

AWS for Solution Architects

Techlanders Solutions

Raman Khanna

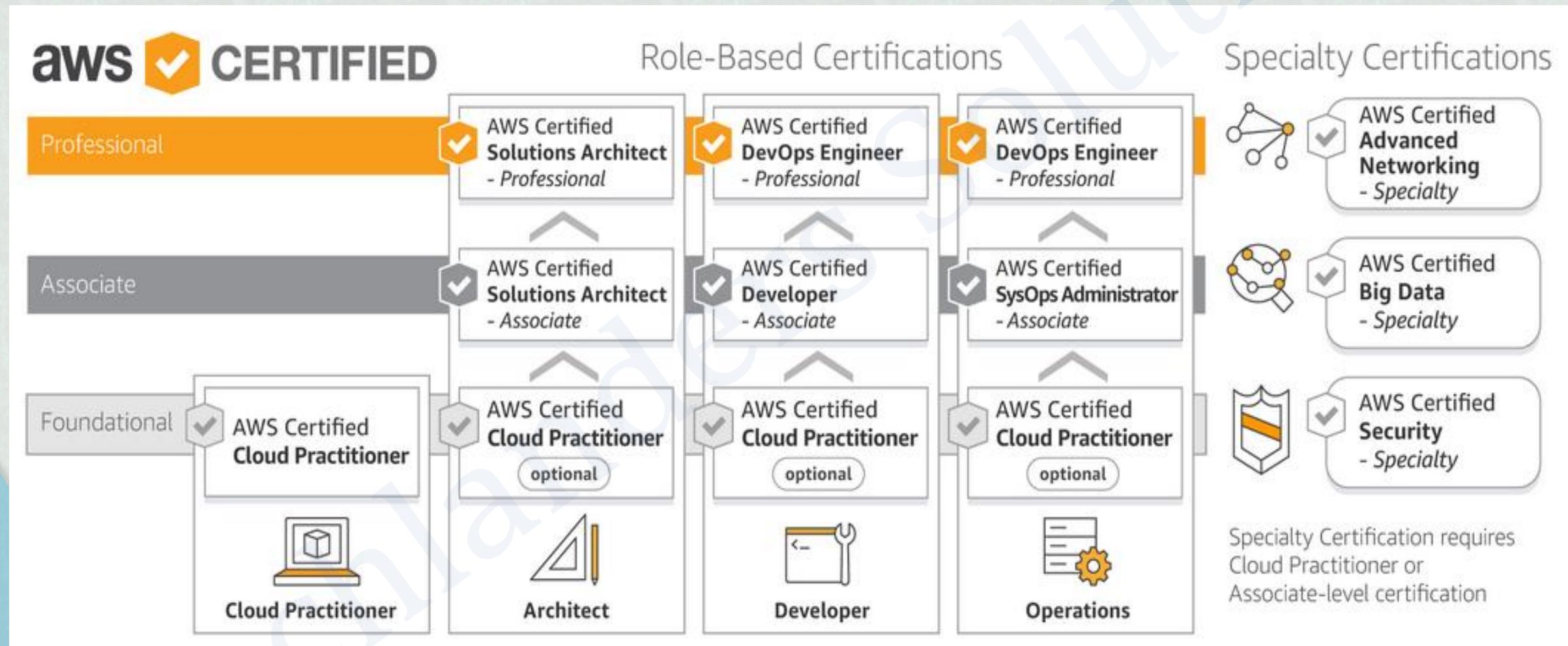
Introduction

Your Name

Background – Development / Infrastructure

Learnings on cloud/AWS

AWS Certification Roadmap



Introduction to Cloud Computing

What is Cloud?

Introduction to Cloud Computing

In simple words, Cloud computing is – Placing your data on someone else's datacenter, letting them manage underline hardware Infrastructure (optionally underline Database or applications too); while having your full control on the data, and accessing that data through Internet or dedicated network.

Cloud computing is a model for enabling **universal, on-demand** access to a **shared pool** of configurable computing resources (e.g., computer networks, servers, storage, applications and services), which can be **rapidly provisioned** and **released** with **minimal management effort** on **Pay-per-use basis**.

Free available cloud Examples:

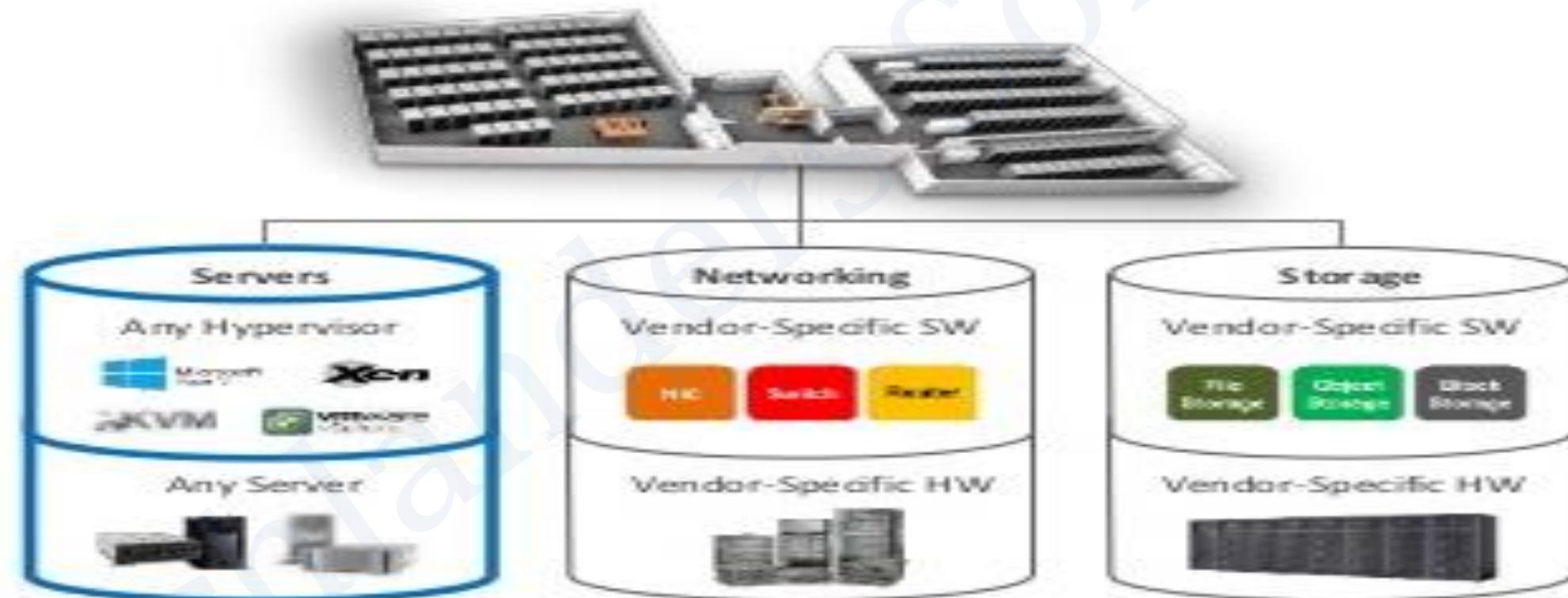
Gmail, IRCTC, WhatsApp/Facebook

Paid available cloud Examples:

AWS, Azure(Microsoft), Oracle Cloud

Traditional DataCenters

Traditional Data Center



Traditional DataCenters

- Main issues with Traditional IT Infrastructure.
 - Infrastructure is not a core business
 - Hard to Scale
 - Dedicated Infrastructure teams
 - Dedicated Datacenters
 - Dependency on vendors (servers, switches, cables etc.)
 - Underutilized Resources
 - High Cost
 - Difficult Capacity Planning
 - On-Spot demands were hard to manage
 - Provisioning resources was very time consuming

Why cloud?

- To overcome all of the discussed challenges, IT infrastructure domain drifted towards Service based model which is a real “cloud computing”
 - No Dedicated Datacenter
 - No Different Infrastructure Teams
 - Higher/Faster Scalability
 - Elasticity
 - Pay per use model
 - Option to adopt high availability
 - Better performance
 - Instant provisioning
 - Optimized use of resources
 - On demand scaling to any extent
 - No to worry about capacity planning

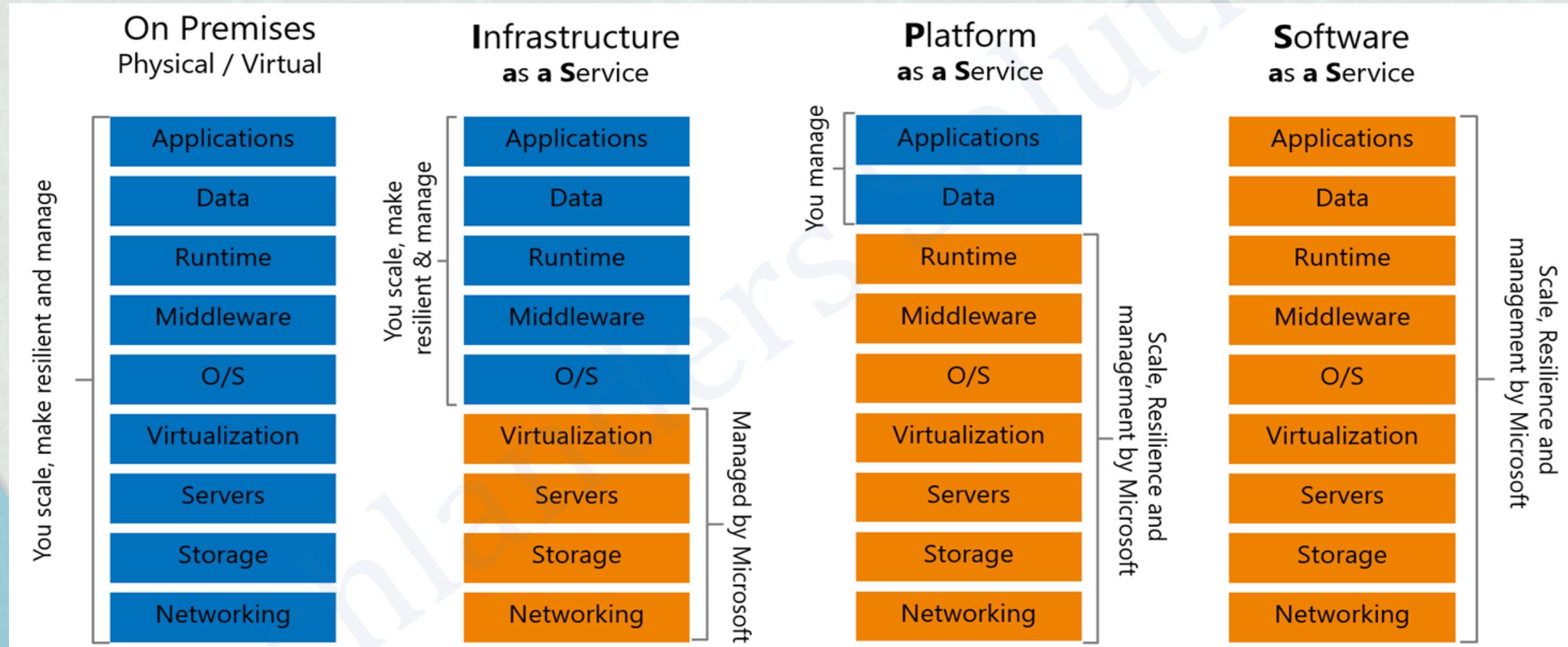
Cloud Advantages



Cloud Service Models

- There are three Cloud Computing Service Models:
 - Infrastructure as a Service (IaaS)
 - Platform as a Service (PaaS)
 - Software as a Service (SaaS)

Responsibility- Who owns What?



Responsibility- Who owns What?



Cloud Service Models - IaaS

IaaS is the most basic Cloud Service Model

It offers Underline Infrastructure for Compute, Storage and Networking

Infrastructure can be selected by customers as per their choice and Pay-per-use model.

- Examples: Bare metal servers, virtual Instances, Load balancers

IaaS - Benefits

- Drastic reduction in capital investment
- Easily Scalable
- Pay only for the used resources
- High Flexibility
- Reduced infrastructure support teams

Cloud Service Models - PaaS

Another service model, where cloud provider manages the OS & middleware part, along with IaaS

Provide capability to deploy applications on cloud infrastructure without managing underline Infra

Consumers are responsible for managing deployed applications and their environment specific configurations

- Examples: webservers and databases

PaaS - Benefits

- Includes all IaaS benefits
- No upfront licensing cost
- More reduction in Infrastructure support team
- Rapid time to market

Cloud Service Models - SaaS

SaaS deliver complete application to the consumers over the internet.

Consumers are not responsible for managing any application or underlying infrastructure.

SaaS application are delivered as “one-to-many” model.

- Examples: office365, Gmail, WhatsApp, JIRA, GIT, Service Now

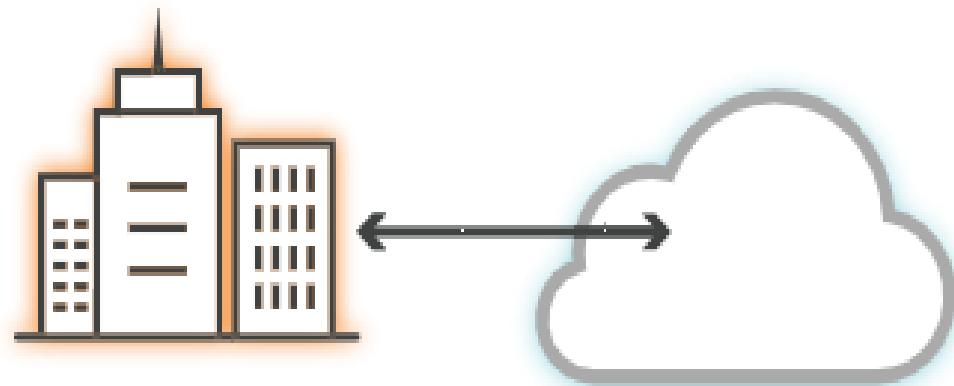
SaaS - Benefits

- Includes all discussed benefits which we get in PaaS
- Ability to access from anywhere
- Ability to access from multiple devices
- No installations and maintenance requirements
- No Application management/Licensing Required

Cloud Essentials Characteristics



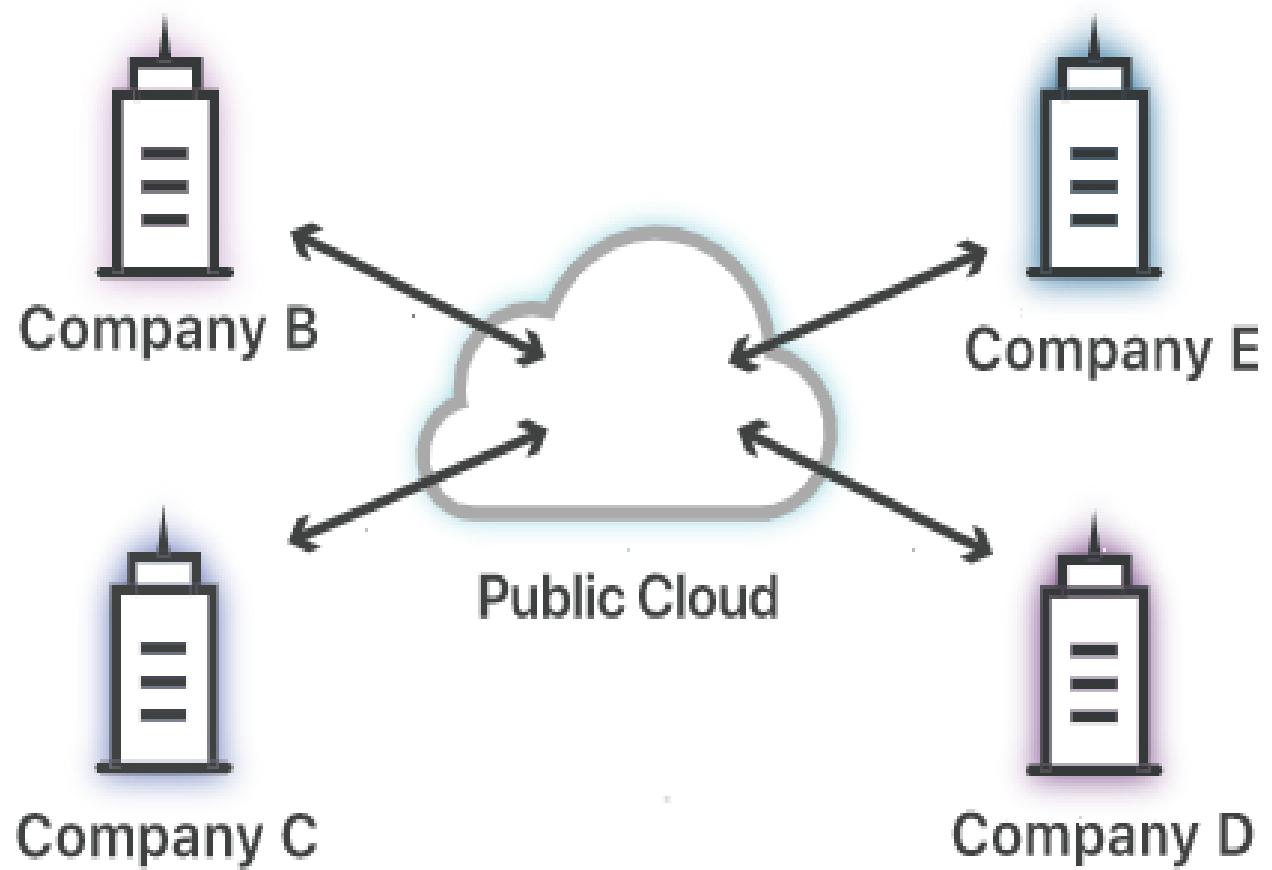
Private cloud



Company A

Company A's
private cloud

Public cloud shared
by multiple companies



Company C

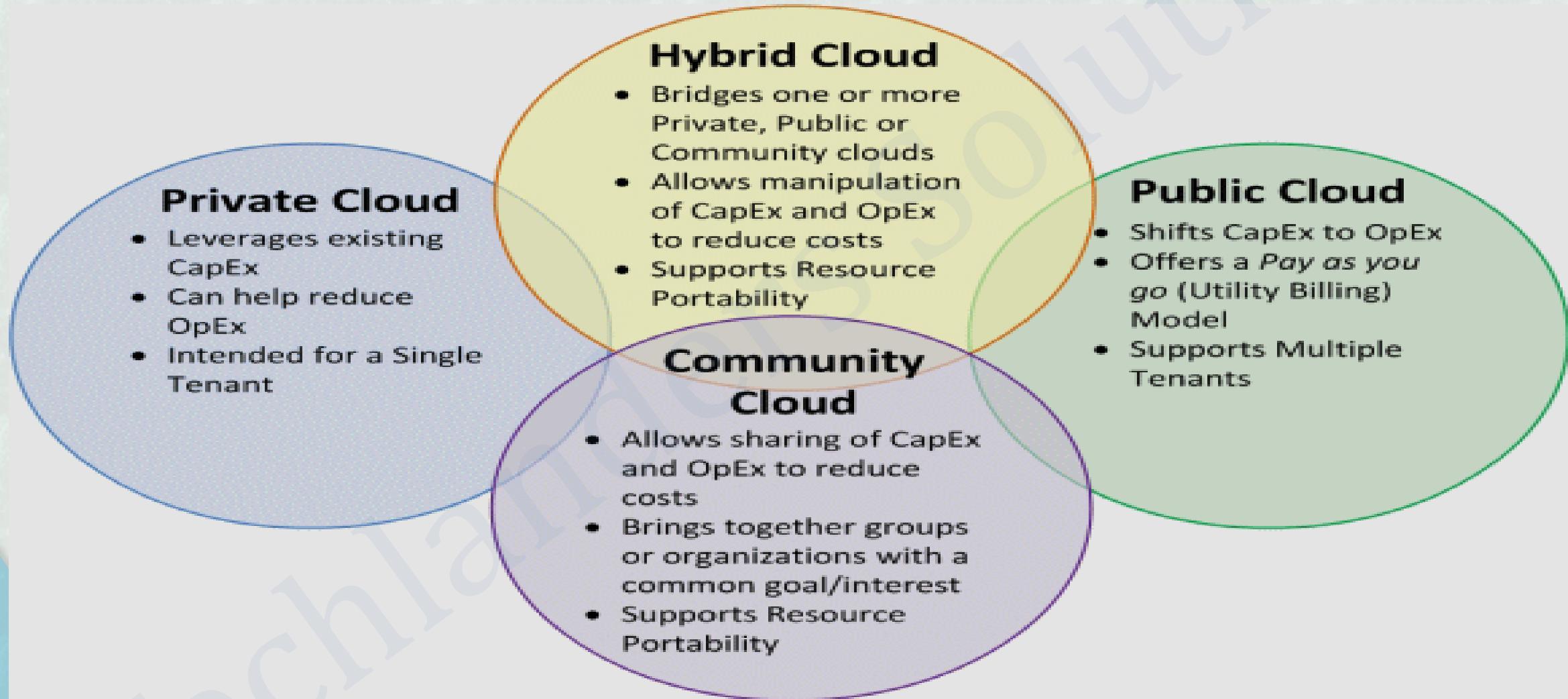
Company D

Company B

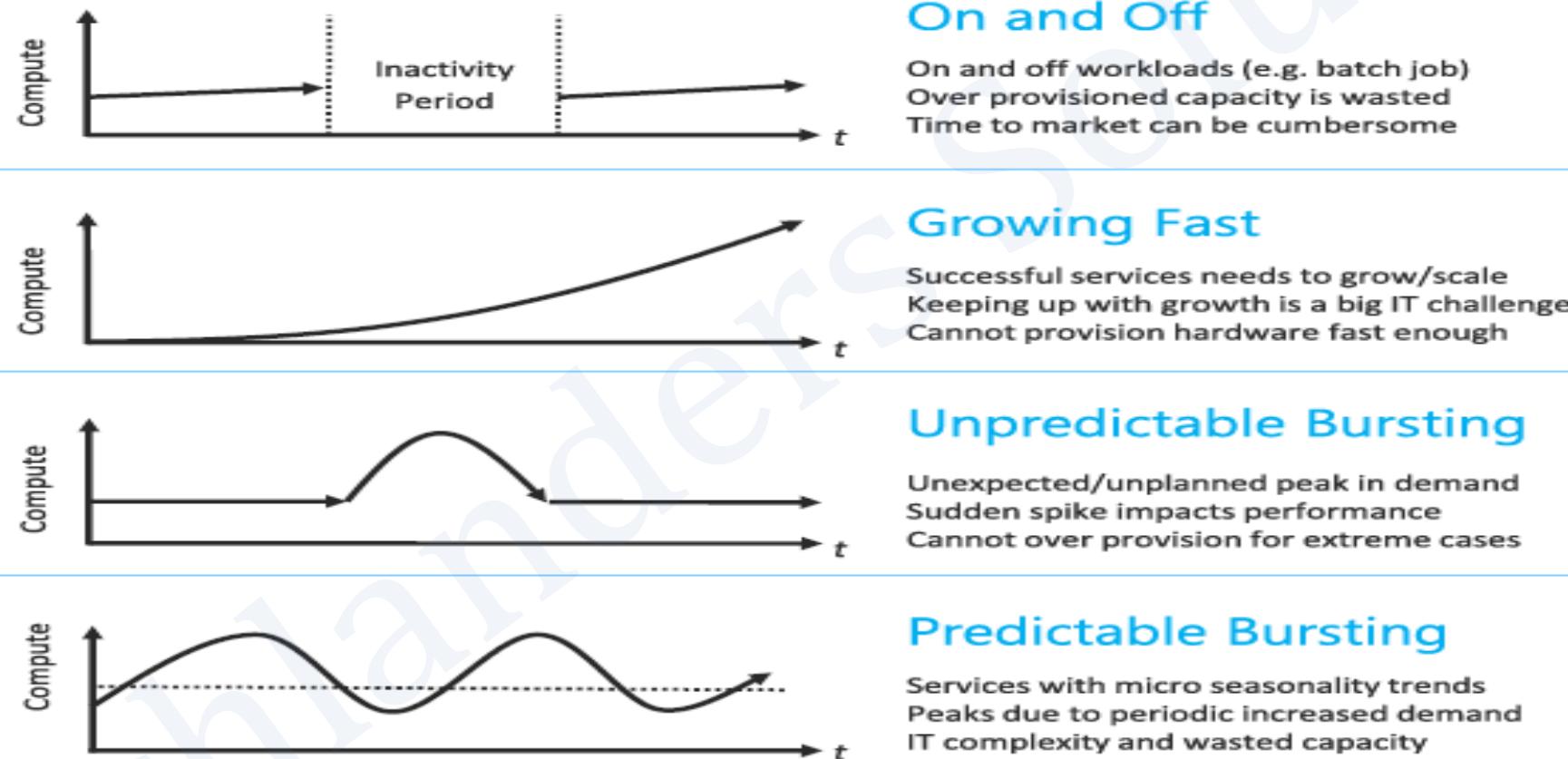
Company E

Public Cloud

Cloud Deployments Types

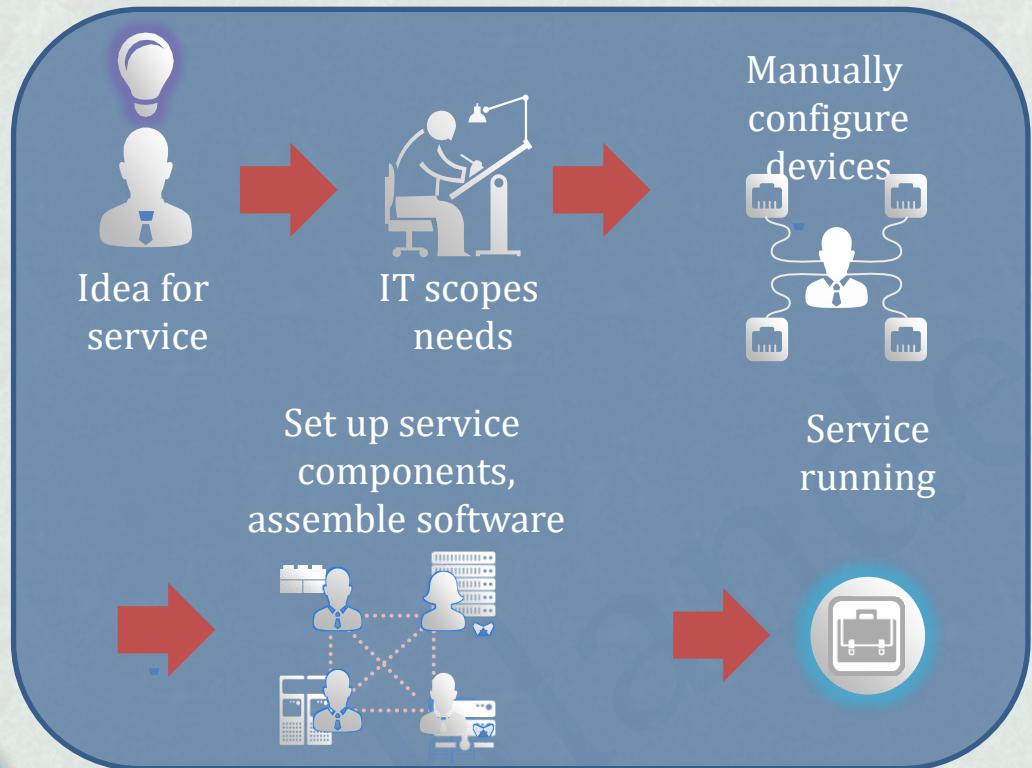


Cloud's Major Use Cases



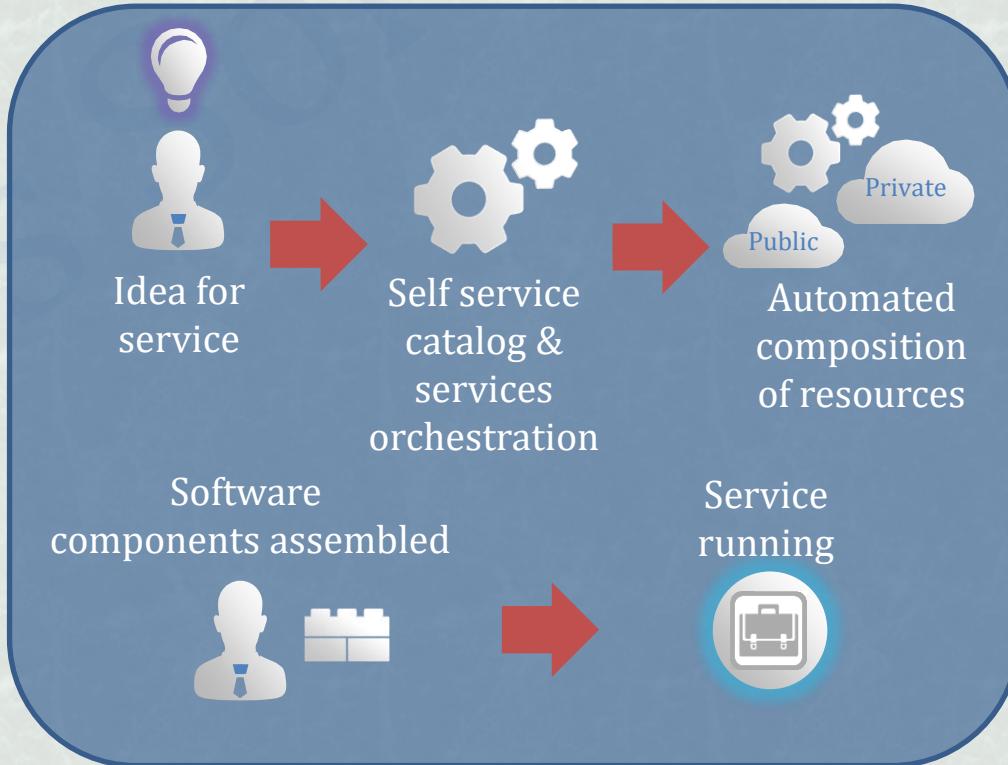
Business Impact of Cloud

Traditional Datacenter



Time to Provision New Service: Months

Cloud Infrastructure



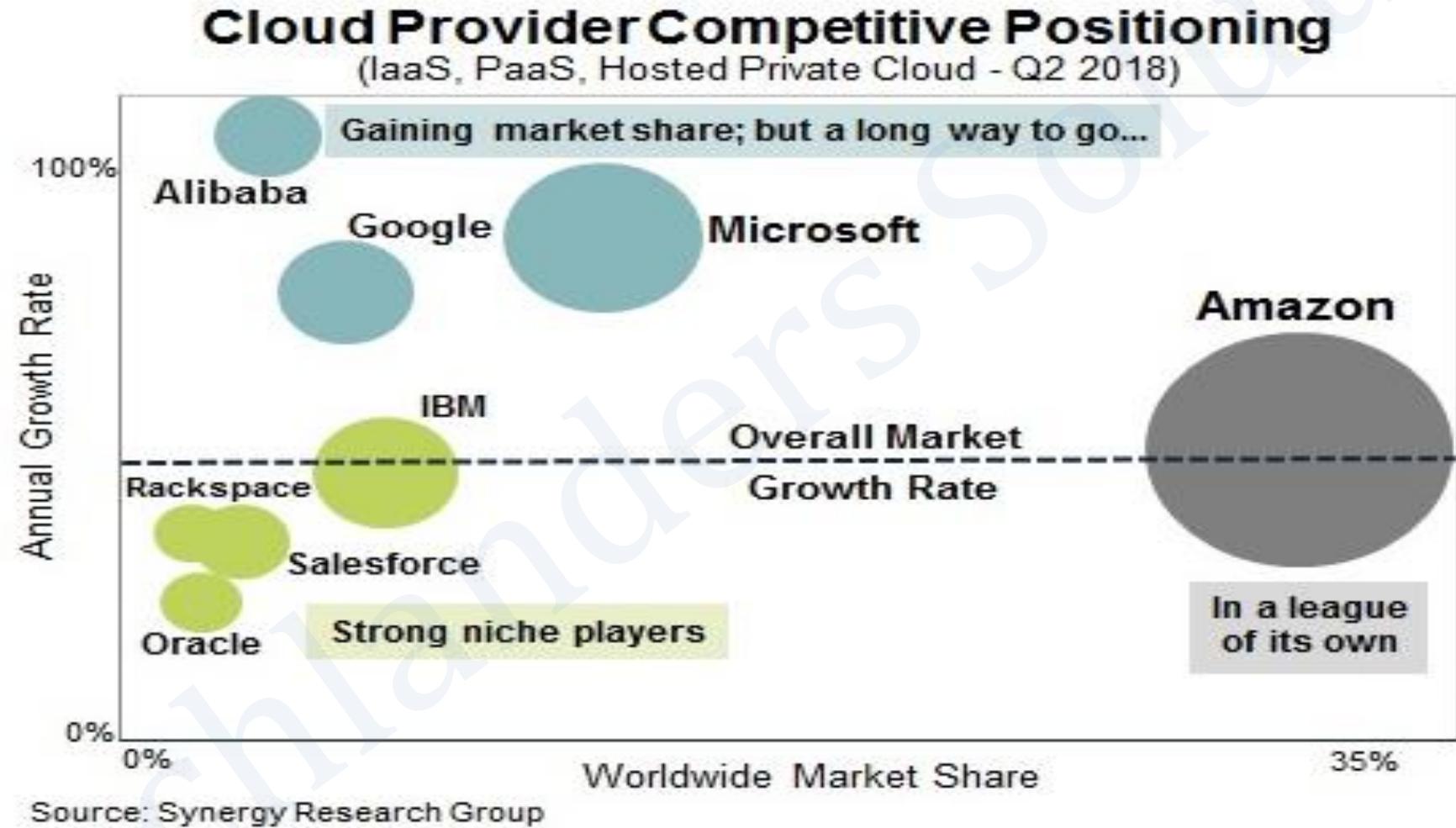
Time to Provision New Service: Minutes

Major Cloud Vendors

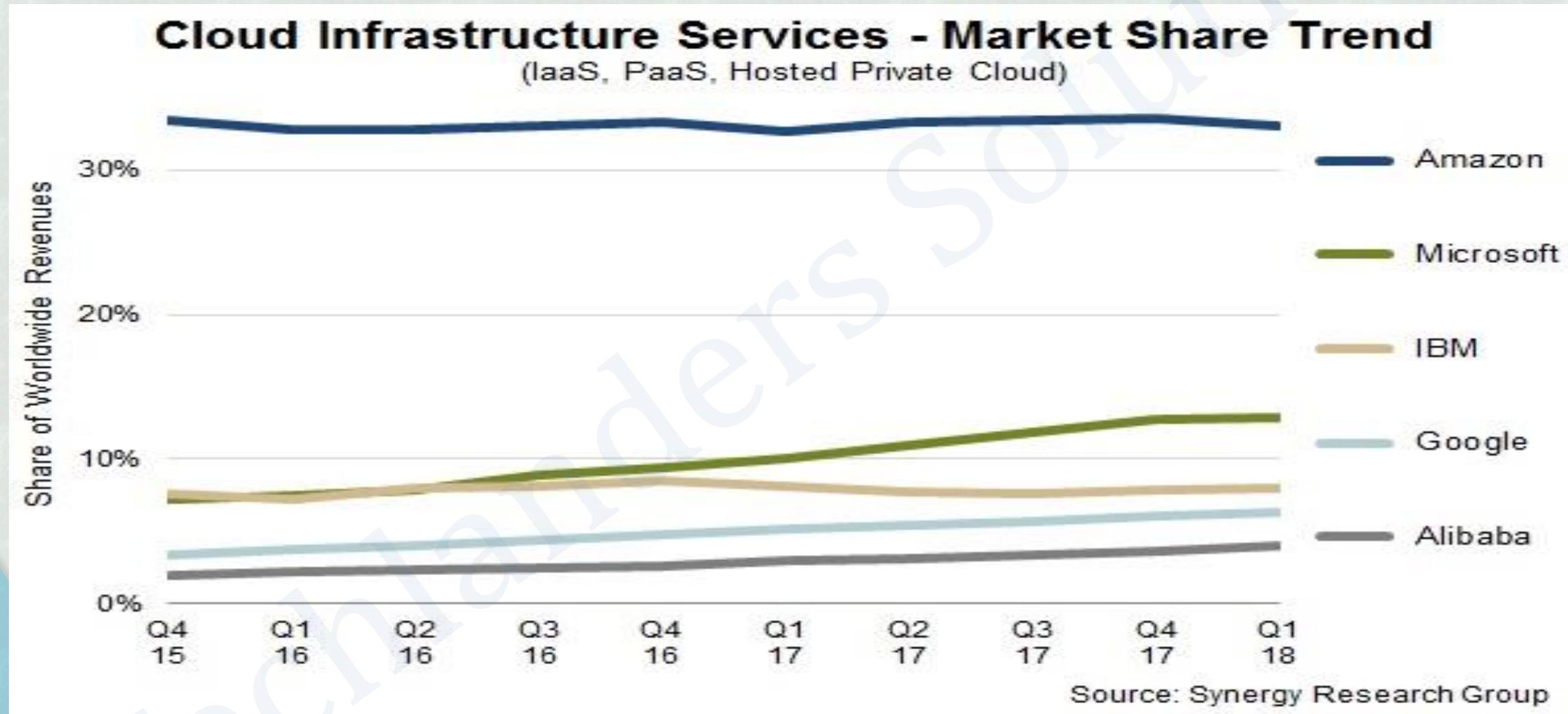
Top Cloud Computing Providers



Who stands where?



Who stands where?



Knowledge Checks

- Which Service Level (IaaS, PaaS, SaaS) provides you most control?
- What is Hybrid Cloud?
- Can two public clouds be connected?
- Connecting two public clouds, will be known as public cloud or Hybrid?
- Cloud provided Database, is a PaaS or SaaS?

AWS **(Amazon Cloud)**

Amazon Web Services

- AWS (Amazon Web Services) is a group of web services (also known as cloud services) being provided by Amazon since 2006.
- AWS provides huge list of services starting from basic IT infrastructure like CPU, Storage as a service, to advance services like Database as a service, Serverless applications, IOT, Machine Learning services etc..
- Hundreds of instances can be build and use in few minutes as and when required, which saves ample amount of hardware cost for any organizations and make them efficient to focus on their core business areas.
- Currently AWS is present and providing cloud services in more than 190 countries.
- Well-known for IaaS, but now growing fast in PaaS and SaaS.

Why AWS?

- **Low Cost:** AWS offers, pay as you go pricing. AWS models are usually cheapest among other service providers in the market.
- **Instant Elasticity:** You need 1 server or 1000's of servers, AWS has a massive infrastructure at backend to serve almost any kind of infrastructure demands, with pay for what you use policy.
- **Scalability:** Facing some resource issues, no problem within seconds you can scale up the resources and improve your application performance. This cannot be compared with traditional IT datacenters.
- **Multiple OS's:** Choice and use any supported Operating systems.
- **Multiple Storage Options:** Choice of high I/O storage, low cost storage. All is available in AWS, use and pay what you want to use with almost any scalability.
- **Secure:** AWS is PCI DSS Level1, ISO 27001, FISMA Moderate, HIPAA, SAS 70 Type II passed. In-fact systems based on AWS are usually more secure than in-house IT infrastructure systems.

AWS Global Infrastructure

AWS Regions:

- Geographic Locations
- Consists of at least two Availability Zones(AZs)
- All of the regions are completely independent of each other with separate Power Sources, Cooling and Internet connectivity.

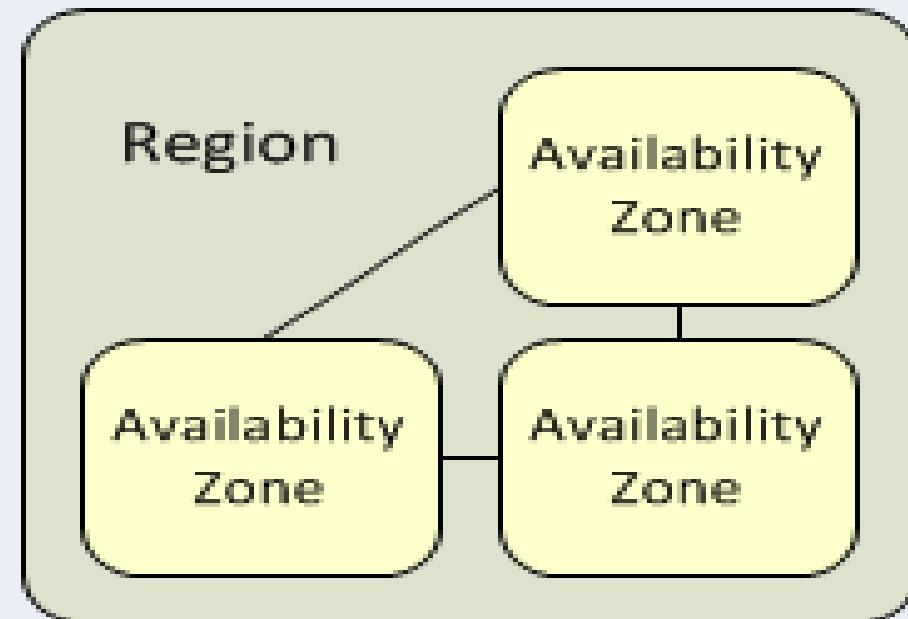
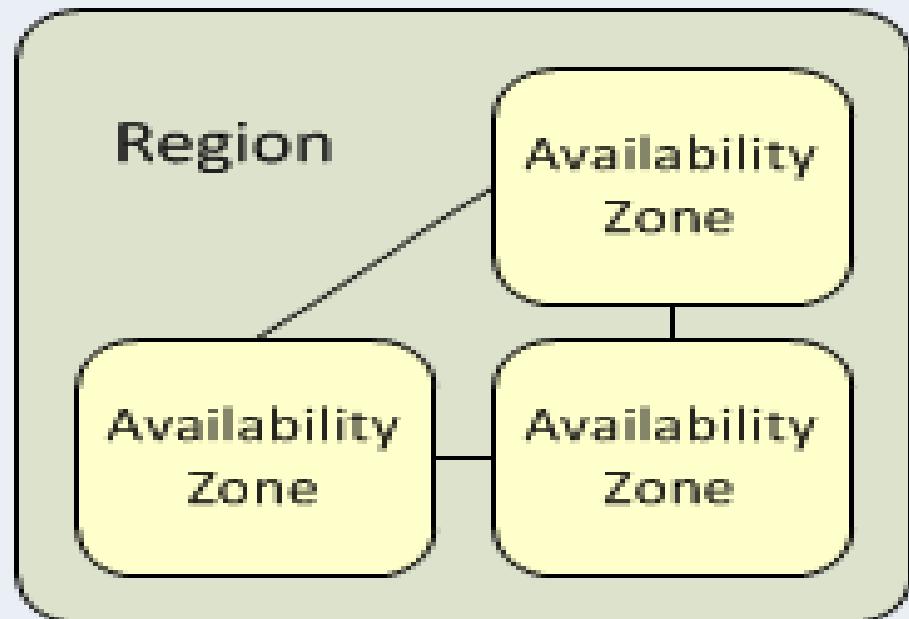
AWS Availability Zones

- AZ is a distinct location within a region
- Each zone is insulated (with low-latency links) from other to support single point of failures
- Each Region has minimum two AZ's
- Most of the services/resources are replicated across AZs for HA/DR purpose.

Note: Resources aren't replicated across regions unless you do so specifically.

AWS Global Infrastructure

Amazon Web Services



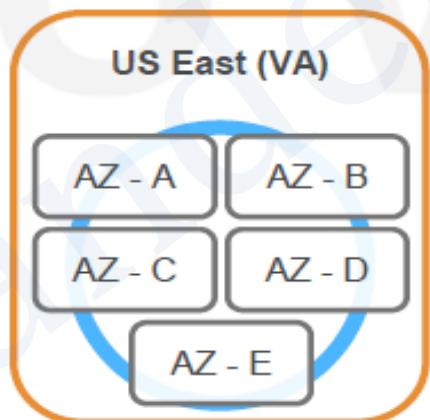
AWS Global Infrastructure

At least 2 AZs per region.

Examples:

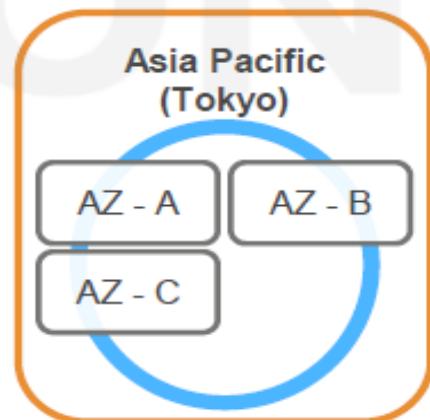
➤ US East (N. Virginia)

- us-east-1a
- us-east-1b
- us-east-1c
- us-east-1d
- us-east-1e



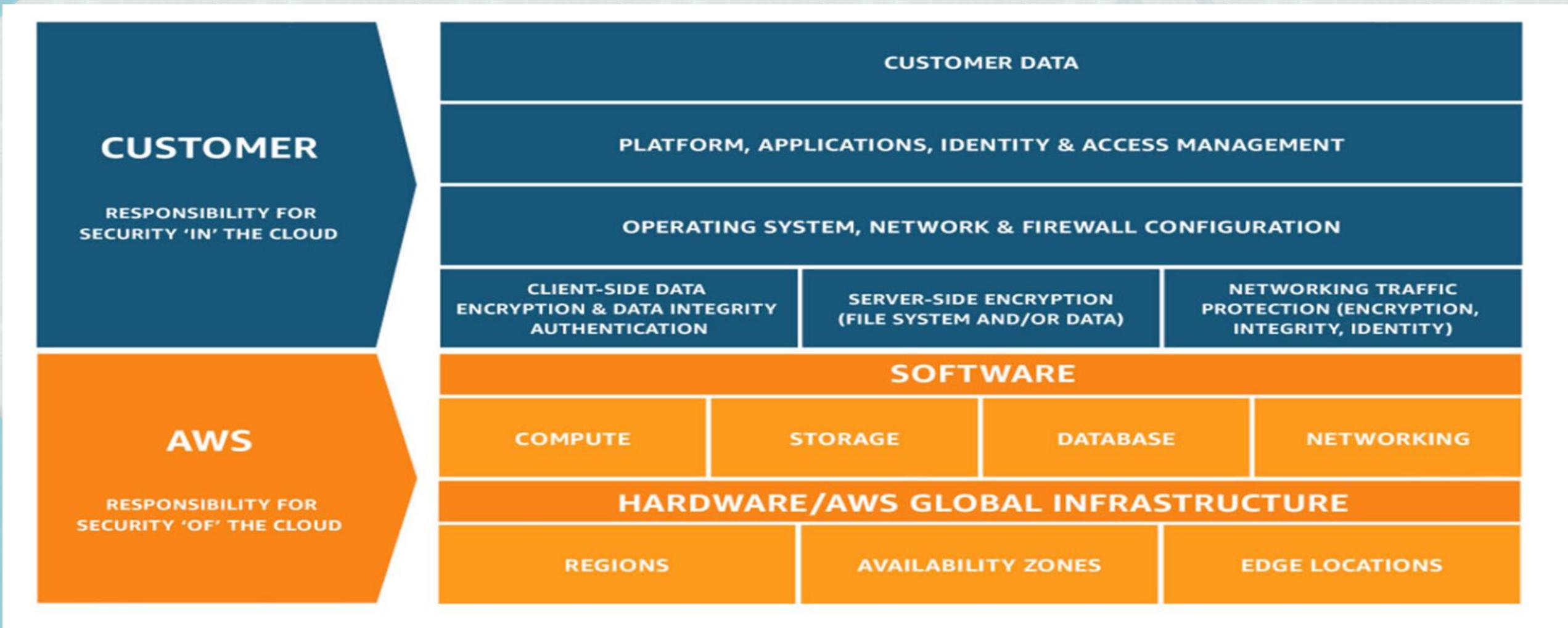
➤ Asia Pacific (Tokyo)

- ap-northeast-1a
- ap-northeast-1b
- ap-northeast-1c



Note: Conceptual drawing only. The number of Availability Zones (AZ) may vary.

Shared Responsibility



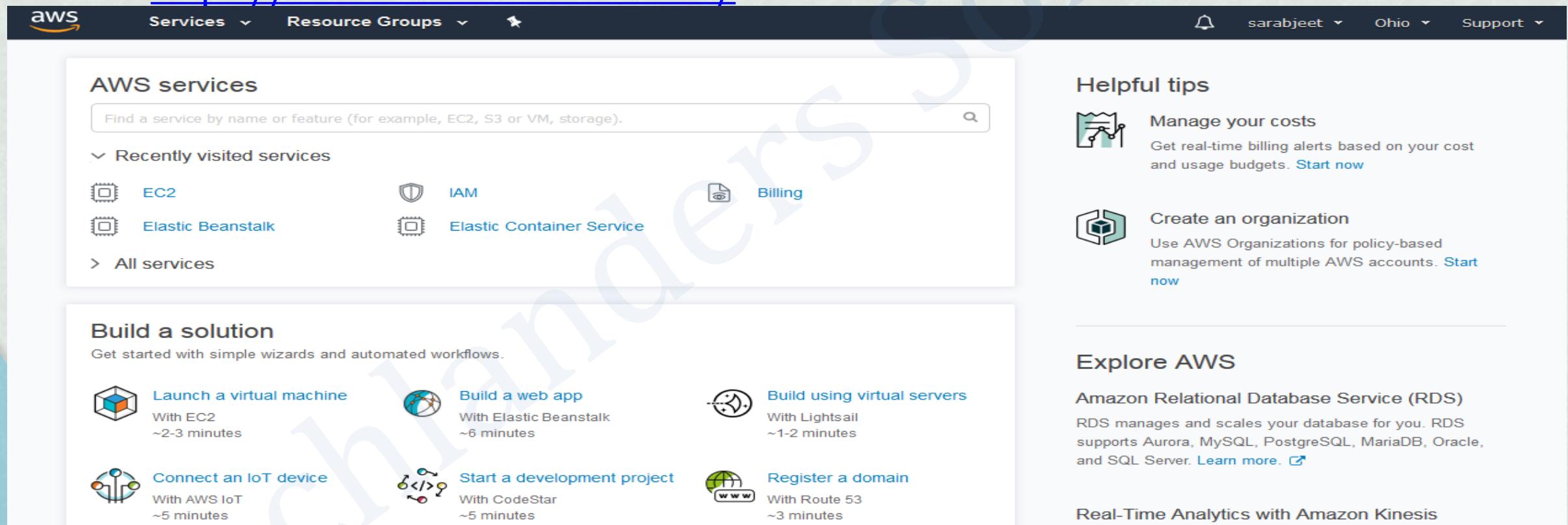
AWS Global Infrastructure

The AWS Cloud operates 44 Availability Zones within 16 geographic Regions around the world, with announced plans for 17 more Availability Zones and six more Regions in Bahrain, China, France, Hong Kong, Sweden, and a second AWS GovCloud Region in the US.



AWS Management Console

- Simple and intuitive web-based user interface.
 - <https://console.aws.amazon.com/>



The screenshot shows the AWS Management Console homepage. At the top, there's a navigation bar with the AWS logo, a search bar, and links for Services, Resource Groups, Notifications, User Profile (sarabjeet), Region (Ohio), and Support.

AWS services: A sidebar with a search bar and a list of recently visited services: EC2, IAM, Billing, Elastic Beanstalk, and Elastic Container Service. There's also a link to All services.

Build a solution: A section with six quick-start options:

- Launch a virtual machine (With EC2, ~2-3 minutes)
- Build a web app (With Elastic Beanstalk, ~6 minutes)
- Build using virtual servers (With Lightsail, ~1-2 minutes)
- Connect an IoT device (With AWS IoT, ~5 minutes)
- Start a development project (With CodeStar, ~5 minutes)
- Register a domain (With Route 53, ~3 minutes)

Helpful tips: Two cards:

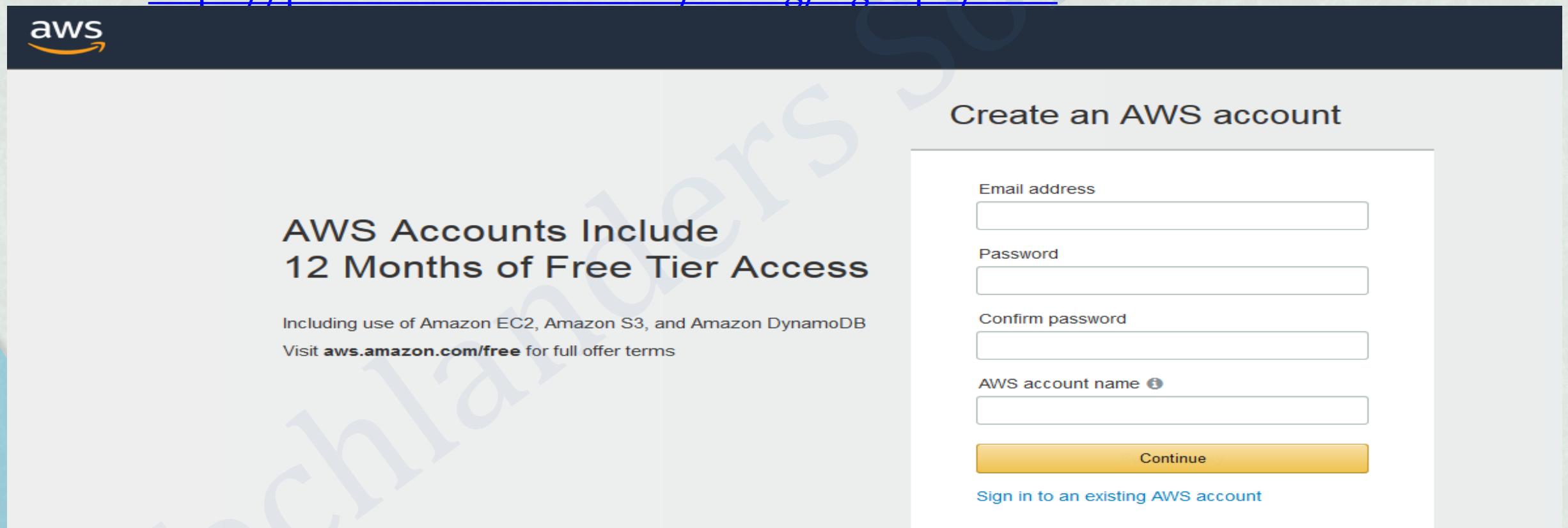
- Manage your costs**: Get real-time billing alerts based on your cost and usage budgets. [Start now](#)
- Create an organization**: Use AWS Organizations for policy-based management of multiple AWS accounts. [Start now](#)

Explore AWS: A section with two cards:

- Amazon Relational Database Service (RDS)**: RDS manages and scales your database for you. RDS supports Aurora, MySQL, PostgreSQL, MariaDB, Oracle, and SQL Server. [Learn more](#).
- Real-Time Analytics with Amazon Kinesis**

LAB 1 : AWS Signup

- Create a new account at
 - <https://portal.aws.amazon.com/billing/signup#/start>



The screenshot shows the AWS Signup process. On the left, there's a promotional message about free tier access, and on the right is the account creation form.

Create an AWS account

Email address

Password

Confirm password

AWS account name

Continue

[Sign in to an existing AWS account](#)

AWS Accounts Include 12 Months of Free Tier Access

Including use of Amazon EC2, Amazon S3, and Amazon DynamoDB

Visit aws.amazon.com/free for full offer terms

AWS SIGNUP

creating the account

dhi Nagar

suite, unit, building, floor, etc

Province or region

e

net Services Pvt. Ltd. Customer

an India contact address are now required to
amazon Internet Service Private Ltd. (AISPL).
cal seller for AWS infrastructure services in

k here to indicate that you have read
gree to the terms of the AISPL
omer Agreement

Create Account and Continue 

Payment Information

We use your payment information to verify your identity and only for usage in excess of the [AWS Free Tier Limits](#). [We will not charge you for usage below the AWS Free Tier Limits](#). For more information, see the [frequently asked questions](#).



As part of our card verification process we will charge INR 2 on your card when you click the "Secure Submit" button below. This will be refunded once your card has been validated. Your bank may take 3-5 business days to show the refund. Mastercard/Visa customers may be redirected to your bank website to authorize the charge.

Credit/Debit card number

Expiration date

10

-

2019

▼

Cardholder's name

Select a Support Plan

AWS offers a selection of support plans to meet your needs. Choose the best plan that best aligns with your AWS usage. [Learn more](#)



Basic Plan

Free

- Included with all accounts
- 24x7 self-service access to AWS resources
- For account and billing issues only
- Access to Personal Health Dashboard & Trusted Advisor
- 12-hour response time for nonproduction systems



Developer Plan

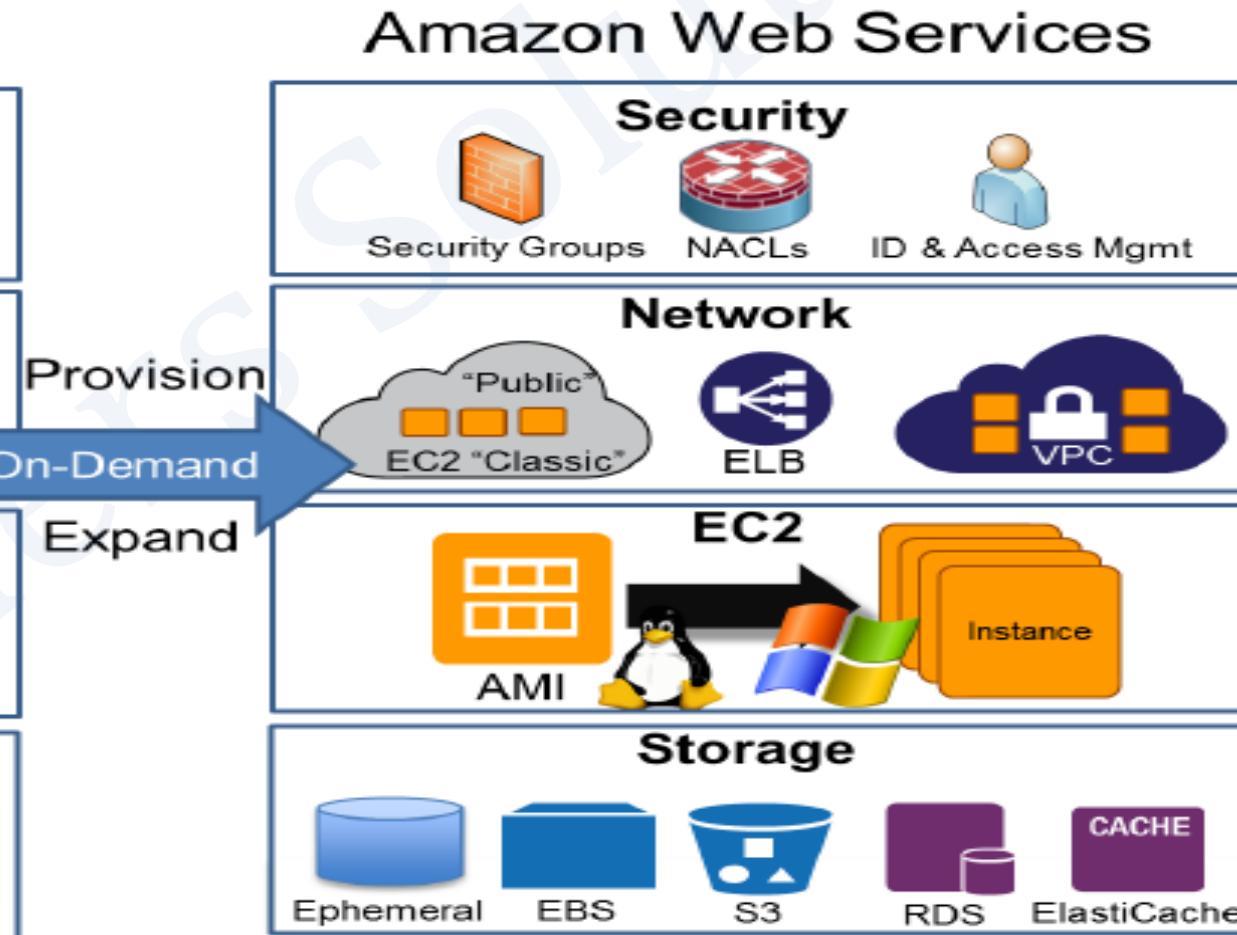
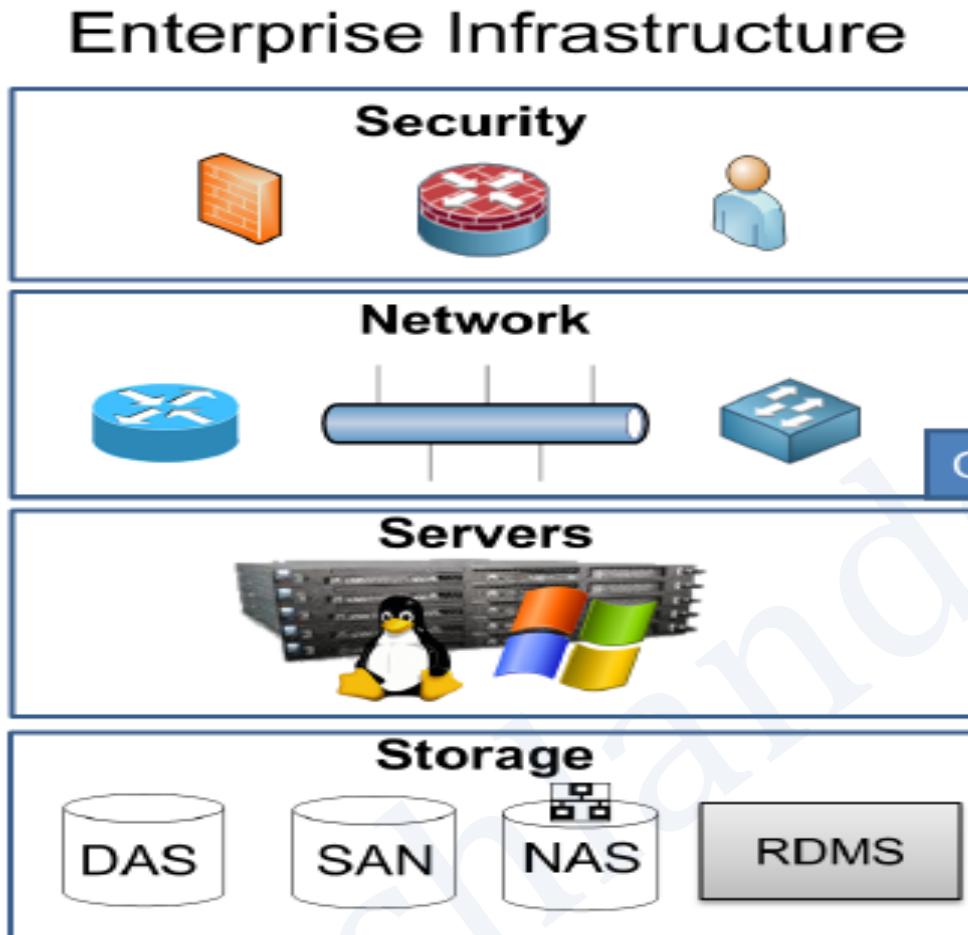
From \$29/month

- For early adoption, testing and development
- Email access to AWS Support during business hours
- 1 primary contact can open an unlimited number of support cases
- 12-hour response time for nonproduction systems

Need Enterprise level support?

Contact your account manager for additional information on our enterprise support options.

AWS Core Infrastructure Services



AWS Security

- **Physical Security:**
- 24/7 trained security staff
- AWS data centers in nondescript and undisclosed facilities
- Two-factor authentication for authorized staff
- Authorization for data center access
- Multiple approval based change process

AWS Security

- **Hardware, Software, and Network :**
- Authentication and authorization in place
- RBAC based access control mechanism
- Firewall and other boundary devices
- Security at Server level, Application level and Network level
- AWS monitoring tools
- Services to log AWS resources access

AWS Security



Amazon Resource Names (ARNs)

Amazon Resource Names (ARNs) uniquely identify AWS resources.

We require an ARN when you need to specify a resource unambiguously across all of AWS, such as in IAM policies, API calls etc.

ARN have a specific format:

arn:partition:service:region:account-id:resourcetype/resource

- IAM user name
 arn:aws:iam::123456789012:user/David
- IAM instance id:

arn:aws:ec2:region:account-id:dedicated-host/host_id

Eg. arn:aws:ec2:us-east-1:123456789012:dedicated-host/h-12345678

AWS Access Credentials

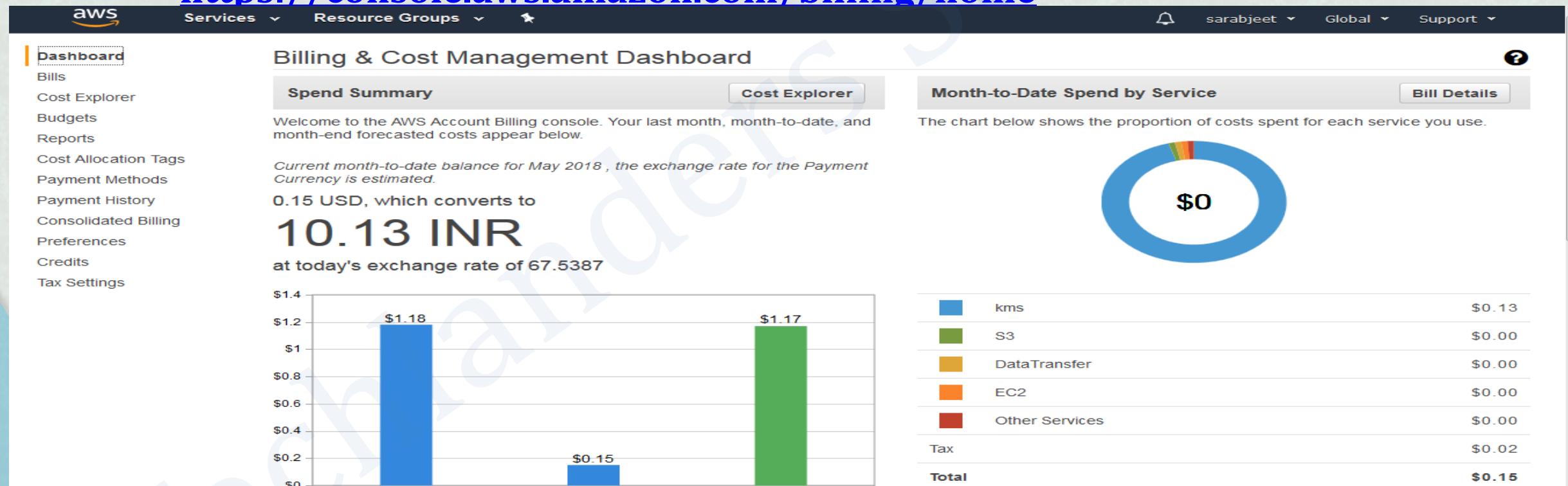
AWS resources can be access using several authentication methods:

- IAM User-id / Password
- Account ID/ AK (Access Key)/ SK (Security Key)
- Certificates
- Key pairs

AWS Billing

AWS Billings history, Previous payments, Current month cost, Budget fixing, Setting usage alarms etc, can be managed from AWS billing page :

<https://console.aws.amazon.com/billing/home>



AWS Pricing calculators

Simple Monthly Calculator:

You can Estimate your expected monthly bill using Simple Monthly Calculator.

<http://calculators.s3.amazonaws.com/index.html>

TCO Calculator:

You can Quickly compare the total cost of ownership (TCO) of your **on-premises infrastructure with a comparable AWS deployment** using TCO Calculator and estimate savings you can realize by moving to AWS. <https://awstcoccalculator.com/#>

Cost Explorer:

With Cost Explorer, you can track your actual account usage and bill, at any time using the billing portal. You can view data for up to the last 13 months, forecast how much you are likely to spend for the next three months, and get recommendations for what Reserved Instances to purchase.

<https://console.aws.amazon.com/billing/home#/costexplorer>

Resource Management Tools

- **AWS Management Console**
- AWS Console Mobile App (View resources)
- **AWS Command line interface**
- AWS Toolkit for PowerShell
- AWS-Shell



AWS Compute Services

Virtual Machines

- What are the bare minimum infrastructure requirements for any application?
- Hardware (RAM, Processor, Processor type, HBA's, etc.)
- OS Disk & Data Disk
- OS Images
- Network
- Security (Firewall, Networking, Access Mechanism)
- Credentials

AWS Elastic Compute Cloud

- Amazon **EC2** stands for **Elastic Compute Cloud**, and is the Primary AWS web service.
- Provides Resizable compute capacity
- **Reduces the time required** to obtain and boot new server instances to minutes
- There are two key concepts to Launch instances in AWS:
 - **Instance Type**
 - **AMI**

EC2 Facts:

- Scale capacity as your computing requirements change
- Pay only for capacity that you actually use
- Choose Linux or Windows OS as per need.
- **Deploy across AWS Regions and Availability Zones for reliability/HA**
- Only X86 based OS supported. So platform specific OS are not supported.

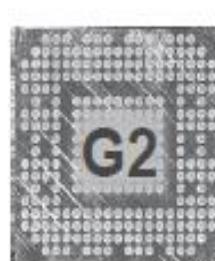
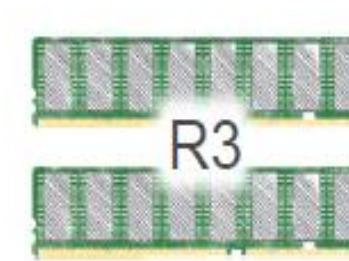
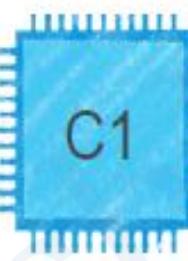
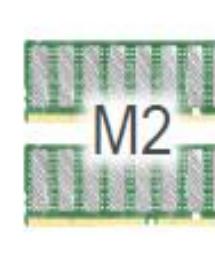
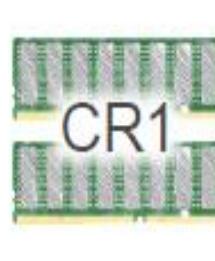
Instance Types (EC2)

- The instance type defines the different virtual hardware Models (sizes) supporting an Amazon EC2 instance.
- There are dozens of instance types available, varying in the following dimensions:
 - Virtual CPUs (vCPUs)
 - Memory
 - Storage (size and type)
 - Network performance
 - Graphical Processing
- Instance types are grouped into families based on the ratio of these values to each other.

Instance Types (EC2)

- EC2 instance types are optimized for different use cases and come in multiple sizes. This allows you to optimally scale resources to your workload requirements.
- AWS uses Intel® Xeon® processors for EC2 instances, providing customers with high performance and value.
- Consider the following when choosing your instances: Core count, memory size, storage size and type, network performance, and CPU technologies.
- Hurry Up and Go Idle - A larger compute instance can save you time and money, therefore paying more per hour for a shorter amount of time can be less expensive.

Instance Types with AWS

General purpose	Compute optimized	Storage and IO optimized	GPU enabled	Memory optimized
		 I2 HS1		
	 			 

EC2 Pricing Models

On-Demand Instances

Pay by the hour.

Reserved Instances

Purchase at significant discount.
Instances are always available.

1-year to 3-year terms.

Scheduled Instances

Purchase a 1-year RI for a recurring period of time.

Spot Instances

Highest bidder uses instance at a significant discount.
Spot blocks supported.

Dedicated Hosts

Physical host is fully dedicated to run your instances. Bring your per-socket, per-core, or per-VM software licenses to reduce cost.

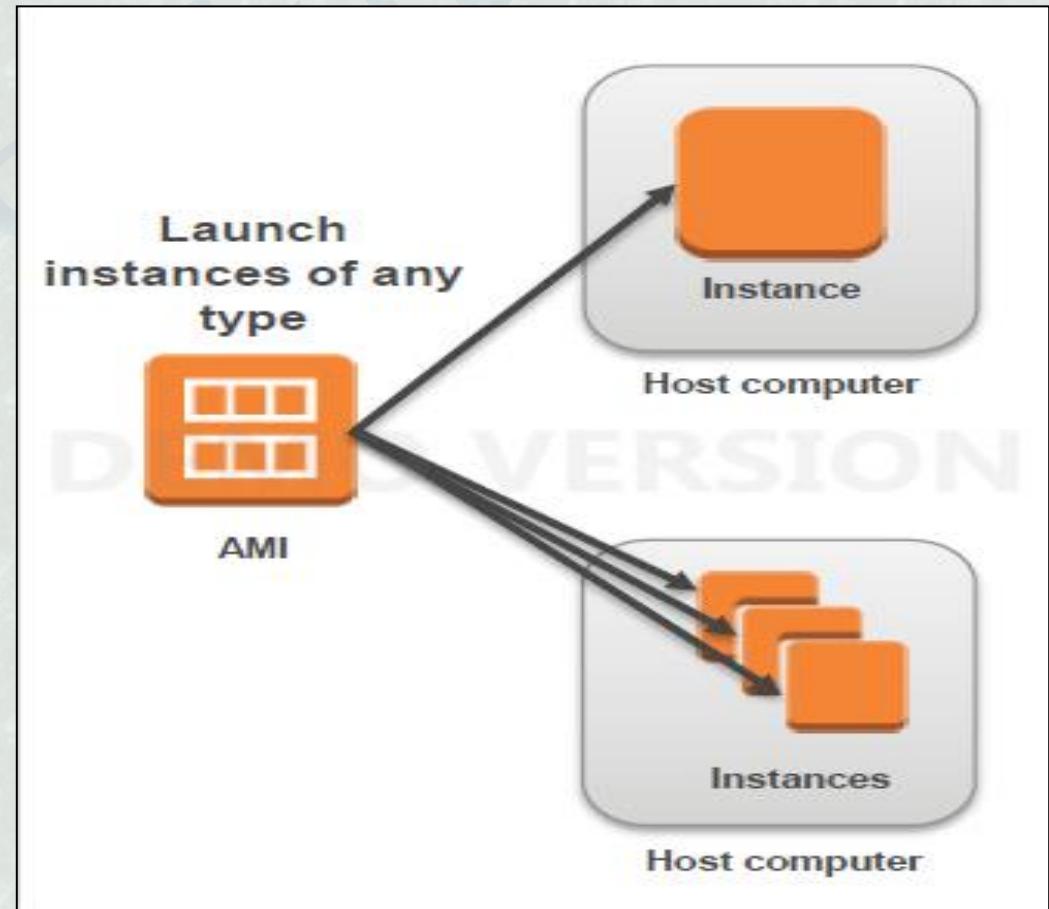
AMI's

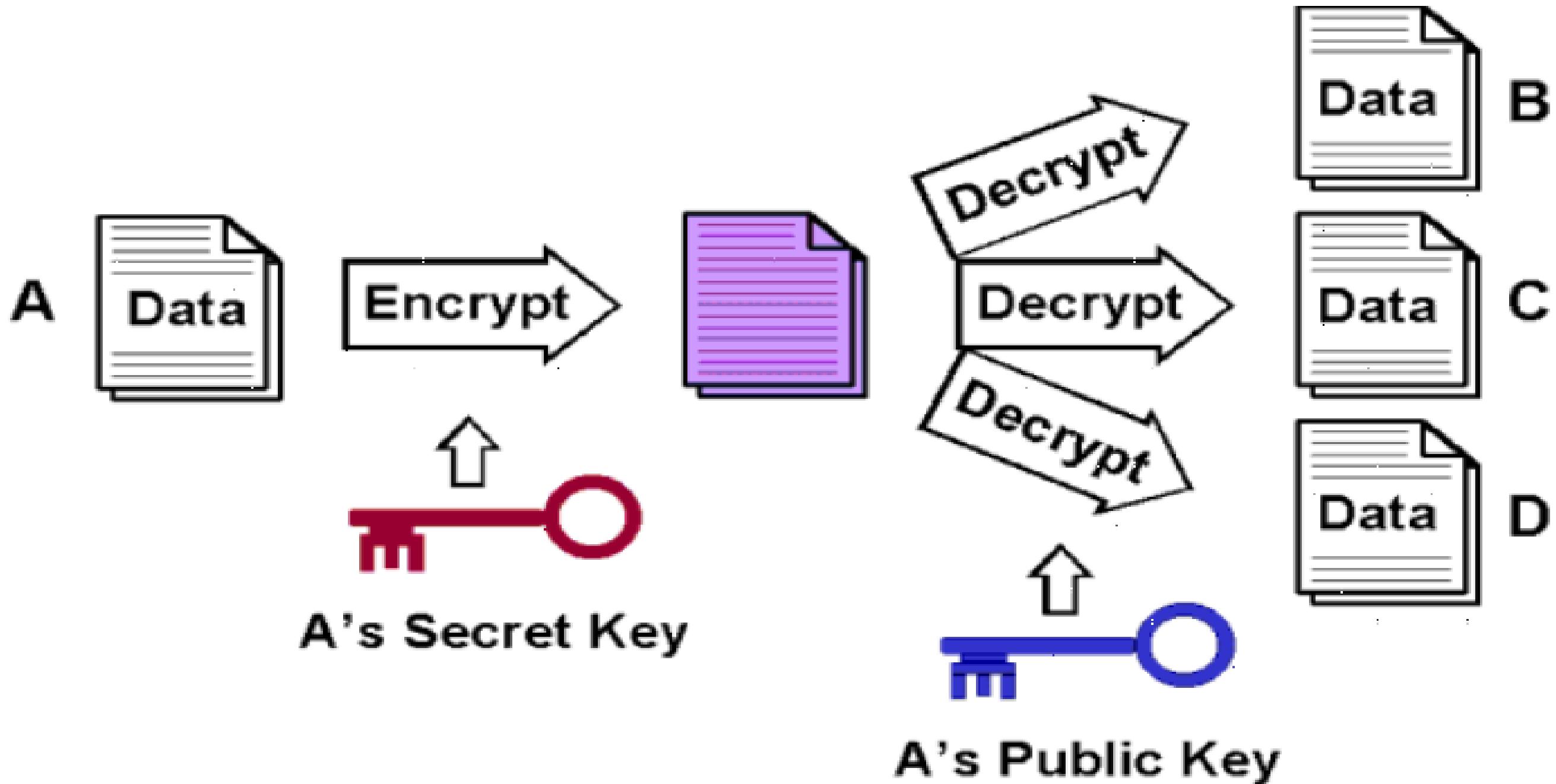
- Stands for **Amazon Machine Image**
- **AMI is a template for the root volume** for the instance (example: an OS image, a webserver, an application server etc.)
- It also includes the launch permissions that control which AWS accounts can use the AMI to launch instances.
- All AMIs are based on x86 OSs, either Linux or Windows.

Instances and AMI's

Select an AMI based on:

- Region
- Operating system
- Architecture (32-bit or 64-bit)
- Launch permissions
- Storage for the root device
- Virtualization Type





Credentials

- Used to access an Instance.
- Amazon EC2 uses public-key cryptography to encrypt and decrypt login information.
- Public-key cryptography uses a public key to encrypt a piece of data and an associated private key to decrypt the data.
- These two keys together are called a *key pair*.
- AWS stores the public key, and the private key is kept by the customer. The private key is essential to acquiring secure access to an instance for the first time.
- When launching a Windows instance, Amazon EC2 generates a random password for the local administrator account and encrypts the password using the public key.

Launching EC2 Instance

- Procedure to launch EC2 instance in minutes:
- Determine the AWS Region in which you want to launch the Amazon EC2 instance.
- Launch an Amazon EC2 instance from a pre-configured Amazon Machine Image (AMI).
- Choose an instance type based on CPU, memory, storage, and network requirements.
- Configure network, security groups, storage volume, tags, and key pair; or directly launch instance post selecting AMI and Instance type.

LAB 3 : Create an EC2 Linux Instance

Console Access: <https://console.aws.amazon.com/>

- Observe different available AMIs and Instance types
- Exercise:

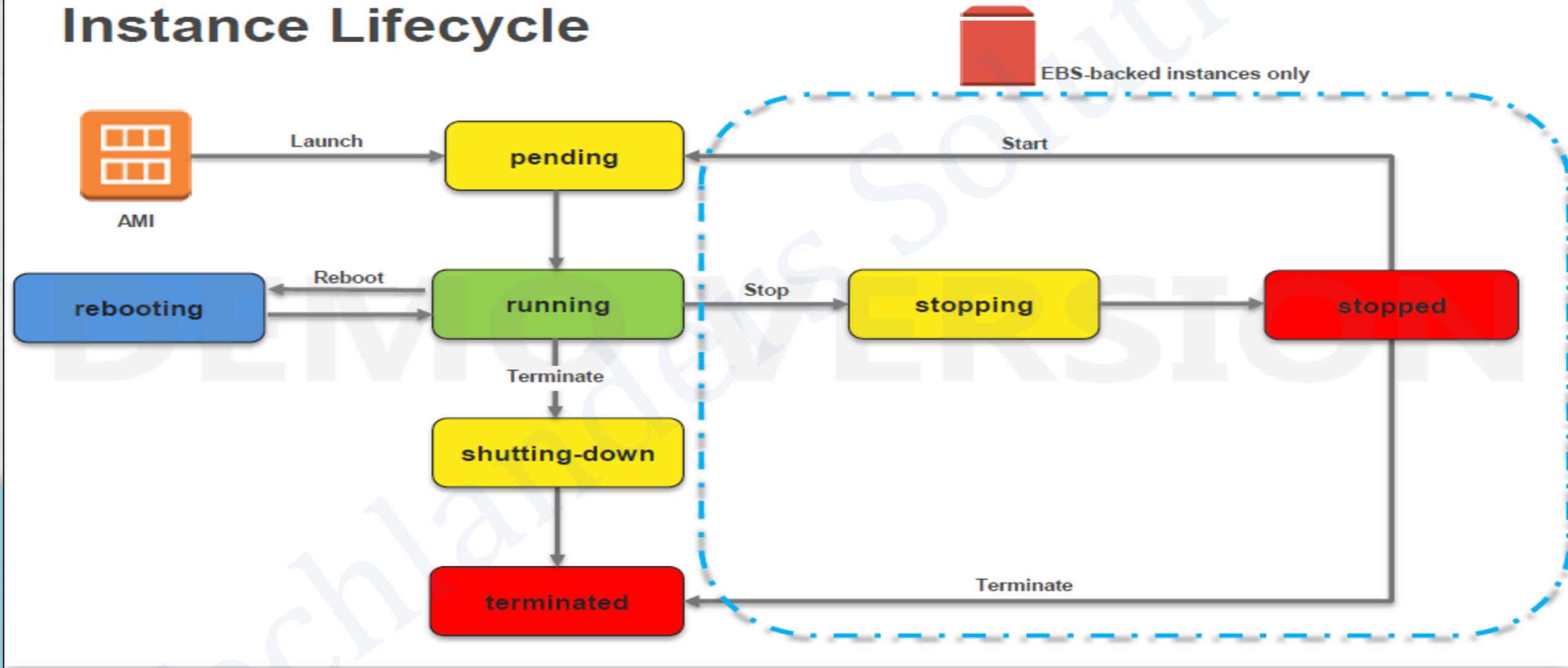
Create a Linux Operating System using mandatory configurations:

- Instance Type : **T2.Micro**
 - AMI : **Amazon Linux 2 AMI**
 - Credentials: **Create/download your own key pair and use same.**
-
- Observe the instance details and navigate through the diff tabs
 - Reboot, Stop, Start your server
 - Access your instance using above downloaded key pair using different platforms i.e , instance connect , using putty , cli.

Note: (For the first instance take the default parameters wherever inputs required)

Instance Lifecycle

Instance Lifecycle



Network part

Private IP: Alike traditional VMs/hosts, each EC2 instance must at least belong to one network (VPC in AWS) and must have at least one Private IP from that network/subnet.

Public IP: A launched instance may also have a public IP address assigned. This IP address is assigned from the addresses reserved by AWS and cannot be specified. This IP address is unique on the Internet, persists only while the instance is running, and cannot be transferred to another instance.

Private/Public Domain Name System (DNS): When you launch an instance, AWS creates one private and one Public DNS name that can be used to access the instance. This DNS name is generated automatically and cannot be specified by the customer. This DNS name persists only while the instance is running and cannot be transferred to another instance.

Elastic IP (EIP): An elastic IP address is an address unique on the Internet that you reserve independently and associate with an Amazon EC2 instance.

EC2 Security – Security Group

Virtual Firewall Protection

AWS allows you to control traffic **in** and **out** of your instances through virtual firewalls called *security groups*.

Security groups allow you to control traffic based on *port*, *protocol*, and *source(inbound)/destination(outbound)*.

Security groups are associated with instances when they are launched. **Every instance must have at least one security group.** Though they can have more.

A security group is default deny.

Amazon Server Storage

Amazon EBS (Elastic Block Store)

- Data stored on an Amazon EBS volume can persist independently of the life of the instance.
- Persistent **block-level storage volumes** for use with Amazon EC2 instances.
- 99% used volume type in AWS.

Amazon EC2 Instance Store

- Like Swap volumes attached to your server for faster processing.
- Data stored on a local instance store persists only as long as the instance is alive.
- Storage is ephemeral.
- Older volume type, used exceptionally in some instances i.e DB,High processing systems.

AWS Tags

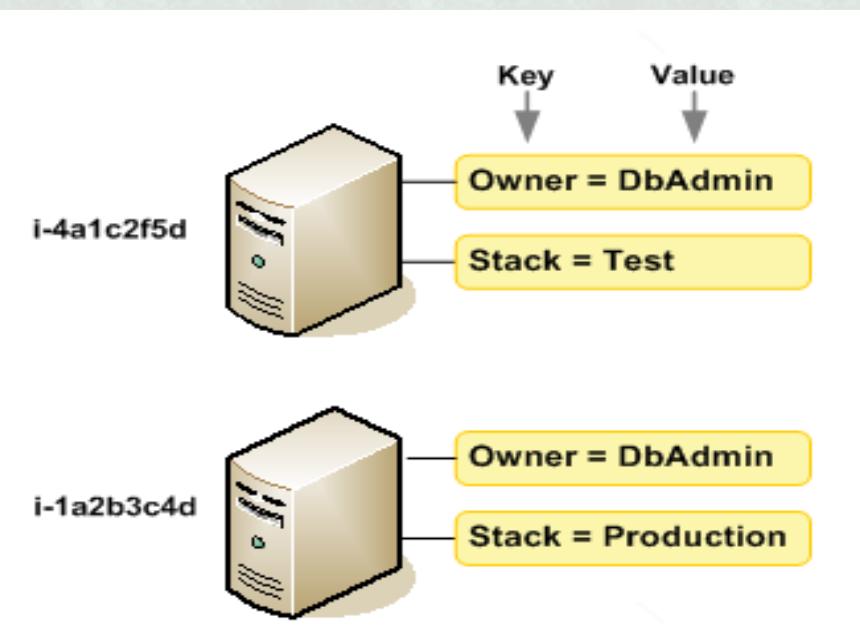
Tags helps you manage your instances, images, and other Amazon EC2 resources.

Its your own metadata to each resource in the form of *tags*.

Each tag consists of a *key* and an optional *value*, both of which you define.

It enable you to categorize your AWS resources in different ways, for example, by purpose, owner, or environment.

You can organize your billing information based on resources that have the same tag key values.



AWS Tags

- Maximum number of tags per resource—50
- Tag keys and values are case-sensitive.
- You can't terminate, stop, or delete a resource based solely on its tags; you must specify the resource identifier : ARN's
- Not all resources supports tags. Pls check the supported list from AWS website.

EC2 Metadata

AWS Instance Metadata –

Instance metadata is data/information about your instance.

An HTTP call to **http://169.254.169.254/latest/meta-data/** will return the top node of the instance metadata tree and will return the information about your instance e.g. Public IP, DNS name, OS type etc..

For Linux systems:

```
curl http://169.254.169.254/latest/meta-data/
```

Dynamic Instance Identity metadata:

```
http://169.254.169.254/latest/dynamic/instance-identity/
```

LAB 4 : Create Windows Instance

Exercise:

- Create a Windows Operating System using custom configurations:
 - Instance Type : **T2.Micro**
 - AMI : **Microsoft Windows Server 2019 Base**
 - Credentials: **Use existing key from Lab 3 or if not create a new one.**
 - Assign tags, **Key = Name, Value= Server-Name**
 - Assign tags, **Key = Owner, Value= Your name**
 - Add inbound port of **RDP**
- Access your instance using Administrator password, which will be retrieved using your downloaded key pair using mstsc or dns

LAB 5 : Work with IP/DNS

Exercise: Elastic ip may charge you if not associated with your instance.

- Use server created in previous exercise.
 - Case-1
 - Reboot and observe the Public IP (there will be no change)
 - Case-2
 - Stop the Instance and Start it back, **observe the Public IP**
- Que is how to make Public IP's Persistent ?
- Exercise: Create an EIP, Allocate the EIP to an Instance and perform both cases again.
- **NOTE : MAKE SURE THE ELASTIC IP IS CHARGEABLE IN FREE TIER WHEN :**

***ELASTIC IP IS NOT ASSOCIATED WITH ANY INSTANCE**

***INSTANCE ATTACHED WITH ELASTIC IP IS STOPPED .**

LAB 6 : Creating server using CLI

Check the details for all running instances using CLI

- aws ec2 describe-instances | grep -i instanceid

Creation of an AWS Instance using CLI:

- aws ec2 run-instances --image-id ami-033b95fb8079dc481 --instance-type t2.micro --key-name virginia-key
- aws ec2 describe-instances | grep -i instanceid
- aws ec2 stop-instances --instance-ids i-052af6a166198df10
- aws ec2 terminate-instances --instance-ids i-052af6a166198df10

LAB 8 : Working with User data

- User data is a feature of EC2, to provide build time commands to EC2 instance.
- Commands provided will run during instance startup.

Create one EC2 Linux instance and provide EC2 user data to run below scripts:

```
#!/bin/bash
sudo yum update -y;
sudo yum install -y httpd;
sudo systemctl start httpd;
sudo systemctl enable httpd;
sudo echo "<h1>Hello World from $(hostname -f)</h1>" > /var/www/html/index.html
```

- If user-data is changed after the creation of server after stopping it then script will not run after modifying and restart as well.

IAM Roles

IAM roles are a secure way to grant permissions to entities that you trust.

- IAM user in another account
- Application code running on an EC2 instance that needs to perform actions on AWS resources
- An AWS service that needs to act on resources in your account to provide its features

LAB 29 : IAM Roles

In this exercise, you will create an IAM Role, which will allow EC2 instance to run codes/access other service on your behalf.

1. While logged in as the root user, Click on IAM → Roles.
2. Click Create Role
3. Select a service that will use this role to get access (select EC2 here).
4. Click “Next – Permissions”.
5. Select “AdministratorAccess”
6. Click on review & provide a Role name
7. Click on Create Role.
8. Connect to instance and using aws commands , you will be able to access s3 , ec2 and other resources .. You will see the resources :
9. aws ec2 describe-instances --region us-east-1 | grep -i instanceid
10. Try to access resources from cmd as well as ec2 instance connect , both will now not ask for access aws configure due to the role attached.



AWS

Storage Services



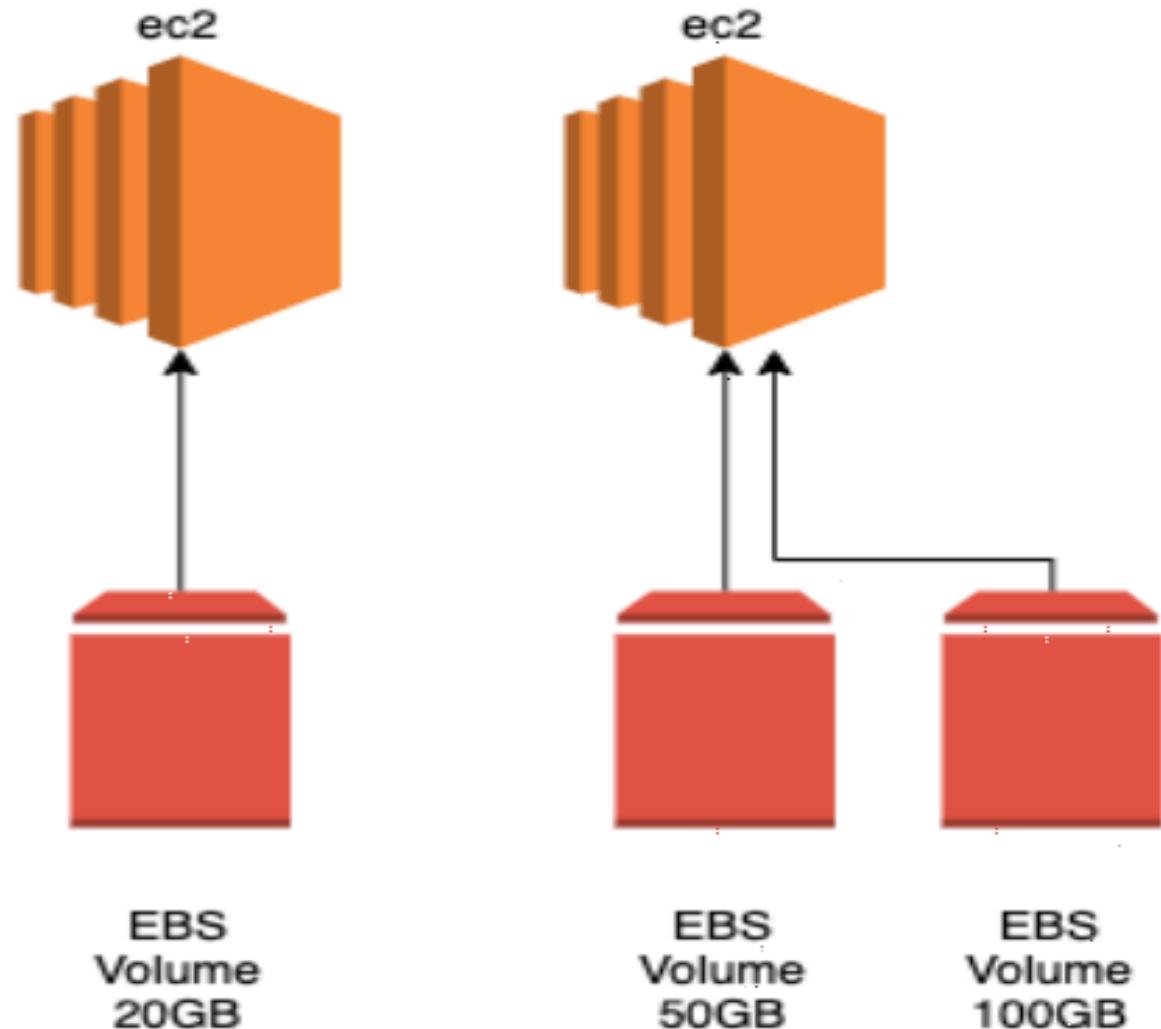
AWS

EBS Service

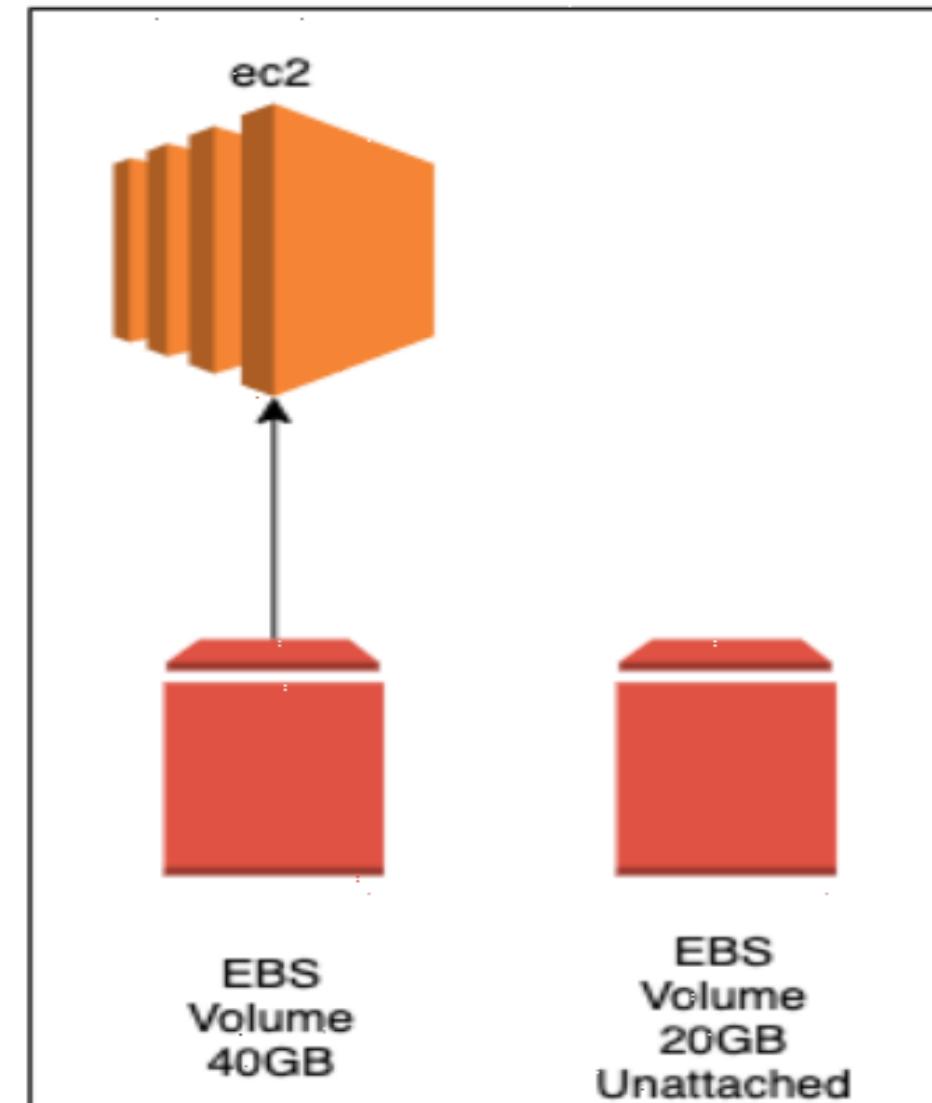
Storage Disk Addition

- Storage Addition, Filesystem Creation are very common tasks in day to day activity for any application.
- In traditional DC we have multiple dependencies on Hardware , Storage and OS teams. Hence a time consuming process.
- With AWS Cloud its no more a pain, we can add, delete EBS volume on the fly.
- Once EBS volume is attached, you can reclaim the storage in few clicks and save the cost of storage by deleting them when not needed.

us-west-1a



us-west-1b



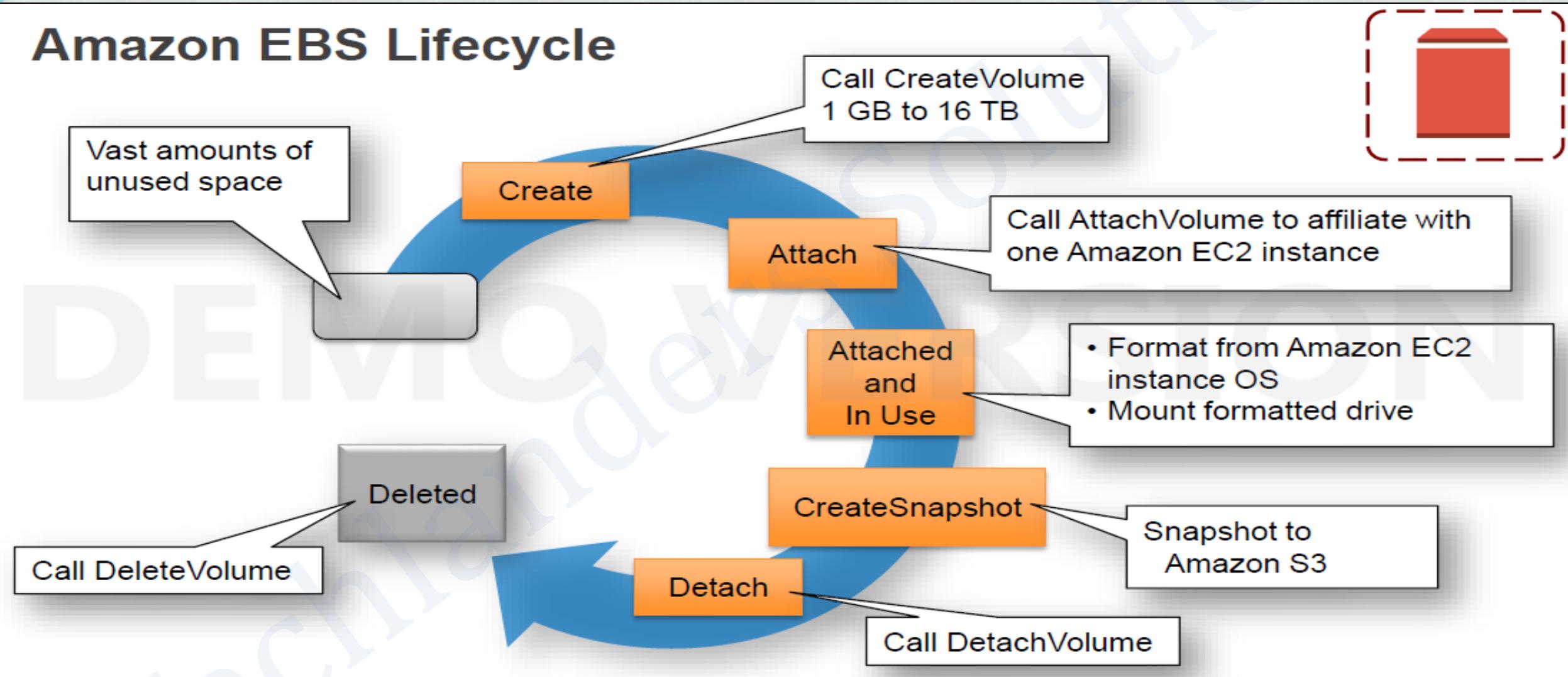
Amazon EBS Volumes

Amazon EBS

- EBS stands for ***Elastic Block Storage***
- **Persistent block level storage** volumes offering consistent and low-latency performance
- Automatically replicated within its Availability Zone
- Strictly bind to Availability Zone.

Amazon EBS Lifecycle

Amazon EBS Lifecycle



EBS Use Cases

- OS – Use for boot/root volume, secondary volumes
- Databases – Scales with your performance needs
- Enterprise applications – Provides reliable block storage to run mission-critical applications
- Business continuity – Minimize data loss and recovery time by regularly backing up using EBS Snapshots.
- Applications – Install and persist any application

EBS Volume Type Comparison

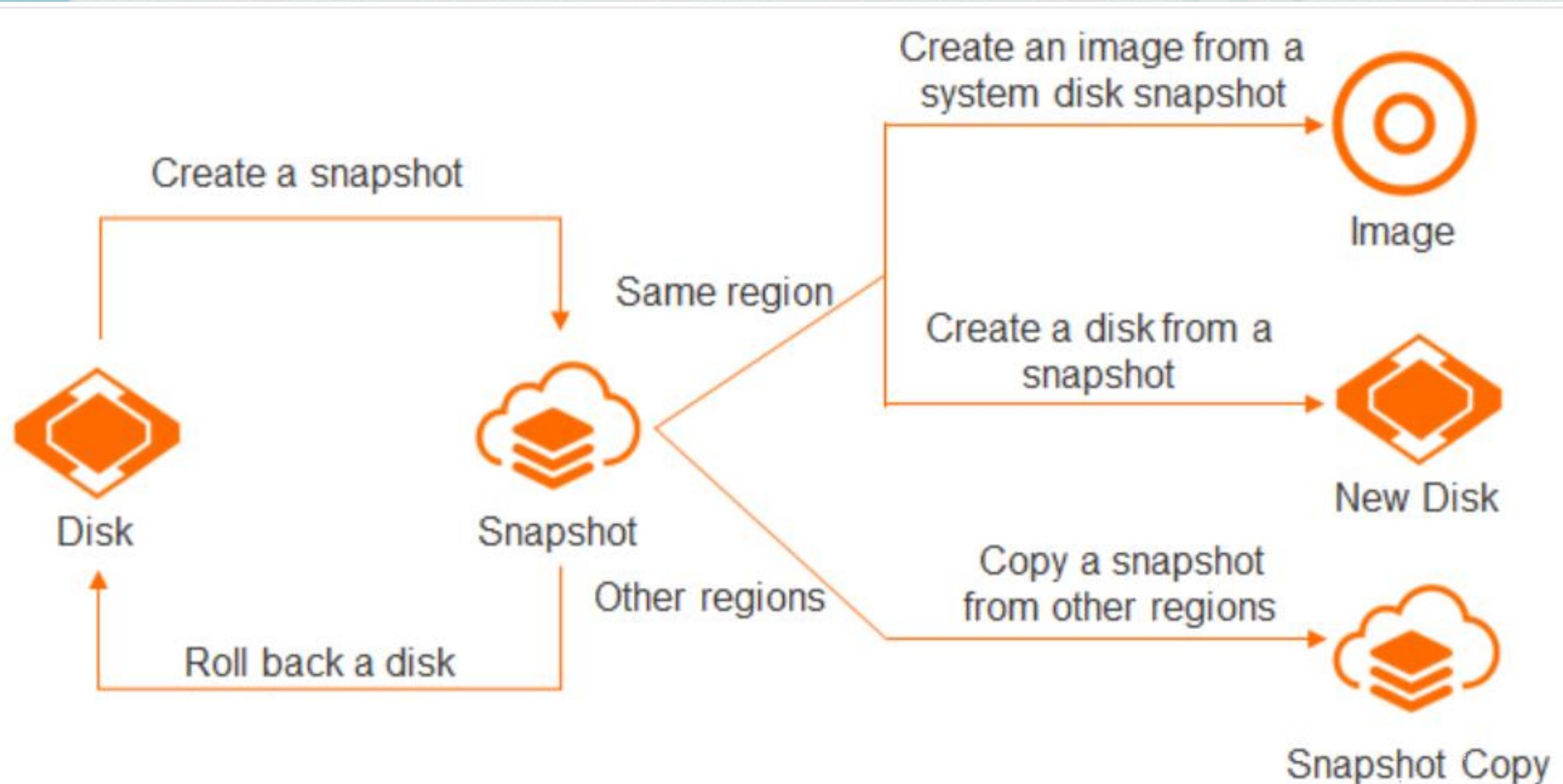
	Solid-State Drives (SSD)		Hard disk Drives (HDD)	
Volume Type	General Purpose SSD (gp2)*	Provisioned IOPS SSD (io1)	Throughput Optimized HDD (st1)	Cold HDD (sc1)
Description	General purpose SSD volume that balances price and performance for a wide variety of workloads	Highest-performance SSD volume for mission-critical low-latency or high-throughput workloads	Low cost HDD volume designed for frequently accessed, throughput-intensive workloads	Lowest cost HDD volume designed for less frequently accessed workloads
Use Cases	<ul style="list-style-type: none"> Recommended for most workloads System boot volumes Virtual desktops Low-latency interactive apps Development and test environments 	<ul style="list-style-type: none"> Critical business applications that require sustained IOPS performance, or more than 10,000 IOPS or 160 MiB/s of throughput per volume Large database workloads, such as: <ul style="list-style-type: none"> MongoDB Cassandra Microsoft SQL Server MySQL PostgreSQL Oracle 	<ul style="list-style-type: none"> Streaming workloads requiring consistent, fast throughput at a low price Big data Data warehouses Log processing Cannot be a boot volume 	<ul style="list-style-type: none"> Throughput-oriented storage for large volumes of data that is infrequently accessed Scenarios where the lowest storage cost is important Cannot be a boot volume
API Name	gp2	io1	st1	sc1
Volume Size	1 GiB - 16 TiB	4 GiB - 16 TiB	500 GiB - 16 TiB	500 GiB - 16 TiB
Max. IOPS**/Volume	10,000	32,000***	500	250
Max. Throughput/Volume	160 MiB/s	500 MiB/s†	500 MiB/s	250 MiB/s
Max. IOPS/Instance	80,000	80,000	80,000	80,000
Max. Throughput/Instance††	1,750 MiB/s	1,750 MiB/s	1,750 MiB/s	1,750 MiB/s
Dominant Performance Attribute	IOPS	IOPS	MiB/s	MiB/s

LAB 9 : Working with Volumes

- Go to EC2 Instance created in → EBS → Volume → Create Volume
- Create a general purpose SSD Volume “Data-volume” of 1 GB
- While creating volume, check different combination of IOPS and Throughputs, by selecting different-2 Volumes types
- Attach volume to a server
- Try to create another 1 gb volume but in a different AZ 2 and try to attach it with instance in AZ 1.
- Detach volume from Server
- Delete the volumes.
- Make sure the volume and EC2 instance should be in same AZ to attach.

Snapshot

- Snapshots are Point-In-Time Copy of Volumes (not Servers)
- You can create snapshots from your existing volumes assigned on dedicated instance and use them on any other instance with same configuration/data.
- Snapshots can be taken for root/Data volumes and no reboot/unmount of volume required for same.
- Snapshots can be Private, can be Shared-with-limited-aws-account or can be public. Permission can be modified for same.
- Further, volumes or Images (Templates) can be created from Snapshots.
- You can copy the snapshot to other regions (charges will be applied).



LAB 10 : Working with Snapshots

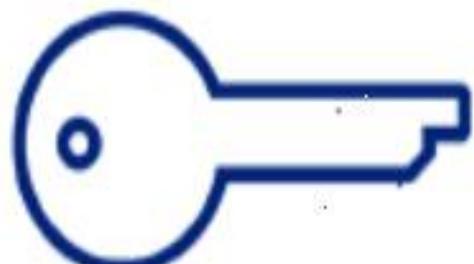
- Go to EC2 Instance in AZ 1 → EBS → Volumes → Create a data volume of 1 gb and attach it to the server in AZ1.
- Create a server in AZ2.
- Create a Snapshot of “Data-volume1” of server in AZ1
- Create “data-volume2” from snapshot in other AZ 2
- Attach “data-volume2” to server in AZ 2
- Delete the volume and delete the snapshot.

Encryption

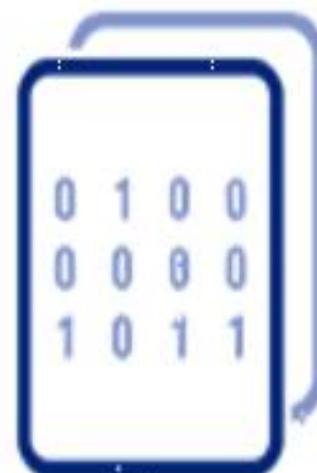
(used to protect sensitive information)



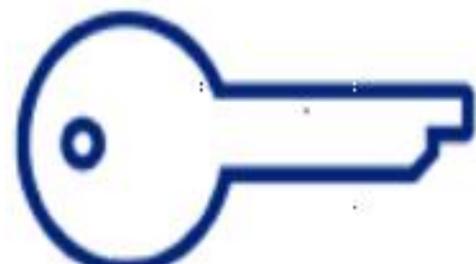
Plain text



Encryption



Encrypted text



Decryption



Plain text

EBS Encryption

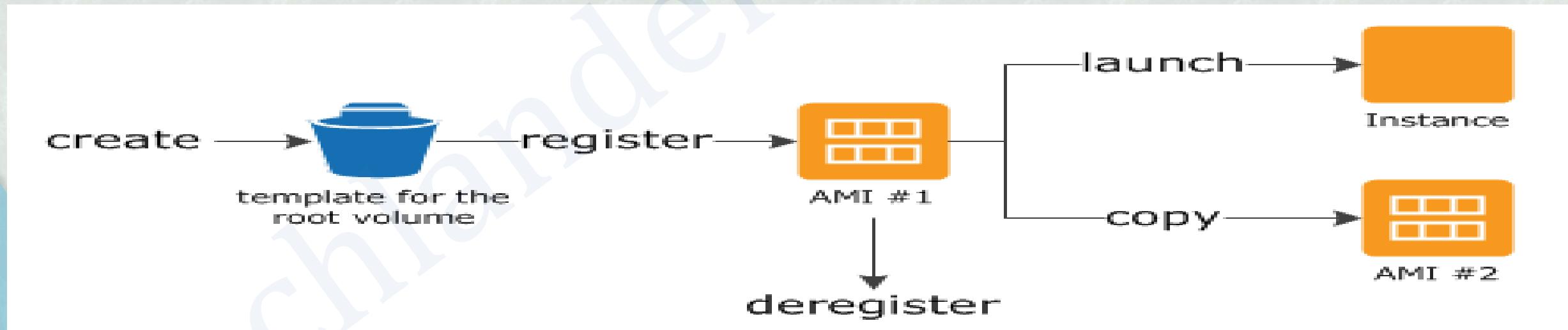
- EBS Encryption allows you to encrypt your data at rest or at flight.
- When you create an encrypted EBS volume and attach it to a supported instance type, the following types of data are encrypted:
 - Data at rest inside the volume
 - All data moving between the volume and the instance
 - All snapshots created from the volume
 - All volumes created from those snapshots
- You can expect the same IOPS performance on encrypted volumes as on unencrypted volumes, with a minimal effect on latency.

EBS Encryption

- EBS encrypts your volume with a data key using the industry-standard AES-256 algorithm.
- There is no direct way to :
 - encrypt an existing unencrypted volume
 - remove encryption from an encrypted volume (However, you can migrate data between encrypted and unencrypted volumes.)
 - remove encryption from an encrypted snapshot.

AMI's

- **AMI is a template for the your server** (example: an OS image, a webserver, an application server etc.)
- AMIs are snapshots of your running system and AWS modify same to make it generic.
- In runtime too, you can create AMI from an instance, but that can be inconsistent.



AMI's

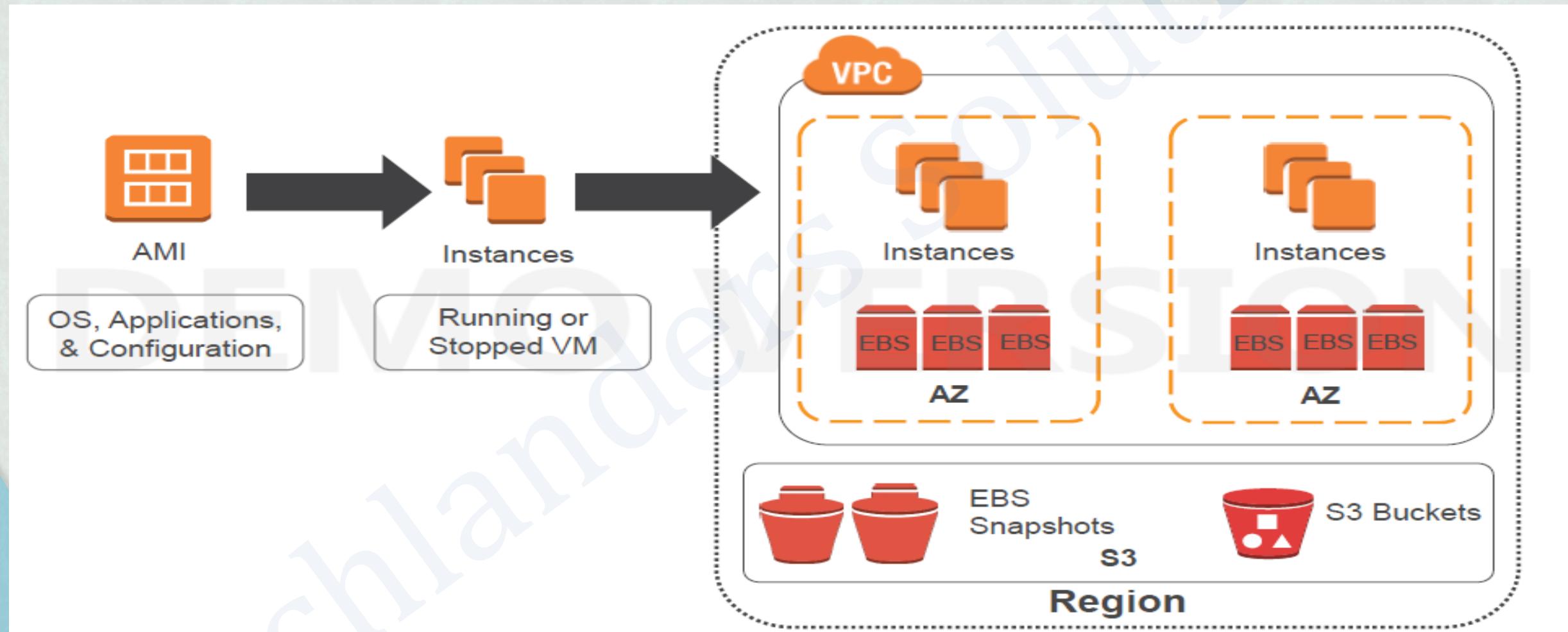
Generalizing an Image:

- Before others can use your Image, you should Generalize your AMI.
- Generalizing means removing few user specific configurations from that Image like:
 - Existing Users & their Credentials
 - Security Keys
 - User specific home directory
 - IP addresses & Hostnames
 - Event logs and volume specific settings

LAB 11 : Working with AMI

- Login to any EC2 instance and make some test file there.
- Now go to EC2 Instance → Select Server → Actions → Image → Create Image
- Type Image name and description
- Select whether Reboot or no reboot.
- Click on create Image
- Once finished, click on image and click launch instance
- Login to new server and check your test files there.

Amazon EC2 Instances



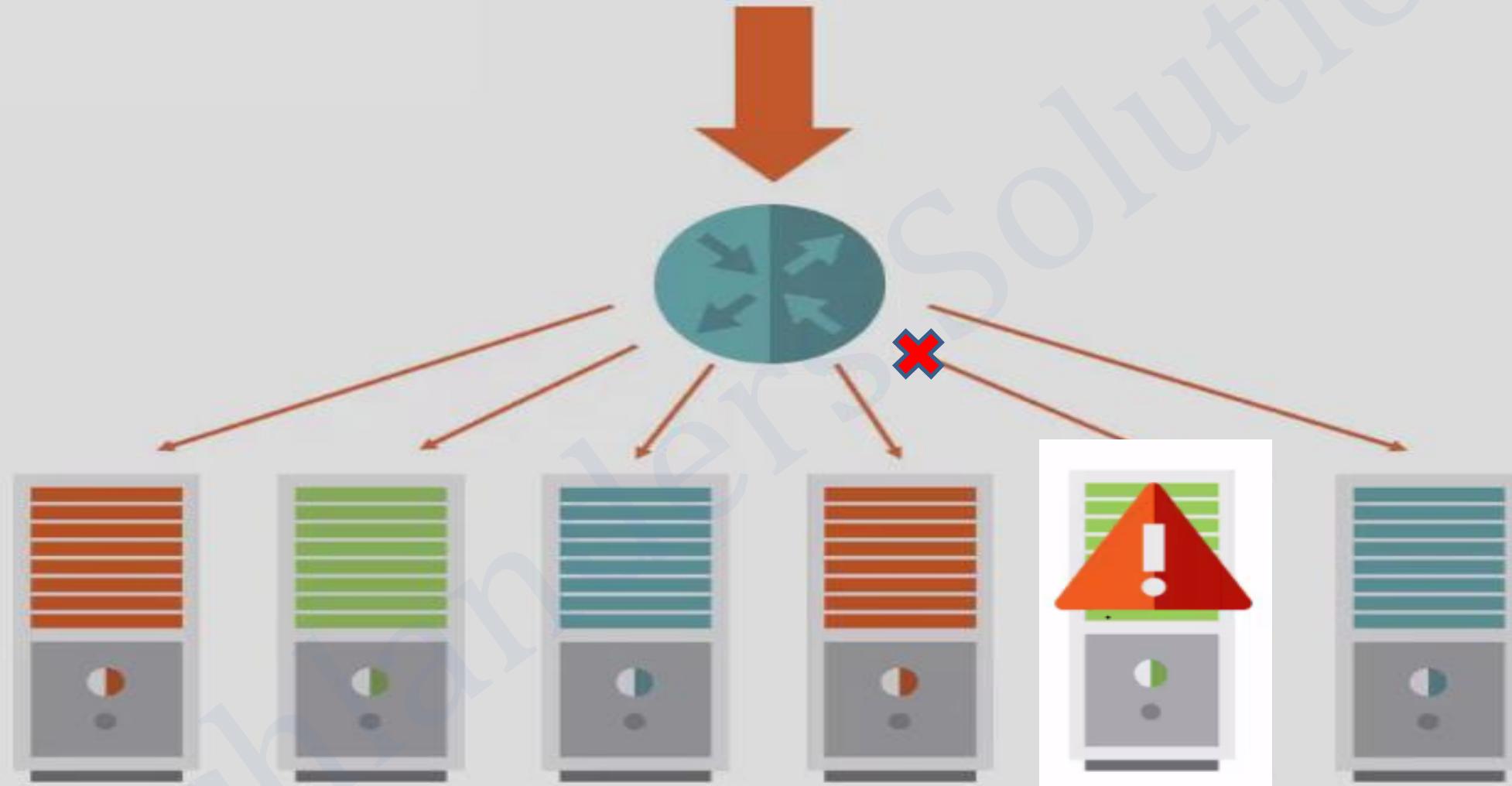
AWS

Load Balancer as a Service - ELB

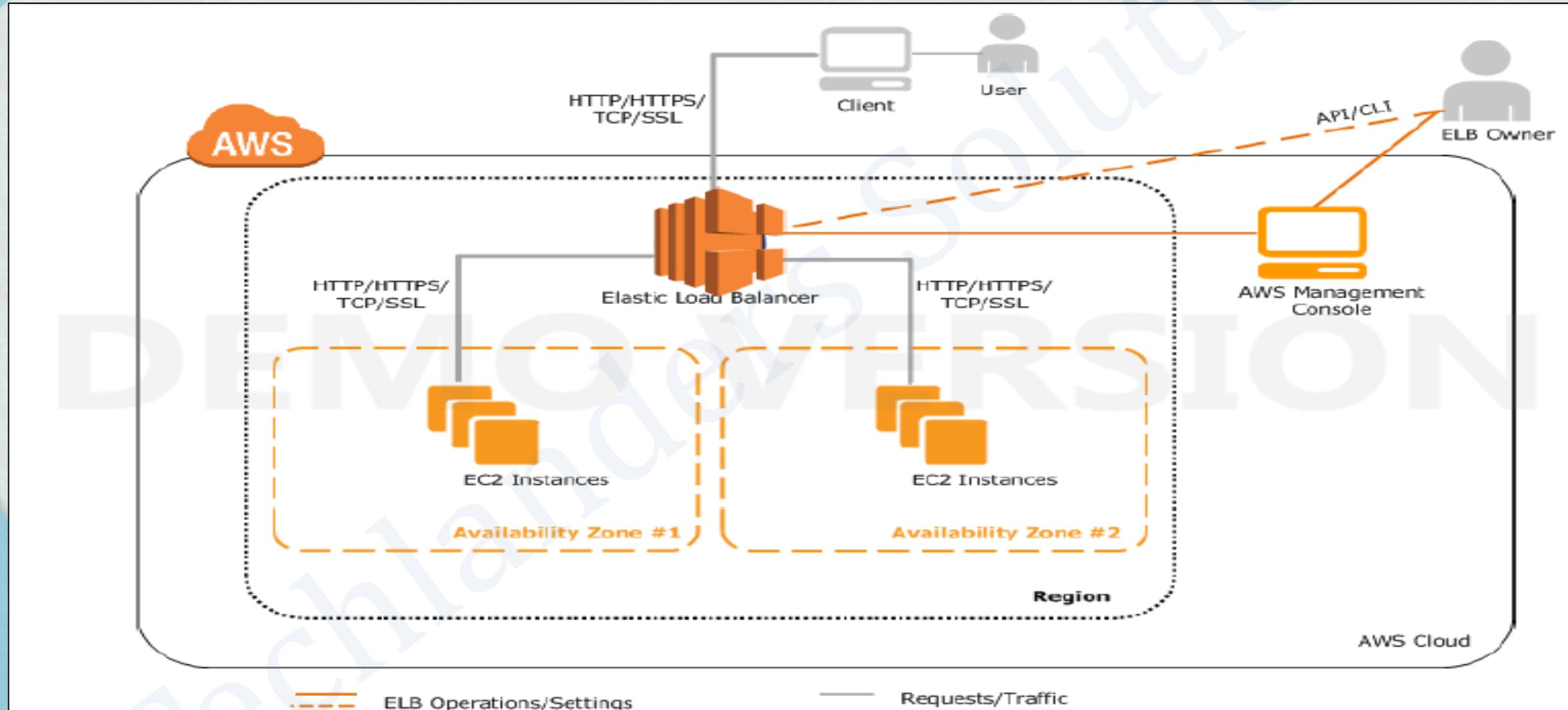
AWS Elastic Load Balancing

- **Distributes** traffic across multiple instances
- Supports **health checks** to detect unhealthy Amazon EC2 instances
- Supports the **routing and load balancing** of HTTP, HTTPS, and TCP traffic to Amazon EC2 instances
- Works within Region and across Availability zones, which means it can divert traffic to servers in different AZs, but within same Region.

Elastic Load Balancers

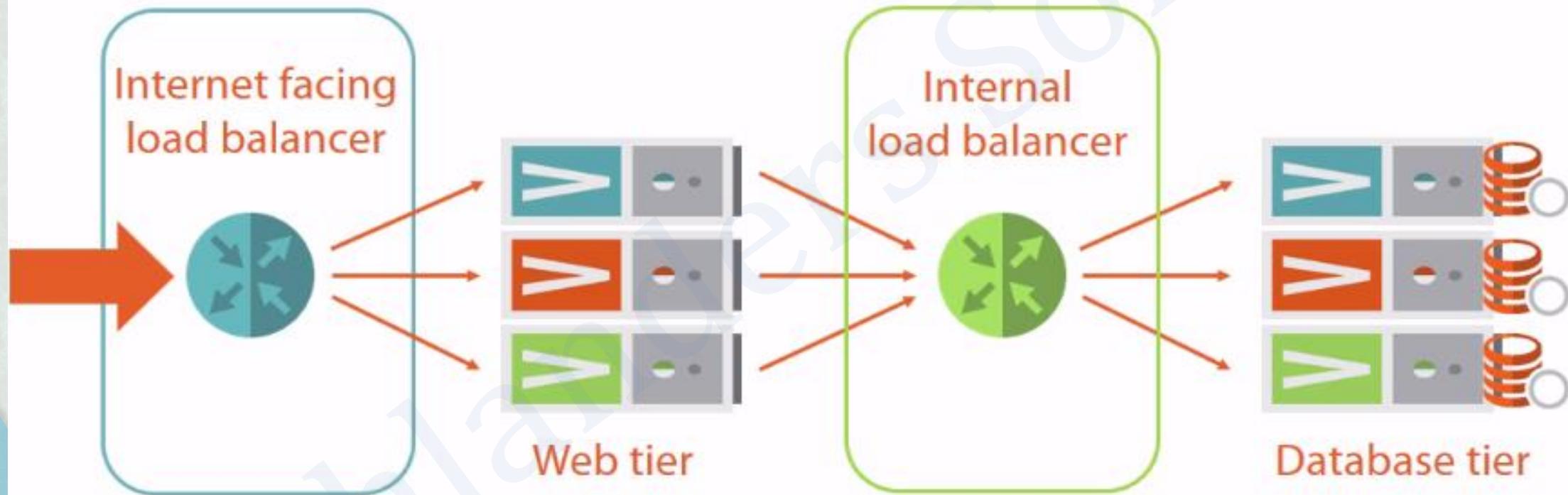


AWS Elastic Load Balancing



Load Balancer Types and Tiering

Tiered Applications & Load Balancers



ELB Terms

- **External/Internet Facing Load Balancer:** Receives Traffic from Open Internet. So carrying a public IP (or DNS name).
- **Internal Load Balancer:** Receives traffic from other VMs in same Network.
- **Frontend IP :** Public IP (Private IP in case of Internal LB) address for the incoming network traffic.
- **Target group :** Backend VM's Group which will receive actual traffic
- **Target:** Backend VM machines
- **Listener:** Port on ELB, which will accept request from end-client
- **Health checks:** Determines if a VM in the pool is healthy.

ELB Types

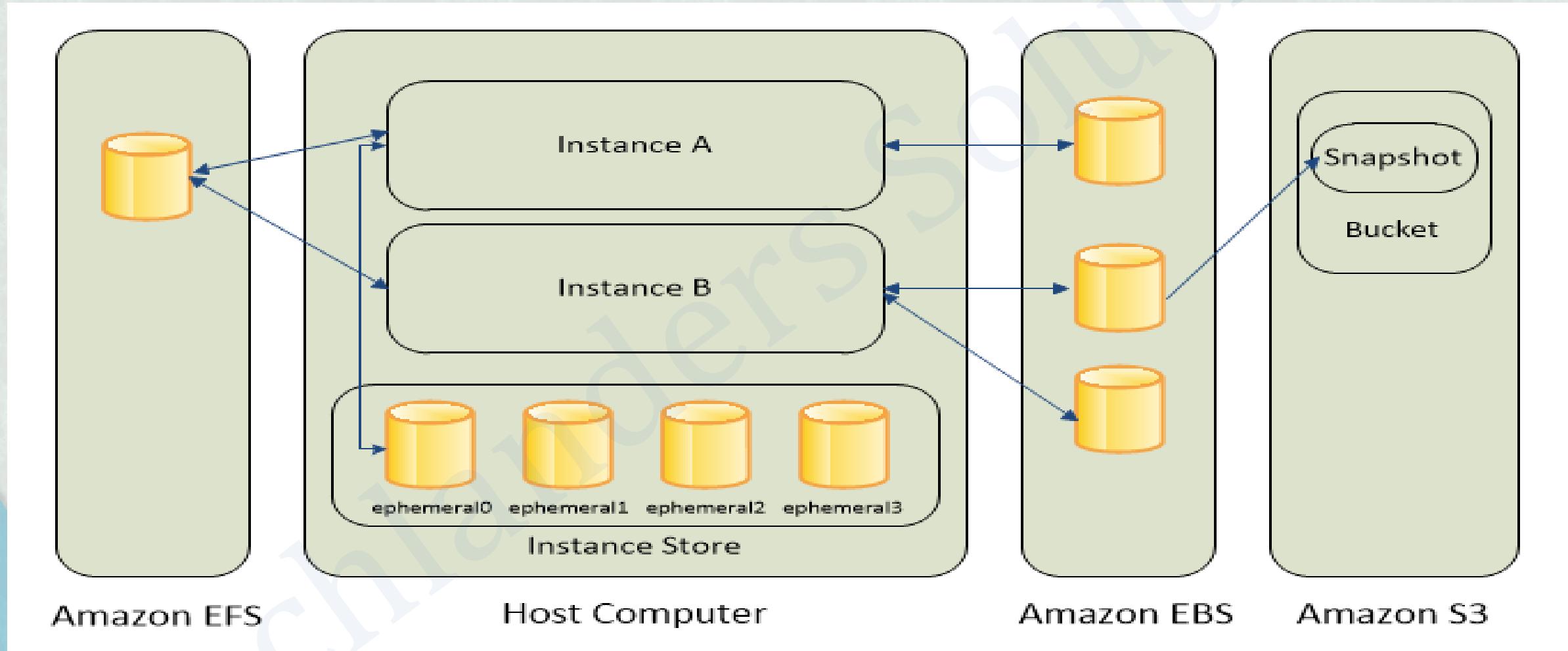
- **Application Load Balancer:**
 - An advance Load balancer which works on OSI Layer 7 (Application layer).
 - It contains flexible feature set (i.e. TLS termination, SSL handling etc.) for your web applications with HTTP and HTTPS traffic.
- **Network Load Balancer:**
 - Most commonly used Load balancer which works at any TCP port.
 - Works on Layer 4 (Transport layer).
 - LB with very high performance and capable of handling millions of requests per second while maintaining ultra-low latencies.
- **Classic Load Balancer:**

Old type of Load-balancer service which works with EC2-Classic network. Not being used any more.

LAB 12 : Working with ELB

- Create two AmazonLinux servers in different-2 Azs (us-east-1a &us-east1b)
- Make sure to add html script user-data for post-verifying on both servers.
- Create a security for alb---ec2 for access of traffic inside ec2 instances from our alb giving port 22 and 80 access to public for now and configure them on both servers us-east-1a and 1b while creating the instances.
- After creating both servers, create the application load balancer and while creating the LB , create a separate security group for ALB to give access to external users to our load balancer.
- Create a target group adding both instances to that target group.
- After adding a target group , click on create load balancer and wait for sometime.
- Now add user—ec2 securitygroup to alb—ec2instances security group to disallow the external users to access our internal instances behind the load balancers directly.

AWS Storage Services





AWS Auto-Scaling

AWS Auto Scaling

What do you understand by “Scaling”?

AWS Auto Scaling

What is the difference between Scale-up and Scale-out?

Scaling your Infra



AWS Auto Scaling

What is Auto-Scaling and Why required?

AWS Auto Scaling

- **Scale (out/in)** your Amazon EC2 capacity **automatically**
- Well-suited for applications that experience **variability in usage**
- Available at no additional charge
- Application instance Scale Out/In is done by service, based on the alerts or defined schedule.
- You can use LB service with Auto-scaling.
- Application should be able to handle scale-out and scale-in servers.
- Auto-scaling can span across-AZ, but remains within Region.

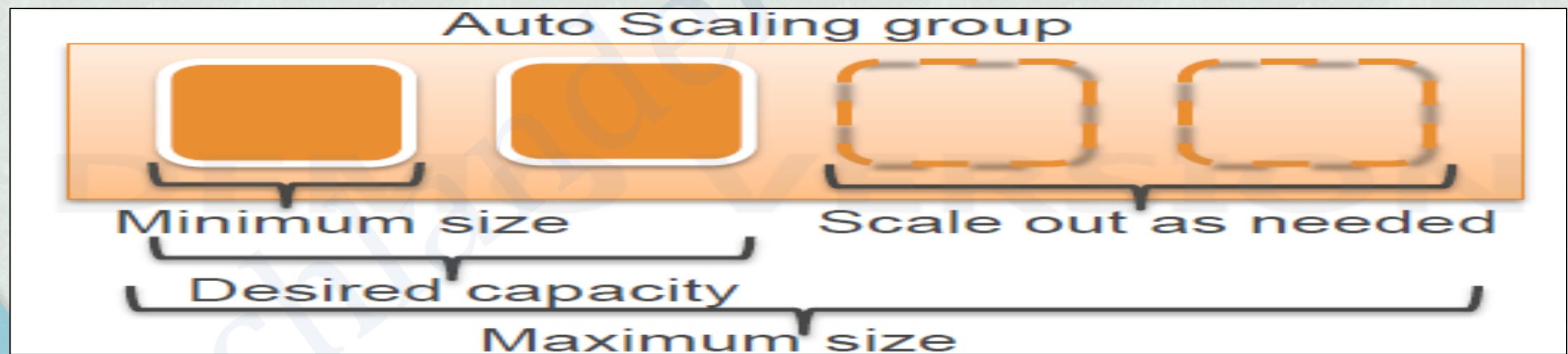
AWS Auto Scaling

Launch template : A launch configuration is a template that an Auto Scaling group uses to launch EC2 instances. When you create a launch template, you can specify:

- AMI ID
- Instance type
- Key pair
- Security groups
- Block device mapping
- User data

AWS AutoScaling

- **Auto Scaling Group:** Contain a collection of EC2 instances that share similar characteristics.
- Instances in an Auto Scaling group are treated as a logical grouping for the purpose of instance scaling and management.



Scaling conditions

Alert Based

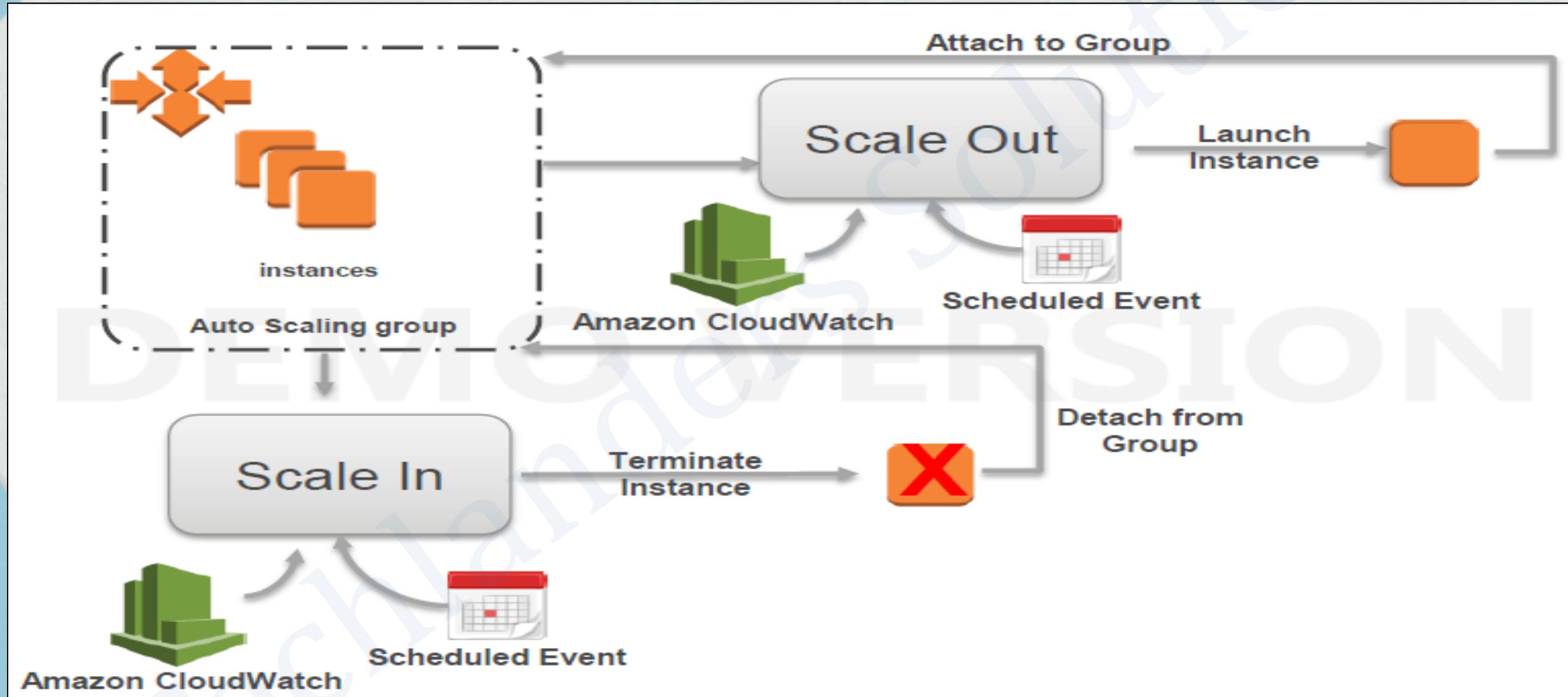
- Based on Health check alerts of Application instances
- Alerts can be based on CPU/Network/Disk IO etc.
- Both Scale-in and Scale out can be done based on Alerts
- Multiple Scale policies can be created and maintained

Scheduled Scaling

- Based on Scheduled Events
- Scaling can be configured to Auto scale once or re-occurrence basis.
- Scheduling can be done for years in advance
- Multiple Schedule policies can be created and maintained.

Note: Both Alert based and Scheduled scaling can co-exist simultaneously.

AWS AutoScaling



LAB 13 : Create AS Group

Console Access: <https://console.aws.amazon.com>

Go to aws console page

- From Services → EC2 → Auto Scaling → Launch temeplate
- Click on Create Launch template after adding required configurations like AMI, Configuration Details, Storage, Security Groups .
- Next step is to create an Autoscaling Group, click on create Auto scaling group from the launched template.
- Group Size is the key here (the number of instances group should have any time, this is also called desired capacity)
- Click next with the policies(optional)
- Click Create AutoScaling group and the instance will start automatically

LAB 14 : Work with AS Group

Console Access: <https://aws.amazon.com/>

Exercice1:

- **Add one more value in Auto scale group** and check one more instance will start getting created.
- Note: In this case we need to modify Desired/Minimum and Maximum attributes values for auto scaling group

Exercice2:

- **Deletion of Auto scale group**
- It will delete all of the instance also (created from Autoscale group).