Welcome to EDGE 2023 Week 1

DevelopIntelligence
A PLURALSIGHT COMPANY

Week 1 - cloud + AWS intro

01

- Accounts, security concepts/policies, building, how the cloud works common services,
- Shared responsibility model in AWS
  Infrastructure Fundamentals:
- Connectivity (availability zones, regions, resiliency)
- Basic AWS services - IAM, API Gateway, S3, Lambdas

Daily Breakdown – week 1

- Day 1:
  - Public vs Private vs Hybrid Cloud
  - Cloud terminology - IaaS vs PaaS vs SaaS vs FaaS
  - AWS intro/overview of common services
  - AWS regions and Availability Zones

- Day 2:
  - Intro to IAM - Users, Roles, Policies
  - EC2 overview and lab
  - Lambdas + API Gateway + other services
  - Serverless Lab - S3 + Lambda + API Gateway

Daily Breakdown – week 1

- Day 3:
  - Lambdas + API Gateway + other services
- Day 4:
  - Serverless Lab - S3 + Lambda + DynamoDB
- Day 5:
  - Intro to RDS, Console vs CLI in AWS
  - RDS lab

Class Logistics

4 hours

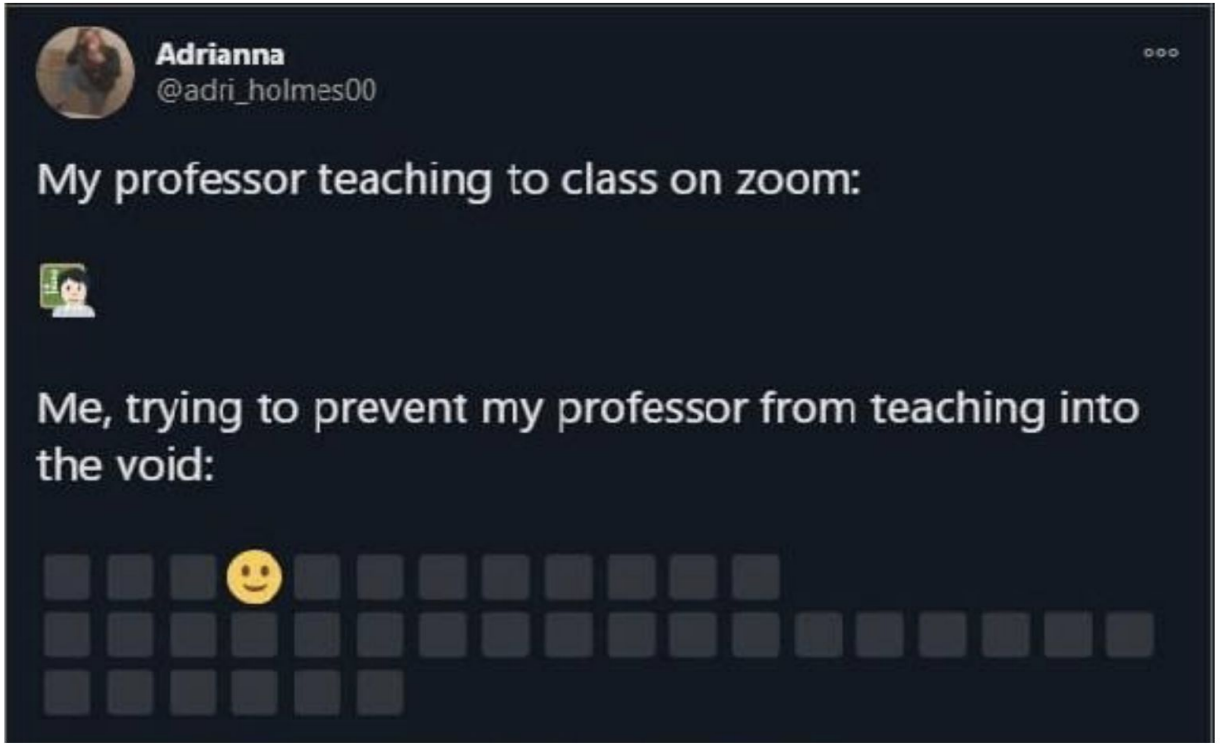
10 min breaks every hour

Class Logistics

Eliminate distractions

Enable video

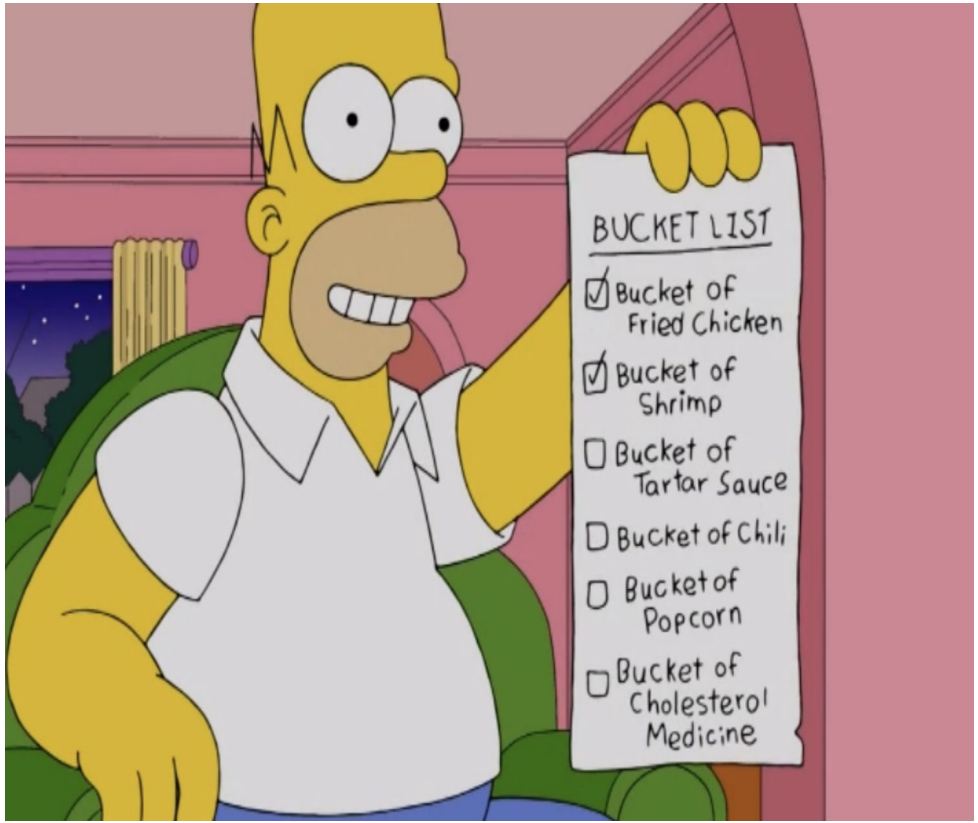
Class Logistics



Week 1 Objectives/Goals

After week 1 of this class you should be able to:

- ❑ Describe cloud computing, its benefits, and its economics
- ❑ Explain AWS global infrastructure and understand considerations for resiliency
- ❑ Understand IAM and the need for it along with other AWS services such as API Gateway, S3, Lambdas, RDS, etc.



Introductions

- Please turn on your camera
- When called on, please share:
- Name, location
- Experience with AWS/other cloud providers
- What motivated you to take the class?
- One fun fact about yourself (like hobbies!)

Notes - Virtual Machine

- We will be doing most of our work in virtual machines
- Plenty of hands on – roughly 70% of the class
- You will need to sign up for a personal Github account if you don’t have one (free of course).

Virtual machine setup

- Virtual machine setup - please open an **incognito window** in Chrome - do File->Incognito Window, then navigate to: https://labs.datacouch.io/pluralsight/
  - Be sure to **“allow”** the clipboard when asked
- Then, login using the credentials provided
- FYI - Your virtual Machines will be up 1 hour before the training start and it will automatically shut down 2 hours after the class end time.

Notes on Region and Security

- We will be doing all of our work in the **us-west-2** region in AWS - it's very important that you only create resources in that specific region
  - We will talk more about regions and what they mean shortly
- Please be very careful with your access keys as well

EDGE week 1 - Cloud intro, AWS intro, and exploring AWS services

Infrastructure Options

Before we get into AWS

- Let’s ensure that we are on the same page cloud terminology wise
- And that we understand all the tradeoffs between public/private and IaaS/PaaS, etc..

Infrastructure Options

Infrastructure is the hardware & software that run our IT workloads and that provide our business users and customers a way to interface with the applications required to complete their daily jobs

What Are the Options?

On-Premise (in a Data Center)

Public Cloud

At the Edge

Hybrid Cloud

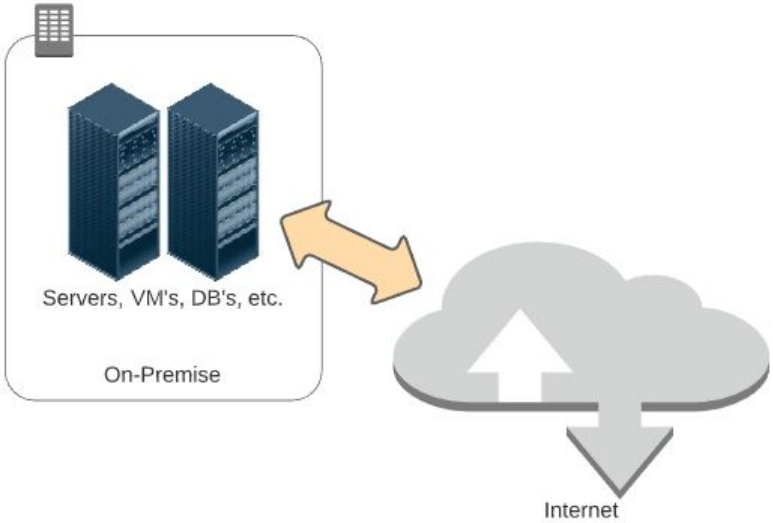
What do they all mean?

On-Premise

Can mean a few different things:

- In a wholly-owned Data Center
- In a COLO (or co-location Data Center)
- Sometimes called a “private cloud”

On-Premise



On-Premise

Why and What?

- How infrastructure has traditionally been done
- With this model, companies try and estimate current & future hardware capacity needed to support business operations

On-Premise

Why and What?

- Stakeholders plan out expected levels of consumption for the next 3 – 5 years (capacity to handle current volumes as well as expected growth)
- Some critical workloads may not be suitable for anything but a physical and directly-managed implementation (e.g., mainframe)

On-Premise – Discussion

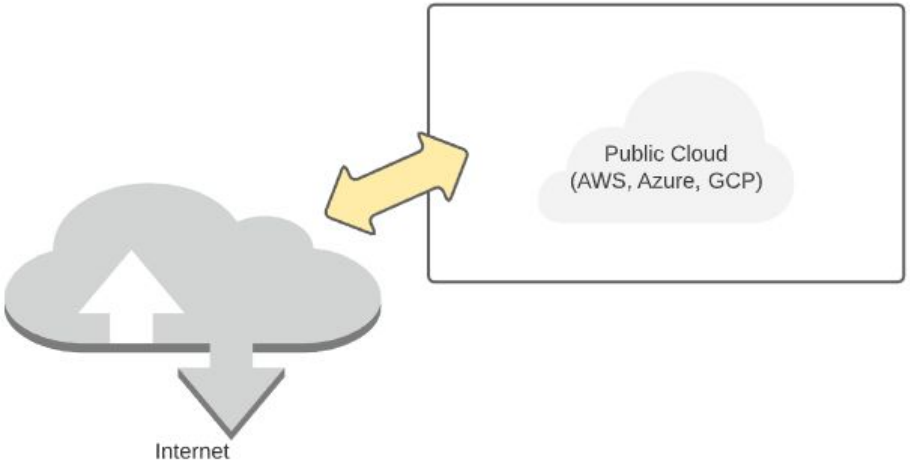
Pros?

- Discrete capacity planning (even if that planning was off)
- Some workloads (e.g., mainframes and certain legacy systems) are tailor-made for a physical data center
- With a move to COLO's, companies could begin to share expenditure

Cons?

- Sometimes difficult to know what is needed and when it is needed – if the plan was off (or unexpected spikes in demand occurred), difficult to adjust quickly
- Some workloads are just as effective (if not more so) in a virtual vs. physical implementation
- Harder to control costs and plan for costs – CAPEX (on-premise - buying servers) vs. OPEX(use public cloud - like AWS - renting servers)

Public Cloud



Public Cloud

Why and What?

- Platform using the standard “Cloud computing model” to provide infrastructure and application services
- Accessed and integrated via the Internet
- May provide a few different types of services – IaaS, PaaS, etc.

Public Cloud

Why and What?

- Usually supports a subscription or “pay as you go” (on-demand) pricing model
- Largest players in this space include Azure, AWS and GCP

Public Cloud - Discussion

Pros?

- Flexibility and elasticity in capacity planning – enables automated schedule-based or metrics-based adjustments to capacity when required
- In some cases, managed services can be leveraged reducing operations overhead
- Because services are PAYG (pay as you go), you're only charged for what you use, and those expenses are OPEX

Cons?

- Requires enough historical data for schedule-based planning or the right configuration for metrics-based planning
- With managed services you lose some levels of granular control
- Because of the flexibility/elasticity, it can be difficult to budget and, if Cloud services are not managed/monitored, costs can be high

At the Edge

Why and What?

- It’s about bringing the power of Cloud computing to you
- Enables additional processing closer to the sources of data while still supporting the offload of higher order processing to the Cloud
- Often involves setting up “Cloud-in-a-box” facilities on-premise

At the Edge

Why and What?

- IoT (Internet of Things) is a good example – devices in a facility reading massive amounts of data can incorporate processing at the edge to improve overall efficiency

- Helps inject lower latency, increased security and improved bandwidth into systems used to aggregate critical data for an enterprise

At the Edge - Discussion

Pros?

- Allows distribution of processing power across a larger surface area
- Can be used to bring critical latency, security and bandwidth improvements to specific types of business workflows
- Efficiencies gained “at the edge” can help with managing the cost of processing data

Cons?

- Requires more infrastructure and more configuration to support that distribution
- Increased distribution of processing power and activity can expand attack surface and requires the right configuration to ensure optimal interaction between system components (i.e., increased complexity)
- More components “at the edge” can lead to increased infrastructure costs

Hybrid Cloud

Why and What?

- In many ways, an amalgamation of the other options
- Supports distribution of system processing across on-premise infrastructure and the public Cloud

Hybrid Cloud

Why and What?

- Allows an enterprise to keep workloads/software applications that are best-suited for on-premise running on-premise while allowing migration of components that can move to the public Cloud

- Can help make an enterprise’s move to the Cloud more gradual and planful

Hybrid Cloud - Discussion

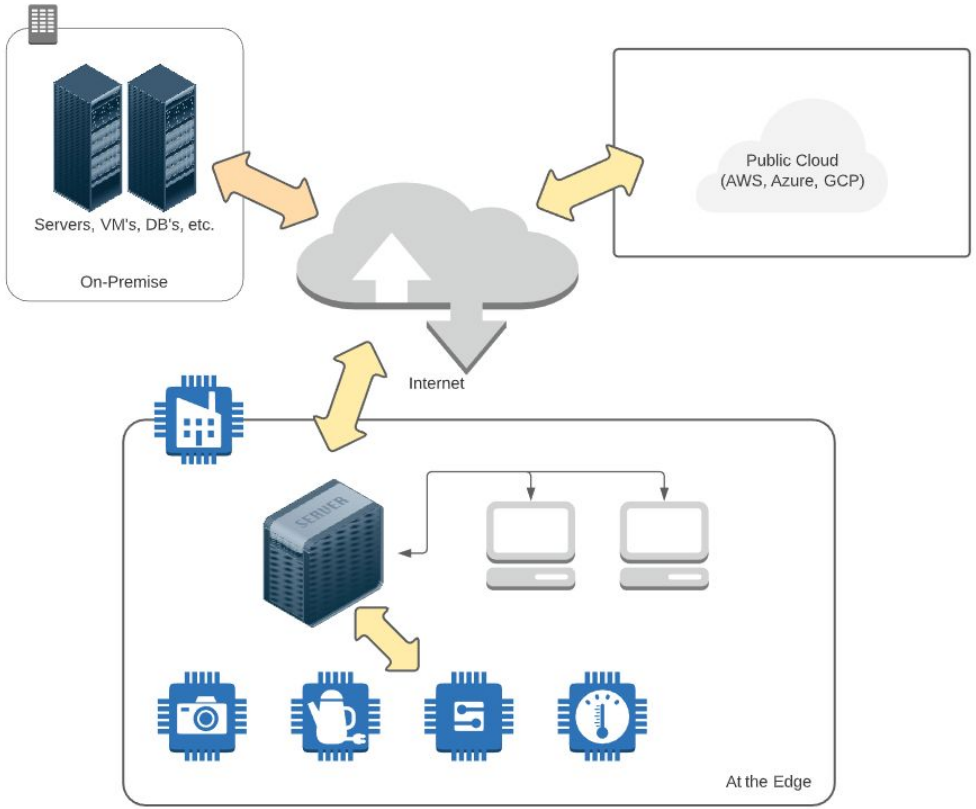
Pros?

- Allows distribution of processing power across a larger surface area
- Can allow a move to the Cloud to be more gradual and allow an enterprise to target optimal deployment platform while making the move
- The ability to support a gradual move enables an enterprise to assess and understand Cloud costs over time

Cons?

- Requires more infrastructure and more configuration to support that distribution
- As with Edge, can lead to increased complexity, often including required setup and maintenance of dedicated, secure connectivity between a data center and the Cloud
- If not managed optimally, costs can be higher due to need to pay for Cloud usage and data center (CAPEX + OPEX)

Hybrid Cloud



LAB:

Infrastructure Options

Scenario: Your company (a global leader in FinTech) is currently hosting all infrastructure used to power the business in an on-premise Data Center. This includes a mainframe for primary business functions (customer management, account management, accounts payable, accounts receivable), several Web Apps (for customer interaction), several Web APIs providing backend data and functionality to the UIs, and a system used to manage data feeds from several security cameras used at corporate offices for observation and security.

As a member of the technical staff, you have been asked to provide thoughts and recommendations on moving from the Data Center to the Cloud.

In your assigned breakout room, discuss as a group and be prepared to provide the following: 1) Potential options for infrastructure in the Cloud for the different types of workload, and 2) considerations that the company should keep in mind as they make the move to ensure awareness and proactive planning.

Nominate someone (or volunteer) to share your group's ideas.

Knowledge Check

With this infrastructure option, stakeholders try to estimate hardware & software needs for the next 3 – 5 years:

A. Public Cloud
B. At the Edge
C. Hybrid
D. On-Premise

Knowledge Check

This infrastructure option focuses on bringing additional power closer to the data sources and application users:

A. Public Cloud
B. At the Edge
C. Hybrid
D. On-Premise

Knowledge Check

This infrastructure option seeks to build a solution by combining the other options into a larger solution:

A. Public Cloud
B. At the Edge
C. Hybrid
D. On-Premise

Application Hosting

Application Hosting

By Application Hosting, we mean the target infrastructure and runtime platform used for deployment and execution of an application or system; can include compute (CPU and server resources), storage, network, data and operating system

What Are the Hosting Options with Cloud?

- IaaS
- PaaS
- Serverless / FaaS
- SaaS
- Containers

What do they all mean?

Infrastructure-as-a-Service (IaaS)

- Involves the building out (and management) of virtual instances of:
  - Compute
  - Network
  - Storage

- Akin to spinning up a server (physical or virtual) in your location or data center complete with disks and required network connectivity

Infrastructure-as-a-Service (IaaS)

- The difference is in the where – instead of in your data center, it is created in a data center managed by one of the public Cloud providers

- Your organization is responsible for patching the OS, ensuring all appropriate security updates are applied and that the right controls are in place to govern interaction between this set of components and other infrastructure

Platform-as-a-Service (PaaS)

- Involves leveraging managed services from a public Cloud provider
- With this model, an enterprise can focus on management of their application and data vs. focusing on management of the underlying infrastructure
- Patching and security of the infrastructure used to back the managed services falls to the CSP (Cloud Service Provider)

Platform-as-a-Service (PaaS)

- Many managed services support automatic scale up or down depending on demand to help ensure sufficient capacity is in place
- Can be considered synonymous with the term “Cloud native”

Serverless / Functions-as-a-Service (FaaS)

- Also represents a type of managed service provided by the CSP
- Cost structure is usually consumption-based (i.e., you only pay for what you use)
- Supports many different coding paradigms (C#/.NET, NodeJS, Python, etc.)

Serverless / Functions-as-a-Service (FaaS)

- Typically, with Serverless (and PaaS), the consumer is only concerned with the application code and data – elements of the CSP's "backbone" used to support are managed by the CSP

- Includes more sophisticated automated scaling capabilities – built for Internet scale

Software-as-a-Service (SaaS)

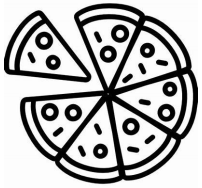
- Subscription-based application services
- Licensed for utilization over the Internet / online rather than for download and install on a server or client machine
- Fully-hosted and fully-managed by a 3rd party

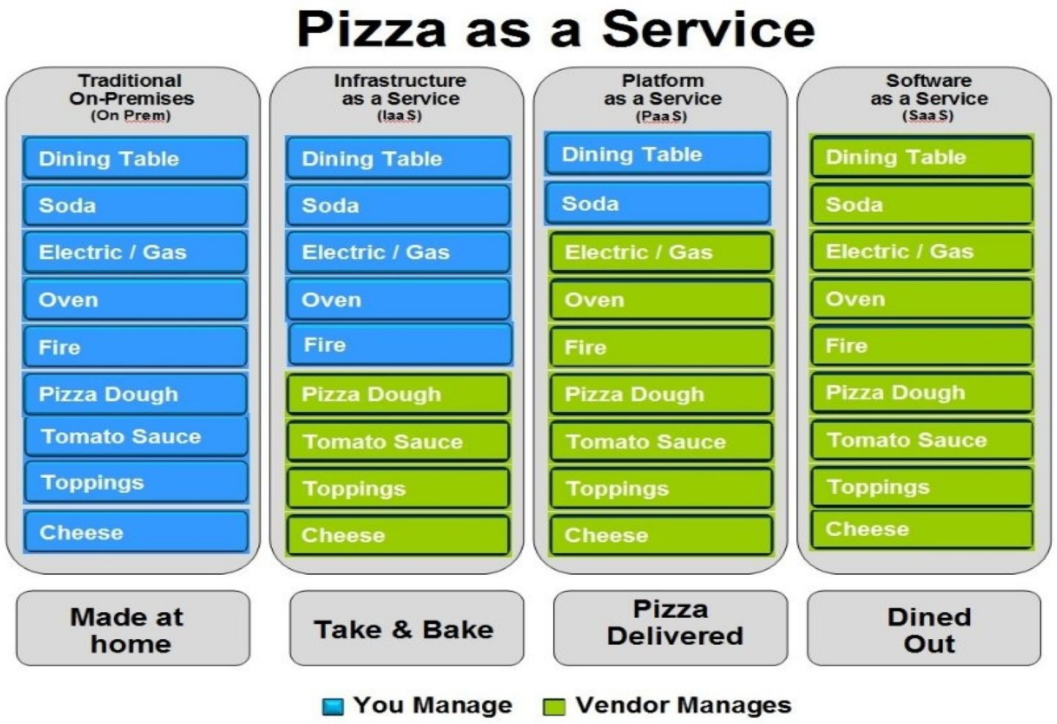
Software-as-a-Service (SaaS)

- Of those discussed, often the cheapest option for service consumers
- However, also offers minimal (or no) control, outside of exposed configuration capabilities

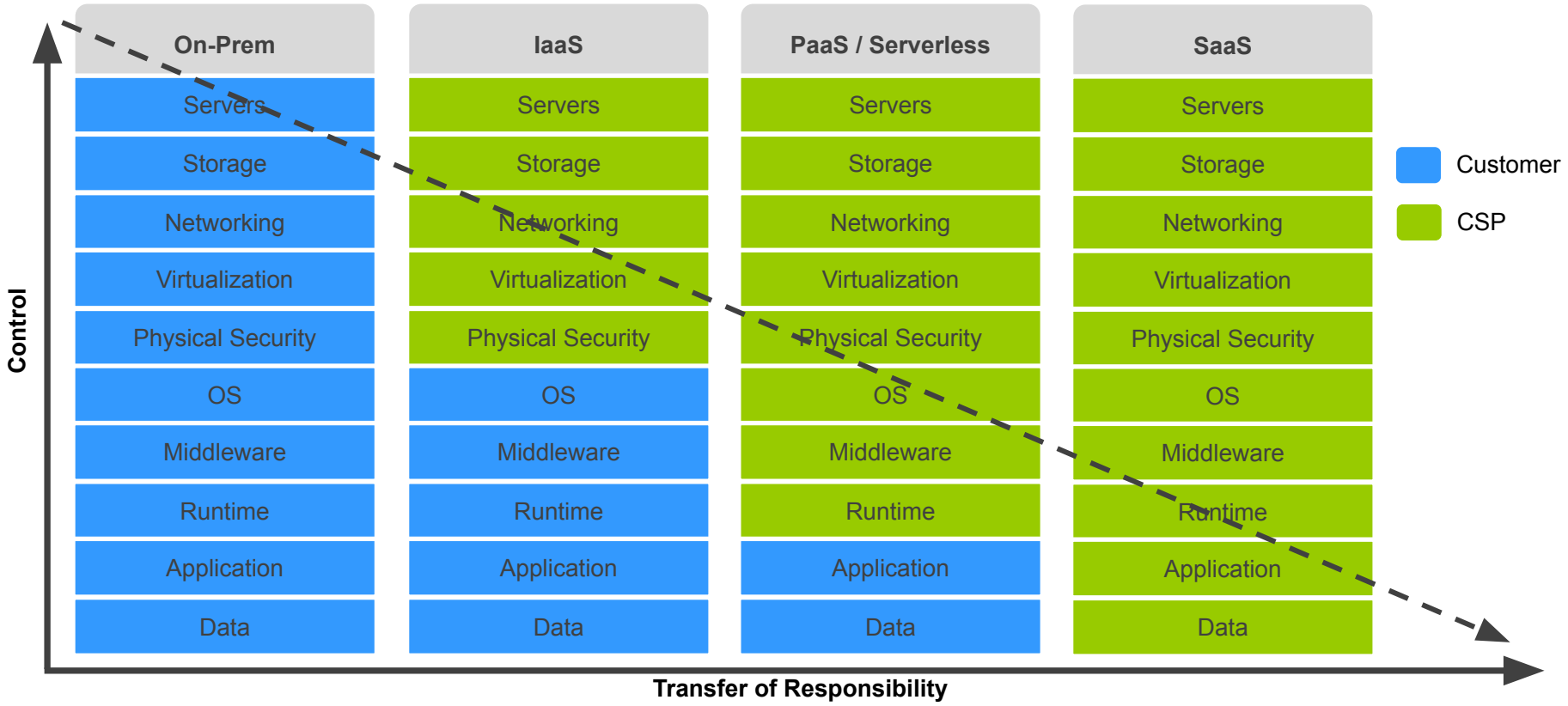
Pizza-as-a-Service

From a LinkedIn post by Albert Barron from IBM
(https://www.linkedin.com/pulse/20140730172610-9679881-pizza-as-a-service/)





Side-by-Side Comparison



Containerization in the Cloud

- One option includes standing up VM's (IaaS) and installing / managing a Kubernetes cluster on those machines or
- Another option includes leveraging a managed service (PaaS) provided by the CSP
- Options in AWS include Elastic Container Service (ECS), Elastic Container Registry (ECR), and EKS (Elastic Kubernetes Service)

Which One is Better?

- The answer is “it depends”
- It depends on the type of application
- It depends on the enterprise

Which One is Better?

- It depends on the skillset and expertise within the organization
- It depends on whether you have budget and opportunity to modernize an application environment (in some cases)
- The best option might be a combination of multiple approaches – right tool for the right job

LAB:

Application Hosting

Scenario: Your company (a global leader in FinTech) is currently hosting all infrastructure used to power the business in an on-premise Data Center. This includes a mainframe for primary business functions (customer management, account management, accounts payable, accounts receivable), several Web Apps (for customer interaction), several Web APIs providing backend data and functionality to the UIs, and a system used to manage data feeds from several security cameras used at corporate offices for observation and security.

As a member of the technical staff, you have been asked to provide thoughts and recommendations on moving from the Data Center to the Cloud.

In your assigned breakout room, discuss as a group and be prepared to provide the following: 1) Recommendations for the types of hosting that could be used for the various application components, and 2) the reasoning behind your recommendations – i.e., what are some benefits you would expect the company to receive by implementing your recommendations.

Nominate someone (or volunteer) to share your group's ideas.

Knowledge Check

With this application hosting option, you primarily only pay for what you actually consume:

A. IaaS
B. PaaS
C. Serverless/FaaS
D. SaaS

Knowledge Check

This application hosting option is used to license certain “commodity” functionality like email, CRM, etc. over the Internet:

A. IaaS
B. PaaS
C. Serverless/FaaS
D. SaaS

Knowledge Check

Of the application hosting options outlined below, which of them provides the combination of greater control but less transfer of responsibility:

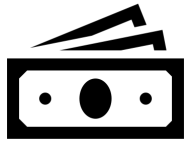
A. IaaS
B. PaaS
C. Serverless/FaaS
D. SaaS

AWS – the basics… history, regions, and AZ's

AWS – brief history

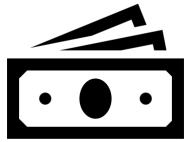
- 2006 – AWS first public launch with S3 – Simple Storage Solution and shortly followed by EC2 – Elastic Cloud Compute
- GCP (Google Cloud Platform) launched in 2008
- 2010 - AWS launches IAM (Identity Access Management) and CloudFormation. Microsoft Azure also launched this year

AWS – revenue



- 2019 Revenue - $35 Billion
- 2020 Revenue - $45 Billion
- 2021 Revenue - $62 Billion
- You can see where this is going – growth of the cloud is tremendous!

AWS – cloud revenue share



- AWS,controlled about **39%** of the cloud infrastructure market in 2021, down from 41% in 2020
- Microsoft Azure is second

AWS – more profitable than amazon.com

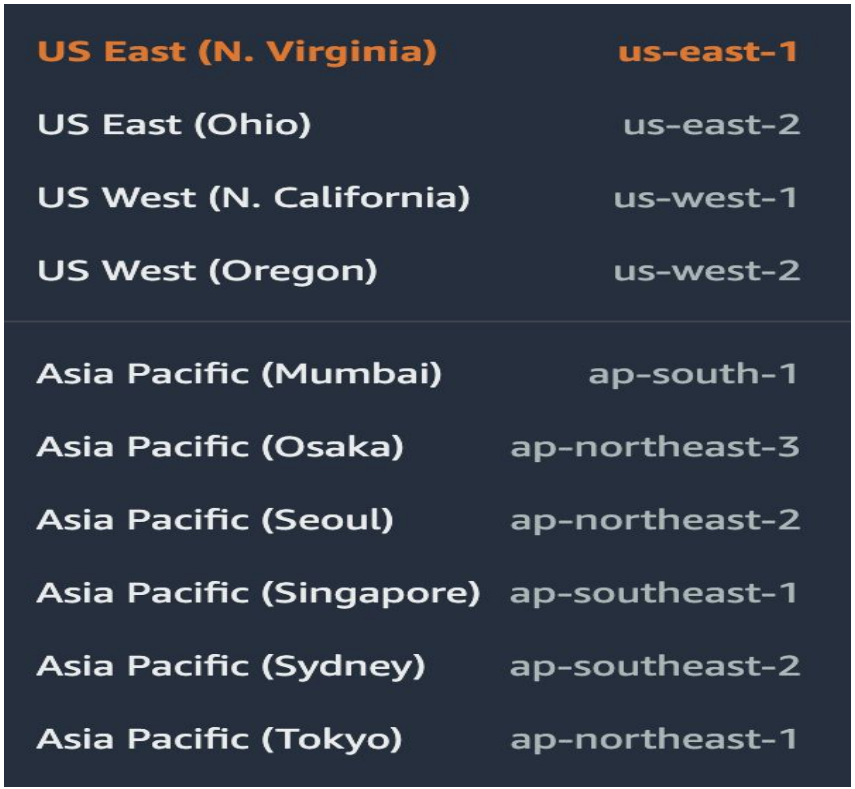
- Amazon overall generated $24.8 billion in operating profits in 2021, and AWS was responsible for $18.5 billion (or 74%) of it.
- Basically, a business segment that contributes 14% of overall revenue is generating roughly three-quarters of Amazon's total operating profits.

AWS Regions

- Where do AWS servers physically live?
- In AWS regions!
- Region == ***cluster of data centers***
- These are different geographic locations where AWS has its data centers – like Ohio, N. California, etc. are sample regions
- *Exact* location of regions is secret - to avoid potential attack (AWS runs a lot of the internet, one data center going down could cause severe damage)

AWS Regions

- These are some sample regions from the AWS console
- Currently 30 regions worldwide (as of early 2023) w/ plans to add 5 more



What regions do we use at CG?

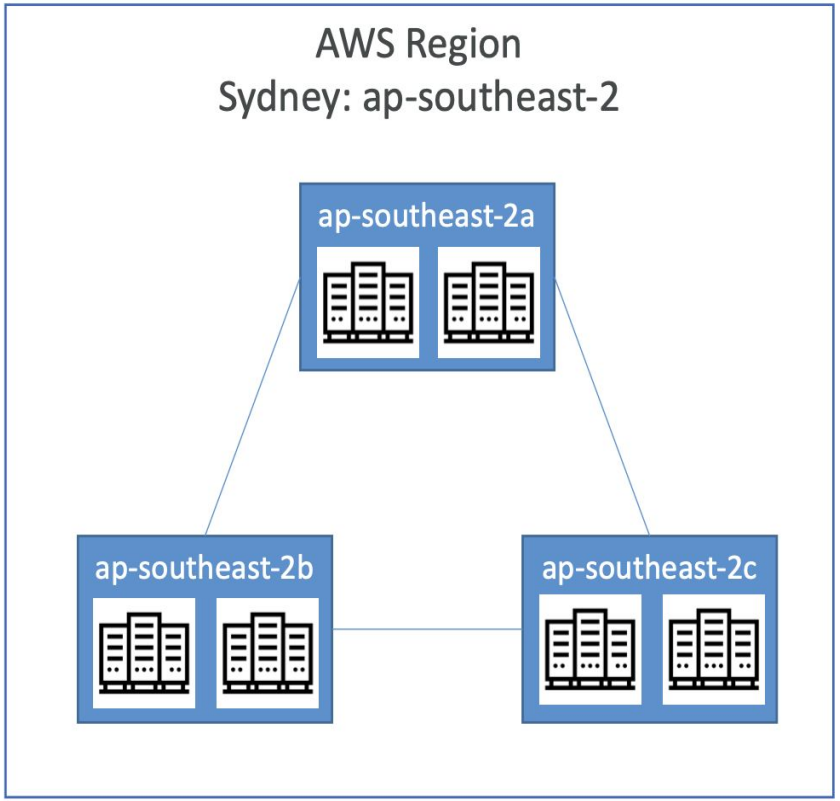
- us-east-1 and us-west-2

How do we choose an AWS region?

- When we launch an application in AWS, we have to choose a region – so how do we choose? There's a few considerations:
- **Location/Proximity:** Typically you want your application to live on servers that are physically closest to your end users
- **Availability within a region:** Some AWS services are not available within certain regions
- **Keeping Compliance:** some governments (like Canada) have strict requirements saying that their citizens data must be stored on servers in the same country

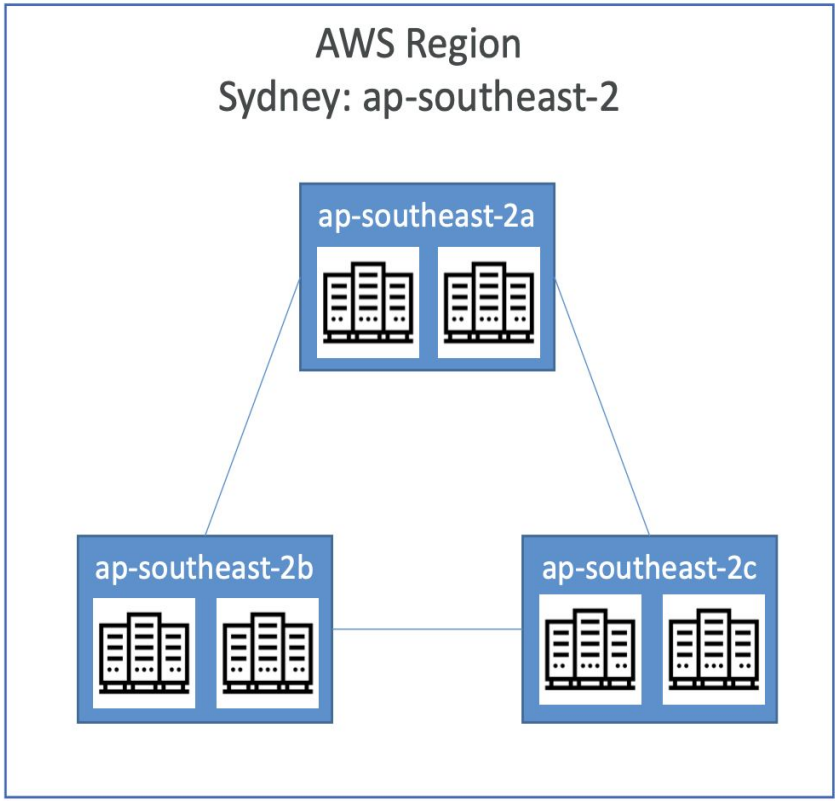
Availability Zones (AZ's)

- AWS has multiple AZs (3-6) within a given region
- On the right you can see the Sydney Australia region of ap-southeast-2 – and within that region there are 3 AZ's
- AZ's distinguished from one another by the trailing letter, ie the "a" in ap-southeast-2a
- Each AZ is a physically separate data center, but they are not completely independent of each other..



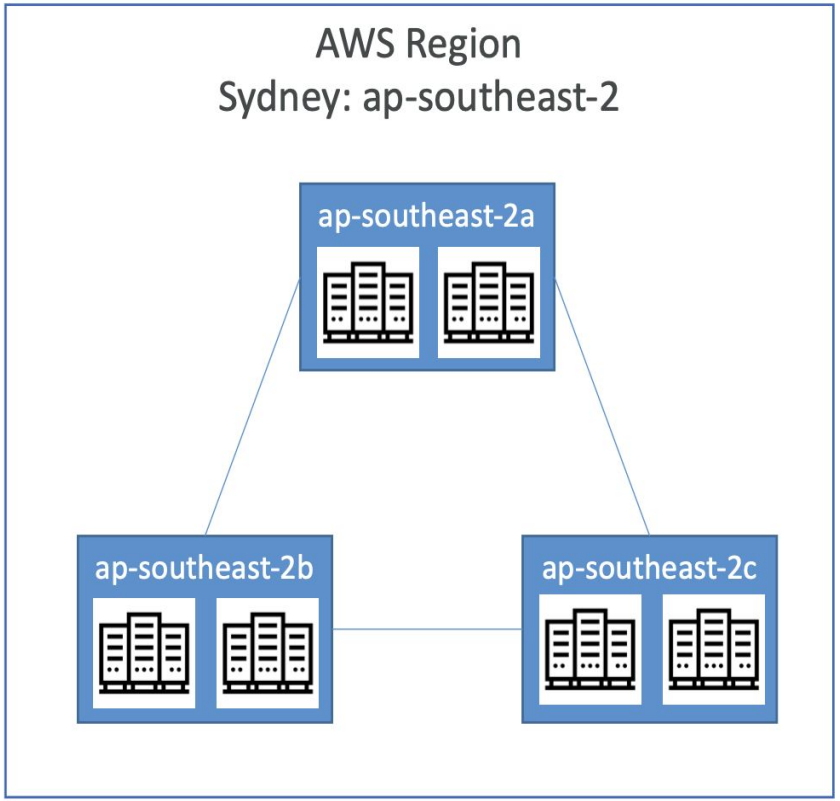
Availability Zones (AZ's) - redundancy

- Although AZ's are separate they are inter-connected via a low-latency network…
- This way if one AZ goes down (maybe a tornado hits), if your workloads reside on multiple AZ's you should not experience downtime



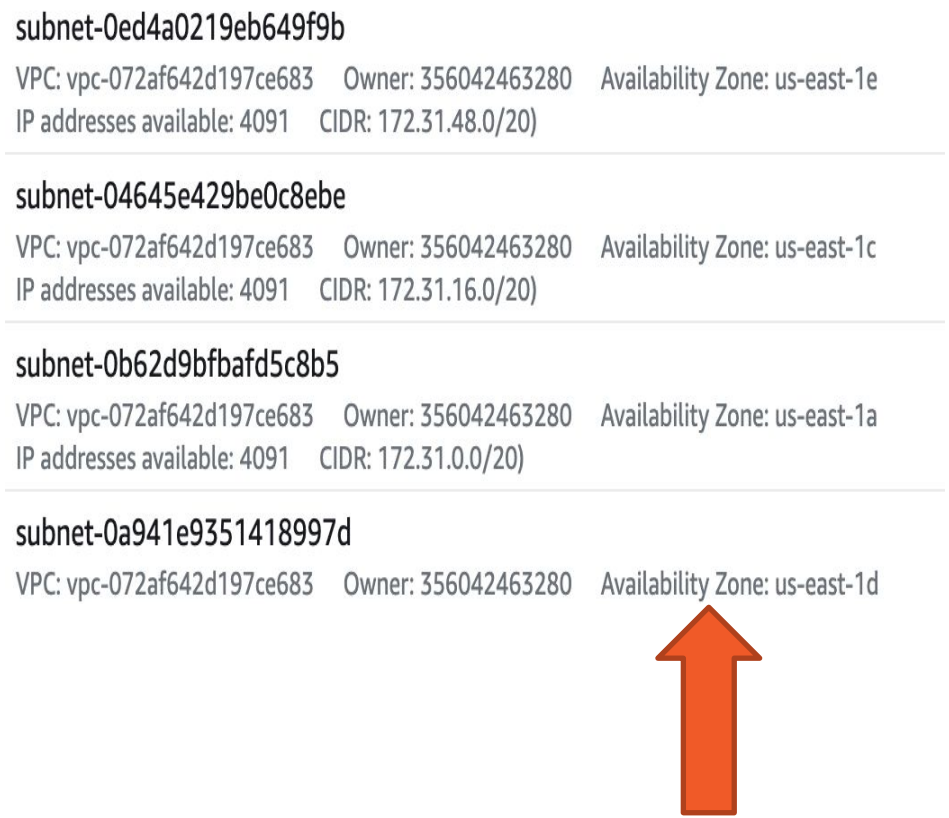
Availability Zones (AZ's) - redundancy

- AZ's are physically separate by many kilometers, although they are all within 100 km of each other
- Key point: AZ's give ***additional redundancy in a region***
- Another example: us-east-1 has 6 AZ's, us-west-1 has 3 AZ's…it varies but most regions have 3 AZ's (for now)
- So, if one AZ goes down in us-east-1 then there's still 5 others. The entire region does not go down



Availability Zones (AZ's)

- Again, here we can see that AZ's are distinguished from one another with a trailing letter in the region name
- So if our region is **us-east-1**, the AZs in that region are us-east-1**a**, us-east-1**b**, us-east-1**c,** us-east-1**d**, us-east-1**e**, and us-east-1**f**



AWS – global and region scoped services

- To build apps on the cloud we need a lot of different services
- AWS has some services that are ***global***, meaning they are not limited by region – like IAM, Route 53 (DNS), CloudFront
- But most AWS services are ***region-scoped*** – like Lambda, EC2, and more!
- Also see: https://aws.amazon.com/about-aws/global-infrastructure/regional-product-services/

IAM - Identity Access Management

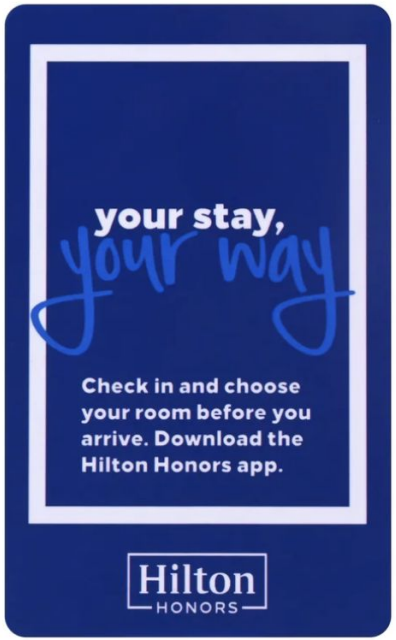
IAM – why?

- IAM helps us to define who can do what
- Also helps define what service can do what
- Some people (users) in your organization will have more permissions than others
- Some users need admin rights and some will need lesser privileges inside of AWS

IAM – more on why?

- AWS costs real money - easy to forget with cloud resources
- So we don’t want everyone to have unlimited access b/c mistakes can be made
- Especially if people don’t have much AWS experience!
- Basically we want to protect our AWS resource and account(s)

IAM – good analogy



IAM – good analogy

- IAM is a bit like a hotel key card which gives you access to some areas of the hotel (like your room, the pool, gym etc)
  - but it denies you access to other areas like kitchen, meeting rooms, other guest rooms, etc
- AWS and it's associated services is the "hotel" in this analogy

IAM – global service

- IAM is a ***global*** service
- Since users, unlike cloud resources (VM's, databases, storage and such), are not tied to a particular region/data center so this makes sense
- Quick demo of this

IAM – root account

- The root account allows full access to your AWS resources
- But if your root user credentials are stolen then anyone can create resources under your account and incur costs
- You should keep the root user account around in case you need to change billing details, add MFA, etc - but never use it for day to day operations.

Even for admin tasks, you can create admin users through IAM to do everything you need – avoid using root

How many AWS accounts do you need?

Typically more than one account is always required.

First account is the Master Account (or Management Account). Bills of all the accounts get consolidated here. You get tiered discount as well.

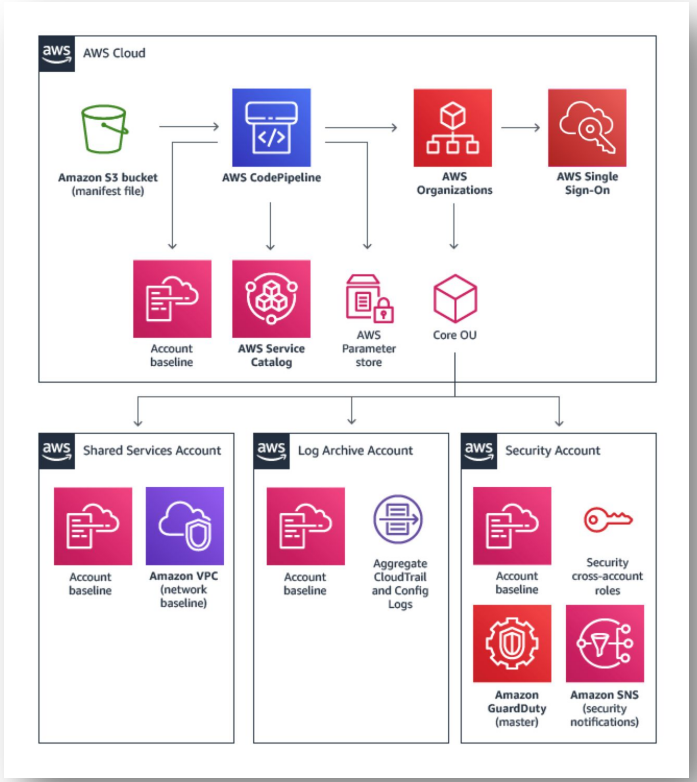
Other accounts (as per the best practices):

Log Archive Account

Security Account

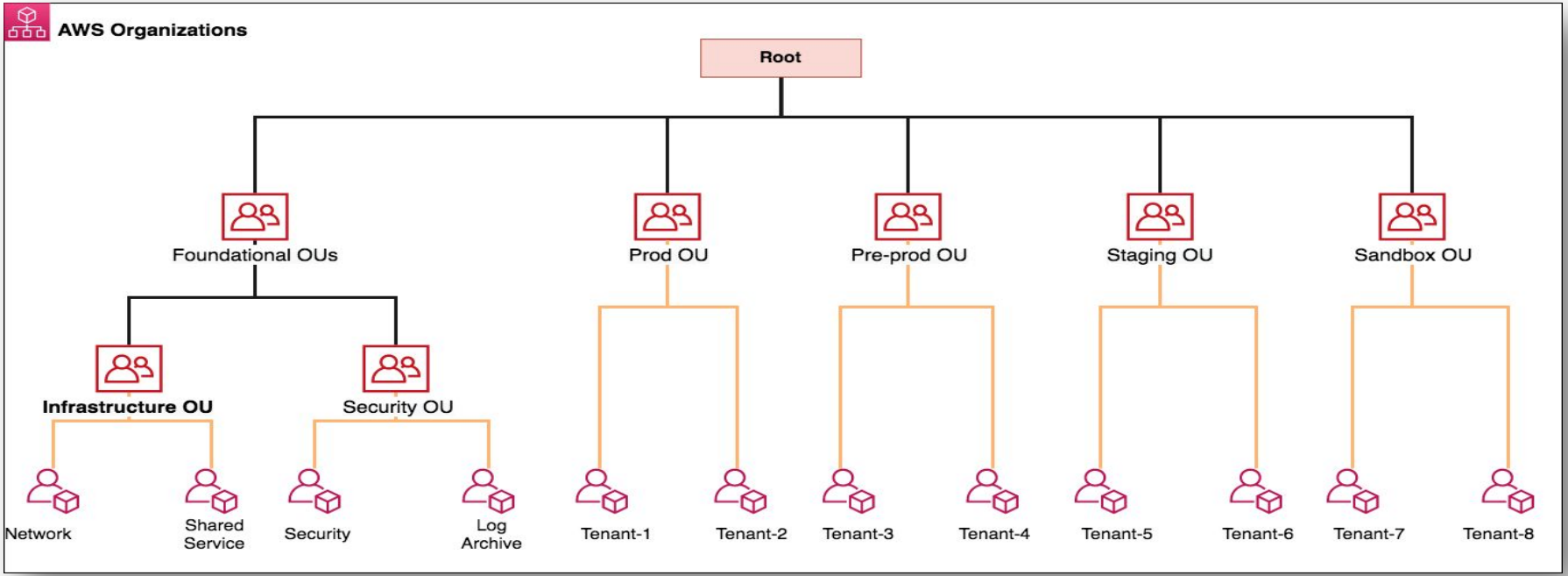
Shared Services Account

Other accounts based on Business Unit, Workload, etc.



Managing multiple accounts – AWS Organizations

- SCPs (Service Control Policy) can be applied at any node.
- **Master/Management Account** remains **unaffected** by **SCPs**.



Creating a New AWS Account

- New AWS account creation process
  - Independent account creation
  - Under AWS Organizations
- Free Tier usage
- Why are Limits/Quota important?

IAM users and groups

- Users are tied to actual people in your organization – Bob, Lisa, Raj, etc..
- Groups allow us to give the same permissions to multiple users

IAM users and groups

- Groups can not contain other groups – only users
- Users can be part of multiple groups, and do not have to be part of a group

IAM Policies – define permissions

- Policies are how we define who can do what in AWS
- Example: Bob can read and delete EC2 instances
- Policies are JSON documents – see on right
- Policies can be applied to users and/or groups…

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "elasticloadbalancing:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloudwatch:ListMetrics",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:Describe*"
            ],
            "Resource": "*"
        }
    ]
}
```

IAM Policies – define permissions

- In example on right, we can see that first entry "allows" us to describe any ec2 instances
- when we see the star (*) by resource that means all resources
  - ie - we could restrict by specific ec2 instances, but in this case we are not doing that
- The third entry allows us to to list, get and describe any cloudwatch resource

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "elasticloadbalancing:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloudwatch:ListMetrics",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:Describe*"
            ],
            "Resource": "*"
        }
    ]
}
```

IAM Policies – define permissions

- Policies by themselves do nothing, they are just JSON documents
- But, once we apply them to a user, group, or service, policies grant or deny certain actions on certain resources
- Also note that the 2 possible options for effect are allow and deny - in the example on right we are only using the "allow" effect
  - We would use deny to explicitly deny permissions

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "elasticloadbalancing:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloudwatch:ListMetrics",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:Describe*"
            ],
            "Resource": "*"
        }
    ]
}
```

IAM Policies – least privilege

- Generally, in AWS and other cloud providers you should apply the least privilege principle
- This means that you do not give more permissions than a user actually needs…

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "elasticloadbalancing:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloudwatch:ListMetrics",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:Describe*"
            ],
            "Resource": "*"
        }
    ]
}
```

IAM Policies – inline and managed

- There are 2 types of policies: inline and managed
- Inline Policies – these are written specific to a particular IAM entity.
- Managed Policies – these can be attached to multiple IAM entities.

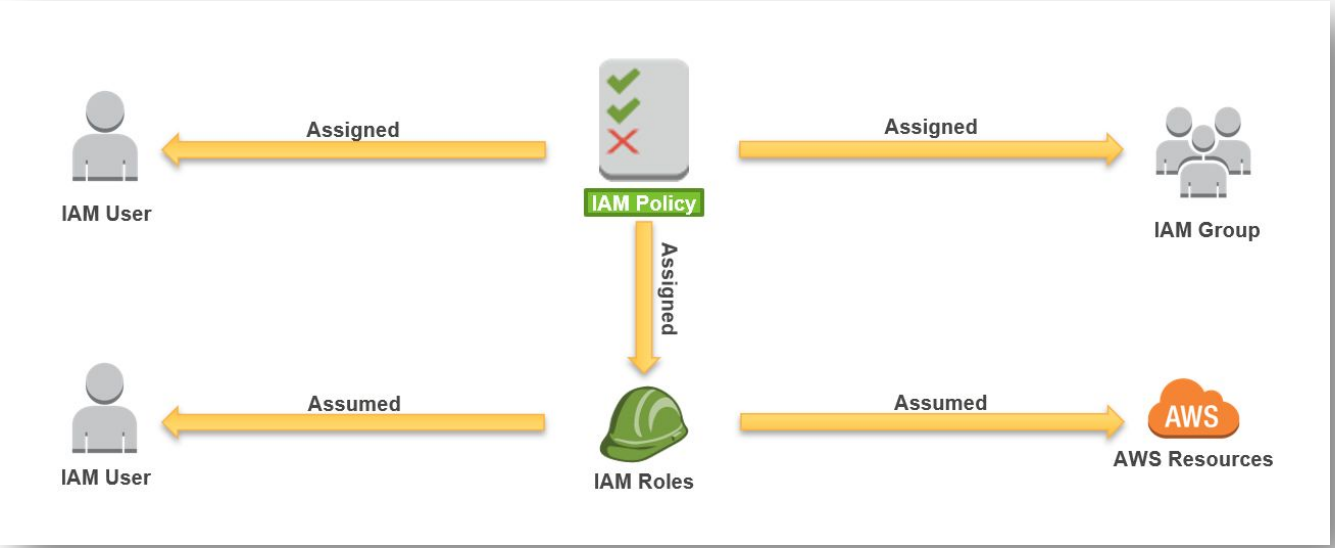
  The 2 types of managed policies:

  - Amazon Managed Policies
  - Customer Managed Policies

IAM Policies – inline and managed

When two conflicting permissions come together,

DENY always WINS instead of ALLOW



IAM Roles and services

- Why do we need roles?
  - Users/human beings are not the only ones who need access to thing
- AWS services will often need to perform some actions on your behalf
- Example: perhaps our EC2 instance needs to list the IAM users. By default, it does not have permissions for this

IAM Roles and services Lab/Hands On

- LAB: perhaps an EC2 instance needs to list the iam users. By default, it does not have permissions for this
- But we can assign an EC2 instance a role and then it will be able to list iam users – let's see this in a lab!

IAM Roles - more

- Roles are similar to users b/c they are IAM identities with permission *policies*
- *Think of it as "assuming a role"*
- Common use-cases for roles: Lambda instance roles, EC2 instance roles, CloudFormation roles

IAM Access Keys

- We have access keys to allow us to connect to our account programmatically (as opposed to the AWS console)
- You NEVER want to put those inside a github repository/any source control - here's an article on what could happen if you expose your access keys:
- https://www.tomshardware.com/news/aws-45000-usd-bill-for-crypto-mining-hack
  - Basically his access keys were stolen and then his AWS account was used to do bitcoin mining via AWS lambdas
  - All for $800 of crypto!

IAM Access Keys

- Let's go ahead and setup our virtual machine to connect to AWS programmatically
- Open up a terminal window and type "aws configure"
- Then put in your AWS access key ID and secret access key and for region put in "**us-west-2**", and for default output format put in **json**
- Now to validate that **aws configure** worked, you can type **aws sts get-caller-identity** and it should return some data that looks like this:

```
{
    "Account": "123456789012",
    "UserId": "AR#####:#####",
    "Arn": "arn:aws:sts::123456789012:assumed-role/role-name/role-session-name"
}
```

IAM Best Practices

- **Never** use Root User's access keys. Delete them.
- Create individual IAM users.
- Use groups to assign permissions to IAM users.
- Grant **least privilege** always.
- Configure a **strong** password policy.
- **Enable MFA** for all the IAM users.
- Use **roles** for applications that run on Amazon EC2 instances.
- **Delegate by using roles** instead of by sharing credentials.
- **Rotate** credentials regularly.
- Remove unnecessary/inactive users and credentials.
- Use policy conditions for extra security.

IAM vs AWS Cognito

- Cognito is an AWS service that allows you to create users and their associated authentication
  - Think: maybe you are building a new social network - AWS Cognito can help you create all the users and manage their passwords/usernames/etc.
- *IAM* however is used to ***manage access to AWS resources (like EC2 instances, Lambdas, etc...)*** - not a specific ***application*** like Cognito

IAM Labs

- **Execute this lab: https://docs.aws.amazon.com/IAM/latest/UserGuide/tutorial_cross-account-with-roles.html**
- **Execute this lab:**
- **https://docs.aws.amazon.com/IAM/latest/UserGuide/tutorial_managed-policies.html**

Shared Responsibility Model

Shared responsibility model

- Security and compliance is not just the responsibility of AWS when you (the customer) use their services
- Rather, it's a ***shared*** responsibility between AWS and you the customer!
- That is what is meant by the shared responsibility model

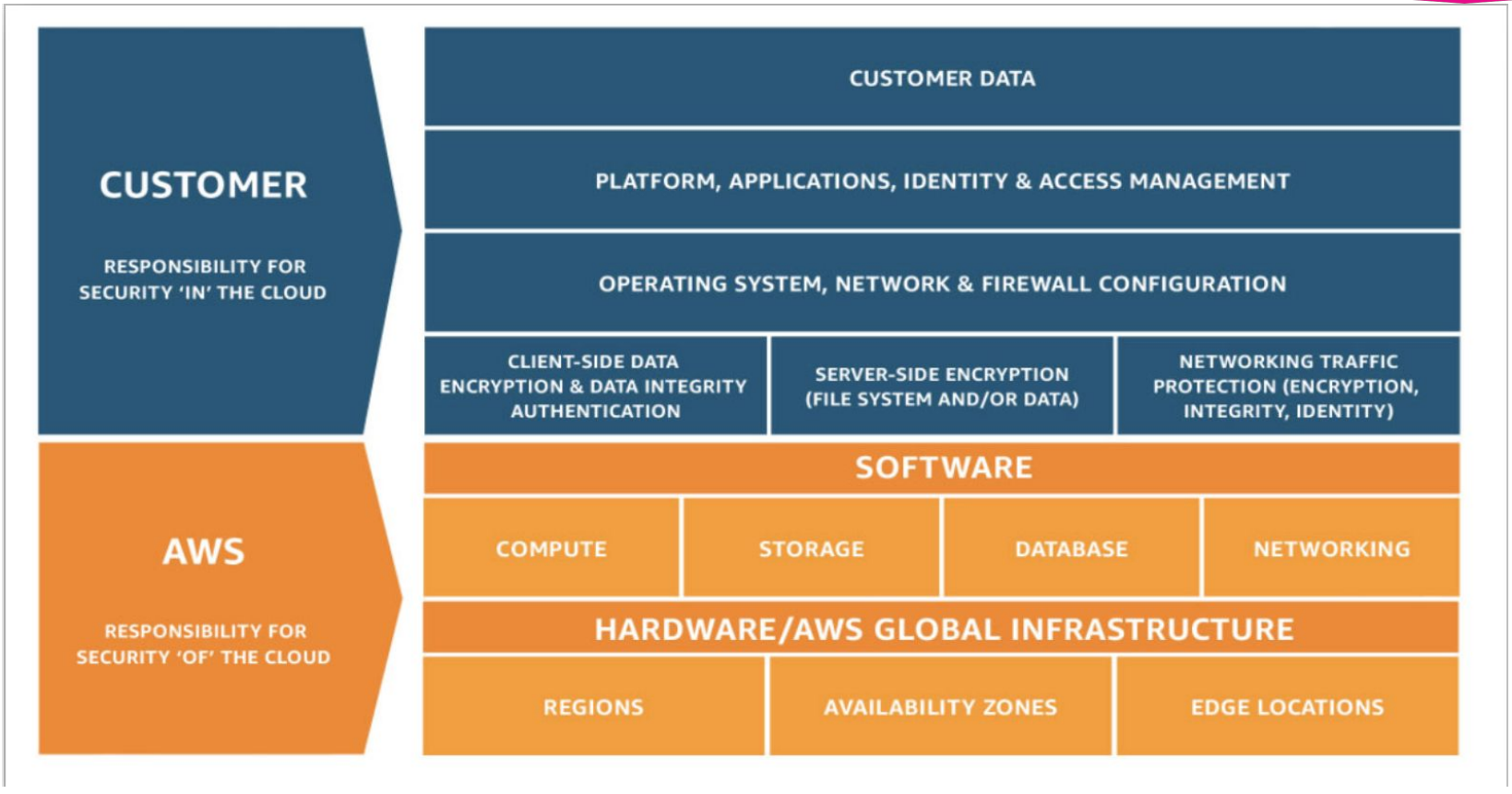
Shared responsibility model - example

- If you use an EC2 instance you (not AWS) are responsible for installing and managing things like operating system updates + security patches
- EC2 gives you a lot of control, but that comes with more responsibility since it’s an IaaS solution

Shared responsibility model - example

- Similarly even with other services where AWS has more control and you have less – like S3, a PaaS solution – you still are responsible for a lot
- Things such as the data, IAM tools to apply permissions, etc..

Shared responsibility model graphic

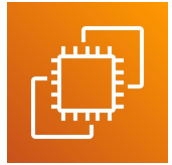


Shared responsibility model – applying it

- Ok great, so what should we do to ensure that we are being responsible customers?
- See here: https://aws.amazon.com/compliance/shared-responsibility-model/

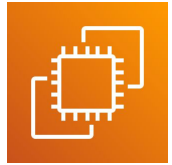
EC2 – IaaS in AWS

EC2 Basics – Elastic Cloud Compute



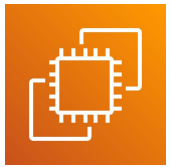
- *Note:* we cover EC2 here (briefly) because it is something that you should be aware of – as it's used in a lot of examples within AWS documentation/tutorials
- But, for ***Capital Group context***, our trajectory for the cloud is more towards ***containerized solutions*** in AWS – so EC2 ***is not necessarily a solution that you should look to implement as it's also expensive***
- That being said, a number of AWS services use EC2 behind the scenes - like ECS and Cloud9 (which we will get plenty of experience with)
    - So, understanding EC2 is still very important
- And, the AWS certification tests require a decent understanding of EC2

EC2 Basics – Elastic Cloud Compute



- **Operating System (OS):** RHEL, SLES, Ubuntu, Amazon Linux, Windows or Mac OS
- How much compute power & cores (CPU)
- How much random-access memory (RAM)
- How much storage space
- Network-attached (EBS)
- Host (Instance Store)
- **Network card:** speed of the card, Public IP address
- **Firewall rules:** security group
- Bootstrap script (runs once during the launch): EC2 User Data

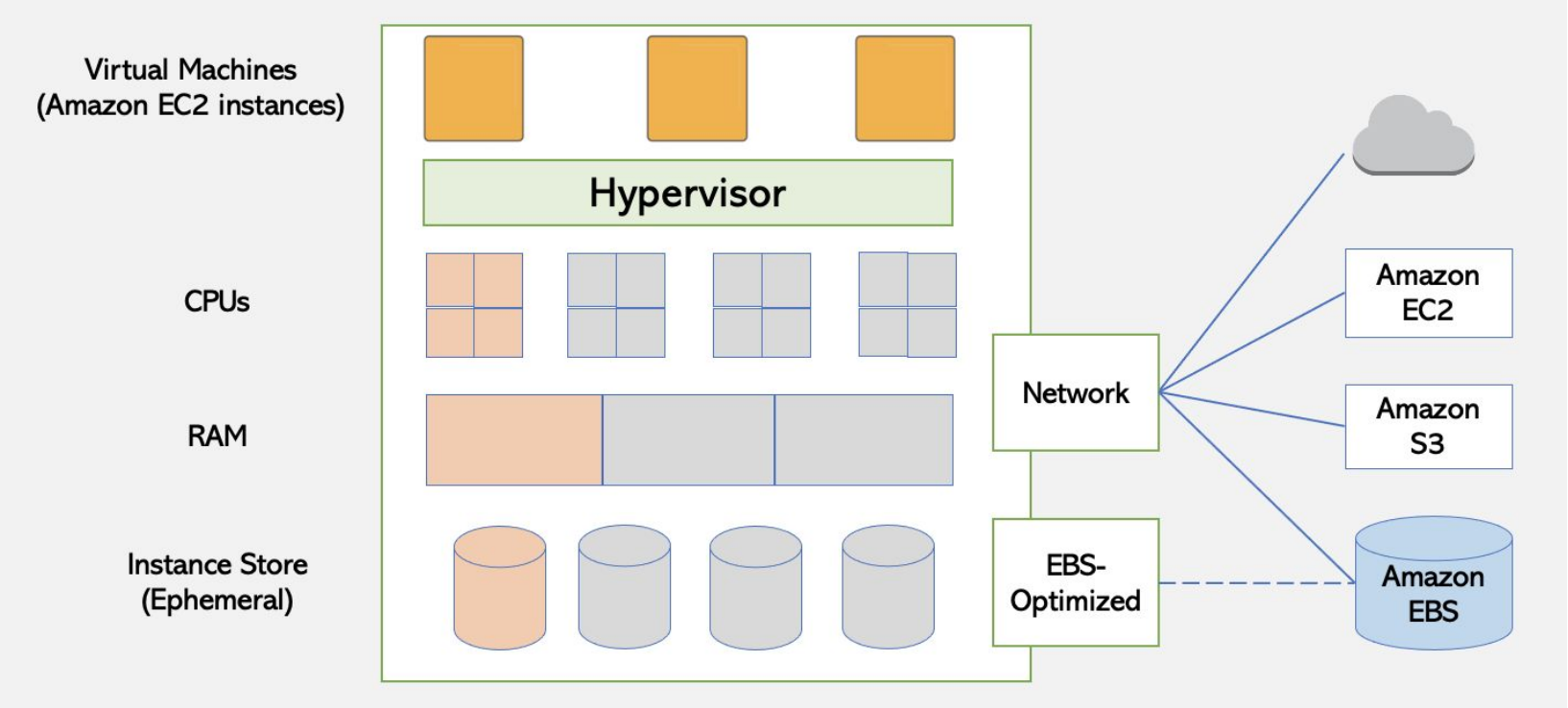
EC2 Basics



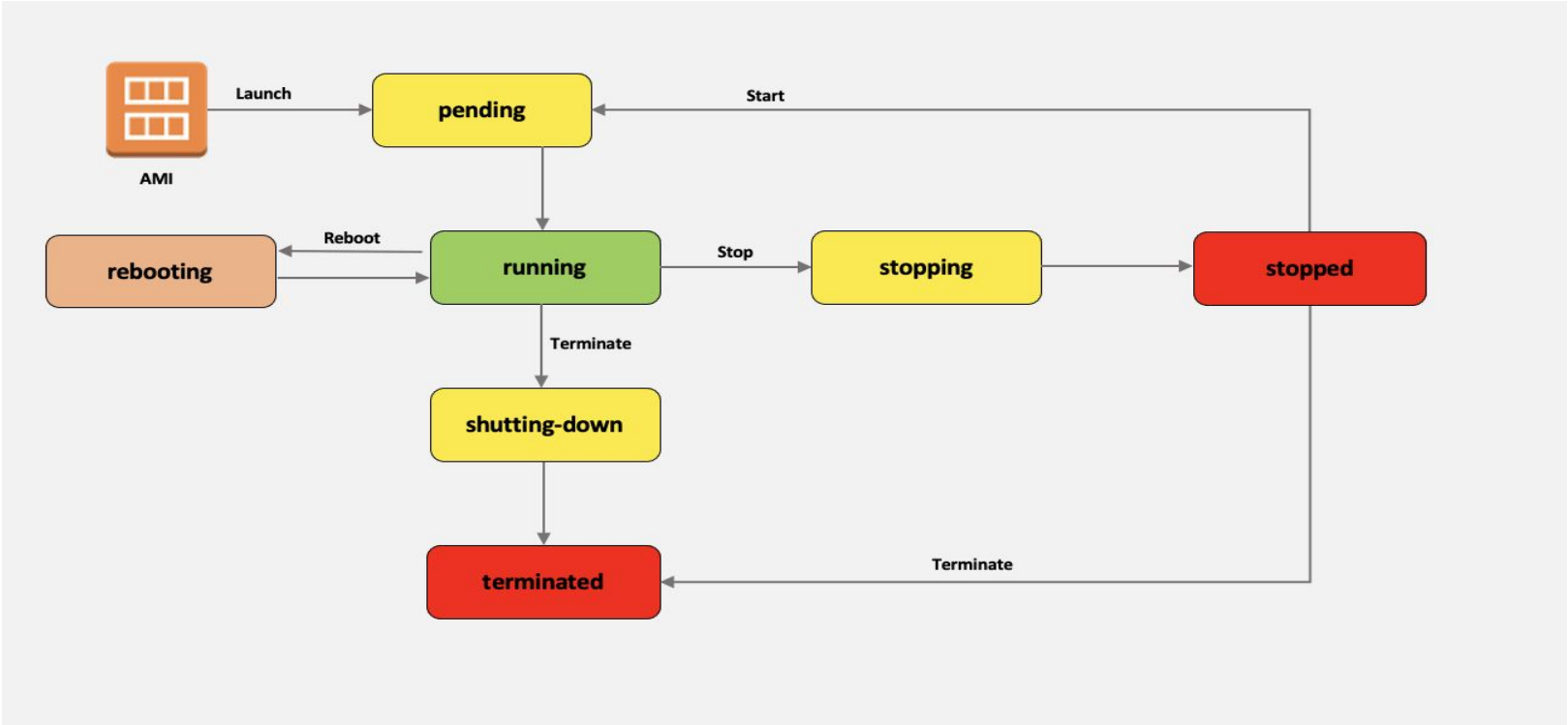
If you use an EC2 instance you (not AWS) are responsible for installing and managing things like operating system updates + security patches

EC2 gives you a lot of control, but that comes with more responsibility since it's an IaaS solution

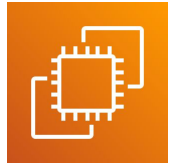
EC2 Architecture



EC2 Lifecycle

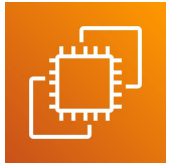


Reboot vs Stop vs Terminate



Characteristic	Reboot	Stop/Start (EBS-backed instances only)	Terminate
Host computer	The instance **stays on the same host computer**.	The instance runs on a **new host computer**.	
Public IP address	No change	**New address assigned**	
Elastic IP addresses (EIP)	EIP remains associated with the instance.	EIP remains associated with the instance.	EIP is **disassociated** from the instance.
Instance store volumes	Preserved	**Erased**	**Erased**
EBS volume	Preserved	Preserved	Boot volume is **deleted by default**.
Billing	Instance billing hour doesn't change.	You **stop incurring charges** as soon as state is changed to *stopping*.	You **stop incurring charges** as soon as state is changed to *shutting-down*.

EC2 Pricing



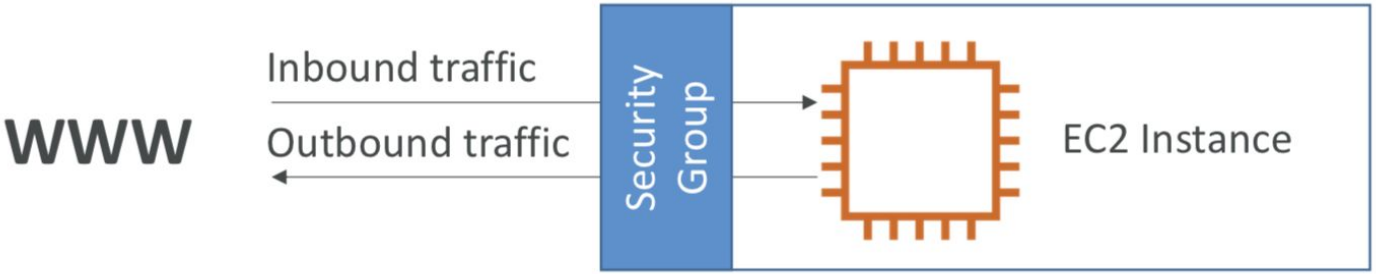
On-Demand	Reserved	Spot Instances
• No commitment • Pay by the hour • Any partial hour converted to full • A new billing cycle starts whenever an instance changes to "Running" state • A billing cycle ends when instance changes to "Stopping" state • Billing cycles don't start at 9am, 10 am etc. • Pay per second (supported for few Operating Systems)	• Two terms available – 1 year or 3 years • 3 Payment options: - Full Upfront - Partial Upfront - No Upfront (not for 3 years term) • Lot of saving in comparison to On-Demand • Gives you Capacity Guarantee as well • You commit the usage for chosen term • You can re-sell on AWS if you choose not to use • Considered for full term	• Unused capacity at AWS is given in market for bidding • Look at pricing history and decide bid price • Instances are terminated with 2 minutes notice when market price goes above bid price • If terminated by AWS, last partial hour is free • Optionally, use Spot Block option with bid to block the instance (maximum 6 hours)

Security Groups in AWS

- Security Groups are a fundamental piece of network security in aws
- They act as virtual firewalls to our AWS resources
- For example, if we have a website running on an EC2 instance
  - how do we specify who can access it and through which protocol (SSH, HTTP, etc…)
- The answer is **security groups!**

Security Groups in AWS

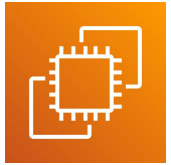
- Security Groups can reference an IP or another security group
- Security groups only contain **allow** rules (ie no deny)



Commonly used ports to know

- 80 - this is for http (not https) - for unsecured websites
- 443 - this is for https - for securied websites
- 22 - SSH - secure shell this is to SSH into a virtual machine, like your EC2 instance for example
- 3306 - the MySQL database default port

EC2 Lab



Execute the lab here:

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EC2_GetStarted.html

https://github.com/varoonsahgal/cg-cloud-foundations/wiki/EC2-Lab

Databases on AWS

Unmanaged vs Managed Services

Unmanaged:

Scaling, fault tolerance, and availability are managed by you.

Amazon Elastic Compute Cloud (EC2)

Managed:

Scaling, fault tolerance, and availability are typically built in to the service.

Amazon Relational Database Service (RDS)

Relational and Non-relational Databases

	Relational	Non-Relational
Data Storage	Rows and Columns	Key-Value
Schemas	Fixed	Dynamic
Querying	Using SQL	Focused on collection of documents
Scalability	Vertical	Horizontal

Relational

ISBN	Title	Author	Format
9182932465265	Cloud Computing Concepts	Wilson, Joe	Paperback
3142536475869	The Database Guru	Gomez, Maria	eBook

Non-Relational

```
{
        ISBN: 9182932465265,
        Title: "Cloud Computing Concepts",
        Author: "Wilson, Joe",
        Format: "Paperback"
}
```

Different Database types and use cases

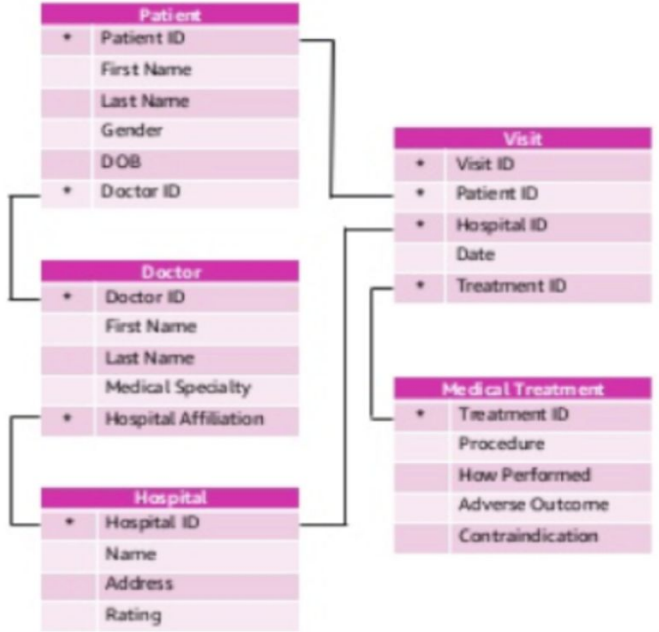
Common data categories and use cases

	Relational	Key-value	Document	In-memory	Graph	Time-series	Ledger
	Referential integrity, ACID transactions, schema-on-write	High throughput, low-latency reads and writes, endless scale	Store documents and quickly access querying on any attribute	Query by key with microsecond latency	Quickly and easily create and navigate relationships between data	Collect, store, and process data sequenced by time	Complete, immutable, and verifiable history of all changes to application data
	Lift and shift, ERP, CRM, finance	Real-time bidding, shopping cart, social, product catalog, customer preferences	Content management, personalization, mobile	Leaderboards, real-time analytics, caching	Fraud detection, social networking, recommendation engine	IoT applications, event tracking	Systems of record, supply chain, health care, registrations, financial

Relational databases

Relational data

- Divide data among tables
- Highly structured
- Relationships established via keys enforced by the system
- Data accuracy and consistency



Different Database services

- Amazon RDS, Aurora – Relational
- Amazon DynamoDB – NoSQL
- Amazon ElastiCache – In memory
- AWS Database Migration Service – For migrating data

CG Context: Dynamo vs RDBMS

CG Context: DynamoDB vs Relational

- Given the fact that CG is in the finance sector, our data is mostly relational in nature
  - Hence, **relational** databases are what we mostly use
- Non-relational databases such as DynamoDB are not as commonly used at CG

AWS RDS

RDS – Relational Database Service

- RDS is not a DB of its own – rather it's a service around a database ***that you choose!***
- RDS gives you 7 databases engines to choose from:
- Including Amazon Aurora MySQL-Compatible Edition, Amazon Aurora PostgreSQL-Compatible Edition, MySQL, MariaDB, PostgreSQL, Oracle, and Microsoft SQL Server.

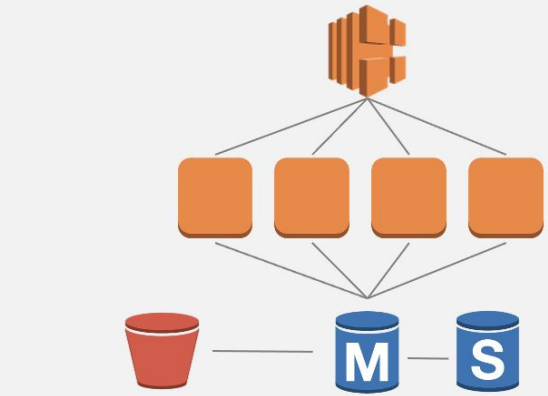
RDS

- RDS could be provisioned as Single-AZ or Multi-AZ
- With Multi-AZ operation, your database is **synchronously** replicated to an instance in another Availability Zone in the same AWS Region.

RDS

- **Failover** to the standby automatically occurs in case of master database failure. There will be delay for few seconds for sure.
- Planned maintenance is applied first to standby databases.

Resilient Durable Application Architecture - RDS



Elastic Load Balancing load balancer instance

Application, in Amazon EC2 instances

Amazon RDS database instances: Master and Multi-AZ standby

DB snapshots in Amazon S3

We could just deploy a DB instance on EC2…

- But why would we want to use RDS instead?
- B/C RDS is PaaS (Managed service) and we can offload the database administration. RDS gives us:
- Dashboards for monitoring
- OS Patching, Automated provisioning
- Read replicas

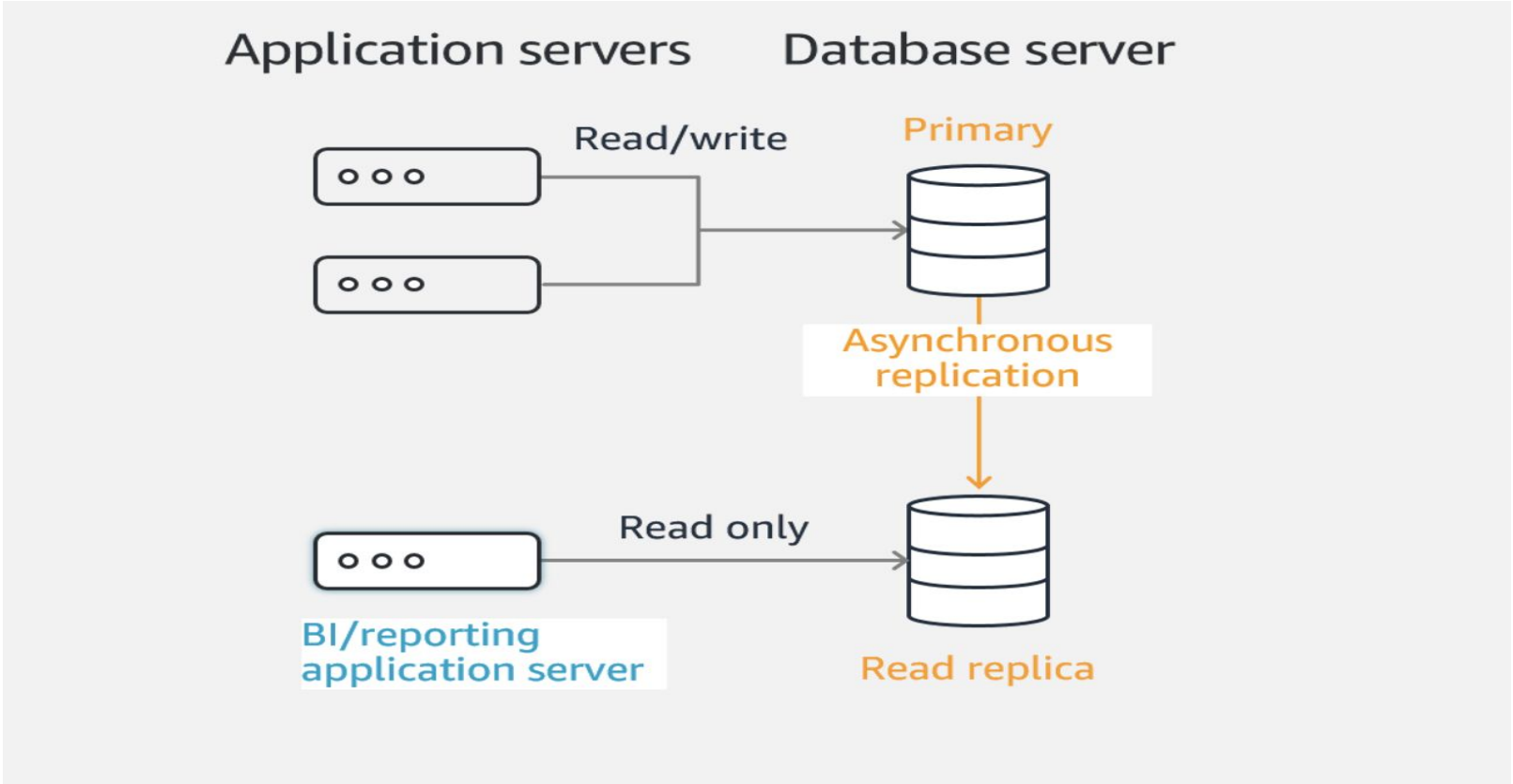
We could just deploy a DB instance on EC2…

- Disaster recovery via multi-AZ setup
- Manual Scaling – both vertical and horizontal..
- However, vertical **auto-**scaling not supported (whats the difference?)
  - But you can enable Storage auto-scaling (horizontal)
- Vertical scaling: changing the DB instance type

RDS Read Replica

- Read replica == copy of the primary DB instance
- Can offload read requests
- Why use them? b/c they make it easy to scale out and improve DB ***read*** performance
- Especially good if your application is read-heavy as opposed to write-heavy

RDS Read Replica - Diagram



RDS Read Replica

- Read replica == copy of the primary DB instance
- Can offload read requests

RDS Read Replica – use case

- Why use them? b/c they make it easy to scale out and improve DB ***read*** performance
- Especially good if your application is read-heavy as opposed to write-heavy
- Great for read-heavy database workloads like reporting applications

RDS Read Replica == SELECT in SQL

- Since they are ***read*** replicas and not write replicas they are used for SELECT kind of statements in SQL
- As opposed to statements which write to a DB – like DELETE, INSERT, UPDATE, etc..

RDS Read Replicas – Network Cost

- When data is transferred from one AZ to another, there is typically a network cost charged by AWS
- However, when we create RDS Read Replicas within the same region, we don’t pay that fee..

Takeaways

- RDS is a PaaS offering allowing developers to focus on their applications as opposed to DB administration
- RDS offers many DB engines to choose from as well
- RDS uses read replicas

RDS Lab

- https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/TUT_WebAppWithRDS.html

- https://github.com/varoonsahgal/cg-cloud-foundations/wiki/RDS-Lab

Amazon Aurora

Aurora walkthrough and demo

- Demo of this service

Aurora Labs

- https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/CHAP_GettingStartedAurora.CreatingConnecting.Aurora.html
- https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/CHAP_GettingStartedAurora.CreatingConnecting.AuroraPostgreSQL.html
- https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/TUT_WebAppWithRDS.html

Dynamo DB

DynamoDB Lab

- Execute labs here:
- https://aws.amazon.com/getting-started/hands-on/create-nosql-table/
- https://aws.amazon.com/getting-started/hands-on/create-manage-nonrelational-database-dynamodb/ (time permitting)
- https://github.com/varoonsahgal/cg-cloud-foundations/wiki/Dynamo-DB-Lab

DynamoDB

- Demo + walkthrough of service…

API Gateway

API Gateway

Example: Building a Serverless API

AWS API Gateway

- AWS Lambda + API Gateway: No infrastructure to manage
- Support for the WebSocket Protocol
- Handle API versioning (v1, v2…)
- Handle different environments (dev, test, prod…)
- Handle security (Authentication and Authorization)
- Create API keys, handle request throttling
- Swagger / Open API import to quickly define APIs
- Transform and validate requests and responses
- Generate SDK and API specifications
- Cache API responses

API Gateway Integrations - High Level

- Lambda Function
  - Invoke Lambda function
  - Easy way to expose REST API backed by AWS Lambda
- HTTP
  - Expose HTTP endpoints in the backend
  - Example: internal HTTP API on premise, Application Load Balancer…
  - Why? Add rate limiting, caching, user authentications, API keys, etc…
- AWS Service
  - Expose any AWS API through the API Gateway
  - Example: start an AWS Step Function workflow, post a message to SQS
  - Why? Add authentication, deploy publicly, rate control…

API Gateway Integrations - High Level

- Edge-Optimized (default): For global clients
  - Requests are routed through the CloudFront Edge locations (improves latency)
  - The API Gateway still lives in only one region
- Regional:
  - For clients within the same region
  - Could manually combine with CloudFront (more control over the caching strategies and the distribution)
- Private:
  - Can only be accessed from your VPC using an interface VPC endpoint (ENI)
  - Use a resource policy to define access

API Gateway - Security

- User Authentication through
  - IAM Roles (useful for internal applications)
  - Cognito (identity for external users – example mobile users)
  - Custom Authorizer (your own logic)

- Custom Domain Name HTTPS security through integration with AWS Certificate Manager (ACM)
  - If using Edge-Optimized endpoint, then the certificate must be in **us-east-1**
  - If using Regional endpoint, the certificate must be in the API Gateway region
  - Must setup CNAME or A-alias record in Route 53

AWS S3

AWS S3 – simple storage solution

- S3 is for storage
- It’s not a database
- What kind of storage?

S3 Use Cases

- Backup + Storage
- Images (eg. for sites where folks upload a lot of pictures)
- Static websites
- Disaster Recovery
- Archives

S3 Case studies

Customers

NASCAR modernizes multi-PB media archive at speed with Amazon S3 »

Snap optimizes cost savings while storing 2 exabytes - over 1.5 trillion photos and videos - on Amazon S3 Glacier Instant Retrieval »

Shutterstock transforms IT and saves 60% on storage costs with Amazon S3 »

Runtastic saves €300,000, stays on track for growth using Amazon S3 »

AWS what is durability?

- Durability ***is not the same as availability***
- Durability is probability that the object will remain intact and accessible after a period of year
- 100% durability – no possibility of object being lost
- 90% - there's a 1 in 10 chance of being lost

AWS S3 11 9s - durability

- 11 9's is **99.999999999%**
- S3 guarantees 11 9's of durability – so **99.999999999%**
- This means that for example, if you store 10,000,000 objects with Amazon S3, you can on average expect to incur a loss of a single object once every 10,000 years

AWS S3 – objects

- In AWS S3, we often use the term ***object***
- An object is a file and any metadata that describes that file
- ***Buckets*** are containers for objects
- To store an object in S3 you create a ***bucket*** and then upload objects into that bucket..

AWS S3 – buckets

- ***Buckets*** are containers for objects
- They must have a globally unique name
- Despite that fact, buckets are defined at the region level
- To store an object in S3 you create a ***bucket*** and then upload objects into that bucket..

AWS S3 Bucket Policies

- S3 buckets have policies as well
- What could we do with S3 bucket policies?
- We can: give access to other accounts, give public access to the bucket, and even force objects to be encrypted..

AWS S3 lab

- Execute the tutorials available at:
- https://docs.aws.amazon.com/AmazonS3/latest/userguide/GetStartedWithS3.html
- https://docs.aws.amazon.com/AmazonS3/latest/userguide/website-hosting-custom-domain-walkthrough.html
- https://docs.aws.amazon.com/AmazonS3/latest/userguide/HostingWebsiteOnS3Setup.html

AWS S3 lab takeaways

- What did we learn from the lab?

Thank you!

If you have additional questions, please reach out to me at:

varoon.sahgal@gmail.com

Linkedin:

https://www.linkedin.com/in/varoon-sahgal-b895b567

DevelopIntelligence
A PLURALSIGHT COMPANY