classification of IPs by the Internet Assigned Numbers Authority, IANA

***Reserved for Private IP Addressing***

* 10.0.0.0 – 10.255.255.255 **(1.0.0.0/8)**: Used for big networks
* 172.16.0.0 – 172.31.255.255 (172.16.0.0/12): Default AWS IPs
* 192.168.0.0 –192.168.255.255 (192.168.0.0/16): Used in Home and Small Networks

All other IPs are public (Routable)

Of all available IP addresses within a subnet, 5 are always reserved for:

10.0.0.0 Network Address

10.0.0.1 Router

10.0.0.2 Mapping

10.0.0.3 Future Use

10.0.0.255 Broadcast (not supported on VPC)

**If you need 29 IP addresses for your subnet, is it OK to use /27? If not, what CIDR will be appropriate?**

(32-27) = 5

2 power 5 = 32 Hosts = 27 host in practice

2 power 6 = 64 Hosts = 64-5 =59 host in practice

/27

255

Remember to unmount all your installed infrastructure to avoid continued charges as we are already being charged for some of our configurations

Identify *the*VPC on **VPC Dashboard**, and then Delete Network Infrastructures in this Order

* Terminate EC2s
* NAT Gateway
* Detach Internet Gateway from VPC
* Delete Internet Gateway
* Dissociateany EIPs
* Release any EIP
* Delete VPC. This will delete all associated subnets, Security Groups and Route Tables