

Cloud computing

RISR CESI

Cloud Computing

Introduction

Introduction

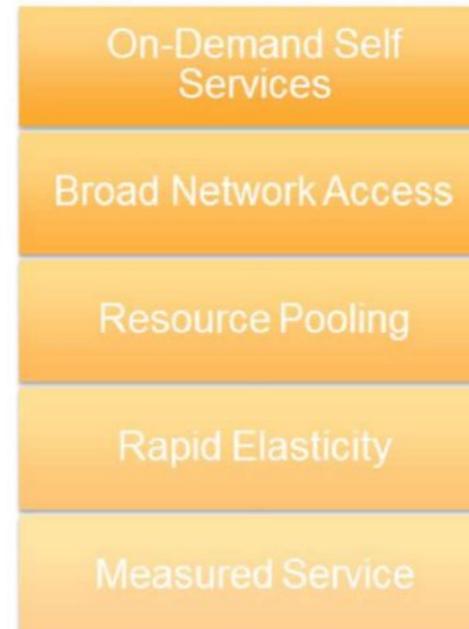
What is cloud computing?

Cloud computing is on-demand delivery of IT resources and applications via the Internet with pay-as-you-go pricing.



Essential Characteristics

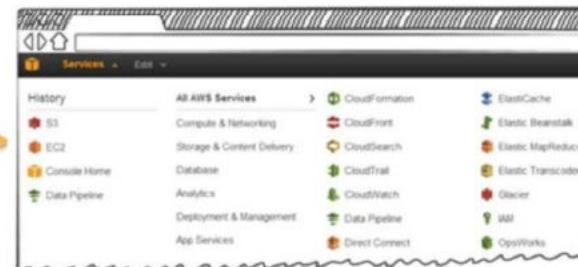
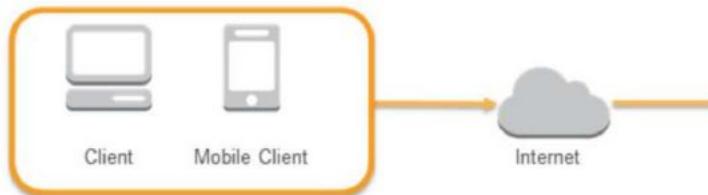
Essential Characteristics of Cloud Computing



Essential Characteristics

On-Demand Self Services & Broad Network Access

- User provisions computing resources as needed.
 - User interacts with cloud service provider through an online control panel.
 - Clear solutions are available through a variety of network-connected devices and over varying platforms.



Essential Characteristics

Resource Pooling

Securely separate resources to service multiple customers.



Essential Characteristics

Rapid Elasticity

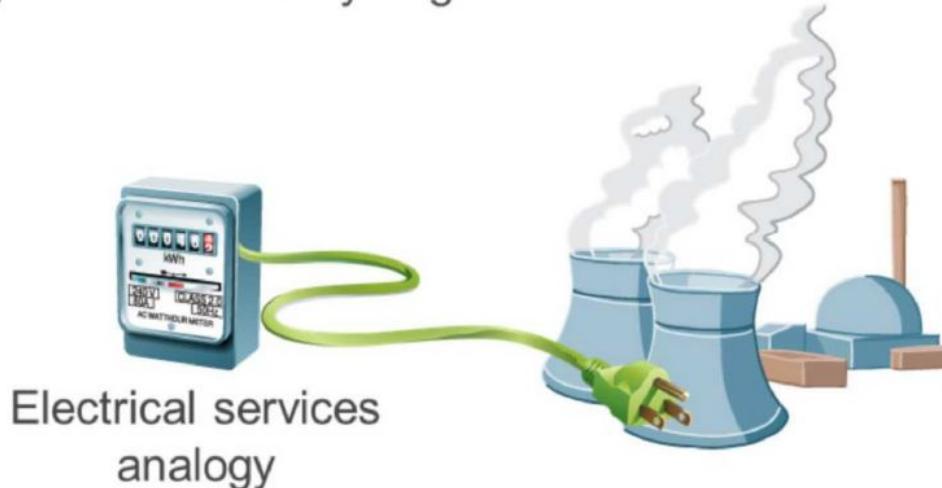
Resources are quickly scalable and flexible based on business needs.



Essential Characteristics

Measured Service

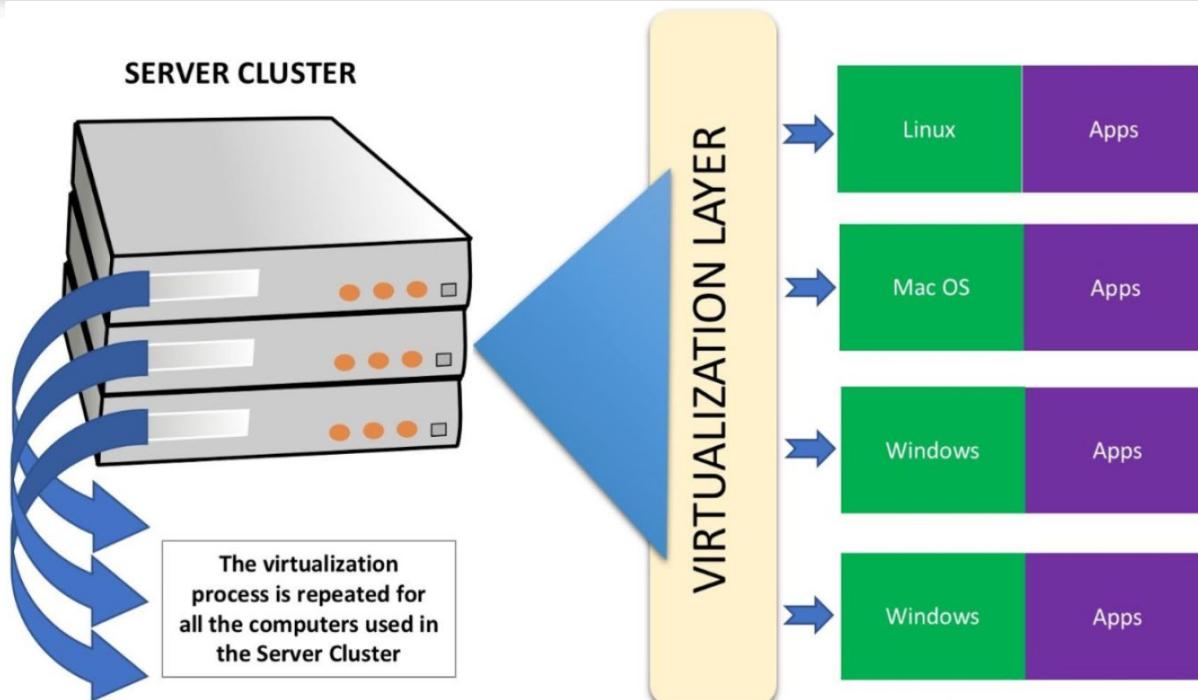
Pay for services as you go.



Reminder

Infrastructure

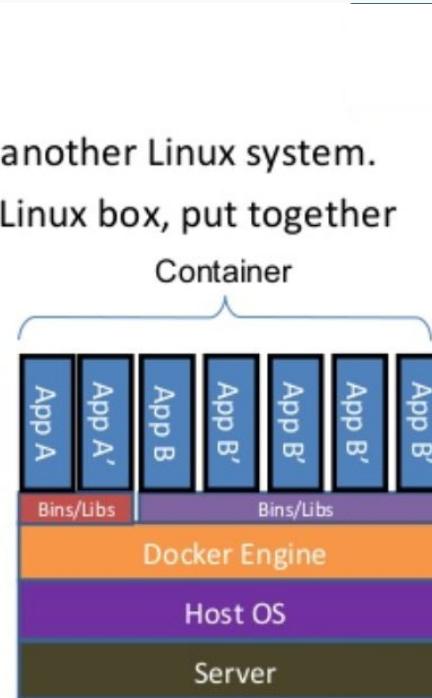
Virtualisation



Containers

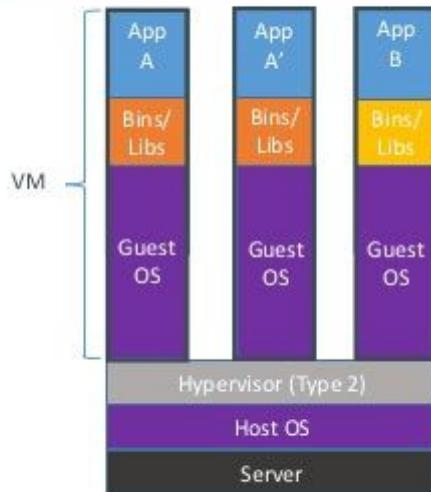
Containers

- It allows us to run a Linux system within another Linux system.
- A container is a group of processes on a Linux box, put together is an isolated environment.
 - From the inside it looks like a VM
 - From the outside, it looks like normal processes

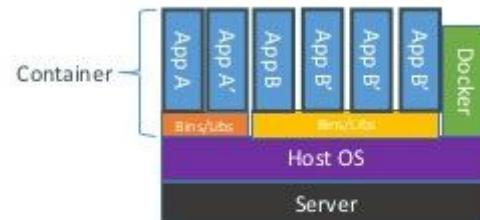


Containers vs VMs

Containers vs. VMs



Containers are isolated,
but share OS and, where
appropriate, bins/libraries



- Continuous Deployment
- Workload Portability
- Software Quality and Compliance
- Cost Optimization
- Infrastructure Agnostic

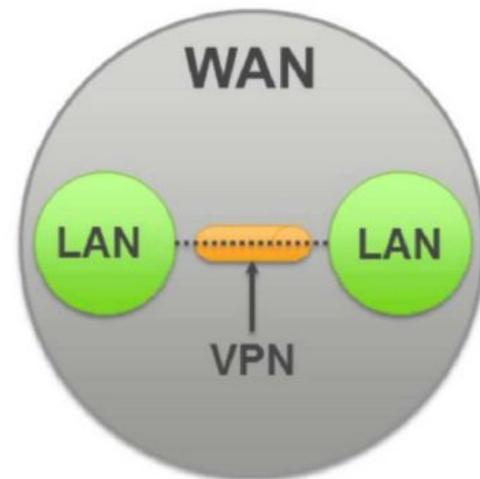
Network

What is a Network?

A network is two or more computers linked to share resources, exchange files, or allow electronic communications.

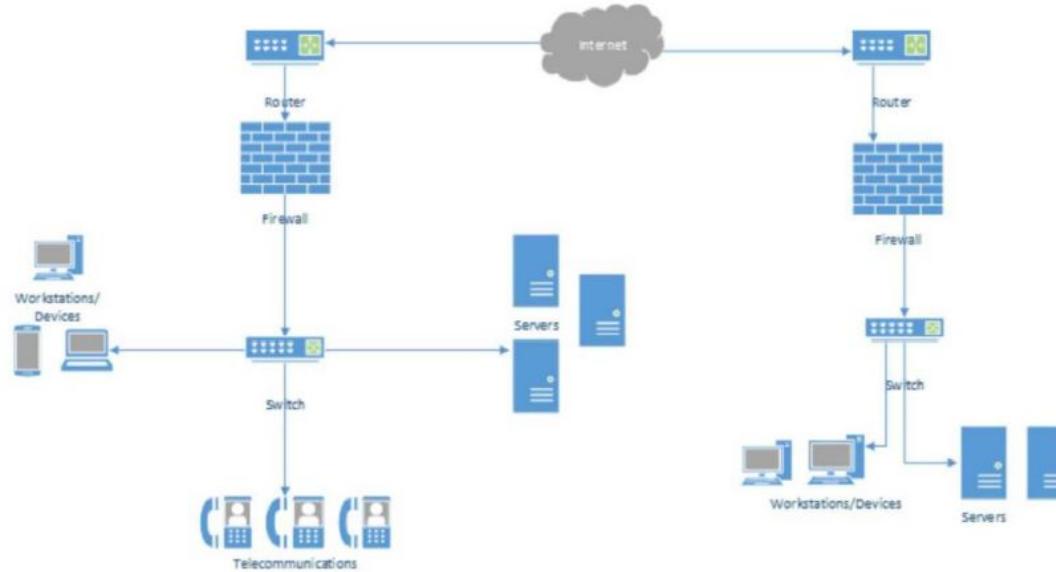
Network Types:

- Local Area Network (LAN)
- Wide Area Network (WAN)
- Virtual Private Network (VPN)

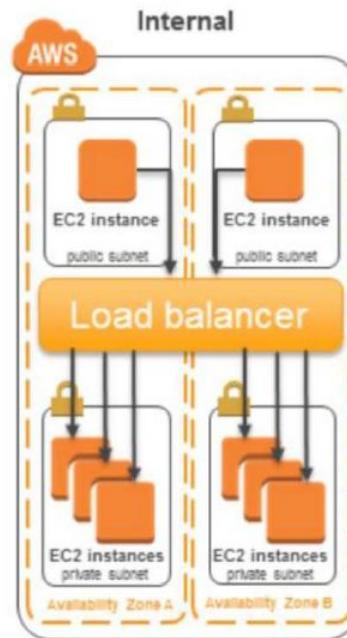
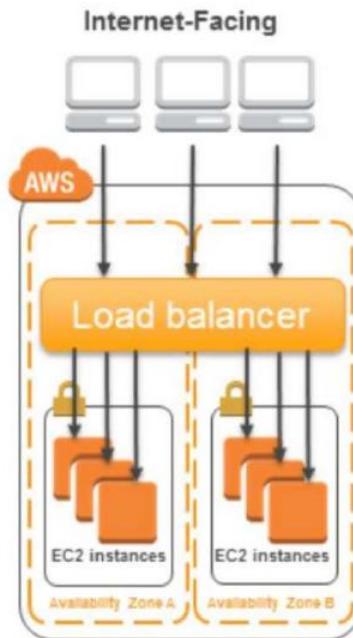


Network

Physical Network Hardware/Devices



Load Balancing

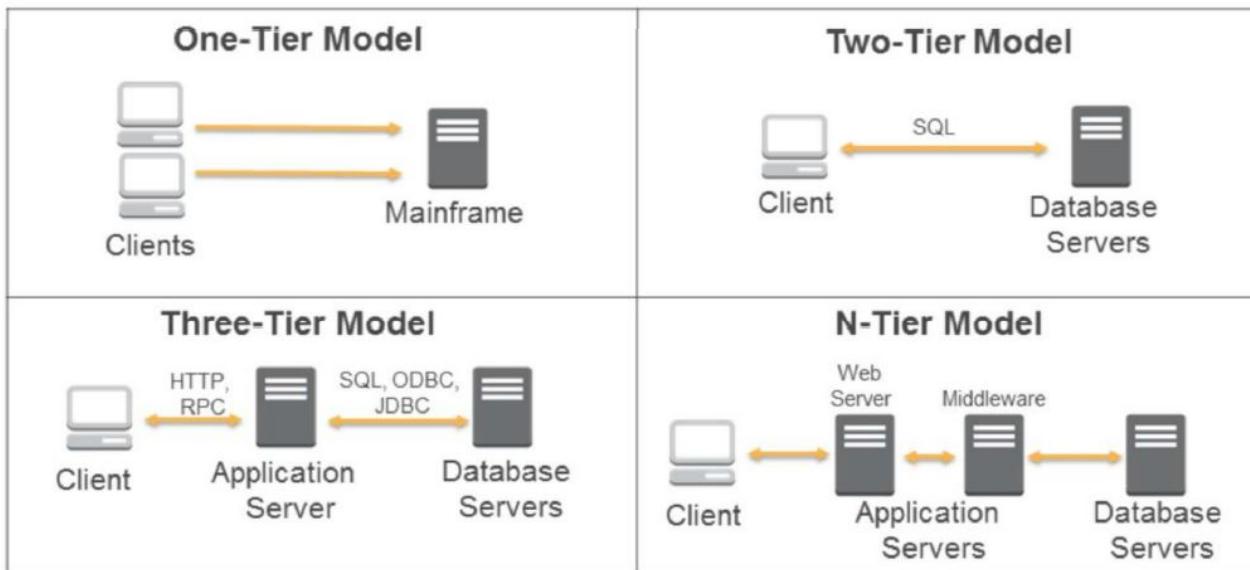


Reminder

Application models

Application models

Application Design Model



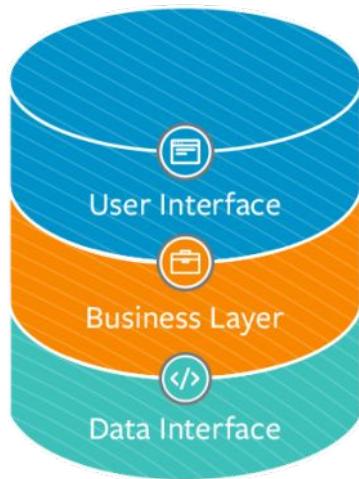
Application models

Web Services Model

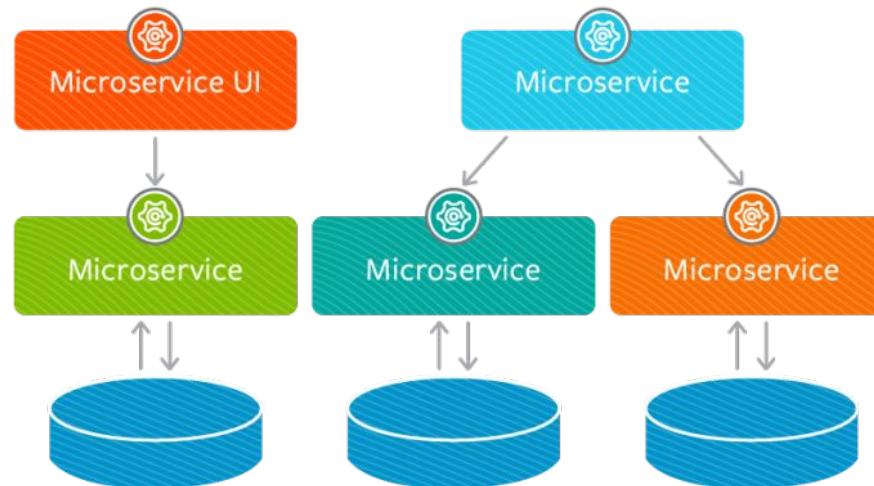


Application models

Monolithic
Architecture

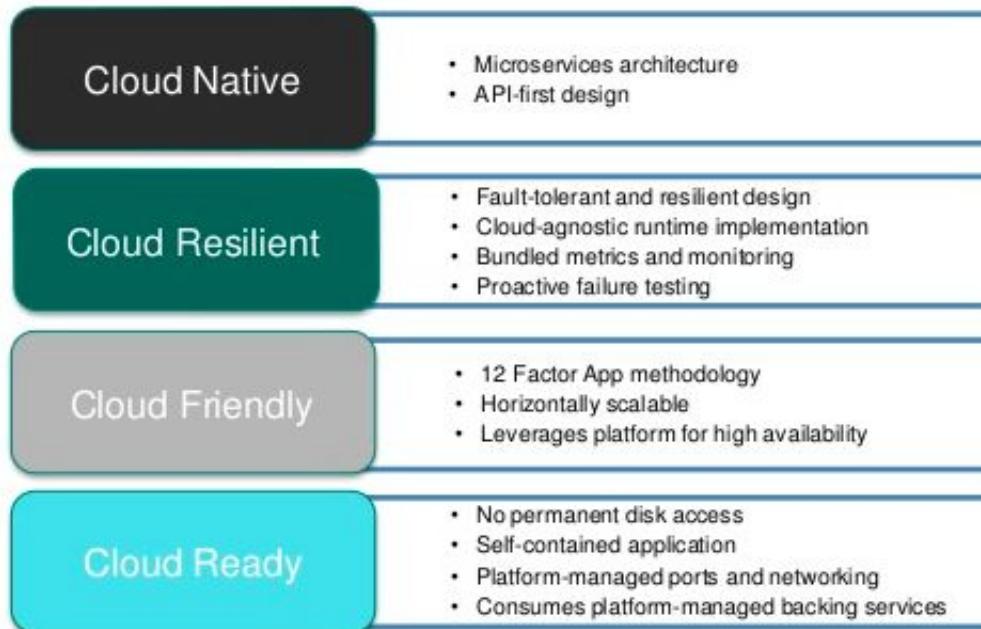


Microservices
Architecture



Application models

Cloud Native Maturity Model



Cloud Computing

Types

Cloud Types

On-premise: Describes software and hardware that is housed and directly managed inside a business' physical location. On-premise hardware and software is usually always on and available.

Cloud: In a hosted cloud, the service provider maintains the systems on their server, accessible by the enterprise at any given time with related processes taken care of by the host-cloud service provider.



Cloud vs On-Prem



Cloud software is delivered and hosted on the Internet.



On-Premise software is managed and hosted locally on a device.

No installation, or increased costs for additional equipment or infrastructure.

Large initial cash outlay for equipment and replacement of outdated equipment.

The vendor handles updates and product enhancements and notifies you of changes.

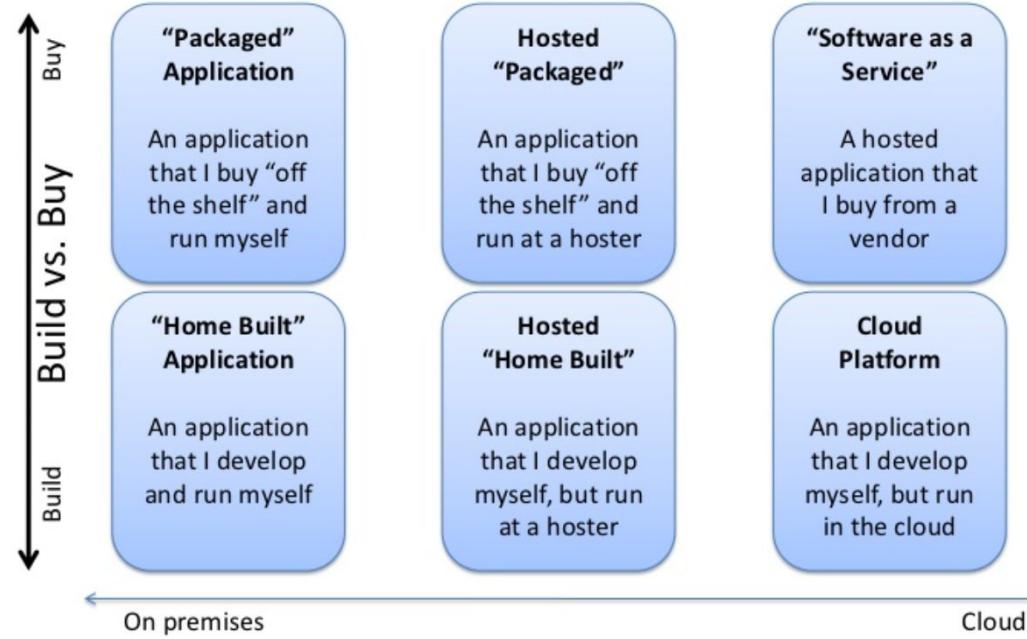
"Feature lock-in"; usually limited to the solution features the product came with at time of purchase.

Accessibility for a broad range of devices through mobile and web applications.

Access generally limited to the device(s) the software was originally installed on.

Types

On premises vs. in the cloud



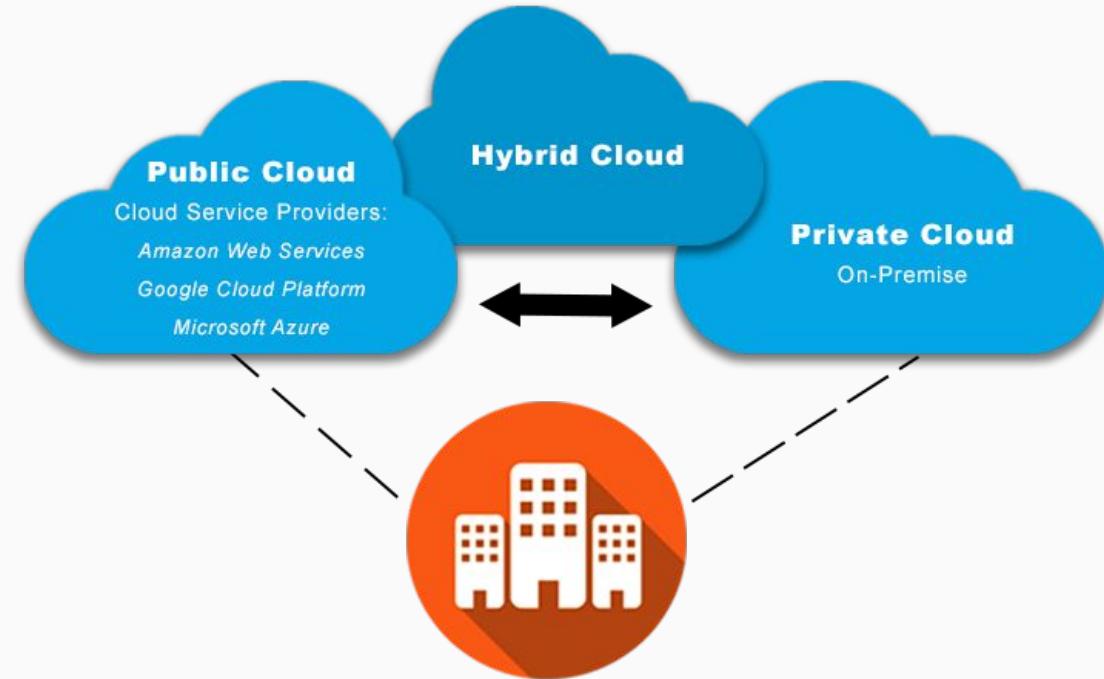
Hybrid Cloud

Hybrid cloud benefits include the following management tools:

Choice: The additional choice of multiple environments provides flexibility and the ability to avoid vendor lock-in.

Disaster Avoidance: Multiple environments ensures compute resources are always available to avoid downtime due to natural disaster or human error.

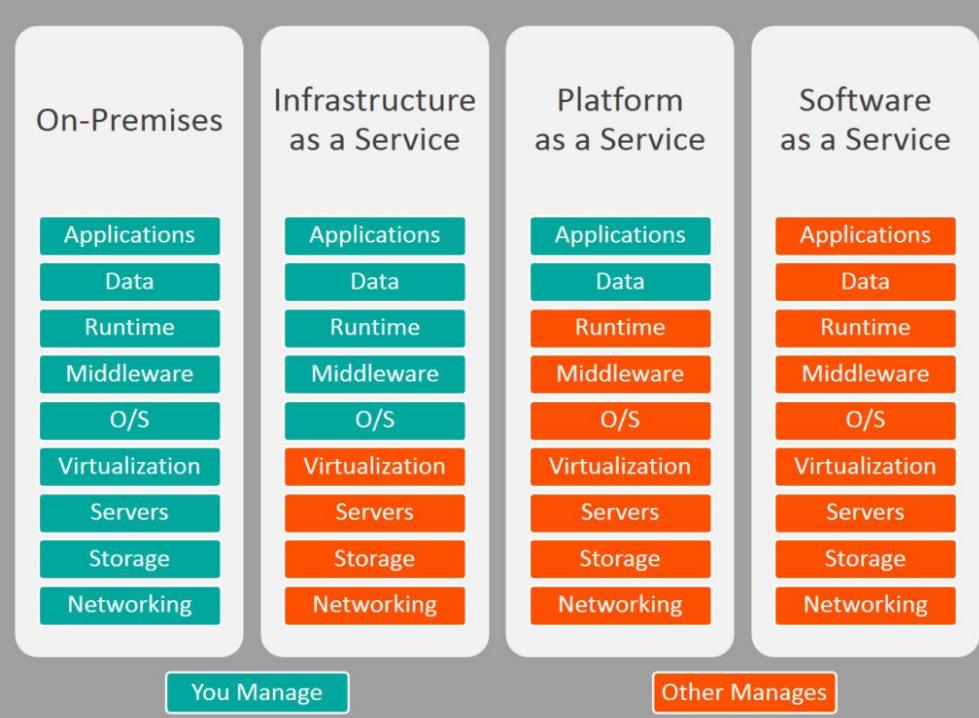
Compliance: Many hybrid cloud environments can help enterprises achieve their goals for governance, risk management and compliance regulations.



Cloud Computing

Models

Cloud computing models



Cloud models come in three types, SaaS , IaaS and PaaS.

Each of the cloud models has their own set of benefits that could serve the needs of various businesses

Infrastructure as a Service

Infrastructure
as a Service

Applications

Data

Runtime

Middleware

O/S

Virtualization

Servers

Storage

Networking

IaaS emulates the hardware that companies traditionally purchased to create internal and external networks.

The chore of installing hardware and connecting it to physical networks has been replaced with the task of creating robust configurations. Almost every part of a traditional network is offered as a cloud service.

Examples: Servers, Firewalls, Routers

Platform as a Service

Platform
as a Service

Applications

Data

Runtime

Middleware

O/S

Virtualization

Servers

Storage

Networking

PaaS products sit in the middle between IaaS and SaaS.

PaaS services allow customers to create a customized experience on top of an existing set of tools.

Examples: Google App Engine, Beanstalk, RDS

Software as a Service

Software
as a Service

Applications

Data

Runtime

Middleware

O/S

Virtualization

Servers

Storage

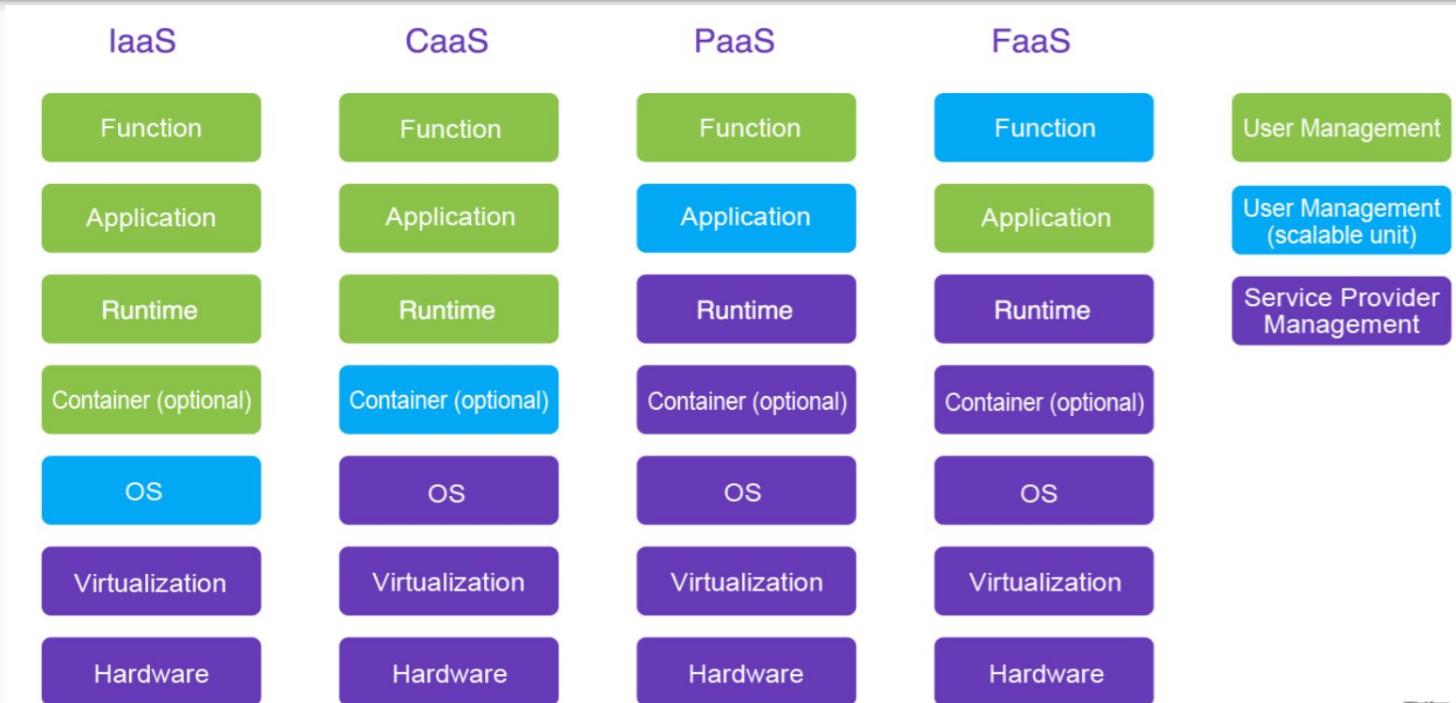
Networking

Software as a Service describes any Internet-based product that provides a specific set of operations. This includes everything from email to contact management.

Interacting with a SaaS product had traditionally been done through a web browser.

Examples: Mail services, online storage (Dropbox)

Advanced Cloud models



Container as a Service

CaaS

Function

Application

Runtime

Container (optional)

os

Virtualization

Hardware

CaaS (also called Container-As-A-Service) is a business model in which the cloud service provider offers container based virtualization services as an online service that is scalable.

CaaS enables the users to use the container services without the need of having the required infrastructure.

Function as a Service

FaaS

Function

Application

Runtime

Container (optional)

OS

Virtualization

Hardware

Function as a service (FaaS) is a cloud computing model that enables users to develop applications and deploy functionalities without maintaining a server, increasing process efficiency.

This is typically utilized when creating microservices such as web applications, data processors, chatbots and IT automation.

Examples: Web apps, Backends, Data/stream processing, Chatbots, Scheduled tasks, IT Automation

Cloud Providers

Introduction

Introduction

A cloud service provider, or CSP, is a company that offers some component of cloud computing -- typically infrastructure as a service (IaaS), software as a service (SaaS) or platform as a service (PaaS) -- to other businesses or individuals.

Some cloud service providers have differentiated themselves by tailoring their offerings to a vertical market requirements.

Major cloud providers



Azure

Great for enterprises that already use a lot of Microsoft software



AWS

The most mature with the widest array of products and services



Google Cloud

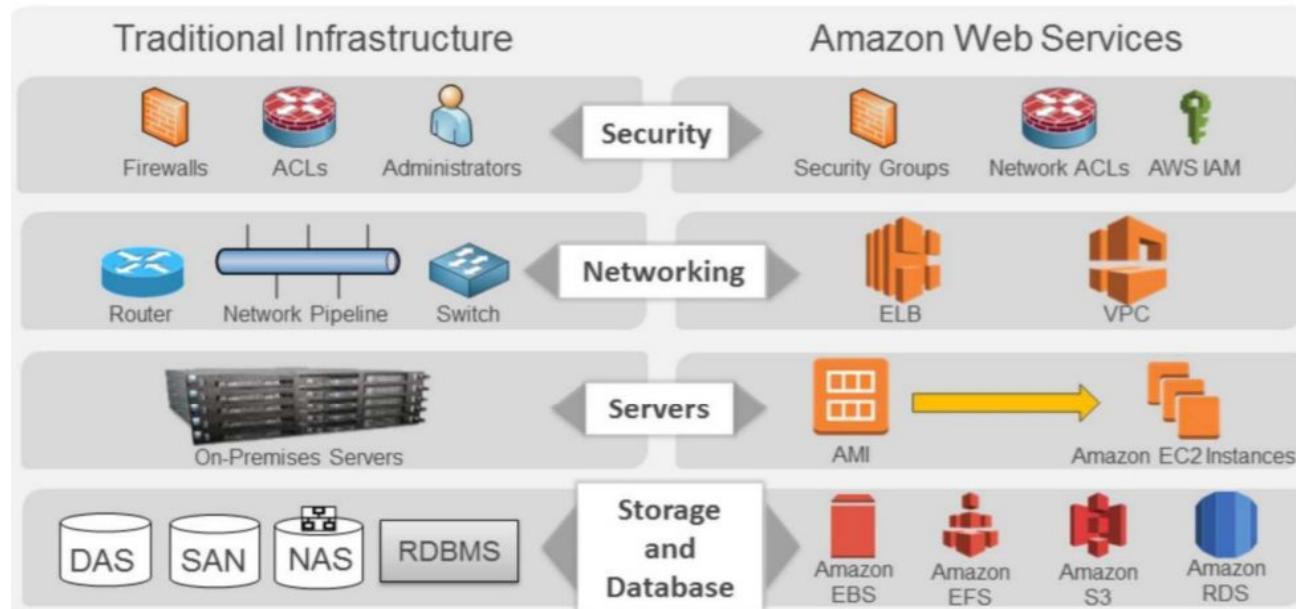
Generally the cheapest of the three

Differents names, same services

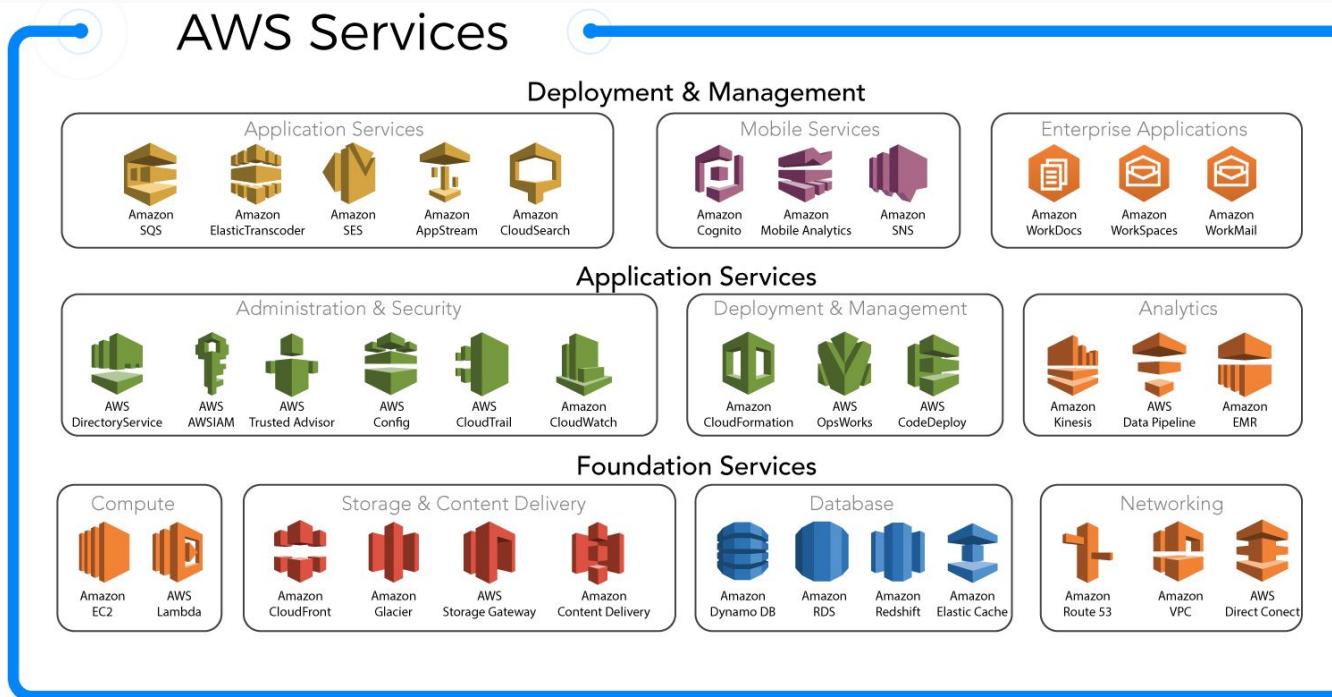
| Category | Service | Amazon Web Services™ | Azure | Google Cloud Platform |
|----------|--------------------------------|--|--|---|
| Compute | Shared Web hosting | |  Azure shared App Services 🔗 | |
| Compute | Virtual Server |  Amazon EC2 🔗 |  Azure Virtual Machine 🔗 |  Compute Engine 🔗 |
| Compute | Bare Metal Server |  Amazon EC2 Bare Metal Instance (Preview) 🔗 |  Azure Bare Metal Servers (Large Instance Only for SAP Hana) 🔗 | |
| Compute | Virtual Dedicated Host |  Amazon EC2 Dedicated Hosts 🔗 | |  Sole Tenant Node (Beta) 🔗 |
| Compute | Container Registration Service |  Amazon EC2 Container Registry 🔗 |  Azure Container Registry 🔗 |  Container Registry 🔗 |
| Compute | Container Management Service |  Amazon EC2 Container Service 🔗  Amazon Elastic Container Service for Kubernetes (EKS) 🔗 |  Azure Kubernetes Service (AKS) 🔗  Azure Container Instances 🔗 |  Kubernetes Engine 🔗 |

On-premise and AWS

AWS Core Infrastructure and Services



Main services



Much more services



Cloud Providers

Regions & Availability Zones

Infrastructure

AWS Global Infrastructure

Regions

- Geographic locations
- Consist of **at least two** Availability Zones

Availability Zones

- Clusters of data centers
- **Isolated from failures** in other Availability Zones

Infrastructure

AWS Global Infrastructure



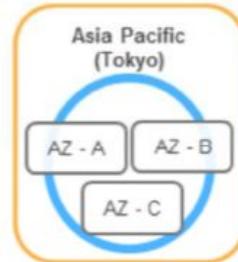
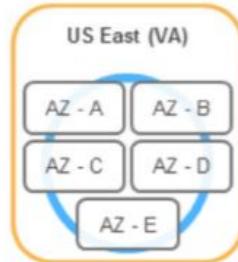
Regions & AZs

AWS Global Infrastructure

At least 2 Availability Zones per region.

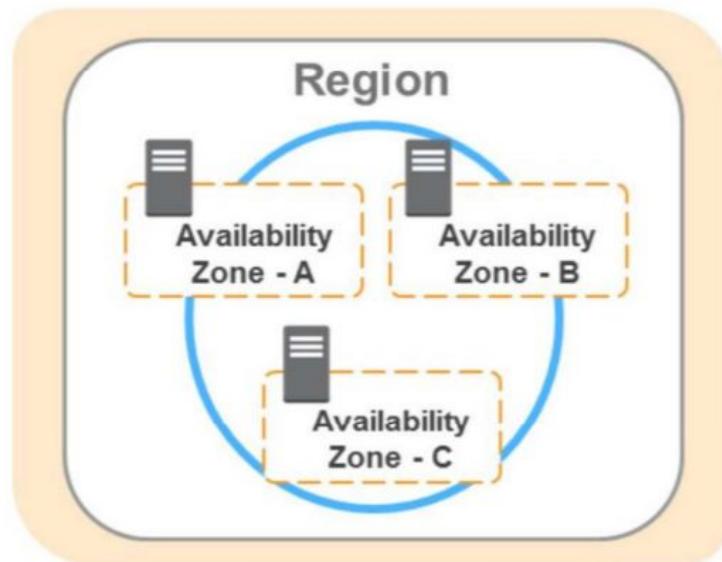
Examples:

- 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



High Availability

High Availability Using Multi-AZ Deployments



Cloud Providers

Servers

Instances

Amazon Elastic Compute Cloud (EC2)



Amazon
EC2

- **Resizable** compute capacity
- Complete control of your computing resources
- **Reduced time required** to obtain and boot new server instances

Instances

Amazon EC2 Facts

- **Scale capacity** as your computing requirements change
- Pay only for capacity that you actually use
- Choose **Linux or Windows**
- Deploy across **AWS Regions** and **Availability Zones** for reliability
- Use **tags** to help manage your Amazon EC2 resources



Instances

Launching an Amazon EC2 Instance via the Management Console



1. **Determine the AWS Region** in which you want to launch the Amazon EC2 instance.
2. **Launch** an Amazon EC2 instance from a pre-configured Amazon Machine Image (AMI).
3. **Choose an instance type** based on CPU, memory, storage, and network requirements.
4. **Configure** network, IP address, security groups, storage volume, tags, and key pair.

Images

Amazon Machine Image (AMI) Details



An AMI includes the following:

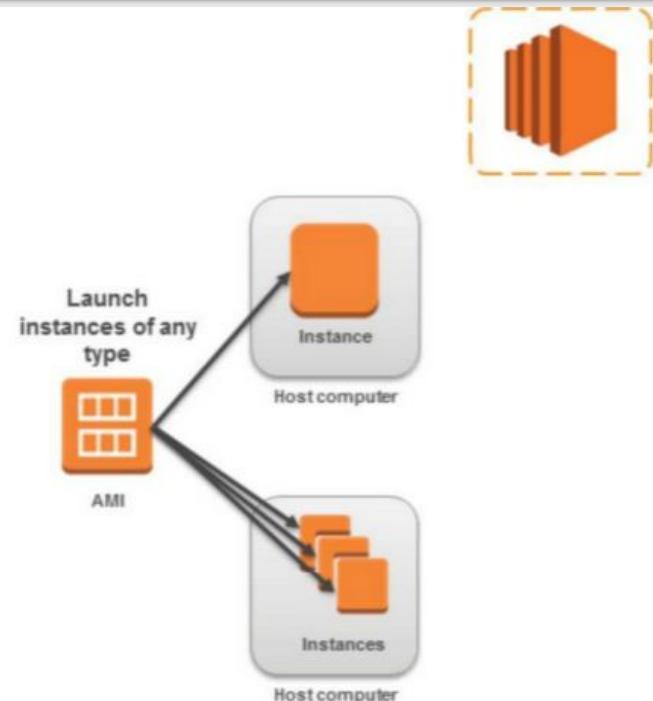
- A template for the **root volume** for the instance (for example, an operating system, an application server, and applications).
- **Launch permissions** that control which AWS accounts can use the AMI to launch instances.
- A block device mapping that specifies the **volumes to attach** to the instance when it is launched.

Images

Instances and AMIs

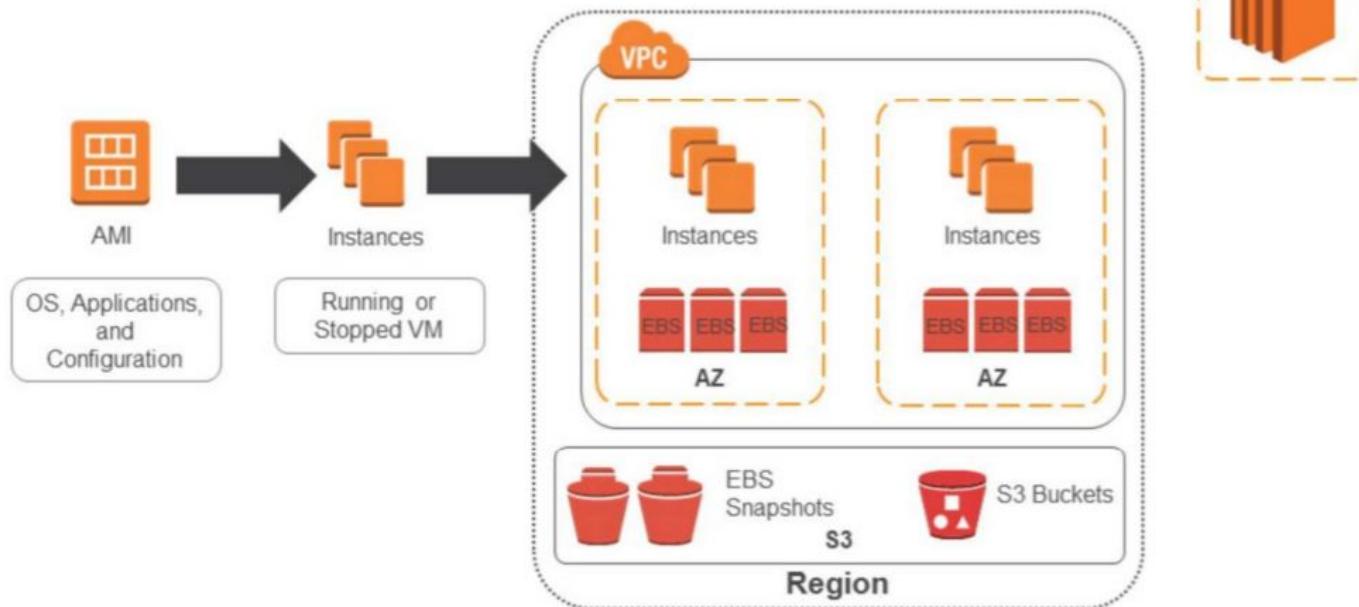
Select an AMI based on:

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



Deployment

Amazon EC2 Instances



Instance types

Current Generation Instances



| Instance Family | Some Use Cases |
|------------------------------|--|
| General purpose (t2, m4, m3) | <ul style="list-style-type: none">Low-traffic websites and web applicationsSmall databases and mid-size databases |
| Compute-optimized (c4, c3) | <ul style="list-style-type: none">High performance front-end fleetsVideo-encoding |
| Memory-optimized (r3) | <ul style="list-style-type: none">High performance databasesDistributed memory caches |
| Storage-optimized (i2, d2) | <ul style="list-style-type: none">Data warehousingLog or data-processing applications |
| GPU instances (g2) | <ul style="list-style-type: none">3D application streamingMachine learning |

User Data

Instance User Data



- Can be passed to the instance **at launch**.
- Can be used to perform common **automated configuration tasks**.
- Runs scripts after the instance starts.

User Data

Adding User Data



- You can specify user data when launching an instance.
- User data can be:
 - Linux script – executed by **cloud-init**
 - Windows batch or PowerShell scripts – executed by **EC2Config** service
- User data scripts run once per instance ID by default.

User Data

User Data Example Linux



```
#!/bin/sh
```

```
yum -y install httpd  
chkconfig httpd on  
/etc/init.d/httpd start
```

User data shell scripts must start with the `#!` characters and the path to the interpreter you want to read the script.

Install Apache web server
Enable the web server
Start the web server

Cloud Providers

Network

Virtual Private Cloud

Amazon Virtual Private Cloud (VPC)



Amazon
VPC

- Provision a **private, isolated virtual network** on the AWS cloud.
- Have complete control over your virtual networking environment.

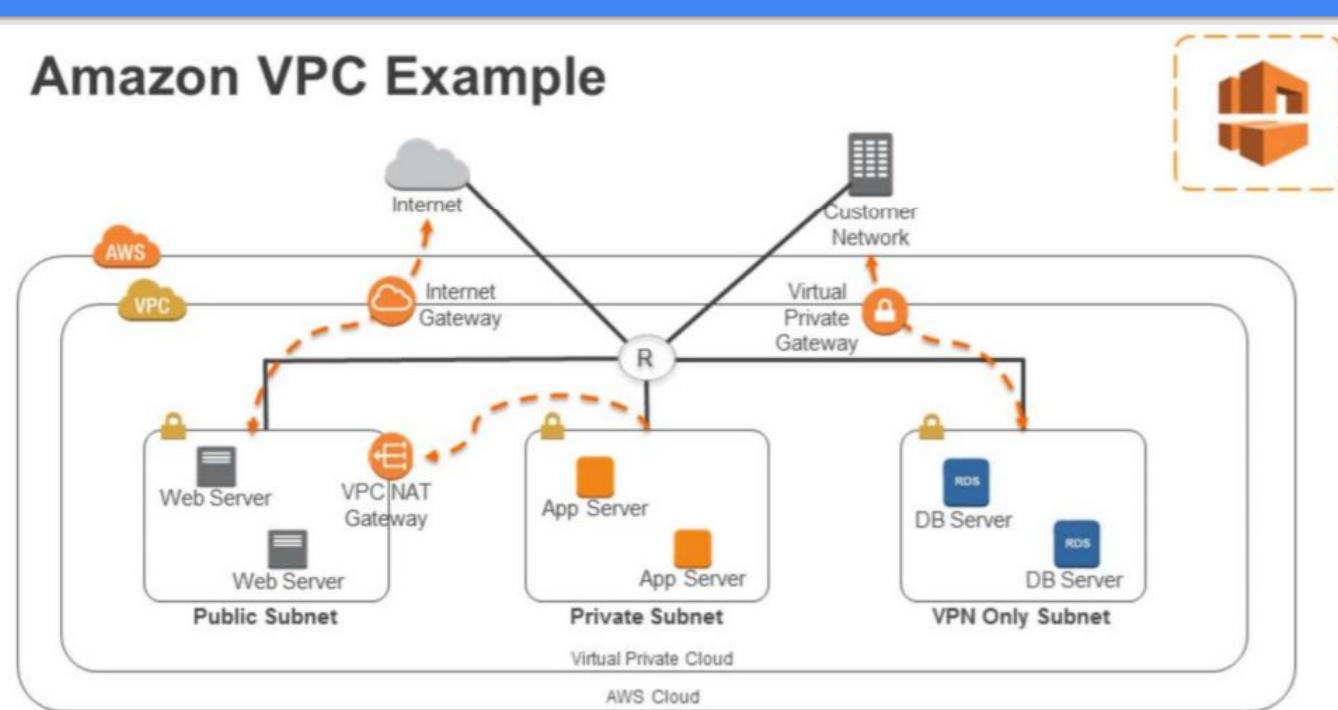
VPC & Subnets

VPCs and Subnets



- A **subnet** defines a range of IP addresses in your VPC.
- You can launch AWS resources into a subnet that you select.
- A **private subnet** should be used for resources that won't be accessible over the Internet.
- A **public subnet** should be used for resources that will be accessed over the Internet.
- Each subnet must reside entirely within one Availability Zone and cannot span zones.

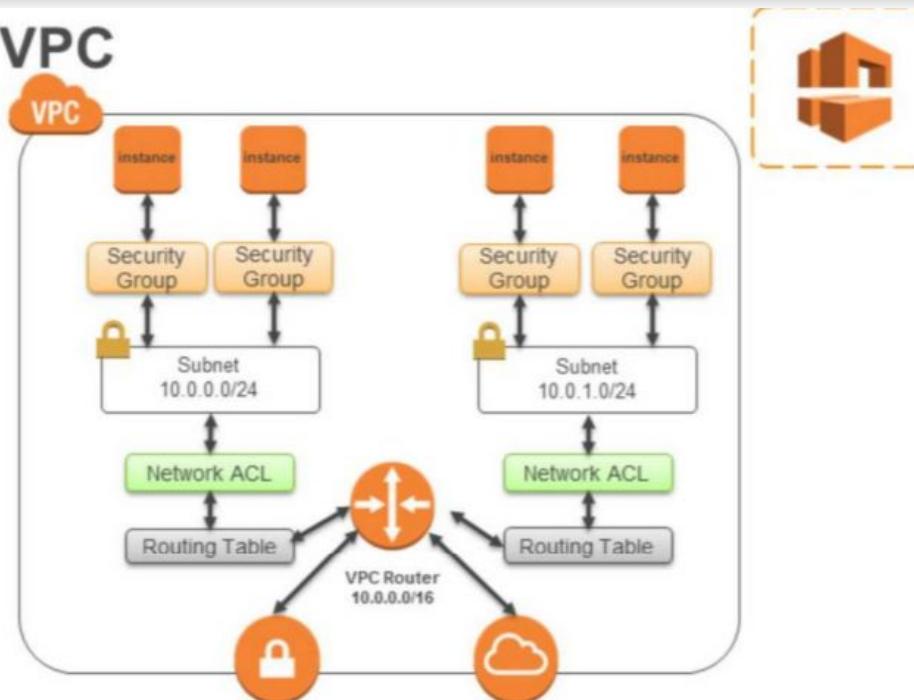
VPC example



Security

Security in Your VPC

- Security groups
- Network access control lists (ACLs)
- Key Pairs



Security groups

Security Groups

Allow access to IP address ranges or Amazon EC2 instances you specify.

Use VPC security groups to control access to a DB instance inside a VPC.

Lab 1: Build VPC & start Web server

Amazon Web Services

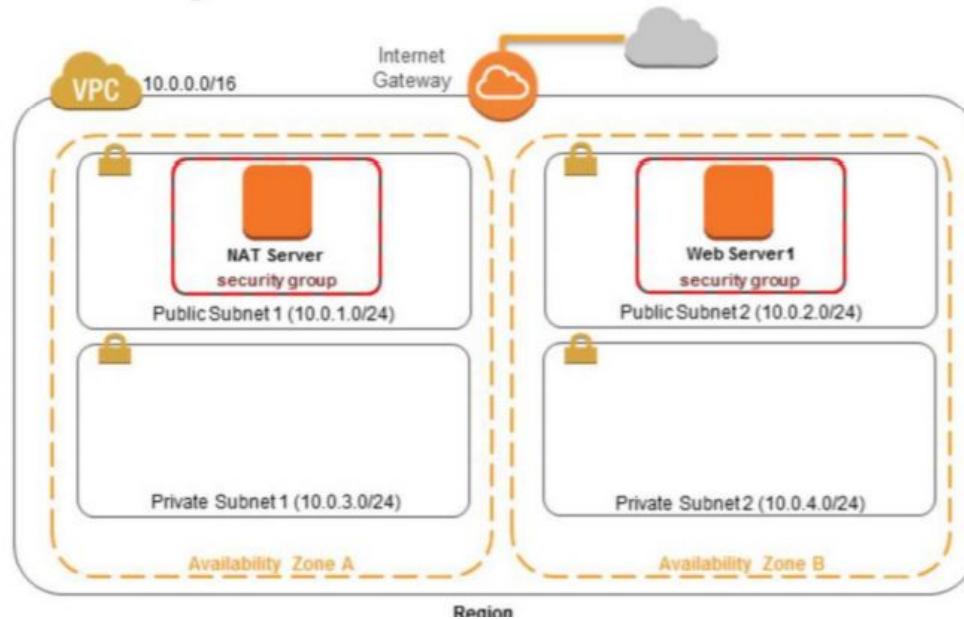
Lab 1: Overview

Lab 1 Overview

-
- 1
 - **Create a VPC**
 - Two public subnets
 - Two private subnets
 - Across two Availability Zones
 - 2
 - **Create an Application Server**
 - Create a security group for your instance.
 - Launch your instance.

Lab 1: Architecture

Lab 1 – Build your VPC and Launch a Web Server



Step-by-step

1. Create four subnets: two in us-east-1a, two in us-east-1b.
2. Create an Internet Gateway and attach it to the VPC.
3. Create a route table; with a route 0.0.0.0/0 → IGW, and attach it to the public subnets.
4. Deploy a Cloud9 environment in the public subnet.
5. Generate a key pair using Cloud9 terminal and the “ssh-keygen” Linux command.
6. Import the key into AWS via the cloud9 terminal and the following command:
`aws ec2 import-key-pair --key-name "ec2-key" --public-key-material file://~/.ssh/id_rsa.pub`
7. Create an EC2 Instance with the following parameters: Ubuntu 18.04 64 bits, t2.micro, 30GB disk size, public-subnet, auto-assign public ip: enable, and choose the “ec2-key” key pair.
8. Edit the associated security group to add HTTP from anywhere in inbound rules.
9. Connect to the instance via Cloud9 terminal: “ssh ubuntu@IP”
10. Try to update your package repositories with the command: sudo apt-get update

Storage

Amazon Web Services

Bucket

Amazon Simple Storage Service (S3)



Amazon S3

- Storage for the Internet
- Natively online, HTTP access
- Storage that allows you to store and retrieve **any amount of data**, any time, from anywhere on the web
- **Highly scalable**, reliable, fast and durable

Bucket

Amazon S3 Facts

- Can store an **unlimited number of objects** in a bucket
- Objects can be **up to 5 TB**; no bucket size limit
- Designed for **99.999999999%** durability and **99.99%** availability of objects over a given year
- Can use **HTTP/S** endpoints to store and retrieve any amount of data, at any time, from anywhere on the web
- Is highly scalable, reliable, fast, and inexpensive
- Can use optional server-side **encryption** using AWS or customer-managed provided client-side encryption
- Auditing is provided by access logs
- Provides standards-based **REST** and **SOAP** interfaces



Bucket

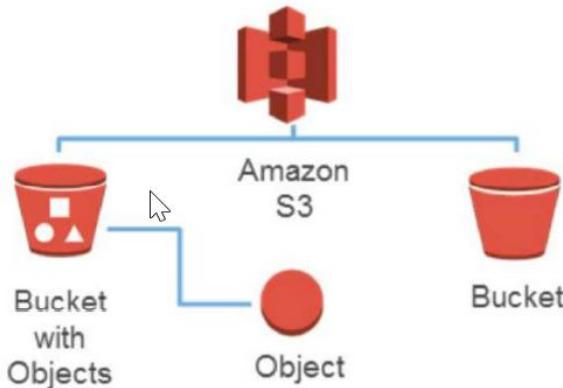
Common Use Scenarios

- Storage and backup
- Application file hosting
- Media hosting
- Software delivery
- Store AMIs and snapshots



Bucket

Amazon S3 Concepts



- Amazon S3 stores data as objects within **buckets**
- An object is composed of a file and optionally any **metadata** that describes that file
- You can have **up to 100 buckets** in each account
- You can **control access** to the bucket and its objects

Bucket versioning

Amazon S3 Versioning

- Protects from **accidental overwrites and deletes** with no performance penalty.
- Generates a **new version with every upload**.
- Allows easily  retrieval of deleted objects or **roll back** to previous versions.
- Three states of an Amazon S3 bucket
 - Un-versioned (default)
 - Versioning-enabled
 - Versioning-suspended



Versioning Enabled

Volumes

Amazon Elastic Block Store (EBS)



Amazon
EBS

- **Persistent block level storage** volumes offer consistent and low-latency performance.
- Stored data is automatically replicated within its Availability Zone.
- Snapshots are stored durably in Amazon S3.

Volumes types

Amazon EBS Volume Types



- SSD-backed volumes are
 - Optimized for **transactional** workloads that involve **frequent read/write** operations with **small I/O** size.
 - Dominant in **IOPS** performance.
- HDD-backed volumes are
 - Optimized for **large streaming** workloads.
 - Dominant in **throughput** (measured in MiB/s).

Volumes

Amazon EBS Facts



- EBS is recommended when data must be **quickly accessible** and requires **long-term persistence**.
- You can launch your EBS volumes as **encrypted** volumes – data stored at rest on the volume, disk I/O, and snapshots created from the volume are all encrypted.
- You can create **point-in-time snapshots** of EBS volumes, which are persisted to Amazon S3.

Volumes

Amazon 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

Database

Amazon Web Services

Database

Amazon Relational Database Service (RDS)



Amazon
RDS

- Cost-efficient and **resizable capacity**
- Manages time-consuming **database administration** tasks
- Access to the full capabilities of **Amazon Aurora, MySQL, MariaDB, Microsoft SQL Server, Oracle, and PostgreSQL** databases

DB types

Database Considerations

| If You Need | Consider Using |
|---|---|
| A relational database service with minimal administration | Amazon RDS <ul style="list-style-type: none">Choice of Amazon Aurora, MySQL, MariaDB, Microsoft SQL Server, Oracle, or PostgreSQL database enginesScale compute and storageMulti-AZ availability |
| A fast, highly scalable NoSQL database service | Amazon DynamoDB <ul style="list-style-type: none">Extremely fast performanceSeamless scalability and reliabilityLow cost |
| A database you can manage on your own | Your choice of AMIs on Amazon EC2 and Amazon EBS that provide scale compute and storage, complete control over instances, and more. |

RDS

Amazon RDS

- Simple and **fast to deploy**
- Manages common database administrative tasks
- **Compatible** with your applications
- Fast, predictable performance
- Simple and **fast to scale**
- Secure
- Cost-effective



Snapshots

Cross-Region Snapshots

- Are a **copy** of a **database** snapshot stored in a **different AWS Region**.
- Provide a backup for disaster **recovery**.
- Can be used as a **base** for **migration** to a different region.



Multi-zones

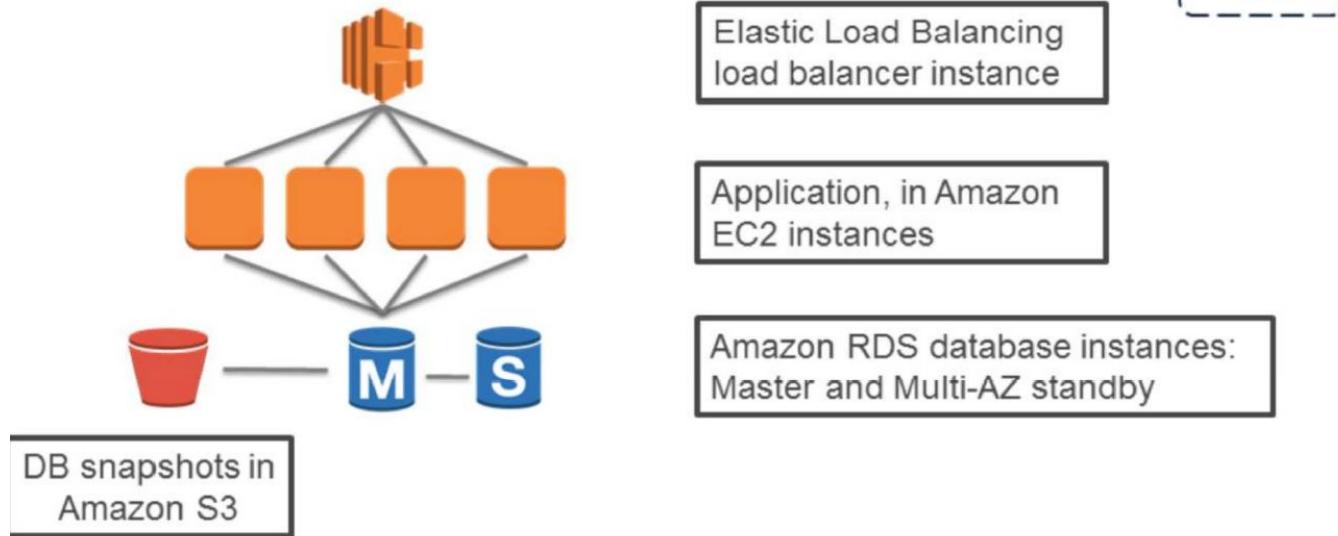
Multi-AZ RDS Deployment



- With **Multi-AZ** operation, your database is **synchronously replicated to another Availability Zone** in the same AWS Region.
- **Failover** to the standby **automatically** occurs in case of master database failure.
- Planned maintenance is applied first to standby databases.

Resilient application

A Resilient, Durable Application Architecture

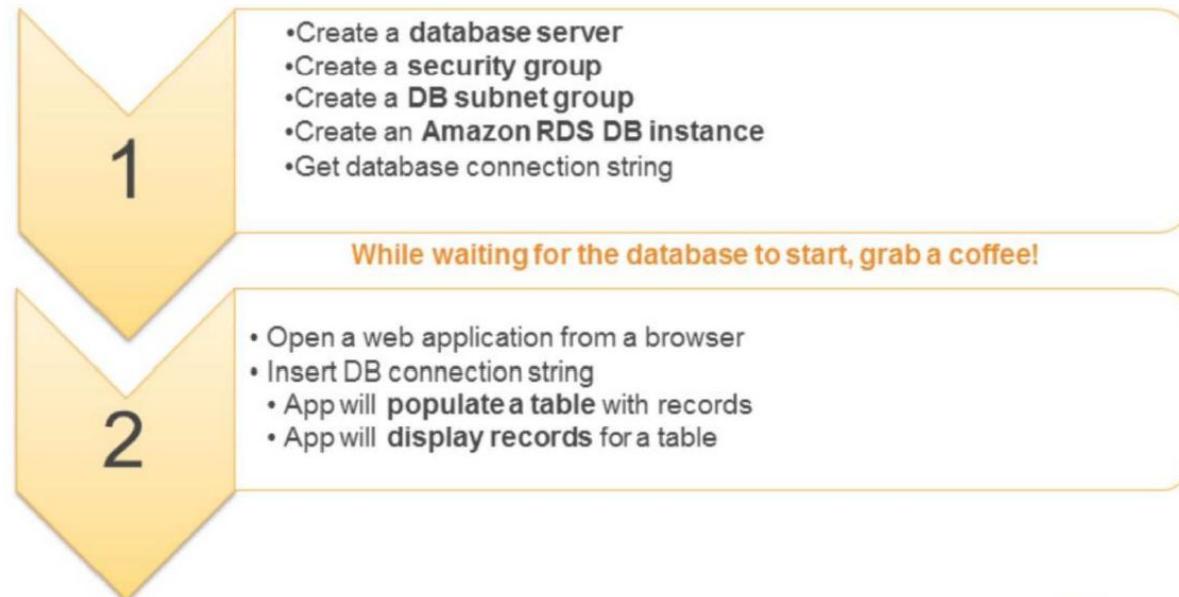


Lab 2: Build DB & configureserver

Amazon Web Services

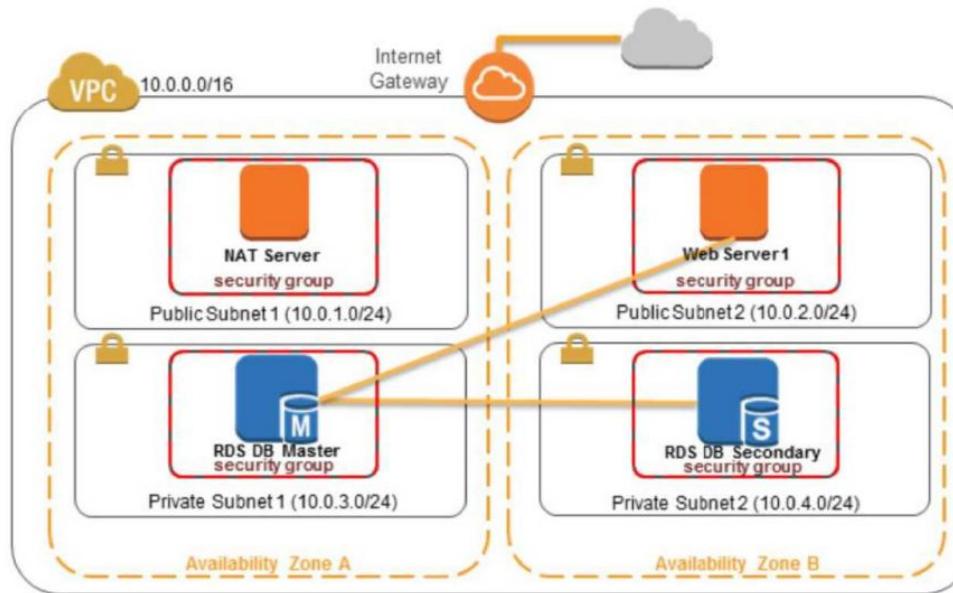
Lab 2: Overview

Lab 2 Overview



Lab 2: Architecture

Lab 2 – Build Your Database Server and Connect to It



Lab 2: Step-by-step

1. Create a DB subnet group with the 2 private subnets
2. Create a SG to authorize the web server to access the RDS instance (3306 port)
3. Create a RDS instance with MySQL engine, Free tier option, choose your previously created subnet group and SG
4. Connect to the web server via Cloud9 terminal and execute the following commands:
5. <https://github.com/skhedim/demo-aws>

```
sudo apt-get update
sudo apt-get install -y apache2 mysql-client php libapache2-mod-php php-cli php-common php-mbstring php-gd
php-intl php-xml php-mysql php-zip php-curl php-xmlrpc unzip
cd /tmp && wget -c http://wordpress.org/latest.tar.gz
tar -xvzf latest.tar.gz && sudo rm -f /var/www/html/index.html && sudo rsync -av wordpress/* /var/www/html/
sudo chown -R www-data:www-data /var/www/html/ && sudo chmod -R 755 /var/www/html/
sudo systemctl restart apache2
```

6. Connect to the RDS instance via the webserver instance using mysql CLI: mysql -u admin -p -h <RDS endpoint> -P 3306
7. Create a database for the website: CREATE DATABASE wordpress; then quit the terminal: QUIT;
8. Configure the database connection from your web browser via the public IP of your web server instance

Elasticity

Amazon Web Services

Load Balancing

Elastic Load Balancing



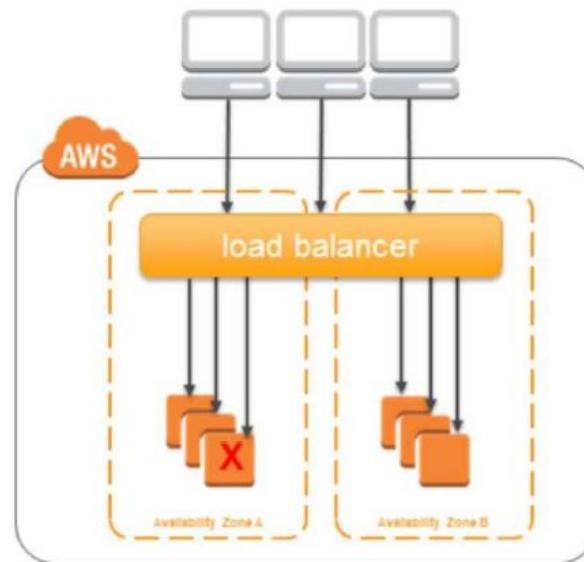
Elastic Load
Balancing

- **Distributes** traffic across multiple EC2 instances, in multiple Availability Zones
- Supports **health checks** to detect unhealthy Amazon EC2 instances
- Supports the **routing and load balancing** of HTTP, HTTPS, SSL, and TCP traffic to Amazon EC2 instances

Load Balancer

Classic Load Balancer - How It Works

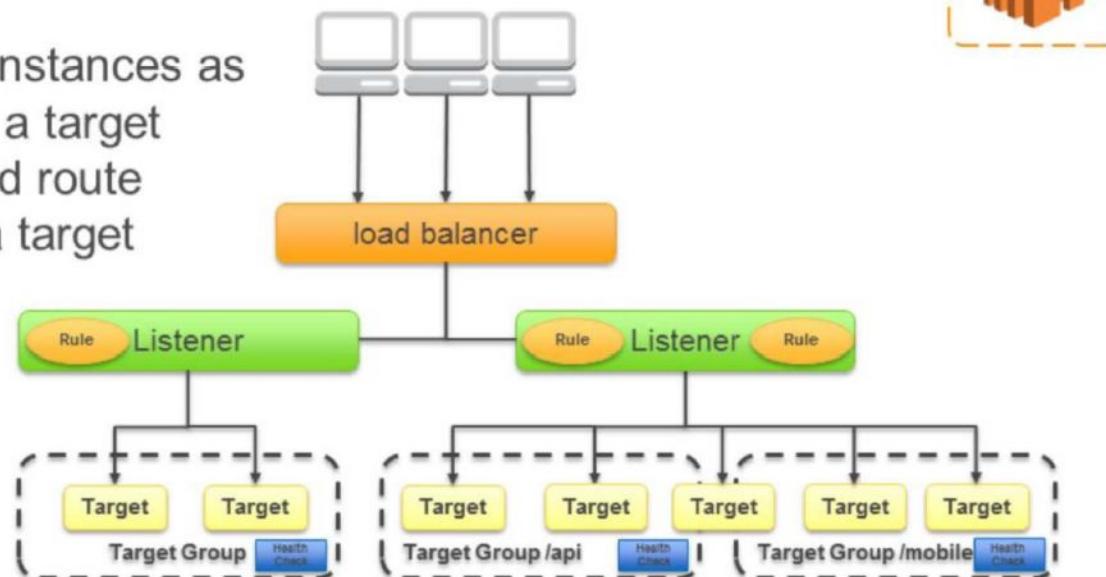
Register instances with your load balancer.



Application load balancer

Application Load Balancer – How It Works

Register instances as targets in a target group, and route traffic to a target group.



Load balancer

Load Balancer Comparison

Classic Load Balancer

benefits include support for:

- EC2-Classic.
- VPC.
- TCP and SSL listeners.
- Sticky sessions.
- OSI Layer 4
(network protocol level)

ALB benefits include support

for:

- Path-based routing.
- Routing requests to multiple services on a single EC2 instance.
- Containerized applications.
- Monitoring the health of each service independently.
- OSI Layer 7
(application level)



Monitoring

Amazon CloudWatch



Amazon
CloudWatch

- A **monitoring service** for AWS cloud resources and the applications you run on AWS
- **Visibility into** resource utilization, operational performance, and overall demand patterns
- **Custom application-specific** metrics of your own
- **Accessible** via AWS Management Console, APIs, SDK, or CLI

Auto-scaling

Auto Scaling

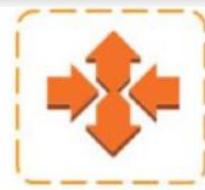


Auto
Scaling

- **Scale your Amazon EC2 capacity automatically**
- Well-suited for applications that experience **variability in usage**
- Available at no additional charge

Auto scaling

Auto Scaling Benefits



Better Fault Tolerance



Better Availability



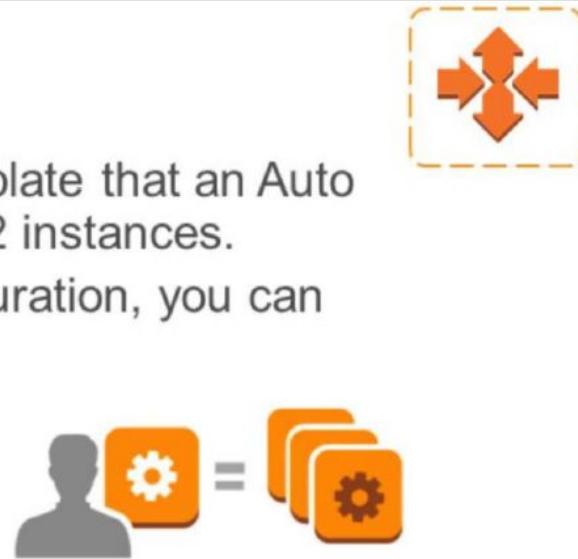
Better Cost Management



Auto-scaling Templates

Launch Configurations

- A **launch configuration** is a template that an Auto Scaling group uses to launch EC2 instances.
- When you create a launch configuration, you can specify:
 - AMI ID
 - Instance type
 - Key pair
 - Security groups
 - Block device mapping
 - User data

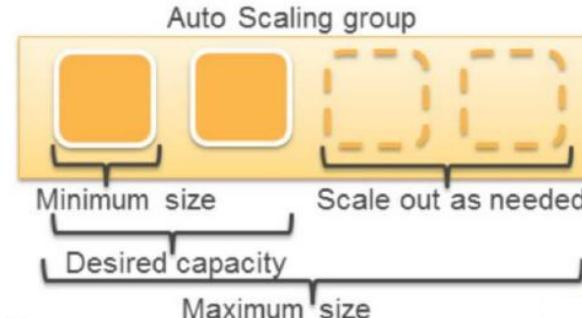


Groups

Auto Scaling Groups

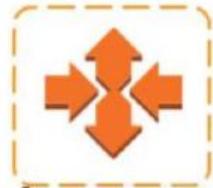


- Contain a collection of EC2 instances that share similar characteristics.
- Instances in an Auto Scaling group are treated as a **logical grouping** for the purpose of instance scaling and management.



Auto scaling

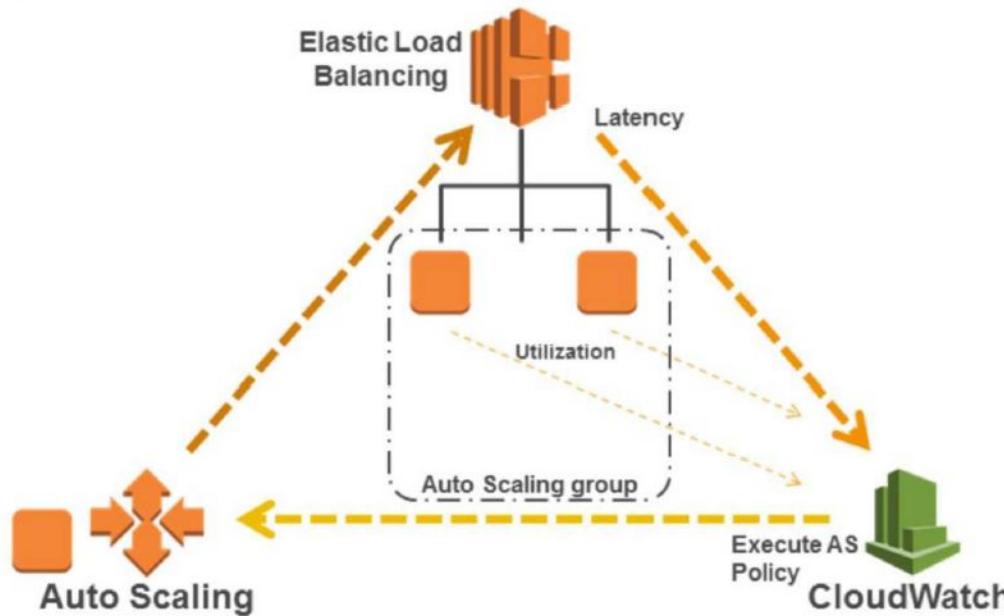
Dynamic Scaling



- You can create a scaling policy that uses **CloudWatch alarms** to determine:
 - When your Auto Scaling group should **scale out**.
 - When your Auto Scaling group should **scale in**.
- You can use alarms to monitor:
 - Any of the metrics that AWS services send to Amazon CloudWatch.
 - Your own **custom metrics**.

A set of services

Triad of Services

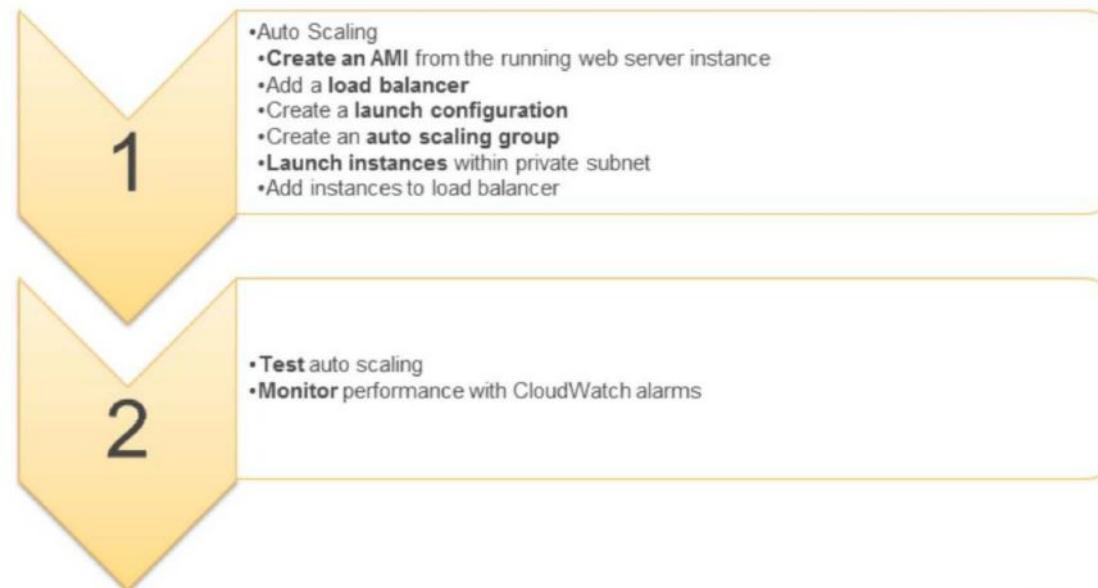


Lab 3: Scale & Load Balance

Amazon Web Services

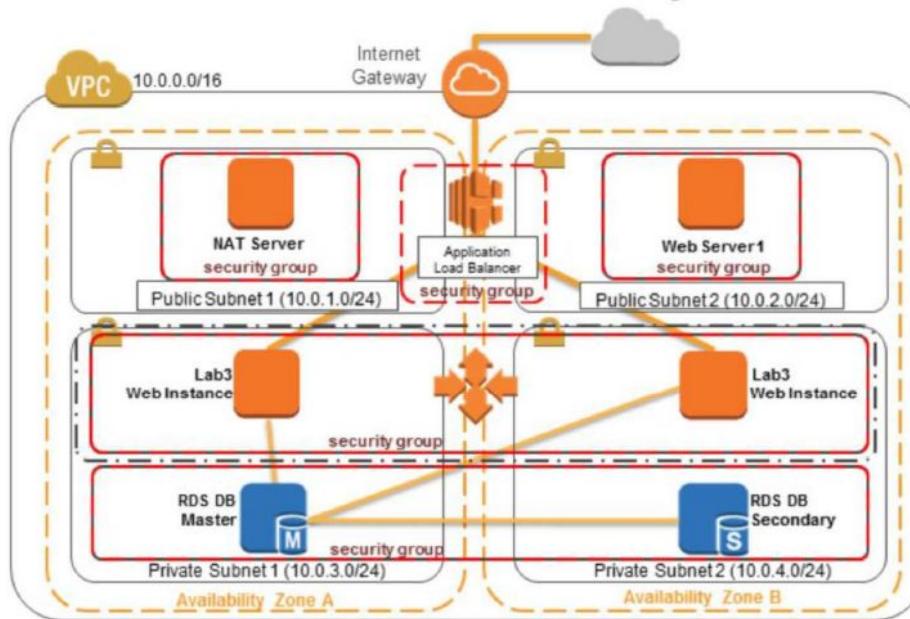
Lab 3: Overview

Lab 3 Overview



Lab 3: Architecture

Lab 3 – Scale and Load Balance your Architecture



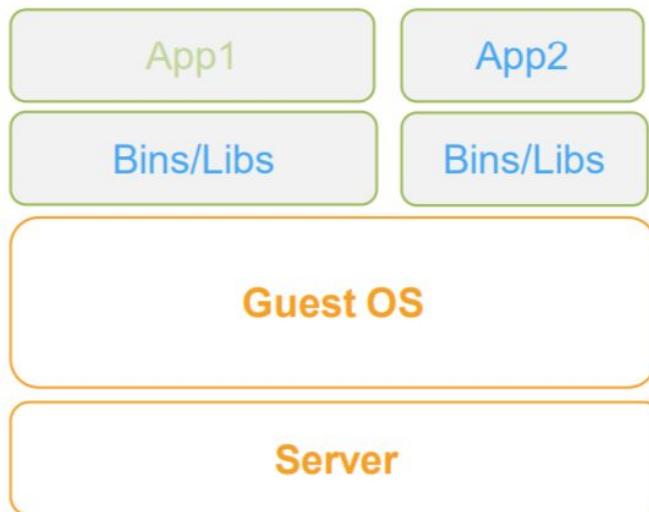
Lab 3: Step-by-step

1. Create an EC2 instance with a minimal web server in a private subnet **without** public IP
2. Create an AMI from the previously created instance
3. Create a NAT GW and attach it to a public subnet.
4. Create a route table; with a route 0.0.0.0/0 → NAT GW, and attach it to the private subnets.
5. Create a Target Group for the autoscaling group
6. Create a Configuration template with the previously created AMI and add the web server security group
7. Create the autoscaling group associated to the configuration template, scale from 1 to 3 based on CPU usage (20%).
8. Edit the details of the autoscaling group and add it to the previously created target group
9. Create an ALB and add the previously created target group to it.
10. Try to access your website via the load balancer DNS name
11. Connect into a scaled machine and generate CPU utilization with this command: `while true;do a=1;done`

Containers

Amazon Web Services

What are containers?



OS virtualization

Process isolation

Images

Automation

How it works?

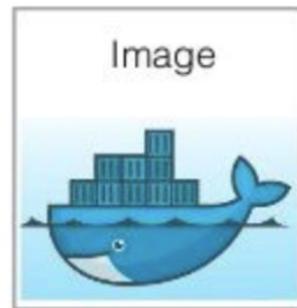
FROM: ubuntu:latest

RUN apt-get update
RUN apt-get install apache2

```
FROM ubuntu:latest
RUN apt-get update
RUN apt-get install apache2
```

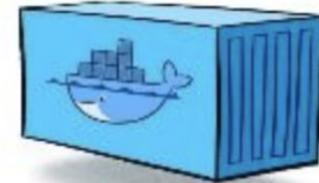
Dockerfile

build



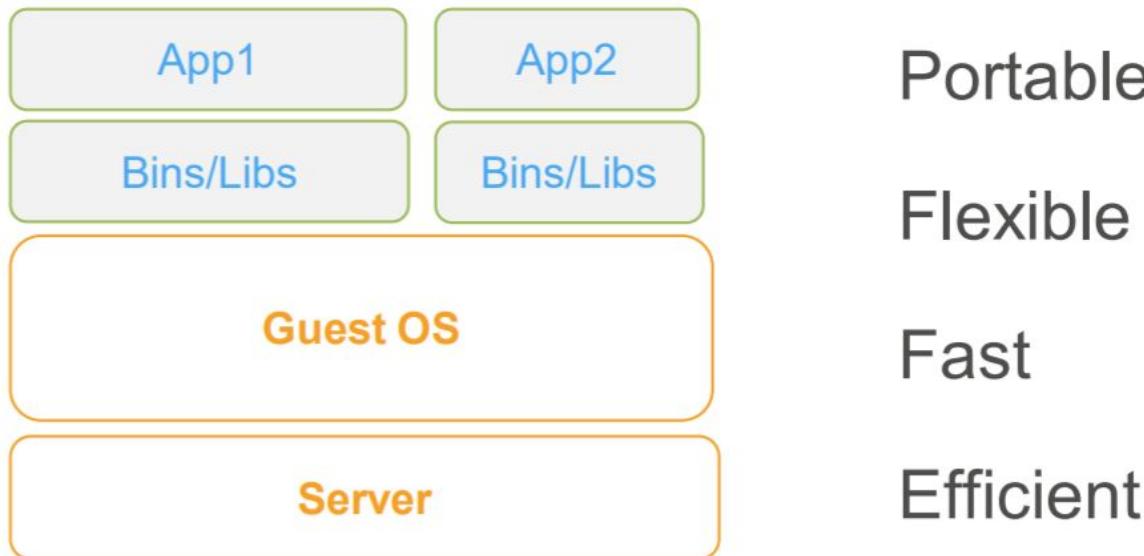
Docker Image

run

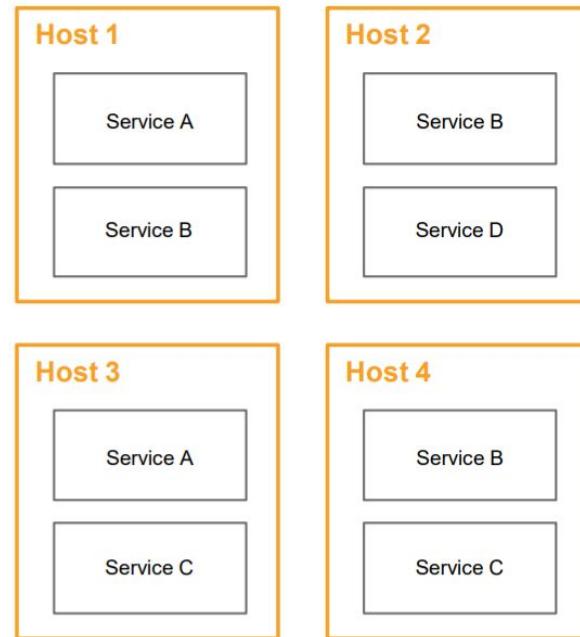
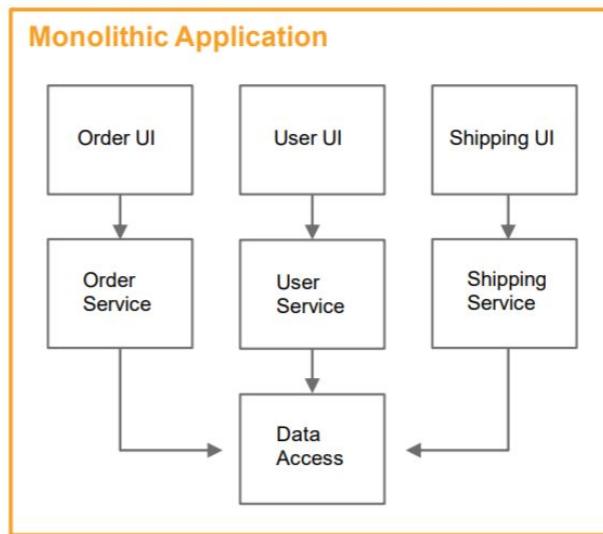


Docker Container

Advantages



Microservice application



Orchestration

Running one container is easy



Running many containers is hard...

Some challenges

Cluster management

Availability

Scheduling

Security

Monitoring

Integration with AWS services



ECS

Amazon EC2 Container Service (ECS)

Highly scalable, high performance container management system.

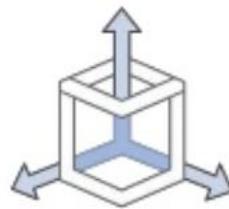
Eliminates the need to install, operate, and scale your own container management infrastructure.



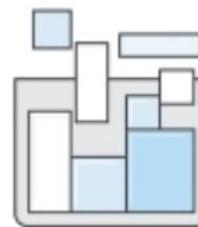
Main features

Amazon EC2 Container Service (ECS)

ECS provides a managed platform for:



Cluster
management



Container
orchestration



Deep AWS
integration

Auto-scaling Templates

How does ECS map to traditional workloads?



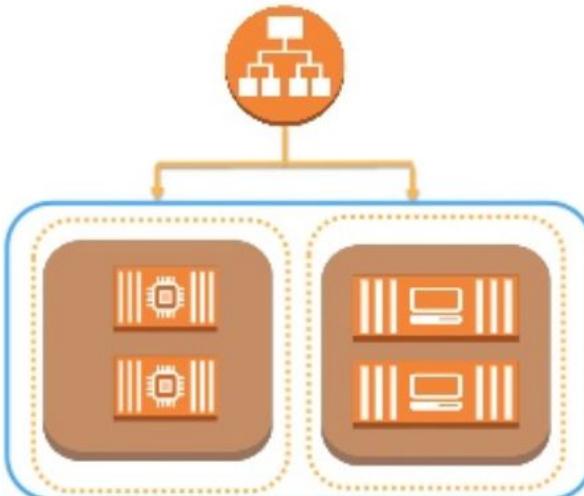
Instances: standard EC2 boxes. Once registered to a Cluster, your Tasks run here

Services: layer that manages and places Tasks

Tasks: container wrapper and configuration around processes running on the instance

Auto-scaling Templates

How does ECS work?



Load balancer: (ALB or EC2 classic) routes traffic to the cluster instances.

Cluster is made up of one or more EC2 instances

Each **cluster instance** runs one or more **Services**

Auto-scaling Templates

How does ECS work?



Each **cluster instance** runs one or more **Services**

A **Service** controls things like the number of copies of a Task you want running (Desired Count), and registers your Service with a load balancer

A **Task Definition** controls things like container image, environment variables, resource allocation, logger, and other parameters

Fargate

Networking only

Resources to be created:

Cluster

VPC (optional)

Subnets (optional)

Powered by AWS Fargate

EC2 Linux + Networking

Resources to be created:

Cluster

VPC

Subnets

Auto Scaling group with Linux AMI

Lab 4: Deploy containers

Amazon Web Services

Lab 4: Step-by-step

1. Try to deploy a nginx container from the ECS get started menu, add an ALB to your service.
2. Access to your nginx container from the ALB DNS name.
3. Create a ECS cluster **EC2 Linux + Networking**
4. Create a Task definition: One nginx container
5. Create a Service In the previously created ECS cluster, attach the previous Task definition, add a Load Balancer.
6. Access to your nginx container from the ALB DNS name.
7. Try to create a new task definition with the container you want (find docker images on dockerhub.com)
8. Deploy a new Service with your new task definition

Cloud Computing

Conclusion