AWS Certified Solutions Architect - Associate (SAA-C02)

Contents

MODULE 1 – INTODUCTION & SCENARIO

Finding and Using the Course Resources

This course is a mixed theory and practical course. To get the most value it's suggested that you at least watch the [DEMO] lessons and ideally follow along step by step in your own AWS environments.

This lesson explains where the course resources can be found and how to best access them

Course GitHub Repo: https://github.com/acantril/AWS-sa-associate-saac02

Course GitHub HTTPS Clone Link: https://github.com/acantril/AWS-sa-associate-saac02.git

Git Utility Downloads: https://git-scm.com/downloads

Visual Studio Code: https://code.visualstudio.com/

Scenario - Animals4life

Animals4life.org is the scenario used throughout the course as a real-world like situation to explain theory and demonstrate practical examples.

This lesson will step you through the scenario and explain the key business issues and requirements. The organisation is fictitious and any resemblance to any real organisations is coincidental and unintended.

Animals4life.org

- Animal Rescue and awareness organization
- They are Global company with HQ in Brisbane Australia - 100 Staff
- They have Call Centre, Admin, IT, Marketing, legal & Accounts department
- 100 Remote workers across Australia and Globally
- Animal care, activists and Lobbyists
- Makor offices in London, NY and Seattle

Now the business's current technical infrastructure is a mess. They have a small, on-premises datacentre in Brisbane, as well as a number of colo facilities where they rent a total of five racks of space. The DC is old and the company who manages them is encouraging customers to migrate out as soon as possible.

Animals4life.org have previously had a vendor helping the business run an AWS pilot in the Sydney AWS region, but it was badly implemented, and it didn't provide the resilience and scalability that the business needed.

Animals4life.org is very cost-conscious, but equally very progressive, and willing to try new things, and adopt new technologies as long as it has a business benefit.

Connect with other students and your instructor

- Twitter: https://twitter.com/adriancantrill
- LinkedIn: https://www.linkedin.com/in/adriancantrill/
- Mail: adrian@cantrill.io
- Slack: https://techstudyslack.com
- https://talk.cantrill.io

MODULE 2 - COURSE FUNDAMENTAL & AWS ACCOUNT

AWS Accounts - The basics

Bigger, more complex projects or businesses will generally use more AWS accounts, and in this course, you're going to use multiple AWS accounts for the demo lessons. And this will help you understand how businesses actually use AWS in the real world.

At a high level, an AWS account is a container for identities is just the technically correct way of referring to things like users. So, the things which you use to log in to systems such as AWS. So, an AWS account contains users which you log in with and resources which you provision inside of that account.

When you create an AWS account, you give the account a name, you need to provide a unique email address and you also need to provide a payment method which is generally a credit card. So, if we're creating a production account, we might name the account PROD for production.

- **Every AWS account has an account ROOT USER**

For example, if you create a production AWS account then the account root user of that account can only login to that one production AWS account.

If you make another AWS account, let's say a developer account, then that account will have its own unique account root user with its own unique email address.

The production account root user can only access the production account and the developer account root user can only access the developer account.

Now, initially the account root user is the only identity, the only user created with an AWS account. So, initially, you have the blank account, the container and inside that is the account root user of that account.

Now the account root user has full control over that one specific AWS account and any resources which are created inside it.

The account root user can't be restricted. It will always have full access to everything within that one AWS account, which it belongs to. now, this is the reason why we need to be really careful with the account root user because if the username and password ever become known the result can be disastrous, because the details can be used to delete everything within the AWS account.

AWS is known as a pay-as-you-consume or pay-as-you-go platform. if you use a service within an AWS account for two min then you pay for two min of that service. Now, I want to touch on the security as it related to AWS accounts. so, you know that the account root user has full control over this one specific AWS account and this can't be restricted.

Well, you can create additional identities inside the AWS account, which can be restricted. this service called **Identity and Access Management known as IAM.** And with IAM you can create other identities inside the account, these can be different types of identities. like IAM users, IAM groups and IAM roles.

It's important to understand that all of these **IAM identities start off with no access to the AWS account**, but they can all be given full or limited access rights over this one specific AWS account.

IAM Basic

- **User:** Identities which represent **humans** or **applications** that need access to your account.
- **Group:** **Collection** of **related users** e.g., development team, finance or HR.
- **Role:** Can be used by **AWS Services**, or for granting **external access** to your account.

Creating an AWS Account [NEWUI]

This lesson demonstrates how to create an AWS account. Because the course is based around a scenario, we will be using a number of isolated accounts, this lesson shows the creation of the GENERAL AWS account.

Direct signup link: https://portal.AWS.amazon.com/billing/signup#/start

Best practices to create AWS account:

- Create two different AWS account.
- Now, the first account that you're going to be creating is the account that you're going to log into. We are going to start by referring to this as the **GENERAL (Management)** AWS account.
- When you create an AWS account, an account root user is created along with it.
- Remember, the account root user is specific to one single AWS account. So, this account root user has full control of the general AWS account.
- Once the account is created along with the root user, you can secure it by adding MFA.
- Second account: **PRODUCTION** secure it with MFA.

You can create different email account with this below tip.

catguy+AWSaccount1@gmail.com

catguy+AWSaccount2@gmail.com

catguy+AWSaccount3@gmail.com

Multi-factor Authentication (MFA)

AWS Multi-Factor Authentication (MFA) is a simple best practice that adds an extra layer of protection on top of your username and password. With MFA enabled, when a user signs in to an AWS Management Console, they will be prompted for their username and password (the first factor—what they know), as well as for an authentication code from their AWS MFA device (the second factor—what they have). Taken together, these multiple factors provide increased security for your AWS account settings and resources.

Why MFA is needed?

Generally, we use Username and password, if leaked, anyone can be you! Anyone can take your username and passwords and use them to log into an application and impersonate you as an identity. So, what we need is a way to improve this.

The term is known as factor or factors.

So, **factors are pieces of evidence which prove identity** and there are different types of factors.

MFA is when you use multiple factors or multiple types of factors to log in to an application.

Now, there are four (4) common factors that you will use to log in to any web application.

1) **Knowledge:** First is knowledge and this is something that you know. So, the knowledge factor includes usernames and passwords.
2) **Possession:** Second type is possession and this is something that you have, as a bank card or an MFA device or an MFA application. So, consider what you do when you use a bank card at an ATM, you have to have the bank card, you put inside the ATM and then you need to enter a pin. So, this is an example of MFA.
   In this case, you are using two factors, something that you have, the bank card, and something that you know, your pin.

3) **Inherent:** The next factor is inherent. So, this is something that you are, so a fingerprint, a face scan, voice identification, or an iris scan. these are all example of inherent factor.
4) **Location:** lastly, we have location. so, this can be rather a physical location, so a particular set of coordinates anywhere in the world or it can be the types of networks that you at currently logged into to access a system.

More factors mean more security & harder to fake.

Enable MFA:

To configure and enable a virtual MFA device for use with your root user (console)

1) Sign in to the AWS Management Console.
2) On the right side of the navigation bar, choose your account name, and choose **My Security Credentials**. If necessary, choose **Continue to Security Credentials**. Then expand the **Multi-Factor Authentication (MFA)** section on the page.
3) Choose **Activate MFA**.
4) In the wizard, choose **Virtual MFA device**, and then choose **Continue**.
   IAM generates and displays configuration information for the virtual MFA device, including a QR code graphic. The graphic is a representation of the secret configuration key that is available for manual entry on devices that do not support QR codes.
5) Open the virtual MFA app on the device.
   If the virtual MFA app supports multiple virtual MFA devices or accounts, choose the option to create a new virtual MFA device or account.
6) The easiest way to configure the app is to use the app to scan the QR code. If you cannot scan the code, you can type the configuration information manually. The QR code and secret configuration key generated by IAM are tied to your AWS account and cannot be used with a different account. They can, however, be reused to configure a new MFA device for your account in case you lose access to the original MFA device.
   - To use the QR code to configure the virtual MFA device, from the wizard, choose Show QR code. Then follow the app instructions for scanning the code. For example, you might need to choose the camera icon or choose a command like Scan account barcode, and then use the device's camera to scan the QR code.
   - In the Manage MFA Device wizard, choose Show secret key, and then type the secret key into your MFA app.

   The device starts generating six-digit numbers.
7) In the **Manage MFA Device** wizard, in the **MFA Code 1** box, enter the six-digit number that's currently displayed by the MFA device. Wait up to 30 seconds for the device to generate a new number, and then type the new six-digit number into the **MFA Code 2** box.
8) Choose Assign MFA, and then choose Finish.

Creating Billing Alerts

Controlling costs within AWS is important if you want to avoid any bill-shock at the end of the month.

Note: Billing Alerts must be enabled on your account before you can create a Billing Alarm.

1. Sign in to your AWS account.
   - Make sure the region is set to U.S. East (N. Virginia). This is the region where AWS stores billing information.
2. Click on your account name in the upper-right corner.

- In the dropdown menu, select **My Billing Dashboard** to open the Billing & Cost Management Dashboard.

3. In the left Dashboard menu, click on **Preferences**.
4. Check the box in front of **Receive Billing Alerts** and then click **Save Preferences**.

Create a Billing Alarm using the CloudWatch Console

1. Click on the AWS logo in the upper left of the Billing & Cost Management Dashboard screen to open the AWS Management Console.
2. Type "CloudWatch" in the AWS Management Console search box and then click on CloudWatch.
3. In the CloudWatch console menu on the left under **Alarms**, click on **Billing**.
4. In the Billing Alarms window, click the **Create Alarm** button.
5. Scroll to the bottom of the Create new alarm page and click on show advanced
6. Alarm details:
   - Add a Name for your new alarm and a Description (optional).
   - Under **Whenever charges for**: set **is**: to >= and **USD $** to the amount in dollars you want to set as a limit.

Note: You will receive an email notification when your account charges have reached or exceeded this amount.

7. Additional settings
   - Under **Treat missing data as**: choose **ignore (maintain the alarm state**)
8. Actions
   - Under **Email list:** enter **your email address**
   - Leave the other settings unchanged.
9. Click on the **Create Alarm** button at the bottom of the Create new alarm page.

Important: You will be sent an email to confirm the email address set in your alarm. Open the email and click on the "Confirm subscription" link to enable notifications.

What Can I do with CloudWatch?

- **Dashboards** – Create awesome dashboard to see what is happening with your AWS environment.
- **Alarms** – Allows you to set Alarms that notify you when particular thresholds are hit.
- **Events** – CloudWatch Events helps you to respond to state changes in your AWS resources.
- **Logs** – CloudWatch Logs helps you to aggregate, monitor, and store logs.

Creating the Production Account

Now, you need to follow the same steps to create a PRODUCTION AWS account.

Setting up the production AWS account:

- New email for the Production AWS account
- Same as the GENERAL account
- Add MFA to account Root user (and test)
- Add billing alert to PRODUCTION account (remember the checkboxes)

Identity and Access Management (IAM) Basics

Identity and Access Management (IAM) is a core AWS service you will use as a Solutions Architect. IAM is what allows additional identities to be created within an AWS account - identities which can be given restricted levels of access.

IAM identities start with no permissions on an AWS Account, but can be granted permissions (almost) up to those held by the Account Root User.

- **NO cost** associated with **IAM**
- **Global service**/ Global resilience
- **ALLOW** or **DENY** its identities on its AWS account
- **No direct control** on external accounts or users
- Identity federation and MFA

Adding an IAM Admin User - PART1 - GENERAL ACCOUNT

To create an administrator user for yourself and add the user to an administrators' group (console)

1. Sign in to the IAM console as the account owner by choosing Root user and entering your AWS account email address. On the next page, enter your password.
2. In the navigation pane, choose Users and then choose **Add user**.
3. For **User name**, enter **Administrator**.
4. Select the check box next to **AWS Management Console access**. Then select **Custom password**, and then enter your new password in the text box.
5. (Optional) By default, AWS requires the new user to create a new password when first signing in. You can clear the check box next to **User must create a new password at next sign-in** to allow the new user to reset their password after they sign in.
6. Choose **Next: Permissions.**
7. Under **Set permissions**, choose **Add user to group**.
8. Choose **Create group**.
9. In the **Create group** dialog box, for **Group name** enter **Administrators**.
10. Choose **Filter policies**, and then select **AWS managed - job function** to filter the table contents.
11. In the policy list, select the check box for **AdministratorAccess**. Then choose **Create group**.
12. Back in the list of groups, select the check box for your new group. Choose **Refresh** if necessary to see the group in the list.
13. Choose **Next: Tags.**
14. (Optional) Add metadata to the user by attaching tags as key-value pairs. For more information about using tags in IAM
15. Choose **Next: Review** to see the list of group memberships to be added to the new user. When you are ready to proceed, choose **Create user**.

You can use this same process to create more groups and users and to give your users access to your AWS account resources.

Adding an IAM Admin User - PART2 - PRODUCTION ACCOUNT

Follow same steps to create IAM user in PRODUCTION account.

IAM Access Keys

- Access keys are how the AWS Command Line Tools (CLI Tools) interact with AWS accounts.
- Temporary security credentials consist of the AWS access key ID, secret access key, and security token. Temporary security credentials are valid for a specified duration and for a specific set of permissions. Temporary security credentials are sometimes simply referred to as tokens. Tokens can be requested for IAM users or for federated users you manage in your own corporate directory.
- You can enable and disable an IAM user's access keys via the IAM APIs, AWS CLI, or IAM console. If you disable the access keys, the user cannot programmatically access AWS services.
- IAM roles for EC2 instances simplifies management and deployment of AWS access keys to EC2 instances. Using this feature, you associate an IAM role with an instance. Then your EC2 instance provides the temporary security credentials to applications running on the instance, and the applications can use these credentials to make requests securely to the AWS service resources defined in the role.

To create access keys for an IAM user

1. Sign in to the AWS Management Console and open the IAM console at https://console.AWS.amazon.com/IAM/
2. In the navigation pane, choose **Users**.
3. Choose the name of the user whose access keys you want to create, and then choose the **Security credentials** tab.
4. In the **Access keys** section, choose **Create access key**.
5. To view the new access key pair, choose **Show**. You will not have access to the secret access key again after this dialog box closes. Your credentials will look something like this:
    - Access key ID: AKIAIOSFODNN7EXAMPLE
    - Secret access key: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
6. To download the key pair, choose **Download .csv file**. Store the keys in a secure location. You will not have access to the secret access key again after this dialog box closes.

   Keep the keys confidential in order to protect your AWS account and never email them. Do not share them outside your organization, even if an inquiry appears to come from AWS or Amazon.com. No one who legitimately represents Amazon will ever ask you for your secret key.
7. After you download the .csv file, choose **Close**. When you create an access key, the key pair is active by default, and you can use the pair right away.

MODULE 3 – CLOUD COMPUTING FUNDAMENTALS

Cloud Computing - what is it...really

NIST definition of Cloud Computing

"Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model is composed of five essential characteristics, three service models, and four deployment models. "

Essential characterise

1) On-demand self-service: You can provision capabilities as needed without requiring human interaction.

2) Broad Network Access: Capabilities are available over the network and accessed through standard mechanisms.

3) Resource Pooling: There is a sense of location independence. no control or knowledge over the exact location of the resources. resources are pooled to serve multiple consumers using a multi-tenant model.

4) Rapid elasticity: Capabilities can be elastically provisioned and released to scale rapidly outward and inward with demand. To the consumer, the capabilities available for provisioning often appear to be unlimited.

5) Measured Service: Resource usage can be monitored, controlled reported and billed.

Public vs Private vs Multi vs Hybrid Cloud

cloud computing can be categorized into three general types:

- **Public cloud** is cloud computing that's delivered via the internet and shared across organizations.

  Advantages of public clouds:

  - **Lower costs**—no need to purchase hardware or software and you pay only for the service you use.
  - **No maintenance**—your service provider provides the maintenance.
  - **Near-unlimited scalability**—on-demand resources are available to meet your business needs.
  - **High reliability**—a vast network of servers ensures against failure.

- **Private cloud** is cloud computing that is dedicated solely to your organization.
  Advantages of a private cloud:

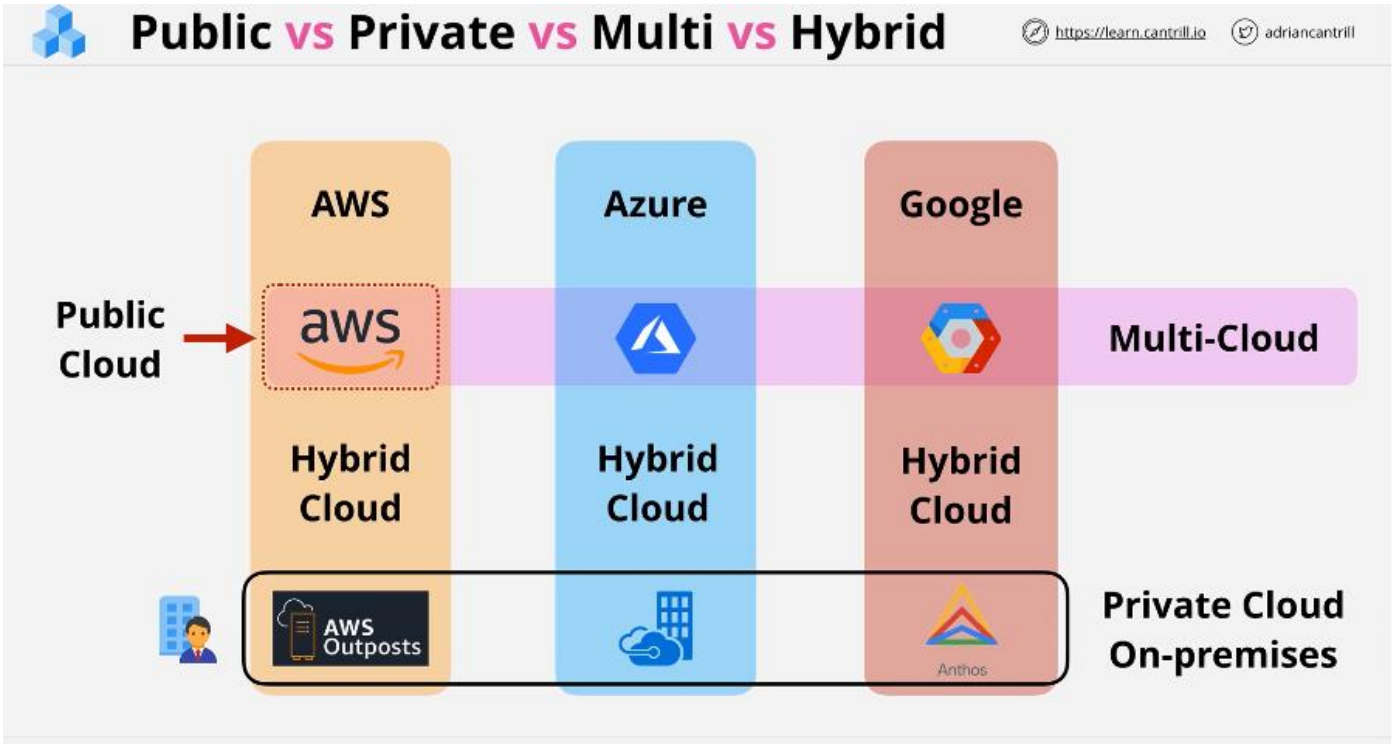
  - **More flexibility**—your organisation can customise its cloud environment to meet specific business needs.
  - **More control**—resources are not shared with others, so higher levels of control and privacy are possible.
  - **More scalability**—private clouds often offer more scalability compared to on-premises infrastructure.

- **Hybrid cloud** is any environment that uses both public and private clouds.
  Advantages of the hybrid cloud:

    - **Control**—your organisation can maintain a private infrastructure for sensitive assets or workloads that require low latency.
    - **Flexibility**—you can take advantage of additional resources in the public cloud when you need them.
    - **Cost-effectiveness**—with the ability to scale to the public cloud, you pay for extra computing power only when needed.
    - **Ease—transitioning** to the cloud does not have to be overwhelming because you can migrate gradually—phasing in workloads over time.
- **Multi-cloud** is a dynamic strategy of mixing and matching workloads across multiple public cloud vendors (e.g., AWS, Azure, Google), managed under one platform, to achieve long term business objectives.

Remember the following:

- Public Cloud = using 1 public cloud
- Private cloud = Using on premises real cloud
- Multiple cloud- using more than 1 public cloud
- hybrid cloud = public and private cloud
- hybrid cloud is not public cloud + legacy on-premises

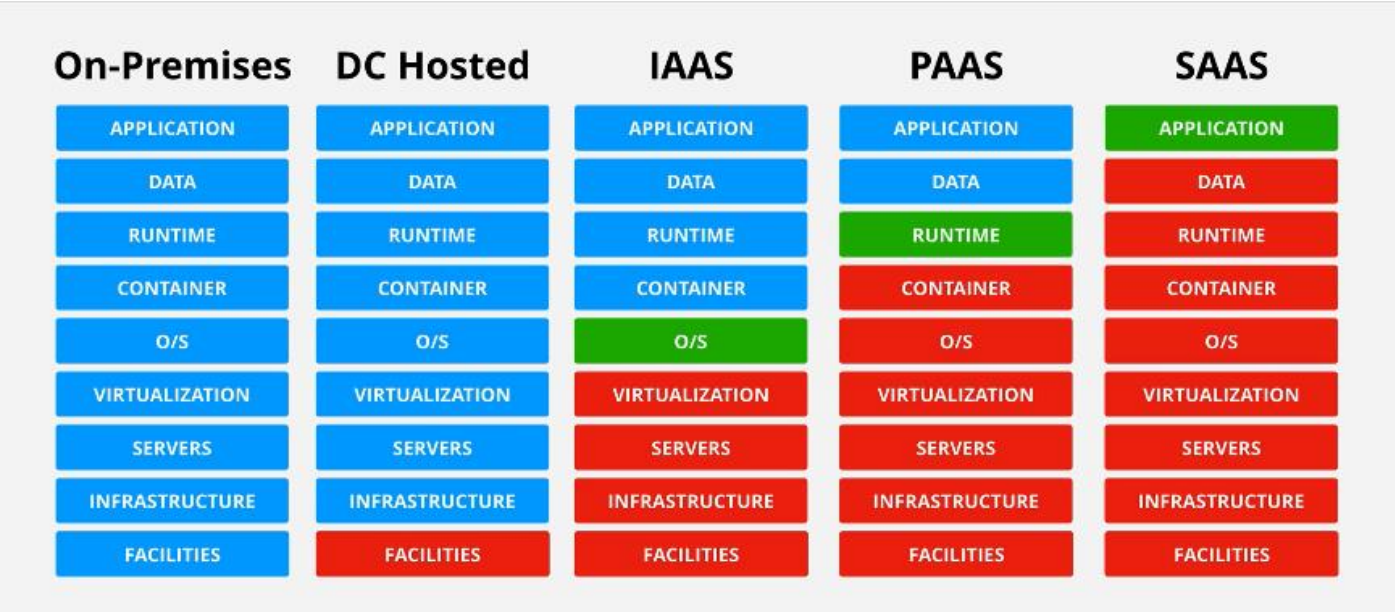


Cloud Service Models:

There are three main models for cloud computing. Each model represents a different part of the cloud computing stack.

On-premises: Deploying resources on-premises, using virtualization and resource management tools, is sometimes called “private cloud”. On-premises deployment does not provide many of the benefits of cloud computing but is sometimes sought for its ability to provide dedicated resources. In most cases this deployment model is the same as legacy IT infrastructure while using application management and virtualization technologies to try and increase resource utilization.

Infrastructure as a Service (IaaS): Infrastructure as a Service, sometimes abbreviated as IaaS, contains the basic building blocks for cloud IT and typically provide access to networking features, computers (virtual or on dedicated hardware), and data storage space. Infrastructure as a Service provides you with the highest level of flexibility and management control over your IT resources and is most similar to existing IT resources that many IT departments and developers are familiar with today.



Platform as a Service (PaaS)

Platforms as a service remove the need for organizations to manage the underlying infrastructure (usually hardware and operating systems) and allow you to focus on the deployment and management of your applications. This helps you be more efficient as you don’t need to worry about resource procurement, capacity planning, software maintenance, patching, or any of the other undifferentiated heavy lifting involved in running your application.

Software as a Service (SaaS)

Software as a Service provides you with a completed product that is run and managed by the service provider. In most cases, people referring to Software as a Service are referring to end-user applications. With a SaaS offering you do not have to think about how the service is maintained or how the underlying infrastructure is managed; you only need to think about how you will use that particular piece of software. A common example of a SaaS application is web-based email where you can send and receive email without having to manage feature additions to the email product or maintaining the servers and operating systems that the email program is running on.

MODULE 4 - AWS FUNDAMENTALS

AWS Public vs Private Services

AWS services can be categorized into two main types, public services, and private services.

When discussing the terms private and public AWS services, these terms relate to the networking only. Whether an AWS service is private or public, from a permission's perspective, aside from the account root user, there is no access by default.

The difference between public and private services, is the connectivity. it's a networking difference. and the difference is which network zone the service is in. So, let's look at an example visually.

Imagine that you are sat in your home office and supposed to be doing some work. You're using your laptop, and this laptop has a connection to the internet. this is the first network zone, the public internet.

Suppose instead of working, you decide to have a small break which turns into a wasted afternoon, because you are catching up on the latest TV show on Netflix.

Now, we have all done this. but when do that, you are accessing on internet-based service. it's a service that's hosted on the public internet, or in the public internet zone. because AWS is a public cloud platform, it's connected to this public internet, but it's adjacent to it.

An AWS public service is one which can be connected to from anywhere where you have an unrestricted internet connection

Now there is a network zone inside AWS, which I refer to as the AWS public zone, and this is directly attached to public internet. For an AWS service to be classified as a public service, it runs from this zone,

An example of this, if you want to store some cat pictures and make them accessible to a large number of people, then one service that you could use, is an S3 buckets. And S3 bucket uses the S3 service which is hosted within the AWS public zone. So, when you're accessing S3, you connect to it generally via the public internet. and data moves from your laptop through the public internet, to the AWS public zone and then back again.

that's what it means to be a public service. And to reiterate, just because you can connect to this service doesn't mean you have the permission to access it.

Now, there is another zone inside AWS, the AWS private zone. This zone has no direct connectivity to the public internet zone or the AWS public zone. by default, no connection is allowed between private zone and anywhere else.

Anything created in this zone is isolated by default. And it's even more flexible, because you can divide this private zone up into individual isolated private networks using a service called Virtual Private Cloud or VPC.

you can create services inside this private zone, such as a virtual server known as an EC2 instance, inside AWS. and by default, this has no way of connecting to the internet, no way of connecting to AWS public services and neither the public internet nor public AWS zones can connect to this EC2 instance that's running in this private zone.

by default, its isolated. you can configure private services so they have public access. An EC2 instance in this example, which is a private service, can be allows to connect out.

To summarize, because AWS is a public cloud, it can be connected to over the public internet.

AWS services are not on the public internet.

AWS has a zone called the AWS public zone, which is where public services run from. this zone is connected to the public internet and it can be accessed from the public internet.

the private zone, which can be itself be subdivided using VPC or Virtual private clouds, is isolated by default from the AWS public zone and from the public internet but private services running from this private zone can be allowed outgoing access, and they can also be configured in some cases to be public.

AWS Global Infrastructure

AWS have created a global public cloud platform which consists of isolated 'regions' connected together by high-speed global networking.

This lesson reviews the main architectural components of AWS, Regions, Edge Locations and Availability Zones.

It also discusses what it means to be Globally Resilient, Regional Resilient and AZ resilient.

Lesson Links

AWS Website used in this lesson https://www.infrastructure.AWS/

The following are the components that make up the AWS infrastructure:

- Availability Zones
- Region
- Edge locations
- Regional Edge Caches

Availability zone as a Data Center

- An availability zone is a facility that can be somewhere in a country or in a city. Inside this facility, i.e., Data Centre, we can have multiple servers, switches, load balancing, firewalls. The things which interact with the cloud sits inside the data centers.
- An availability zone can be a several data centers, but if they are close together, they are counted as 1 availability zone.

Region

- A region is a geographical area. Each region consists of 2 more availability zones.
- A region is a collection of data centers which are completely isolated from other regions.
- A region consists of more than two availability zones connected to each other through links.
- Availability zones are connected through redundant and isolated metro fibres.

Edge Locations

- Edge locations are the endpoints for AWS used for caching content.
- Edge locations consist of CloudFront, Amazon's Content Delivery Network (CDN).
- Edge locations are more than regions. Currently, there are over 150 edge locations.
- Edge location is not a region but a small location that AWS have. It is used for caching the content.
- Edge locations are mainly located in most of the major cities to distribute the content to end users with reduced latency.
- For example, some user accesses your website from Singapore; then this request would be redirected to the edge location closest to Singapore where cached data can be read.

Regional Edge Cache

- AWS announced a new type of edge location in November 2016, known as a Regional Edge Cache.
- Regional Edge cache lies between CloudFront Origin servers and the edge locations.
- A regional edge cache has a large cache than an individual edge location.
- Data is removed from the cache at the edge location while the data is retained at the Regional Edge Caches.
- When the user requests the data, then data is no longer available at the edge location. Therefore, the edge location retrieves the cached data from the regional edge cache instead of the Origin servers that have high latency.

AWS Default Virtual Private Cloud (VPC)

VPC is the service you will use to create private networks inside AWS that other private services will run from. VPCs are also the service which is used to connect your AWS private network to your on-premises networks when creating a hybrid environment, or it's the service which lets you connect to other cloud platforms when you're creating a multi-cloud deployment.

Key points:

- A VPC or Virtual Private Cloud is a virtual network inside of AWS. when you create a VPC you are doing so within an AWS accounts, and specifically, within a region of that AWS account.

- VPCs are regional services, meaning they're regionally resilient. VPCs operate from multiple availability zones in a specific AWS region.

- VPC by default is private and isolated. Services deployed into the same VPC can communicate, but the VPC is isolate from other VPCs and from the public AWS zone and public internet. that's of course unless you configure otherwise.

- There are two types of VPC available inside a region,

- Default VPC
- Custom VPC

- You only have one default VPC per region. But you can have many custom VPCs in a region. as a name suggest are custom, as you can configure them in any way that you want. Custom VPCs require that you configure everything end-to-end in detail, and thy are also a 100% private by default.

-Every VPC is allocated a range of Ip addresses called the VPC CIDR. and the VPC CIDR defines the start and end range of IP addresses that VPC can use. Everything inside a VPC uses the CIDR range of that VPC. if anything needs to communicate with a VPC and assuming you allow it, it needs to communicate to that VPC CIDR.

- Custom VPCs can have multiple CIDR ranges, but the default VPC only gets one, and it always the same. it’s 172.31.0.0/16.

- The way in which VPC provides resilience is that it can be subdivided into subnets, which is short for subnetworks. Each subnet inside of VPC is located in one availability zone.

This is set on creation and can never be changed.

-Each of these subnets use part of the VPCs range of IP addresses, it's CIDR range. these cannot be the same as other subnets in the VPC, and they can't overlap with any of the subnets inside the VPC.

- Each subnet is inside one availability zone. so, if that availability zone fails, then the subnet in that availability zone also fails. But any of the subnets inside the VPC that are located in other availability zones, they will continue to operate normally as long as the other availability zones are unaffected.

Elastic Compute Cloud (EC2) Basics

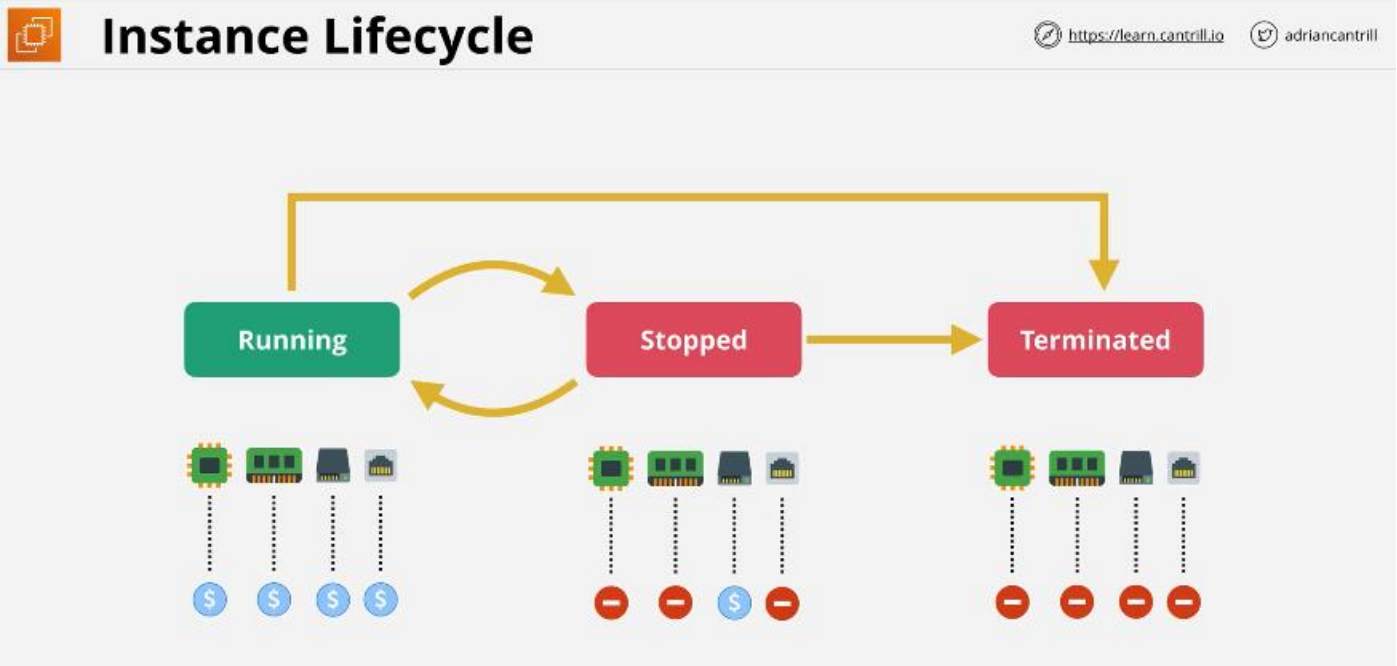
The Elastic Compute Cloud or EC2 is AWS's implement of IAAS - Infrastructure as a service.

It allows you to provision virtual machines known as instances with resources you select and an operating system of your choosing.

Now let's quickly review some important features that EC2 provides before we talk about some important architectural points.

- EC2 is IAAS. It provides access to virtual machines known as EC2 instances. Instance is just an operating system configures in a certain way with a certain set of allocated resources.
- EC2 is a private AWS service, which as you now know means that by default, it runs in the private AWS zone.
- Specifically, an EC2 instance is configured to launch into a single VPC subnet. You set it when you launch the instance. and when you launch the instance, you also have to configure any public access should you want that because it is by default a private service. if you do want to allow public access for an EC2 instance, then the VPC that it's running within needs to support that public access. because the instance is launched into a specific subnet and because a subnet is in a specific availability zone, it means that EC2 is AZ resilient.
- If the AZ that an instance is launched into fails, then instance itself will likely fail. Remember this right from the start, because it will help you as you move through the course.
- EC2 is IAAS you as a consumer manager the OS and upwards on the infrastructure stack, meaning the AWS handle the virtualization, physical hardware, networking, storage and facilities.
- EC2 offers on-demand billing. and this is either by the second or by the hours, depending on the software that is launched within the instance.
- You only pay for what you consume. There are a few components to an instance charge, there is the charge for running the instance.
- So, an amount for CPU and memory that the instance consumes. there’s a charge for the storage that the instance uses, and then extras for any commercial software that the instance is launched with. instance can use a number of different types of storage, and two popular types of storage that's on the local host.
- So, the EC2 host that the instance runs on, as well as another AWS service called Elastic Block Store, or EBS, which is network storage made available to the instance.
- Now, an EC2 instance has an attribute called a state. The state of the instance provides an indication into its condition. just like a computer can be powered on or powered off, an instance can be in one of a few states.
- The most important states to remember at this point are running, stopped and terminated. There are some states in between those such as stopping, or shutting down and pending, but at this stage, just focus on these three.
- it's also important to understand the relationship between these states. when you launched an instance, after it finishes provisioning, it moves into a running state. and instance can be moved from running to stopped. if you shut down the instance, or they can be moved from stopped to running when you start the instance up again.
- If you terminate an instance, either when it's running or stopped, that's it, it's terminated. This is a non-reversible action. the instance is fully deleted.

Now the reason why these states matter so much is they influence the charges for the instance.



- At high level an instance is composed of CPU, memory, disk and networking. The CPU determines how much processing can be achieved. The memory is a super-fast area to store data that's currently been worked on the instance. The disc generally provided by EBS is where more medium-term data is stored. The networking is how the instance communicates with other entries in AWS and beyond.
- Now when instance is in the running state, you're being charges for all four of these categories. When an instance is stopped, though, things are slightly different. being in the stopped state means that no CPU resources are being consumed and likewise, no memory is being used by the instance. when the instance is stopped, you're nor being charged any costs for the running of that instance, because it's not consuming resources. The instance also doesn't generate any network data because it's not running.
- Now the key point that tends to confuse most people, especially in exam question is that storage is still allocated to the instance regardless of whether it's in a running or stopped state. The storage made available to the instance. So that's really important to understand the stopped instance still generate storage charges.
- The only way to truly have no EC2 costs for an instance is to terminate that EC2 instance. and when you do that, not only does it stop the resource usage, so CPU, memory and networking but it also delates the disks. but you need to pay special care and attention because it is not reversible.

Connecting to EC2 instance

Prerequisites

Before you connect to your Linux instance, complete the following prerequisites.

Check your instance status: After you launch an instance, it can take a few minutes for the instance to be ready so that you can connect to it. Check that your instance has passed its status checks. You can view this information in the Status check column on the Instances page.

Get the public DNS name and user name to connect to your instance: To find the public DNS name or IP address of your instance and the user's name that you should use to connect to your instance.

Install an SSH client on your local computer as needed

To connect to your instance using SSH

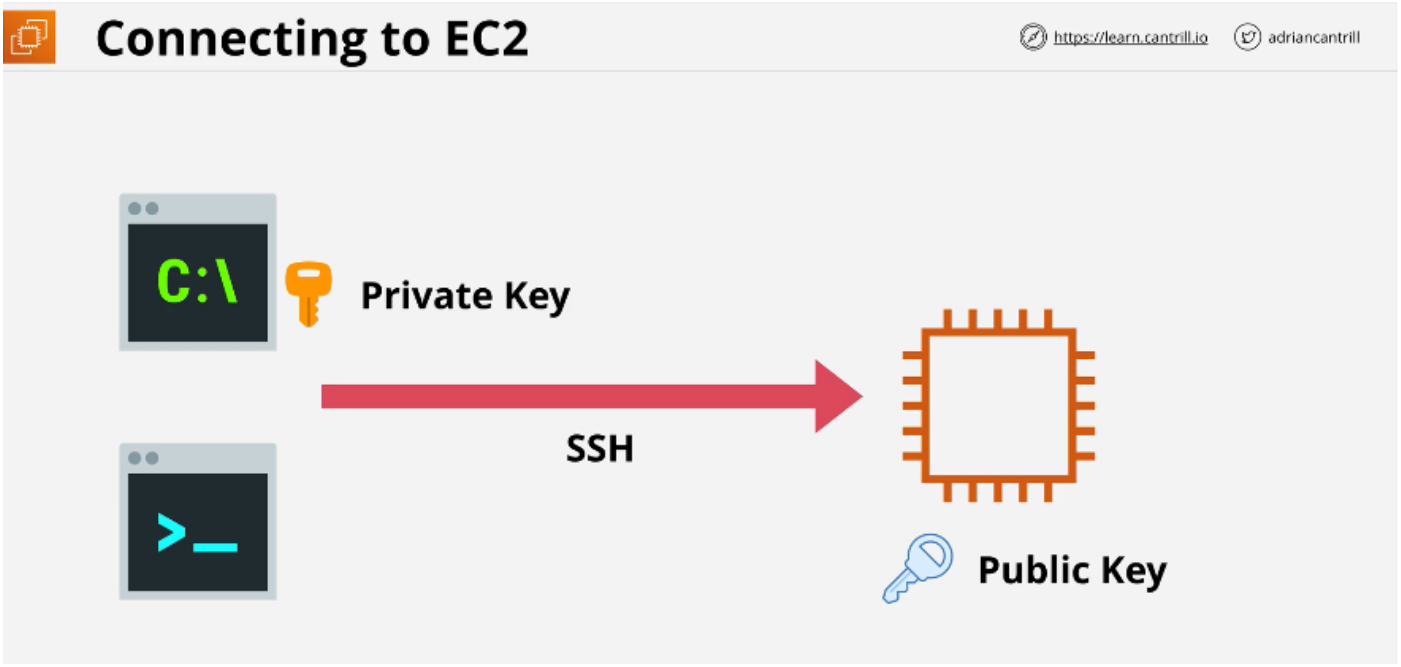
1. In a terminal window, use the **ssh** command to connect to the instance. You specify the path and file name of the private key (.pem), the user name for your instance, and the public DNS name or IPv6 address for your instance. For more information about how to find the private key, the user name for your instance, and the DNS name or IPv6 address for an instance, see Locate the private key and Get information about your instance. To connect to your instance, use one of the following commands.
    - (Public DNS) To connect using your instance's public DNS name, enter the following command.

    ssh -i /path/my-key-pair.pem my-instance-user-name@my-instance-public-dns-name

2. (Optional) Verify that the fingerprint in the security alert matches the fingerprint that you previously obtained in (Optional) Get the instance fingerprint. If these fingerprints don't match, someone might be attempting a "man-in-the-middle" attack. If they match, continue to the next step.
3. Enter **yes**.

   You see a response like the following:

```
Warning: Permanently added 'ec2-198-51-100-1.compute-1.amazonAWS.com' (ECDSA) to the
list of known hosts.
```



AMI

- An AMI, or Amazon Machine image, as the name suggested is an image of an EC2 instance. now an AMI can be used to create an EC2 instance or an AMI can be created from an EC2 instance.

AMI contains a few important things which you should be aware of.

- Firstly, an AMI contains attached permissions. and these permissions control which accounts can and can't use the AMI.

- AMI can be set as a public AMI; in which case everyone is allowed to launch instances from that AMI.
- The second type is the owner of an AMI is implicitly allowed to create EC2 instances from the AMI. as the owner, he or she owns it.
- And finally, you can add explicit permissions to that AMI where the owner explicitly grants access to the AMI is either private, so only the owner can make use of it, or you can explicitly add other AWS account, so they're allowed access, or it can be set to public where everyone is allowed.

As well as permissions, an AMI handles two other important things

- It contains the boot volume of the instance. So, this is the C-drive in winnows and the root volume in Linux.

- It also provides what's known as a block device mapping. and this is a configuration which links the volumes that the AMI has and how they're presented to the OS.

so, it determines which volume is the boot volume and which is a data volume,

EC2 instances can run different OS. they can run a distribution or a version of Linux, as well as various different versions of windows. You can connect to Windows instances using RDP, the remote desktop protocol. And this runs in port 3389. With Linux instances you use the SSH protocol which runs on port 22.

My First EC2 Instance [Practical]

To create your key pair

1. Open the Amazon EC2 console at https://console.AWS.amazon.com/ec2/.
2. In the navigation pane, choose **Key Pairs**.
3. Choose **Create key pair**.
4. For **Name**, enter a descriptive name for the key pair. Amazon EC2 associates the public key with the name that you specify as the key name. A key name can include up to 255 ASCII characters. It can't include leading or trailing spaces.
5. For **File format**, choose the format in which to save the private key. To save the private key in a format that can be used with OpenSSH, choose **pem**. To save the private key in a format that can be used with PuTTY, choose **ppk**.
6. Choose **Create key pair**.
7. The private key file is automatically downloaded by your browser. The base file name is the name you specified as the name of your key pair, and the file name extension is determined by the file format you chose. Save the private key file in a safe place.

Important: This is the only chance for you to save the private key file.

To create a security group with least privilege

1. Open the Amazon EC2 console at https://console.AWS.amazon.com/ec2/.
2. From the navigation bar, select a Region for the security group. Security groups are specific to a Region, so you should select the same Region in which you created your key pair.
3. In the navigation pane, choose **Security Groups**.
4. Choose **Create security group**.
5. For **Basic details**, do the following:

a. Enter a name for the new security group and a description. Use a name that is easy for you to remember, such as your username, followed by _SG_, plus the Region name. For example, *me*_SG_*uswest2*.
b. In the **VPC** list, select your default VPC for the Region.

6. For **Inbound rules**, create rules that allow specific traffic to reach your instance. For example, use the following rules for a web server that accepts HTTP and HTTPS traffic. For more examples, see Security group rules for different use cases.
    a. Choose **Add rule**. For **Type**, choose **HTTP**. For **Source**, choose **Anywhere**.
    b. Choose **Add rule**. For **Type**, choose **HTTPS**. For **Source**, choose **Anywhere**.
    c. Choose **Add rule**. For **Type**, choose **RDP**. For **Source**, do one of the following:
        - Choose **My IP** to automatically add the public IPv4 address of your local computer.
        - Choose **Custom** and specify the public IPv4 address of your computer or network in CIDR notation. To specify an individual IP address in CIDR notation, add the routing suffix /32, for example, 203.0.113.25/32. If your company or your router allocates addresses from a range, specify the entire range, such as 203.0.113.0/24.

    Warning: For security reasons, do not choose Anywhere for Source with a rule for RDP. This would allow access to your instance from all IP addresses on the internet. This is acceptable for a short time in a test environment, but it is unsafe for production environments.

7. For **Outbound rules**, keep the default rule, which allows all outbound traffic.
8. Choose **Create security group**.

To launch an instance

1. Open the Amazon EC2 console at https://console.AWS.amazon.com/ec2/.
2. From the console dashboard, choose **Launch Instance**.
3. The **Choose an Amazon Machine Image (AMI)** page displays a list of basic configurations, called *Amazon Machine Images (AMIs)*, that serve as templates for your instance. Select the AMI for Windows Server 2016 Base or later. Notice that these AMIs are marked "Free tier eligible."
4. On the **Choose an Instance Type** page, you can select the hardware configuration of your instance. Select the t2.micro instance type, which is selected by default. The t2. micro instance type is eligible for the free tier. In Regions where t2. micro is unavailable, you can use a t3. micro instance under the free tier. For more information, see AWS Free Tier.
5. On the **Choose an Instance Type** page, choose **Review and Launch** to let the wizard complete the other configuration settings for you.
6. On the **Review Instance Launch** page, under **Security Groups**, you'll see that the wizard created and selected a security group for you. You can use this security group, or alternatively you can select the security group that you created when getting set up using the following steps:
    a. Choose **Edit security groups**.
    b. On the **Configure Security Group** page, ensure that **Select an existing security group** is selected.
    c. Select your security group from the list of existing security groups, and then choose **Review and Launch**.

7. On the **Review Instance Launch** page, choose **Launch**.
8. When prompted for a key pair, select **Choose an existing key pair**, then select the key pair that you created when getting set up.

   Warning: Don't select **Proceed without a key pair**. If you launch your instance without a key pair, then you can't connect to it.

   When you are ready, select the acknowledgement check box, and then choose **Launch Instances**.
9. A confirmation page lets you know that your instance is launching. Choose **View Instances** to close the confirmation page and return to the console.
10. On the **Instances** screen, you can view the status of the launch. It takes a short time for an instance to launch. When you launch an instance, its initial state is pending. After the instance starts, its state changes to running and it receives a public DNS name. (If the **Public IPv4 DNS** column is hidden, choose the settings icon ( ) in the top-right corner, toggle on **Public IPv4 DNS**, and choose **Confirm**.
11. It can take a few minutes for the instance to be ready so that you can connect to it. Check that your instance has passed its status checks; you can view this information in the **Status check** column.

Simple Storage Service (S3) Basics

- S3 is a Global Storage Platform. It's global, because it runs from all of the AWS regions and can be accessed from anywhere with an internet connection.
- It's a public service. it's regional based because your data is stored in a specific region. And it never leaves that region unless you explicitly configure it to.
- S3 is regionally resilient, meaning the data is replicated across availability zones in that region.
- S3 can tolerate the failure of an AZ and it also has some ability to replicate data between regions.
- S3 might initially appear confusing. if you utilize it from the UI, you appear not to have to select a region. Instead, you select the region when you create things inside S3.
- S3 is a public service so it can be accessed from anywhere as login as you have an internet connection.
- The service itself runs from the AWS public zone. It can cope with unlimited data amounts, and it’s designed for multi user usage of that data.
- S3 is perfect for hosting large amount of data. so, think movies, audio distribution, large scale photo storage like stock images, large textual data or big datasets.
- S3 is economical & accessed via CLI/UI/API/HTTP
- S3 has two main things to delivers, **objects** and **buckets**.
- Objects are the data that S3 stores, your cat pictures, the latest episode of Game of Thrones, which you have stored legally. you can think about objects like files, conceptually, most of the time they're interchangeable.
- Buckets are containers for objects.
- An object in S3 is made up of two main components and some associated metadata.
  First, there is the object key. and for now, you can think of the object key as similar to a file name. The key identifies the object in a bucket. So, if you can uniquely access the object assuming that you have permissions, remember, by default, even for public services, there is no access in AWS initially, except for the account root user,
  Now the other main component of an object is its value. and the value s the data or the contents of the objects. the value of an object, in essence, how large the object is, can range from zero bytes up to five terabytes in size. So, you can have an empty object or you can have one that is huge 5TB. object also have version ID, metadata, some access control, as well as sub resource.

My First S3 Bucket

In this [DEMO] Lesson I step through the process of creating a simple S3 bucket and uploading objects. I demonstrate the block public access settings, talk about the bucket ARN and go into some detail about permissions on objects and how folders are really objects :)

First, you need to create an Amazon S3 bucket where you will store your objects.

1. Sign in to the preview version of the AWS Management Console.
2. Under **Storage & Content Delivery**, choose **S3** to open the Amazon S3 console.

   If you are using the **Show All Services** view, your screen looks like this:
3. From the Amazon S3 console dashboard, choose **Create Bucket**.
4. In **Create a Bucket**, type a bucket name in **Bucket Name**.

   The bucket name you choose must be globally unique across all existing bucket names in Amazon S3 (that is, across all AWS customers).
5. In **Region**, choose **Oregon**.
6. Choose **Create**.

   When Amazon S3 successfully creates your bucket, the console displays your empty bucket in the **Buckets** pane.

CloudFormation (CFN) Basics

CloudFormation is an Infrastructure as Code (IaC) product in AWS which allows automation infrastructure creation, update and deletion. Templates created in YAML or JSON can be used to automate infrastructure operations Templates are used to create stacks, which are used to interact with resources in an AWS account.

- CloudFormation is a tool which lets you create update and delete infrastructure in AWS in a consistent and repeatable way using templates so rather than creating and updating resources manually you can create templates and cloud formation we'll do the rest on your behalf.
- CloudFormation uses templates you can use a template to create AWS infrastructure. you can also update a template and reapply it which causes cloud formation to update infrastructure and eventually you can use cloud formation to delete that very some infrastructure.
- CloudFormation template is written either in **YAML** or **JSON**.

  I'll focus on YAML in this course.

- All templates have a list of resources at least one. it's the resources section of a CloudFormation template that tells CloudFormation what to do. if resources are added to it, then cloud formation creates resources. if resources are updated then it updates those resources if resources are removed from a template and that template this reapplied then physical resources are removed.
- The **Resources section** of a template is the only mandatory part of a CloudFormation template, which makes sense without resources the template wouldn't actually do anything.
- **Description section** that is free text field which let the author of the template add as the name suggests a description. generally, you would use this give some details about what the temperature does what resources get changed the cost of per template.
- **Metadata section** in the template is the next part that I want to talk about it got many functions including some pretty advance one but one of the things that it does it can control how the different thing in CloudFormation templates are presented through the console UI so through the AWS console if you are applying it so you can specify grouping, you can control the order, you can add descriptions and label, it's a way that you can force how the UI presents the template. generally, the bigger your template the wider audience and more likely it's going to have a meta data section.
- **Parameters section** of a template is when you can add fields which prompt the user for more information. if you apply the template from the console UI you will see boxes that you need to type in or select from dropdown. now things that you might use this for, which size of instance to create, the name of something, the number of availability zones to use. Parameters can even have settings for which are valid entries. so, you can apply criteria for values that can be added as parameters and you can also apply default values.
- The next section is **Mapping** and this is another optional section of the CloudFormation templates and something that we won't use as much especially when you just get started with cloud formation it allows you to create lookup table.
- **Conditions section** allows decision making in the template so you can set certain things in a template that will only occur if the condition is met now using conditions is a two-step process step one is to create the condition.
- **Output section** can present output based on what's being created updated or deleted.

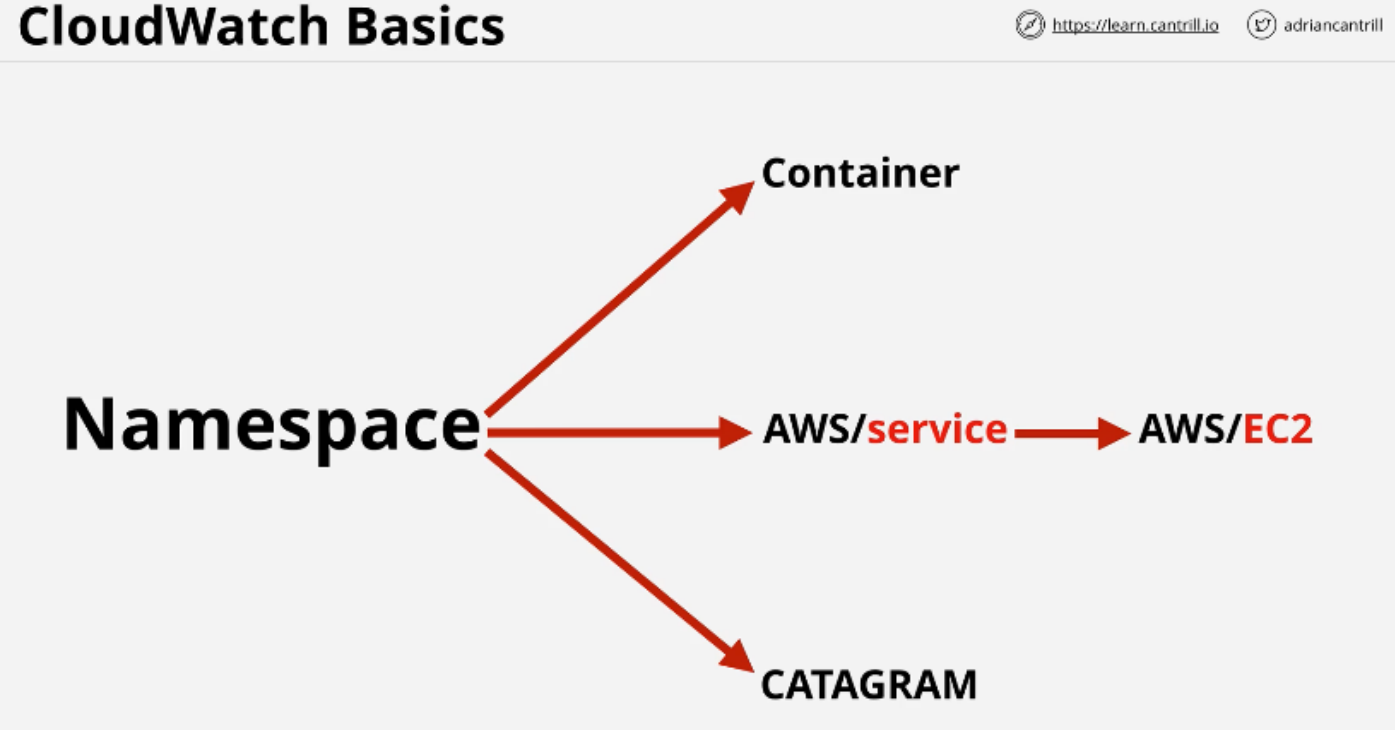
CloudWatch (CW) Basics

- CloudWatch is a core product inside AWS. it's a support service which is used by almost all other AWS services, especially for operational management and monitoring.
- CloudWatch performs three main jobs and it's important that you understand all three.
- At a high level, CloudWatch is a product which collects and manages operational data on your behalf. operational data is any data generated by an environment, either detailing how it performs, how it nominally runs, or any logging data that it generates.
- CloudWatch allows the collection of metrics, monitoring of metrics and action based on metrics.
- **Metrics** are simple data relating to AWS products, application, or on-premise systems. Some common examples are CPU utilization of ec2 instances, disk space usage on an on-premises server or may be the number of visitors per second to your website.

- CloudWatch is a public service and it can be sued for metric gathering, either inside AWS, within on-premises environments, or even within other cloud platforms.

  Some types of metrics collection need an extra piece of software called **CloudWatch agent**.

  So, example might be to collect metrics outside of AWS in other cloud environments or within on-premise environments, you ned to install CloudWatch agent. also monitoring certain things inside product which aren't exposed to AWS needs the agent. For example, if you want to monitor which processes are running on an ec2 instance or the memory utilization of these processes, you will need to install CloudWatch agent.



- Now, the second part of a CloudWatch is called **CloudWatch Log** and this allow for the collection monitoring and actions based on logging data now this might be Windows event logs, web server logs, Firewall logs, Linux server log almost anything which is log can be ingested by CloudWatch logs.
- lastly, we have got **CloudWatch Event** and this function as an event hub. CloudWatch event provides two powerful features. firstly, if AWS service does something maybe an ec2 instances is terminated started or stopped then CloudWatch events will generate an event which can perform another action. and the second type of thing that CloudWatch event can do is to generate an event to do something at a certain time of a day or Saturday of a week.
- Now, because CloudWatch manages lots of different services it needs a way of keeping things separated so the first concept I want to talk about is a **Namespace** and you can think of a namespace as a container for monitoring data its way to keep things from becoming Messy its way to separate things in two different areas now namespace have a name and this can be almost anything as long as it stays within the rules set for namespace is names.
- Namespaces contain related metrics a metric is a collection of related **data points** in a time ordered structure so to give you a few examples we might have CPU utilisation network in and out or disk utilisation they are all Matrix if you imagine a set of service logging data for CPU utilisation it will be time ordered it will start when you enable monitoring and it will finish when you disable it a metric the and this is a fairly nouns point to understand a metric is not for a specific server CPU utilisation is the metric and that metric might be receiving data for lots of different ec2 instances.
- I want to talk about data points. let's say we have got a metric called CPU utilisation every time any server measured is utilisation and send it into a CloudWatch that goes into a CPU utilisation metric and each one of those measurements so every time the server reports it CPU that measurement it’s

called **data point**. Now a data point and its simplest form consists of two things first a timestamp which include the year month day are minute second and time zone when the measurement was conducted and secondly a value in this case 98.3 which represent 98.3 CPU utilisation now, I mentioned earlier that the CPU utilisation metric could contain data for many servers.

- **Dimensions** separate datapoints for different things or perspectives within the same metric.
- CloudWatch also allows us to take actions based on Matrix and this is done using alarm as a concept a pretty simple alarms are created and their link to a specific Matrix then based on how you can figure out the alarm it will take an action based on that metric.

[Demo] Simple Monitoring with CloudWatch

To create an alarm based on CPU usage:

1. Open the CloudWatch console at https://console.AWS.amazon.com/cloudwatch/.
2. In the navigation pane, choose **Alarms**, **Create Alarm**.
3. Choose **Select metric**.
4. In the **All metrics** tab, choose **EC2 metrics**.
5. Choose a metric category (for example, **Per-Instance Metrics**).
6. Find the row with the instance that you want listed in the **InstanceId** column and **CPUUtilization** in the **Metric Name** column. Select the check box next to this row, and choose **Select metric**.
7. Under **Specify metric and conditions**, for **Statistic** choose **Average**, choose one of the predefined percentiles, or specify a custom percentile (for example, **p95.45**).
8. Choose a period (for example, **5 minutes**).
9. Under **Conditions**, specify the following:
    a. For **Threshold type**, choose **Static**.
    b. For **Whenever CPUUtilization is**, specify **Greater**. Under **than...**, specify the threshold that is to trigger the alarm to go to ALARM state if the CPU utilization exceeds this percentage. For example, 70.
    c. Choose **Additional configuration**. For **Datapoints to alarm**, specify how many evaluation periods (data points) must be in the ALARM state to trigger the alarm. If the two values here match, you create an alarm that goes to ALARM state if that many consecutive periods are breaching.

       To create an M out of N alarm, specify a lower number for the first value than you specify for the second value. For more information, see Evaluating an alarm.
    d. For **Missing data treatment**, choose how to have the alarm behave when some data points are missing. For more information, see Configuring how CloudWatch alarms treat missing data.
    e. If the alarm uses a percentile as the monitored statistic, a **Percentiles with low samples** box appears. Use it to choose whether to evaluate or ignore cases with low sample rates. If you choose **ignore (maintain alarm state)**, the current alarm state is always maintained when the sample size is too low. For more information, see Percentile-based CloudWatch alarms and low data samples.
10. Choose **Next**.

11. Under **Notification**, choose **In alarm** and select an SNS topic to notify when the alarm is in ALARM state

    To have the alarm send multiple notifications for the same alarm state or for different alarm states, choose **Add notification**.

    To have the alarm not send notifications, choose **Remove**.
12. When finished, choose **Next**.
13. Enter a name and description for the alarm. The name must contain only ASCII characters. Then choose **Next**.
14. Under **Preview and create**, confirm that the information and conditions are what you want, then choose **Create alarm**.

Shared Responsibility Model

AWS responsibility "Security of the Cloud" - AWS is responsible for protecting the infrastructure that runs all of the services offered in the AWS Cloud. This infrastructure is composed of the hardware, software, networking, and facilities that run AWS Cloud services.

Customer responsibility "Security in the Cloud" – Customer responsibility will be determined by the AWS Cloud services that a customer selects. This determines the amount of configuration work the customer must perform as part of their security responsibilities.

Below are examples of controls that are managed by AWS, AWS Customers and/or both.

Inherited Controls – Controls which a customer fully inherits from AWS.

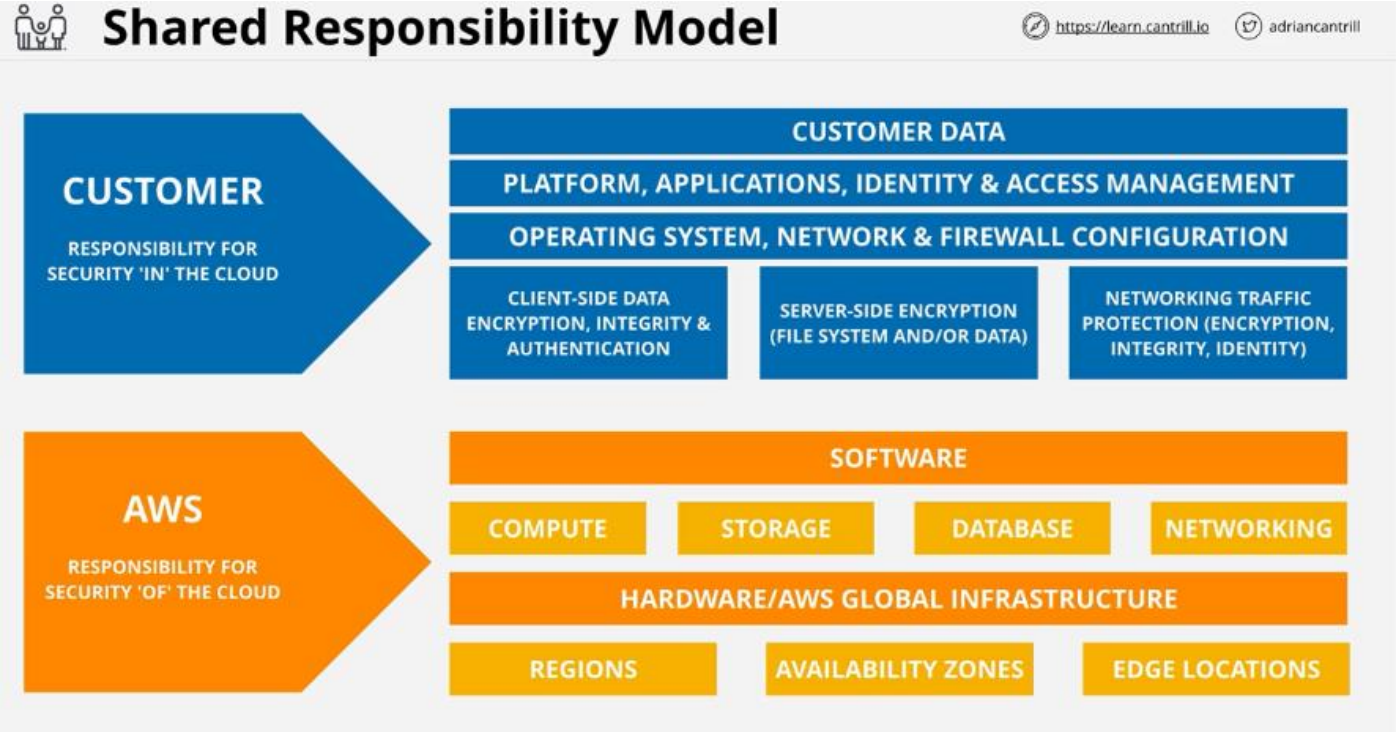
- Physical and Environmental controls

Shared Controls – Controls which apply to both the infrastructure layer and customer layers, but in completely separate contexts or perspectives. In a shared control, AWS provides the requirements for the infrastructure and the customer must provide their own control implementation within their use of AWS services. Examples include:

- **Patch Management** – AWS is responsible for patching and fixing flAWS within the infrastructure, but customers are responsible for patching their guest OS and applications.
- **Configuration Management** – AWS maintains the configuration of its infrastructure devices, but a customer is responsible for configuring their own guest operating systems, databases, and applications.
- **Awareness & Training** - AWS trains AWS employees, but a customer must train their own employees.

Customer Specific – Controls which are solely the responsibility of the customer based on the application they are deploying within AWS services. Examples include:

- Service and Communications Protection or Zone Security which may require a customer to route or zone data within specific security environments.



High-Availability vs Fault-Tolerance vs Disaster Recovery

High-Availability (HA):

- Formally, the definition is that highly availability aims to **ensure an agreed** level of **operational performance**, usually **uptime**, for a **higher-than-normal period**.
- Most of the student have an assumption that making s system highly available means ensuring that the system never fails or that the user of the system never experience any outages, and that is not true.
- HA isn't aiming to **stop failure** and it definitely doesn't mean that customers won't experience outage
- A highly available system is one designed to be online and providing services as often as possible. it's a system designed so that when it fails its components can be replaced or fixed as quickly as possible, often using automation to bring systems back into service.
- **high availability** is about maximizing a system's online time and that's it.
- highly availability is about keeping a **system operational**. it's about **fast** or **automatic recovery** of issues. It's not about preventing user disruption. While that's a bonus, a highly available system can still have user disruption to your user base when there is a failure.
- Now, high availability has **costs required** to implement it. it needs some design decisions to be made in advance, and it requires a certain level of automation.
- Sometimes high availability needs **redundant service** or **redundant infrastructure** to be in place ready to switch customers over to in the event of a disaster to minimize downtime.

Fault Tolerance (FT):

- Now let's take this a step further and talk about fault tolerance and how it differs from high availability. when most people think a high availability, they're actually mixing it up with fault tolerance.
- fault tolerance in some ways is very similar to high availability but it is much more.
- fault tolerance is defined as the property that enables a system, to **continue operating properly** in the event of a **failure of some** of its **components**, so one or more fault within the system.

- fault tolerance means that if a system has faults and this could be one fault or multiple faults, then it should continue to operate properly even while those faults are present and being fixed.it means it has a failure without impacting customers.
- HA is just about **maximizing uptime**. fault tolerance is what means to operate through failure.
- fault tolerance can be expensive because it's much more complex to implement versus high availability.

Disaster Recovery (DR):

- The definition of disaster recovery is a set of policies, tools and procedures to **enables the recovery** or **continuation** of **vital** technology infrastructure and systems **following a natural or human-induces disaster.**
- So, while high availability and fault tolerance are a way designing systems to cope or operate through disaster, disaster recovery is about what to plan for and do when disaster occurs, which knocked out a system.
- The worst time for any business is recovering in the event of a major disaster. In that type of environment bad decisions are made, decisions based on shock, lack of sleep and fear of how to recover.
- So, a good set of DR processes need to pre-plan for everything in advance.
- Build a set of processes and documentation.
- Plan for staffing and physical issues when a disaster happens, you have a business premises with some staff, then part of a good DR plan might be to have a standby premise ready. And this standby premises can be used in the event of a disaster. That why done in advance your staff, unaffected by the disaster, know exactly where to go.
- You might need space for IT systems or you might use a cloud platform such as AWS as a backup location. but in any case, you need the idea of a backup premises or a backup location that's ready to go in the event of a disaster. If you have local infrastructure then make sure you have resilience. make sure you have plans in place and ready during a disaster.
- this might be extra hardware sitting at the backup site ready to go. Or it might be virtual machines or instances operating in a cloud environment ready when you need them.
- A good DR plan means taking regular backups. this is essential. but the worst thing you can do is to store these backups at the same site as your system. it's dangerous.
- If your main site is damaged, your primary data and your backups are damaged at the same time, and that's a huge problem.

Summary:

high availability - Minimise any outage

Fault Tolerance - Operate Through Faults

Disaster recovery - used when these don't work

Domain Name System (DNS) Fundamentals

DNS

- DNS is a **discovery service**
- Translates machine into human and vice-versa
- When using say, amazon.com, you might access it using www.amazon.com. but this isn't what your computer uses, that requires IP addresses. And so, one function of DNS, is to find the IP address for a given domain name.
- For example, www.amzone.com, Now, there are two crucial things to realize about DNS and its requirements. Because of the number of services on the internet, and on private networks, and because of the importance of DNS, it's actually a huge database, and it has to be distributed and resilient.
- DNS is a **huge scale database**. it's **distributed** and its **global**.
- it translates the information which machine need to and from information which human needs.
- when you use your laptop to browse www amazon.com from your prospective it just loads but that some massive abstraction and simplification computer don't work with domain names for your laptop to communicate to the ww.com web server it uses IP addresses how this conversation happen is transparent to you. and that's because either your laptop or device is communicating directly to the DNS system or it's been configured to talk to a DNS resolver server on your behalf and potentially this resolver server is running within your internet provider or on your internet router. DNS is a huge scale database it distributed and its Global but somewhere in that Global platform is one piece of information on single database which has information in it that we need to convert between the name and IP addresses so in this example amazon.com.
- Now the database that piece of information that we are looking for is called as zone and the way that that zone is stored is often referred to as a zone file and somewhere on the internet is one zone file for amazon.com now that amazon.com zone file has a record inside it a record a DNS record which links the name, www and IP addresses that your laptop needs to communicate with that website.
- this zone file is hosted by a DNS server that's known as a name server or NS for shorts so if you can query this zone for the record www.amazon.com then use the result of that query which has an IP address, your laptop can communicate with the web server.
- This zone file could be allocated anywhere on potentially one or two out of millions of DNS name servers also one of the core pieces of functionality that DNS provides is it allow a DNS resolver server which sitting either on your internet router or in your internet provider to find this zone.
- let's quickly summarise firstly we have the DNS client and the DNS client refers to the device or thing which want to the data the DNS has so it wants IP address for amazon.com and generally the DNS client is a piece of software running inside operating system on a device that you use so a laptop, Mobile Phone or Tablet or PC.
- next we have got a DNS resolver and that could be a piece of software running also on the DNS client for your laptop PC Tablet or it could be a separate server running inside your internet router or a physical server running inside your internet provider and it is the DNS resolver that queries the DNS system on your behalf. so general is a DNS client talks to the DNS server and it as the DNS resolver to query DNS on its behalf.
- next we have got a DNS zone. and a DNS zone is a part of the Global DNS data. for example, amazon.com, netflix.com they are all example of zones, and they live inside the DNS system. a zone file is how the data for that zone is physically stored. so there going to be a zone file, for amazon.com, netflix.com so if I talk about a DNS zone, I refer to what the data is, its substance. if I talk about zone file and talking about the physically that data is stored.
- lastly, we have a name server or DNS server this is a server which shows these zone files. so, the point that we need to get to when using DNS, is to find the name server, which host the particular zone file, and then query that name server for a record that is in that zone file. that's the job of DNS.

so, the DNS resolver server needs to locate the correct name server for a given zone query that name server, retrieve the information it needs, and then passed it back to the DNS client. that is the flow of DNS.

Remember these!

- DNS Client=> Your laptop, phone, tablet, PC.
- Resolver=> software on your device, or a server which queries DNS on your behalf.
- DNS zone => A part of the DNS databases (e.g., amazon.com)
- Zone file => physical database for a zone
- Nameserver => where zone files are hosted
- DNS has to have a starting point, and that point is the DNS root. DNS is structure like a tree.
- DNS root is hosted on 13 special name servers, known as the root servers. The root servers are operated, by 12 different large global companies or organization.

Route53 (R53) Fundamentals

- route 53 provides two main services first it's a service in a w s which allows you to **register domain**, second and it can **Host zone file** for you on Managed name servers which it provides.
- route 53 is a **global service** with a single database it's one of a very few a w a services which operates as a single global service. So don't need to pick a region, when using it from the console UI.
- the data that route53 stores and manages is distributed globally as a single set and it's replicated between regions and so it's the globally resilient service
- route 53 can tolerate the failure of one or more regions and continue to operate without any problems now it's one of the most important AWS products it needs to be able to scale stay highly performance while remaining reliable and continue work through failure.
- Route53 provides **DNS zones**, as well as hosting for those zones. it's basically DNS as a service.
- it lets you create and manage zone files, and these zone files are called hosted zones, in Route53 terminology, because they're hosted on AWS managed name servers.
- from route 53 prospective, every hosted zone also has a number of allocated manage name servers. now a hosted zone can be public, which means that the data is accessible on the public internet. the names servers for a public hosted zone live logically in the A W S Public zone and that is accessible everywhere with the public internet connection.
- a hosted zone could also be private, which means that it linked to one or more VPCs, and only accessible from within those VPCs and you might use this type of zone if you want to host sensitive DNS records that you don't want to be publicly accessible.

DNS Record Types

DNS servers create a DNS record to provide important information about a domain or hostname, particularly its current IP address. The most common DNS record types are:

- **Address Mapping record (A Record)—**also known as a DNS host record, stores a hostname and its corresponding IPv4 address.
- **IP Version 6 Address record (AAAA Record)—**stores a hostname and its corresponding IPv6 address.
- **Canonical Name record (CNAME Record)—**can be used to alias a hostname to another hostname. When a DNS client requests a record that contains a CNAME, which points to another hostname, the DNS resolution process is repeated with the new hostname.
- **Mail exchanger record (MX Record)—**specifies an SMTP email server for the domain, used to route outgoing emails to an email server.

- **Name Server records (NS Record)—**specifies that a DNS Zone, such as “example.com” is delegated to a specific Authoritative Name Server, and provides the address of the name server.
- **Reverse-lookup Pointer records (PTR Record)—**allows a DNS resolver to provide an IP address and receive a hostname (reverse DNS lookup).
- **Certificate record (CERT Record)—**stores encryption certificates—PKIX, SPKI, PGP, and so on.
- Service Location (SRV Record)—a service location record, like MX but for other communication protocols.
- **Text Record (TXT Record)—**typically carries machine-readable data such as opportunistic encryption, sender policy framework, DKIM, DMARC, etc.
- **Start of Authority (SOA Record)—**this record appears at the beginning of a DNS zone file, and indicates the Authoritative Name Server for the current DNS zone, contact details for the domain administrator, domain serial number, and information on how frequently DNS information for this zone should be refreshed.

MODULE 5 - IAM, ACCOUNTS AND AWS ORGANISATIONS

IAM Identity Policies

Let's talk about how AWS handle security?

IAM policies are a type of policy, which get attached to identities inside AWS. Identities are IAM users, IAM groups and IAM roles.

Now, you'll use IAM policies constantly, you'll need to understand them for the exam, as well as if you design and implement solution inside AWS.

Understanding policies comes in three main stages. Firstly, you'll need to understand their architecture and how they work. Second, you need to gain the ability to read and understand policy. And finally, you'll need to lean to write your own.

An IAM identity policy or an IAM policies is just a set of security statements to AWS. it grants access or denied access to AWS products and features to any identity, which uses that policy. identity policy is also known as policy statements are created using JSON.

Above is an example of an identity policy document. and this is the type of thing that you would use with the user, group or a role. At the higher level, a policy document is just one or more statements. so inside the statement block, there are multiple statements each of them is inside a pair of curly braces.

When an identity attempts to access AWS resources that identity needs to prove who it is to AWS a Process known as **Authentication**. once authenticated that identity is known as **authenticated identity**.

AWS knows which policies and identity has, and it could be multiple, and each of these policies can have multiple statements in it, so AWS has a collection of all of the statements which apply to the given identity.

AWS also knows which resources or resources you are attempting to interact with as well as what actions you want to perform on those resources

Let's step through what make a statement. the first part of statement is a **statement ID or a Sid** and this is an optional field which lets you identify a statement and what it does see how in this case it's States full access. In the second statement, it states denycatbucket. This is just a way that we can inform the reader.

A statement only applies if the interaction that you are having with AWS match the action and the resources.

The **action** part of a statement match is one or more actions it can be very specific and list the specific individual action.

Every interaction that you have with AWS is a combination of two main things. The resources that you're interacting with, and the action that you're attempting to perform on that resource.

A statement only applies if the interaction that you're having with AWS match the action and the resource.

The action part of a statement matches one or more actions.

It can be every specific and list a specific individual action.

Effect controls what AWS does if the action and the resource parts of a statemen match the operation what you are attempting to do with AWS.

IAM Users and ARNs

IAM users are an identity used for anything requiring long-term AWS access e.g., **Humans**, **Applications or Service accounts.**

- IAM starts with a **principal**. And this is a word which represent an entity trying to access an AWS account. Principals can be individual people, computers, services or a group have any of these things.
- For a principal to be able to do anything it needs to **authenticate** and be **authorised** and that is the process that I want to step through now.
- a principle which in this example is a personal or an application make request to IAM to interact with resources now to be able to interact with resources it needs to authenticate against an identity within IAM.
- Authentication is the first step and the authentication is a process where the principle on the left proves to IAM that it is an identity that it claims to be.
- Authentication for IAM user is done either using username and password or access keys these are both example of long-term credential
- Generally, username password use if human is accessing AWS and accessing the console you why access keys are used if it's an application
- Now once the principal goes through the authentication process the principal is known as unauthenticated ID and authenticated ID has been able to prove to AWS that it is intended the attitude that claim to be.

- The principal becomes an authenticated identity than AWS knows which policy is applied to that identity. This process known as **Authorization**.
- Authentication is how principal can prove to IAM that it is the identity that it claims to be using username and password or access key and authorisation is IAM checking the statement that apply to that identity and either following or denying that access
- Amazon resource names or ARN do one thing that that's to uniquely identify resources within any a w s when you are working with resources using the command line or API is you need a way to refer to these resources in an ambiguous way and allow you to refer to a single resource if needed or in some cases a group of resources using wildcards

IAM Groups

- IAM groups are containers for IAM users they exist to make organising large sets of IAM users easier. you can't login to IAM groups, and IAM group have no credential of their own.
- IAM user can be a member of multiple IAM groups so that's the important thing to Remember.
- Groups gives us to main benefits as solution architect firstly, they allow effective administration style management of user so we can make group that present team or projects or any other functional group inside a business and we can put IAM user into those groups so it can help us organise now the second benefit which builds of the first group can actually have policy is attached to them and this is the both inline policy and managed policy.
- IAM user can be a member of 10 groups, and there was a 5000 IAA user limit for account. neither of those are changeable, they're hard limits.
- limitation of groups, is that you can't have any nesting, you can have group withing groups.
- IAM groups contain users, and IAM groups can have permission attached, that's it.
- groups cannot be logged into; they don't have any credentials.
- Now, there is limit of 300 groups per account, but this can be increase with a support ticket.
- Groups are not a true identity. they can't be reference as a principal in a policy

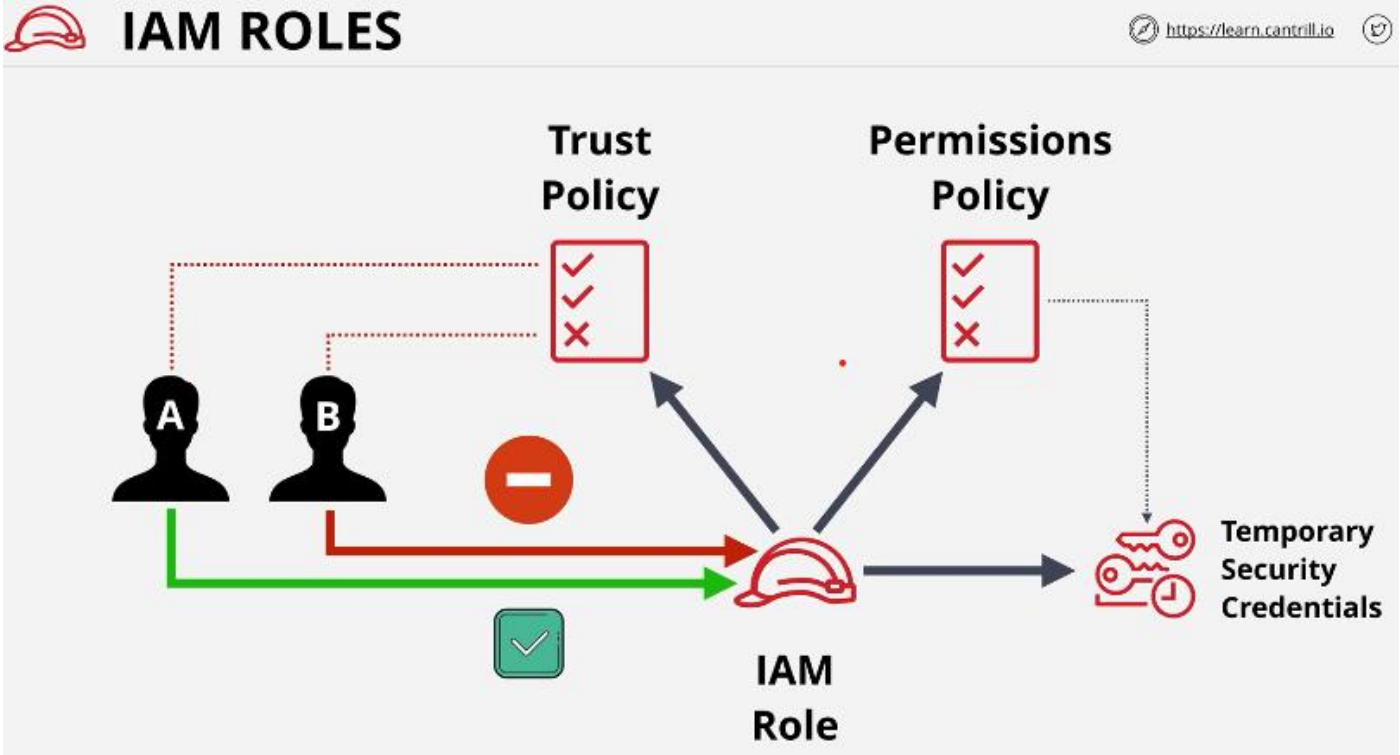
IAM Roles - The Tech

An IAM role is an IAM identity that you can create in your account that has specific permissions. An IAM role is similar to an IAM user, in that it is an AWS identity with permission policies that determine what the identity can and cannot do in AWS. However, instead of being uniquely associated with one person, a role is intended to be assumable by anyone who needs it. Also, a role does not have standard long-term credentials such as a password or access keys associated with it. Instead, when you assume a role, it provides you with temporary security credentials for your role session.

You can use roles to delegate access to users, applications, or services that don't normally have access to your AWS resources.

IAM roles have two types of policies which can be attached. The **trust policy** and **permission policy**.

The trust policy controls which **identities** can **assume that role**.



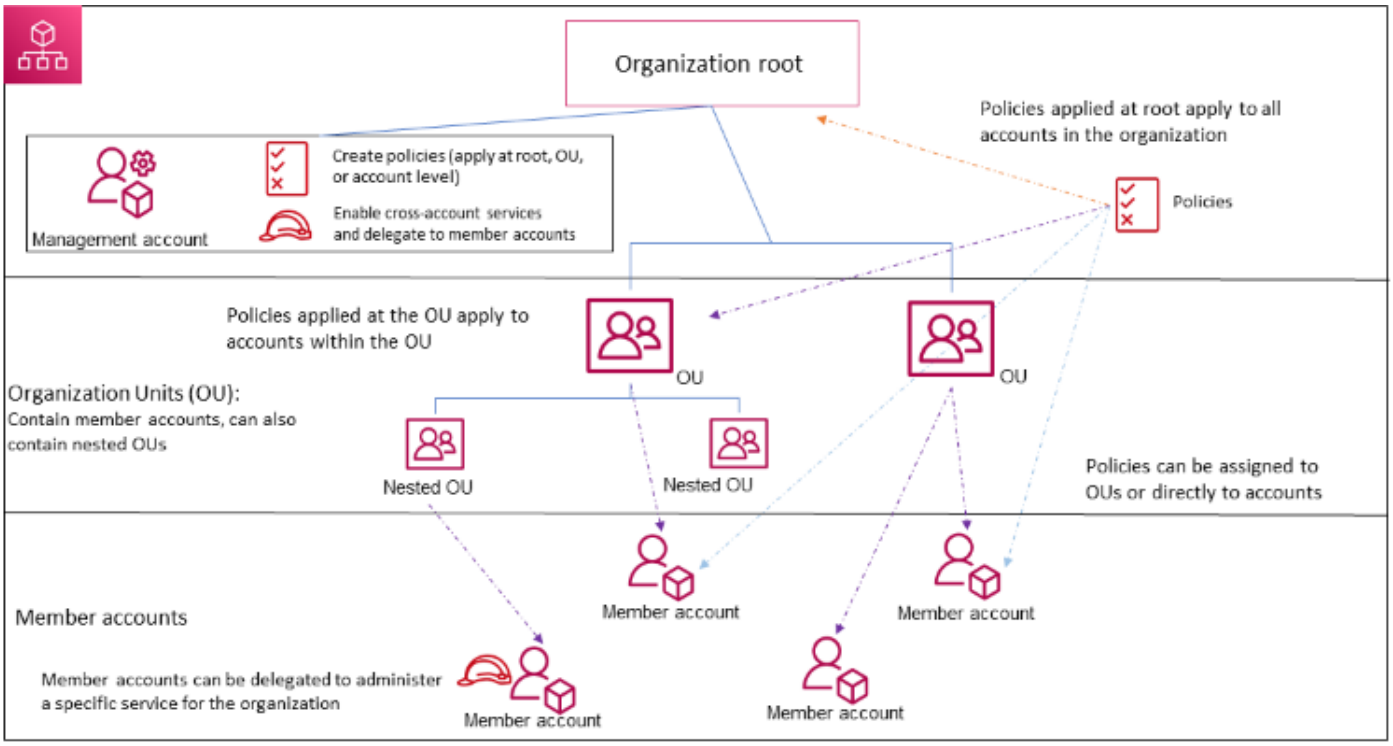
- With the above example, identity A is allowed to assume the role because identity A is allowed in the trust policy. Identity B is not specified as being allowed to assume the role in the trust policy.
- Now, the trust policy can reference identities in the same account, so other IAM user, or the roles and even AWS services such as EC2, and the trust policy can also reference identities.
- If a role gets assumes by something which is allowed to assume it, then AWS generates temporary security credential, and these are made available to the identity which assumed the role.
- temporary credentials are much like access keys but instead of being long-term, they're time limited.
- Once they expire, the identity will need to renew them by reassuming the role, and at that point new credentials are generated and given to the identity, again which assumed that role.
- Now, these **temporary credentials** will be able to access whatever AWS resources specified within the **permission policy**. Every time the temporary credentials are used, the access is checked against this permission policy.
- If you change the permission policy, the permission of those temporary credentials also changes.
- Now when you assume a role, temporary credentials are generated by an AWS service called STS or the Secure Token Service.

When to use IAM Roles

- One of the most common uses of roles within the same AWS account are for AWS services themselves. AWS services operate on your behalf, and they need access rights to perform certain actions.
- An example of this is AWS Lambda. It's a function as a service product.
- This function when it runs might do things like start and stop EC2 instances or it might perform backups.
- The key thing is a lambda function as with most AWS things has no permissions by default.
- A lambda function is not an AWS identity. It's a component of a service and so it needs some way of getting permissions to do things when it runs.
- Running lambda function is known as function invocation or a function execution using Lambda terminology.

AWS Organizations

- AWS organization is a product which allows larger business to manage multiple AWS accounts in a cost-effective way, with little to no management overhead.
- Without AWS organizations many large businesses would face a situation where they need to manage many AWS accounts.
- AWS Organizations includes account management and consolidated billing capabilities that enable you to better meet the budgetary, security, and compliance needs of your business. As an administrator of an organization, you can create accounts in your organization and invite existing accounts to join the organization.
- The following diagram shows a basic organization that consists of seven accounts that are organized into four organizational units (OUs) under the root. The organization also has several policies that are attached to some of the OUs or directly to accounts. For a description of each of these items, refer to the definitions in this topic.



Root

- The parent container for all the accounts for your organization. If you apply a policy to the root, it applies to all organizational units (OUs) and accounts in the organization.
- Organization unit (OU)
- A container for accounts within a root. An OU also can contain other OUs, enabling you to create a hierarchy that resembles an upside-down tree, with a root at the top and branches of OUs that reach down, ending in accounts that are the leaves of the tree. When you attach a policy to one of the nodes in the hierarchy, it flows down and affects all the branches (OUs) and leaves (accounts) beneath it. An OU can have exactly one parent, and currently each account can be a member of exactly one OU.

Account

An account in Organizations is a standard AWS account that contains your AWS resources and the identities that can access those resources.

There are two types of accounts in an organization: a single account that is designated as the **management account**, and one or more **member account**.

- The **management account** is the account that you use to create the organization. From the organization's management account, you can do the following:
    - Create accounts in the organization
    - Invite other existing accounts to the organization
    - Remove accounts from the organization
    - Manage invitations
    - Apply policies to entities (roots, OUs, or accounts) within the organization
    - Enable integration with supported AWS services to provide service functionality across all of the accounts in the organization.

The management account has the responsibilities of a *payer account* and is responsible for paying all charges that are accrued by the member accounts. You can't change an organization's management account.

- **Member accounts** make up all of the rest of the accounts in an organization. An account can be a member of only one organization at a time. You can attach a policy to an account to apply controls to only that one account.

Invitation

- The process of asking another account to join your organization. An invitation can be issued only by the organization's management account.
- The invitation is extended to either the account ID or the email address that is associated with the invited account.
- After the invited account accepts an invitation, it becomes a member account in the organization. Invitations also can be sent to all current member accounts when the organization needs all members to approve the change from supporting only consolidated billing features to supporting all features in the organization.
- Invitations work by accounts exchanging handshakes. You might not see handshakes when you work in the AWS Organizations console. But if you use the AWS CLI or AWS Organizations API, you must work directly with handshakes.

Handshake

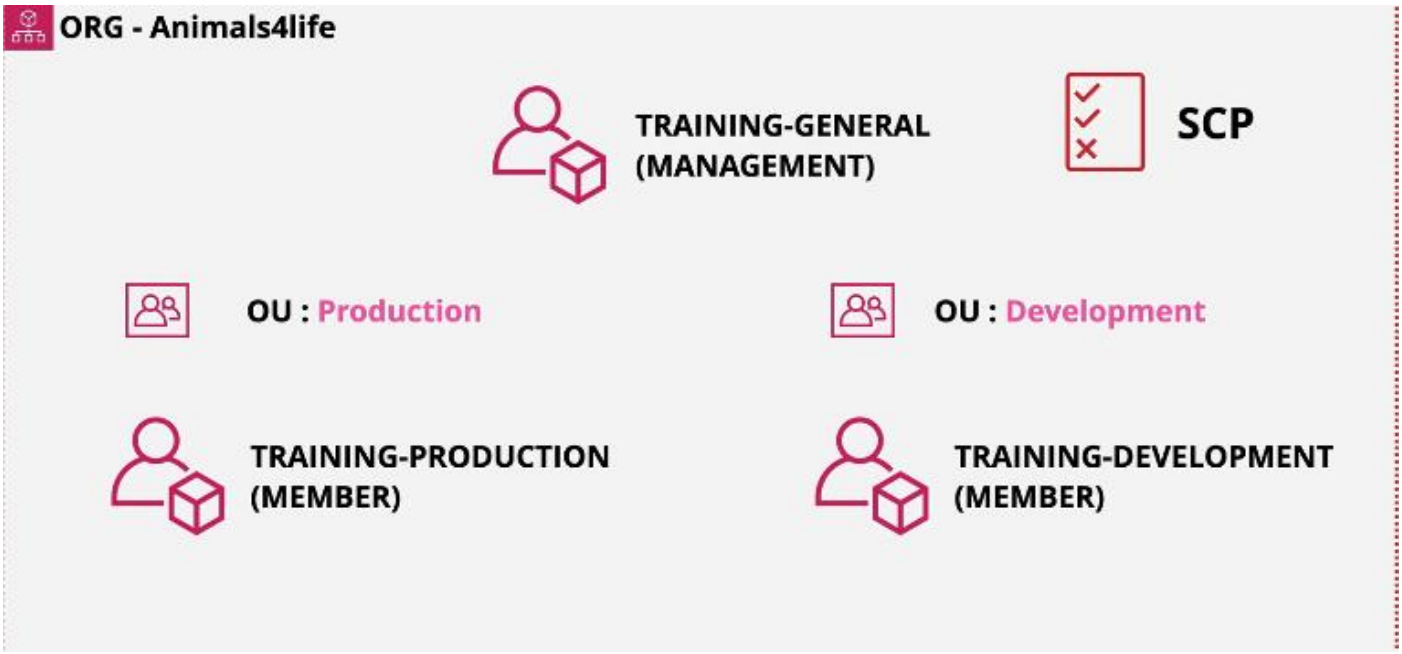
- A multi-step process of exchanging information between two parties. One of its primary uses in AWS Organizations is to serve as the underlying implementation for invitations.
- Handshake messages are passed between and responded to by the handshake initiator and the recipient. The messages are passed in a way that helps ensure that both parties know what the current status is.
- Handshakes also are used when changing the organization from supporting only consolidated billing features to supporting all features that AWS Organizations offers. You generally need to directly interact with handshakes only if you work with the AWS Organizations API or command line tools such as the AWS CLI.

AWS Organizations DEMO

- The GENERAL account will become the MANAGEMENT account for the organisation
- We will invite the PRODUCTION account as a MEMBER account and create the DEVELOPMENT account as a MEMBER account.
- Finally - we will create an **OrganizationAccountAccessRole** in the production account, and use this role to switch between accounts.
- WARNING: If you get an error "You have exceeded the allowed number of AWS Accounts" then you can go here https://console.AWS.amazon.com/servicequotas/home?region=us-east-1#!/services/organizations/quotas/L-29A0C5DF and request a quote increase for the number of member accounts in an ORG

Service Control Policies (SCPs)

- SCPs are a features of AWS organizations which can be used to restrict AWS accounts. They're an essential feature to understand that if you're involved in the design and implementation of larger AWS platforms.
- We've created an organization for Animal4life and inside it we have the genera account which from now on, Management account, and then two member accounts, Prod and Dev
- All of these AWS accounts are witing the root container of the organization. we're going to be adding organizational units, one for production and one for development. The concept of the Service Control Policy is simple. it's a policy document written in JSON. and these service control policies can be attached to the organizations as a whole, by attaching them to the root container or they can be attached to one or more organizational units. they can even be attached to individual AWS account.



- Service control policies inherit down the organizational tree. this means if they're attached to the organization as a whole, then they affect all of the account inside the organization.
- if they're attached to an organizational unit then they impact all accounts directly inside that organizational unit,
- Management account is special because it never affected by Service control policies.
- It's the only AWS account within AWS organizations which can't be restricted using service control policies.

- Service control policies are account permissions boundaries. means they limit what the AWS account can do including the Account root user within that account.
- You can't directly restrict what the account root user of an AWS account can do. The account root user always has full permissions over that entire AWS account but with a Service control policy you can restrict that specifically any identities within that account.
- Service control policies define the limit of what is, and isn't allowed just like a boundary, but they don't grant permissions.
- You still need to give identities within that AWS account permissions to AWS resources but any SCPs will limit the permissions that can be assigned to individual identities.
- you can use SCPs in two ways, you can block by default and allow certain services, which is an allowable list. Or you can allow by default and block access to certain services, which is a deny list.
- When you enable SCPs on your organization AWS apply a default policy, which is called fullAWSaccess. This is applied to the organization and all OUs within that organization.

Allow lists vs. deny lists

Allow lists and deny lists are complementary strategies that you can use to apply SCPs to filter the permissions that are available to accounts.

- **Allow list strategy** – You explicitly specify the access that ***is*** allowed. All other access is implicitly blocked. By default, AWS Organizations attaches an AWS managed policy called FullAWSAccess to all roots, OUs, and accounts. This helps ensure that, as you build your organization, nothing is blocked until you want it to be. In other words, by default all permissions are allowed. When you are ready to restrict permissions, you ***replace*** the FullAWSAccess policy with one that allows only the more limited, desired set of permissions. Users and roles in the affected accounts can then exercise only that level of access, even if their IAM policies allow all actions. If you replace the default policy on the root, all accounts in the organization are affected by the restrictions. You can't add permissions back at a lower level in the hierarchy because an SCP never grants permissions; it only filters them.
- **Deny list strategy** – You explicitly specify the access that ***is not*** allowed. All other access is allowed. In this scenario, all permissions are allowed unless explicitly blocked. This is the default behavior of AWS Organizations. By default, AWS Organizations attaches an AWS managed policy called FullAWSAccess to all roots, OUs, and accounts. This allows any account to access any service or operation with no AWS Organizations–imposed restrictions. Unlike the allow list technique described above, when using deny lists, you leave the default FullAWSAccess policy in place (that allow "all"). But then you attach additional policies that explicitly ***deny*** access to the unwanted services and actions. Just as with IAM permission policies, an explicit deny of a service action overrides any allow of that action.

CloudWatch Logs

- It is a **public service** usable from AWS or **on-premises**
- It Stores, Monitor and Access logging data.
- It also integrated with AWS services like EC2, VPC Flow logs, Lambda, Cloud Trail, R53 and more.
- It is often the default place where AWS Services can output their logging too.
- CloudWatch Logs is a public service and can also be utilised in an on-premises environment and even from other public cloud platforms.
- CloudWatch logs are also capable, of taking logging data and generating a metric from it. this is known as metric filter.
- Now the starting point are logging sources, which can be include AWS products and services, mobile or server-based applications, external computer services, or virtual or physical servers, databases, or even external APIs. these sources inject data into CloudWatch logs as log events.
- Log events have a time stamp and a message block. CloudWatch logs treated this message as a row block of data.
- Log events are stored inside log streams and log streams are essentially, a sequence of log events from the same source.
- We also have log groups. log groups are containers for multiple log streams, for the same type of logging. It also a place that stores configuration setting, where we define things like retention setting and permissions.

CloudTrail

- CloudTrail Is a product which logs API calls and account events. It logs API calls/activities as a CloudTrail Event
- **90 days** stored by default in Event History
- It's very often used to **diagnose security or performance issues**, or to provide quality account level traceability.
- It is enabled by default in AWS accounts and logs free information with a **90-day retention.**
- It can be configured to store data indefinitely in **S3** or **CloudWatch Logs**.
- CloudTrail events can be two types, Management event or data events.
- **Management events** provide information about management operations that are performed on resources in you AWS account.
- These are also known as control plane operations. Think of things like creating an EC2 instance, terminating an EC2 instance and creating a VPC, these are all control plane operations.
- Now, data events contain information about resource operations performed on or in a resource. So, examples of this might be object being uploaded to S3, or object being accessed from S3, or when a lambda function is being invoked.
- By default, Cloud Trail only logs management events because data events are often much higher volume.
- CloudTrail trail is the **unit of configuration** withing the CloudTrail product. It's a way you provide configuration to CloudTrail on how to operate.
- A trial logs events for the AWS region that it's created in.
- CloudTrail is a **regional service** but when you create a trial, it can be configured to operate in one or two ways.
- You can create a trial which is a one region trail, or a trial can be set to all regions.
- A single region trail only logs events for that region.
- All **region trail** you can think of as a collection of trials in every AWS region, but it's managed as one logical trail.

S3 Security

- S3 is **private** by **default**.
- The only identity which has any initial access to an S3 bucket is the account root user of the account which own that bucket, so, the account which created it.
- Anything else so any of the permissions have to be explicitly granted and there are a few ways that this can be done.
- The first f S3 bucket policy. and S3 bucket policy is a types of **resource policy**.
- A resource policy is just like an identity policy. but as the name suggests, they're attached to resources instead of identities.
- Resource policies provide a resource perspective on permissions.
- The difference between resource policies and identity policies is all about this perspective.
- with identity policies, you're controlling what that identity can access, with resource policies you're controlling who can access that resource.
- So, it's from an inverse perspective one is identities and one is resources.
- Identity policies have one pretty significant limitation, you can only attach identity policy to identities in your own account. So, identity policies can only control security inside your account.
- Resource policies can access from the same account or different account because the policy is attached to the resource. and it can reference any other identities inside that policy. so, by attaching the policy to the resource, and then having flexibility to be able to reference any other identity, whether they are in the same account, or different accounts, resource policies, therefore are a great way of controlling Access for a particular resource, no matter what the source of that access is.
- They also have another benefit; resource policies can allow or deny anonymous principles. identity policies, by design have to be attached to a valid identity in AWS. you can't have one attached to nothing. resource policies can be used to open a bucket to a world by referencing all principles even those not authenticated by AWS.

- Bucket policies can be used to grant anonymous access.
- Resource policies have one major difference to identity policies and that's the presence of an explicit principal component. the principal part of a resource policies define which principle are affected by the policy. so, the policy is attached to a bucket in this case, but we need a way to say who is impacted by the configuration of that policy because the bucket policy can contain multiple statements, there might be one statement which affects your account, and one which affect another account as well as the one which affect the specific user. the principal part of a policy or more specifically the principal part of a statement in a policy defined who that statement applies to, which identities which principals.

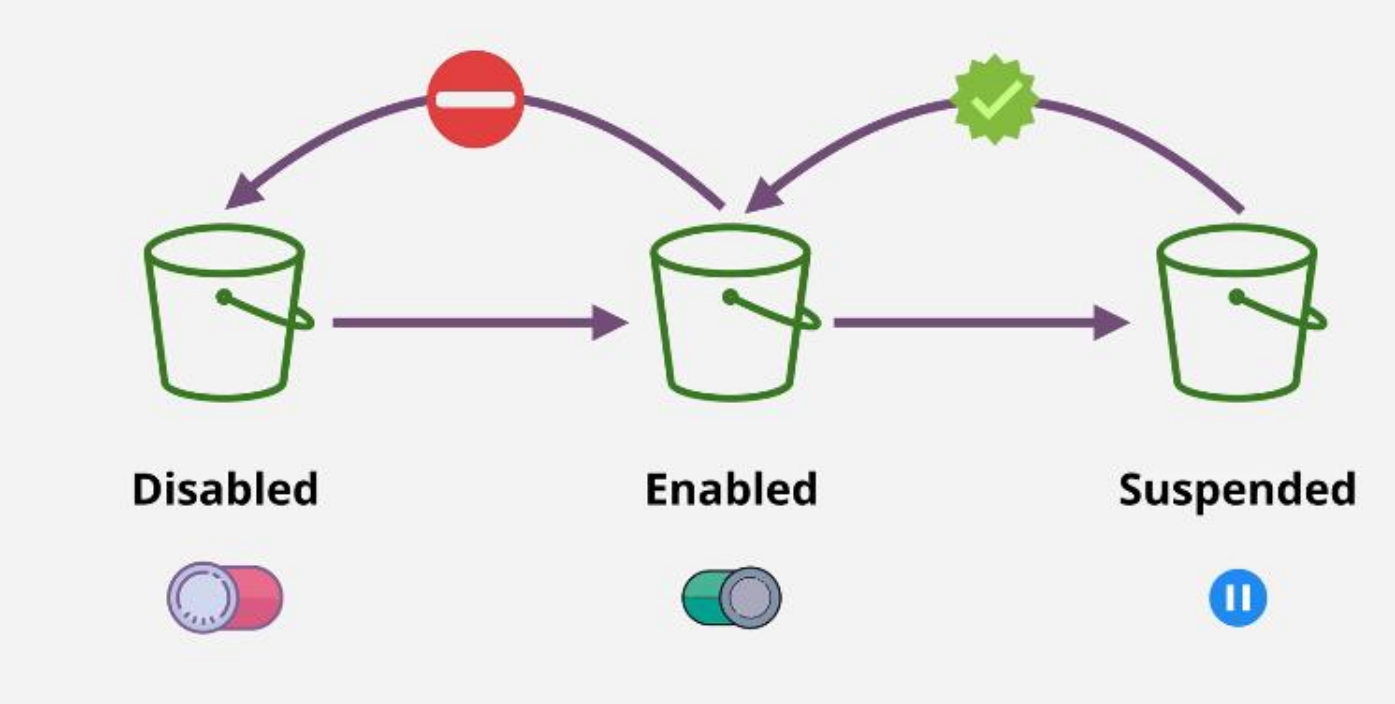
- Your identity policy, by definition applies to you. so you are the principal. so a good way of identifying if a policy is a resource policy or identify policy is the presence of this principal component.

S3 Static Hosting

- Accessing S3 is generally done via APIs
- Static Website Hosting is a feature of the product which lets you define a HTTP endpoint, set index and error documents and use S3 like a website.
- You can only use a custom domain with a bucket if the name of the bucket matches the domain.
- Let's understand the pricing structure for S3
- Firstly, we've got the cost to store data on S3, and this is generally expressing as a per gigabyte month fee.
- There's also a data transfer fee. So, every gigabyte of data that you transfer in and out of S3, there's a cost.
- Now, to transfer data into S3 is always free, so you're never charged for transferring data into S3. TO transfer data out of S2, there's is a per gigabyte charge.
- Storage and data transfer are incredibly cheap options available.
- Free trier: 5GB storage and 20,000 GET request and 20,000 PUT request provided by AWS.

Object Versioning & MFA Delete

- Stored all versions of the object (including all writes and even if you delete an object)
- Great backup tool.
- Once enabled, versioning cannot be disabled only suspended.
- Integrate with lifecycle rule of S3.
- I have a bucket and inside the bucket is a picture of my cat, Winkie. So, the object is called winkie.jpg, it's identified in the bucket by the key, essentially its name and the key are unique. If I modify winkie.jpg object or delete it, these changes impact this object.
- Now, there's an attribute of an object which is ID of the object. when versioning on a bucket is disabled, the IP of the objects in that bucket are set to null.
- If you upload or put a new object into a bucket with versioning enabled, then S3 allocates an ID to that object. if any modifications are made to this object, let's say somebody accidentally overrides the winkie.jpg object with a dog picture but still calls it winkie.jpg, S3 doesn't remove the original object, it allocated a new ID to the new version and it retains the old version.
- The newest version of any object in a version enabled bucket is known as the current version of that bucket.
- If we delete the object and we don't give any specific version ID, then S3 will add a new special version of that object, known as a delete marker. Now the delete marker essentially is just a new version of that object and hidden.



- If you want to delete a version of an object then you need to delete an object and specify the particular version ID that you want to remove.
- Space is consumed by **ALL versions**.
- you are **billed** for **ALL versions**.
- Only way to **0 costs**- is to **delete the bucket**

MFA Delete

- Enabled in versioning configuration
- MFA is required to change bucket versioning state
- MFA is required to delete versions
- In order to change a bucket's versioning state or delete a particular version of an object, you need to provide the serial number of your MFA token as well as the code that it generates with API CALLs.

S3 Performance Optimization

Understanding Performance characteristics of S3 is essential as a solution architect.

We know from the Animal4life scenario that remote worker need to upload large data sets and do so frequently.

By default, when you upload an object to S3, it uploaded as a single blob of data in a single stream.

A file becomes an object and it's uploaded using the PUT object API call and places in a bucket and this all happens in a single stream. This method has a problem if a stream fails the whole upload fails and the only recovery from it a full restart of the entire upload.

If the upload fails at 4.5GB of a 5GB upload, that's 4.5GB of data wasted and probably a significant amount of time.

S3 performance optimization

- When using the single PUT method, the speed and reliability of the upload with always be Limited because of this single stream of data.
- Data transfer protocol such as BitTorrent have been developed to allow a speedy, distributed transfer of data.
- Using data transfer with only a single stream is just a bad idea. now there is a limit within AWS if you utilise a single PUT upload then you are limited to 5 GB of data as a maximum.
- Solution is multipart upload. multipart upload improves the speed and reliability of upload to S3 and it does this by breaking data up into individual parts.
- The minimum size for using multipart upload is 100 MB. you can't use multipart upload if you are uploading data smaller than this.
- Multipart upload is so effective is that each individual part is treated as its own isolated upload. each individual part can fail in isolation and be restarted in isolation rather than needing to restart the whole thing.
- This means that the risk of uploading large amount of data to S3 is significantly reduced.
- The transfer rate of the whole upload is the sum of all the individual parts. so you get much better transfer rates by splitting this original blob of data into smaller individual parts and then uploading them in parallel.
- Transfer acceleration uses the network of AWS edge locations which located in lots of convenient locations globally. S3 bucket needs to be enabled for transfer acceleration. the default is that it's switched off and there are some restrictions for enabling it.
- The bucket name cannot contain periods and it needs to be DNS compatible in its naming, so keep in mind those two restrictions

Encryption 101

Encryption at rest is designed to protect against physical theft and physical tampering.

Encryption at rest is also useful commonly within cloud environment you are data stored on shared hardware, and it’s done so in an encrypted form

The Other approach to encryption is known as **encryption in transit** and this is aimed at protecting data while its being transferred between two places.

Plaintext: information that can be directly read by humans or a machine (this article is an example of plaintext). Plaintext is a historic term pre-dating computer, when encryption was only used for hardcopy text, nowadays it is associated with many formats including music, movies and computer programs

Algorithm: algorithms are used for important tasks such as data encryption, authentication, and digital signatures

Key: a key is a variable value that is applied using an algorithm to a string or block of unencrypted text to produce encrypted text, or to decrypt encrypted text.

Ciphertext: Ciphertext is the unreadable output of an encryption algorithm

Encryption: Encryption is the process of converting normal message (plaintext) into meaningless message (Ciphertext).

Decryption: Decryption is the process of converting meaningless message (Ciphertext) into its original form (Plaintext).

Symmetric Key Cryptography: It is an encryption system where the sender and receiver of message use a single common key to encrypt and decrypt messages. Symmetric Key Systems are faster and simpler but the problem is that sender and receiver have to somehow exchange key in a secure manner. The most popular symmetric key cryptography system is Data Encryption System (DES).

Asymmetric encryption: Under this system a pair of keys is used to encrypt and decrypt information. A public key is used for encryption and a private key is used for decryption. Public key and Private Key are different. Even if the public key is known by everyone the intended receiver can only decode it because he alone knows the private key.

Steganography: Steganography is the practice of hiding a secret message inside of (or even on top of) something that is not secret. That something can be just about anything you want. These days, many examples of steganography involve embedding a secret piece of text inside of a picture. Or hiding a secret message or script inside of a Word or Excel document.

key management service (KMS)

- KMS is regional and public service. Now it's a public service which means it occupies the AWS Public zone and so it can be connected to from anywhere with access to this public zone. but you need permissions in order to access it.
- KMS lets you create, store and manage cryptographic key. These are the keys which can be used to convert plain text to cypher text and vice versa.
- Kms is capable of handling both symmetric and asymmetric keys.
- Kms is also capable of performing actual cryptographic operations which include encryption and decryption but also many other
- Now the fundamental things to understand about kms is that cryptographic key is never leave the product. kms can create keys, keys can be imported, it manages keys, it can use this key to perform operations but the keys are locked inside the kms.
- KMS provides a FIPS 140-2 compliant service. That's a US security standard.
- Now the main thing that KMS manages as known as CMKs or customer master keys.
- These CMKs are used by KMS within cryptographic operations.
- CMKs are logical. think of them as a container for the actual physical master key. so, a CMK is logical and it contains a few things. it contains a key ID, which is unique identifier for the key. it contains a creation date, a key policy, which is the type of resource policy, a description and a state of the key, whether its active or not.
- CMK backed by physical key material which can actually use to encrypt and decrypt.
- A CMK can only be used to directly encrypt or directly decrypt data that is a maximum of 4 KB in size.
- KMS does not store the data encryption key in any way. It provides it to you or the service using KMS, and then it discards.
- The reason it discards it that KMS doesn't actually perform the encryption and decryption of data using data encryption keys. you do that or the service using KMS does that.
- let's look how his works. when a data encryption key is generated, KMS provides you with two versions of that data encryption key.
- First, a plaintext version of the key and also a ciphertext or an encrypted version of same encryption key.
- The data encryption key is encrypted by the customer master key that generates it So, in the future it can be decrypted by KMS using that same customer master key.

key concept

- **CMKs** are isolated to a **region** & never leave.
- **AWS Managed** or **Customer Managed CMKs**
- Customer managed key are more configurable.
- CMKs support **rotation**.
- CMK itself contains the current **backing key**, so the physical material that's used to encrypt and decrypt, as well as previous keys caused by rotating that material.
- you can create **Alias**, which is essentially a shortcut to a particular CMK.

Key policy and Security

The starting point for KMS security is the key policy, which is a type of resource policy. It's like a bucket policy on S3 bucket, only a key policy is on a key. Every customer master key has a key policy.

For CMKs you can adjust that policy.

Now there are two components to server-side encryption the first is the actual encryption and decryption process itself.

So, taking plaintext, a key, an algorithm and generating ciphertext and the reverse which is taking that ciphertext and the key and using the algorithm to produce the plaintext.

so, one half of the process that server-side encryption can do is the actual encryption operation and the second part is generation and the management of the cryptographic keys.

Now these 3 methods, handle each of these differently.

Server-Side Encryption with Amazon S3-Managed Keys (SSE-S3)

When you use Server-Side Encryption with Amazon S3-Managed Keys (SSE-S3), each object is encrypted with a unique key. As an additional safeguard, it encrypts the key itself with a master key that it regularly rotates. Amazon S3 server-side encryption uses one of the strongest block ciphers available, 256-bit Advanced Encryption Standard (AES-256), to encrypt your data.

Server-Side Encryption with Customer Master Keys (CMKs) Stored in AWS Key Management Service (SSE-KMS)

Server-Side Encryption with Customer Master Keys (CMKs) Stored in AWS Key Management Service (SSE-KMS) is similar to SSE-S3, but with some additional benefits and charges for using this service. There are separate permissions for the use of a CMK that provides added protection against unauthorized access of your objects in Amazon S3. SSE-KMS also provides you with an audit trail that shows when your CMK was used and by whom. Additionally, you can create and manage customer managed CMKs or use AWS managed CMKs that are unique to you, your service, and your Region.

Server-Side Encryption with Customer-Provided Keys (SSE-C)

With Server-Side Encryption with Customer-Provided Keys (SSE-C), you manage the encryption keys and Amazon S3 manages the encryption, as it writes to disks, and decryption, when you access your objects

Object Encryption

- Encryption in S3 is simple.
- Buckets are not encrypted you define encryption at an object level and each object inside a bucket could be using different encryption settings.
- S3 is capable of supporting two main encryption methods
- Clint side encryption
- Server-side encryption both of these refer to encryption at rest.
- There are three types of server-side encryption available for S3 objects.
- Server-side Encryption with Customer-provided keys (SSE-C)
- Server-side Encryption with Amazon S3-managed keys (SSE-S3)
- Server-side Encryption with Customer Master keys (CMKs) stored in AWS key Management Service (SSE-KMS)

Method	Key Management	Encryption Processing	Extras
Client-Side	YOU	YOU	
SSE-C	YOU	S3	
SSE-S3	S3	S3	
SSE-KMS	S3 & KMS	S3	Rotation Control Role Separation

S3 Object Storage Classes

Amazon S3 offers a range of storage classes designed for different use cases. These include **S3 Standard** for general-purpose storage of frequently accessed data; S3 Intelligent-Tiering for data with unknown or changing access patterns; **S3 Standard-Infrequent Access (S3 Standard-IA)** and **S3 One Zone-Infrequent Access (S3 One Zone-IA)** for long-lived, but less frequently accessed data; and Amazon S3 Glacier (S3 Glacier) and Amazon S3 Glacier Deep Archive (S3 Glacier Deep Archive) for long-term archive and digital preservation. If you have data residency requirements that can't be met by an existing AWS Region, you can use the S3 Outposts storage class to store your S3 data on-premises. Amazon S3 also offers capabilities to manage your data throughout its lifecycle. Once an S3 Lifecycle policy is set, your data will automatically transfer to a different storage class without any changes to your application.

Amazon S3 Standard (S3 Standard): S3 Standard offers high durability, availability, and performance object storage for frequently accessed data. Because it delivers low latency and high throughput, S3 Standard is appropriate for a wide variety of use cases, including cloud applications, dynamic websites, content distribution, mobile and gaming applications, and big data analytics.

Key features:

- Low latency and high throughput performance
- Designed for durability of 99.999999999% of objects across multiple Availability Zones
- Resilient against events that impact an entire Availability Zone
- Designed for 99.99% availability over a given year
- Backed with the Amazon S3 Service Level Agreement for availability
- Supports SSL for data in transit and encryption of data at rest
- S3 Lifecycle management for automatic migration of objects to other S3 Storage Classes

Amazon S3 Intelligent-Tiering (S3 Intelligent-Tiering): Amazon S3 Intelligent-Tiering (S3 Intelligent-Tiering) is the only cloud storage class that delivers automatic cost savings by moving objects between four access tiers when access patterns change.

Key features:

- **Automatically optimizes storage** costs for data with changing access patterns
- Stores objects in four access tiers, optimized for frequent, infrequent, archive, and deep archive access
- Frequent and Infrequent Access tiers have same low latency and high throughput performance of S3 Standard
- Activate optional automatic archive capabilities for objects that become rarely accessed
- Archive access and deep Archive access tiers have same performance as Glacier and Glacier Deep Archive
- Designed for **durability** of **99.999999999%** of objects across multiple Availability Zones
- Designed for **99.9% availability** over a given year
- Backed with the Amazon S3 Service Level Agreement for availability
- Small monthly monitoring and auto-tiering fee
- No operational overhead, no retrieval fees, no additional tiering fees apply when objects are moved between access tiers within the S3 Intelligent-Tiering storage class.

Amazon S3 Standard-Infrequent Access (S3 Standard-IA): S3 Standard-IA is for data that is accessed less frequently, but requires rapid access when needed. S3 Standard-IA offers the high durability, high throughput, and low latency of S3 Standard, with a low per GB storage price and per GB retrieval fee. This combination of low cost and high performance make S3 Standard-IA ideal for long-term storage, backups, and as a data store for disaster recovery files. S3 Storage Classes can be configured at the object level and a single bucket can contain objects stored across S3 Standard, S3 Intelligent-Tiering, S3 Standard-IA, and S3 One Zone-IA. You can also use S3 Lifecycle policies to automatically transition objects between storage classes without any application changes.

Key Features:

- Same low latency and high throughput performance of S3 Standard
- Designed for **durability of 99.999999999%** of objects across multiple Availability Zones
- Resilient against events that impact an entire Availability Zone
- Data is **resilient** in the event of one entire Availability Zone destruction
- Designed for **99.9% availability** over a given year
- Backed with the Amazon S3 Service Level Agreement for availability
- Supports SSL for data in transit and encryption of data at rest
- S3 Lifecycle management for automatic migration of objects to other S3 Storage Classes

Amazon S3 One Zone-Infrequent Access (S3 One Zone-IA): S3 One Zone-IA is for data that is accessed less frequently, but requires rapid access when needed. Unlike other S3 Storage Classes which store data in a minimum of three Availability Zones (AZs), S3 One Zone-IA stores data in a single AZ and costs 20% less than S3 Standard-IA. S3 One Zone-IA is ideal for customers who want a lower-cost option for infrequently accessed data but do not require the availability and resilience of S3 Standard or S3 Standard-IA. It's a good choice for storing secondary backup copies of on-premises data or easily re-creatable data. You can also use it as cost-effective storage for data that is replicated from another AWS Region using S3 Cross-Region Replication.

Key Features:

- Same low latency and high throughput performance of S3 Standard
- Designed for **durability of 99.999999999%** of objects in a single Availability Zone†
- Designed for **99.5% availability** over a given year
- Backed with the Amazon S3 **Service Level Agreement** for availability
- Supports SSL for data in transit and encryption of data at rest
- S3 Lifecycle management for automatic migration of objects to other S3 Storage Classes

- † Because S3 One Zone-IA stores data in a single AWS Availability Zone, data stored in this storage class will be lost in the event of Availability Zone destruction.

Amazon S3 Glacier (S3 Glacier): 3 Glacier is a secure, durable, and low-cost storage class for data archiving. You can reliably store any amount of data at costs that are competitive with or cheaper than on-premises solutions. To keep costs low yet suitable for varying needs, S3 Glacier provides three retrieval options that range from a few minutes to hours. You can upload objects directly to S3 Glacier, or use S3 Lifecycle policies to transfer data between any of the S3 Storage Classes for active data (S3 Standard, S3 Intelligent-Tiering, S3 Standard-IA, and S3 One Zone-IA) and S3 Glacier.

Key Features:

- Designed for **durability of 99.999999999%** of objects across multiple Availability Zones
- Data is resilient in the event of one entire Availability Zone destruction
- Supports SSL for data in transit and encryption of data at rest
- Low-cost design is ideal for long-term archive
- Configurable retrieval times, from minutes to hours
- S3 PUT API for direct uploads to S3 Glacier, and S3 Lifecycle management for automatic migration of objects.

Amazon S3 Glacier Deep Archive (S3 Glacier Deep Archive): S3 Glacier Deep Archive is Amazon S3's lowest-cost storage class and supports long-term retention and digital preservation for data that may be accessed once or twice in a year. It is designed for customers — particularly those in highly-regulated industries, such as the Financial Services, Healthcare, and Public Sectors — that retain data sets for 7-10 years or longer to meet regulatory compliance requirements. S3 Glacier Deep Archive can also be used for backup and disaster recovery use cases, and is a cost-effective and easy-to-manage alternative to magnetic tape systems, whether they are on-premises libraries or off-premises services.

Key Features:

- Designed for **durability of 99.999999999%** of objects across multiple Availability Zones
- Lowest cost storage class designed for long-term retention of data that will be retained for 7-10 years
- Ideal alternative to magnetic tape libraries
- **Retrieval** time within **12 hours**
- S3 PUT API for direct uploads to S3 Glacier Deep Archive, and S3 Lifecycle management for automatic migration of objects

S3 Outposts storage class: Amazon S3 on Outposts delivers object storage to your on-premises AWS Outposts environment. Using the S3 APIs and features available in AWS Regions today, S3 on Outposts makes it easy to store and retrieve data on your Outpost, as well as secure the data, control access, tag, and report on it. S3 on Outposts provides a single Amazon S3 storage class, named S3 Outposts, which uses the S3 APIs, and is designed to durably and redundantly store data across multiple devices and servers on your Outposts. S3 Outposts storage class is ideal for workloads with local data residency requirements, and to satisfy demanding performance needs by keeping data close to on-premises applications.

Key Features:

- S3 Object compatibility and bucket management through the S3 SDK
- Designed to durably and redundantly store data on your Outposts
- Encryption using **SSE-S3** and **SSE-C**
- Authentication and authorization using **IAM**, and **S3 Access Points**
- Transfer data to AWS Regions using **AWS DataSync**
- S3 Lifecycle expiration actions

S3 Lifecycle Configuration

To manage your objects so that they are stored cost effectively throughout their lifecycle, configure their Amazon S3 Lifecycle. An S3 Lifecycle configuration is a set of rules that define actions that Amazon S3 applies to a group of objects. There are two types of actions:

Transition actions—Define when objects transition to another Using Amazon S3 storage classes. For example, you might choose to transition objects to the S3 Standard-IA storage class 30 days after you created them, or archive objects to the S3 Glacier storage class one year after creating them.

Expiration actions—Define when objects expire. Amazon S3 deletes expired objects on your behalf. The lifecycle expiration costs depend on when you choose to expire objects. There are costs associated with the lifecycle transition requests.

Managing object lifecycle

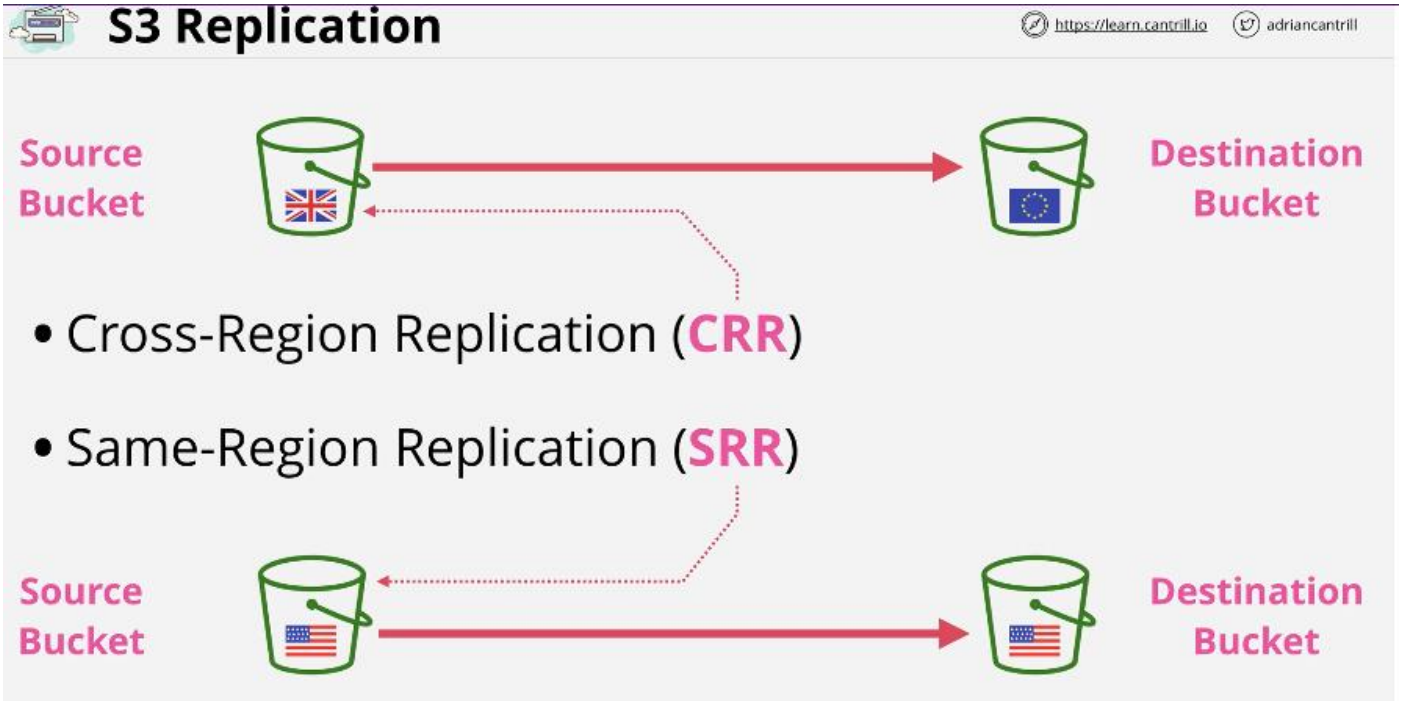
Define S3 Lifecycle configuration rules for objects that have a well-defined lifecycle. For example:

- If you upload periodic logs to a bucket, your application might need them for a week or a month. After that, you might want to delete them.
- Some documents are frequently accessed for a limited period of time. After that, they are infrequently accessed. At some point, you might not need real-time access to them, but your organization or regulations might require you to archive them for a specific period. After that, you can delete them.
- You might upload some types of data to Amazon S3 primarily for archival purposes. For example, you might archive digital media, financial and healthcare records, raw genomics sequence data, long-term database backups, and data that must be retained for regulatory compliance.

With S3 Lifecycle configuration rules, you can tell Amazon S3 to transition objects to less expensive storage classes, or archive or delete them.

S3 Replication

- S3 replication, a feature which allows you to configure the replication of objects between a source and destination S3 bucket.
- There are two types of replications supported by S3.
- the first type, which has been available for some time is Cross-Region Replication or CRR, and that allow the replication of objects from a source bucket to a destination bucket in different AWS regions.
- The Second types of replications is Same-Region Replication or SRR, which as the name suggests, in the same process where both the source and destination buckets ate in the same AWS region.



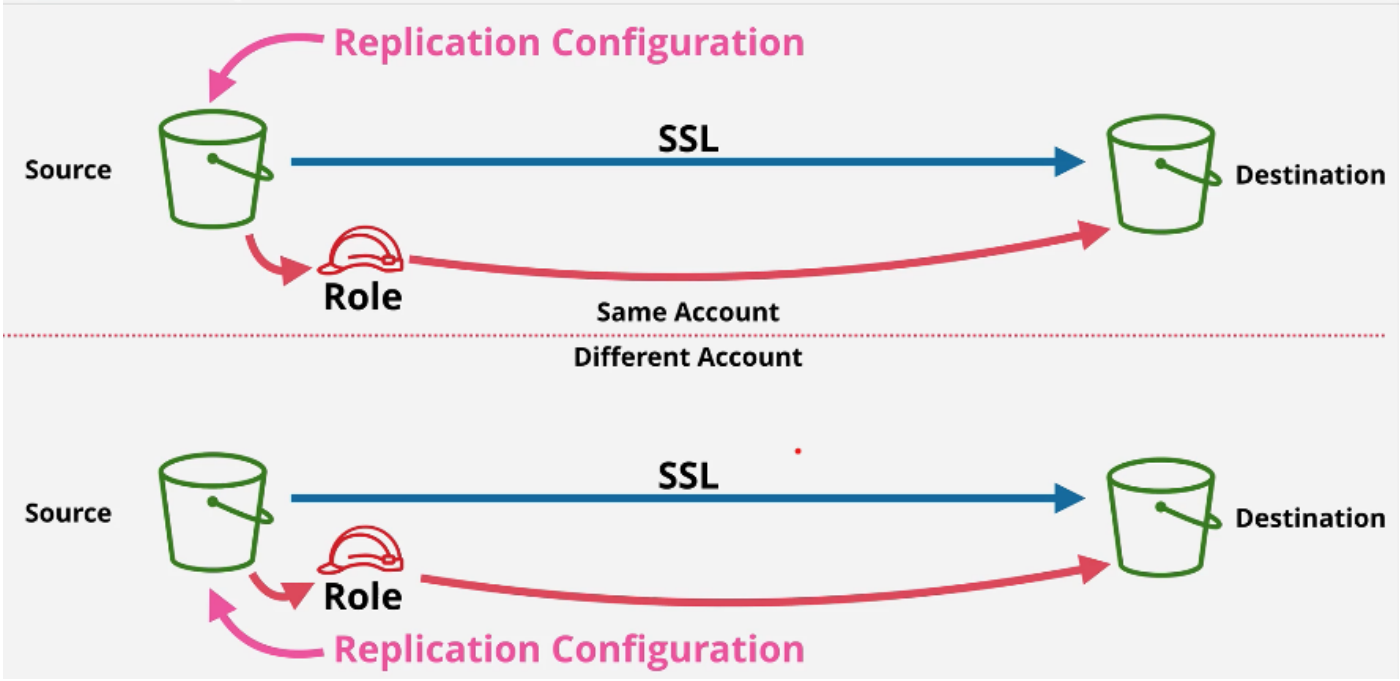
Replication configuration is applied to the source bucket. The replication configuration, configures S3 to replicate from this source bucket to a destination bucket.

Another thing that's configured in the replication configuration is an IAM role to use for the replication process.

The role is configured to allow the S3 service to assume it so that's defined in its trust policy.

The roles permission policy gives it the permission to read objects on the source bucket and permissions to replicate those objects to the destination bucket.

This is how replication is configured between source and destination buckets, and of course that replication is encrypted.



There is one crucial difference between replication which occurs in the same AWS account versus different AWS account. inside one account, both S3 buckets are owned by the same AWS account so they both trust the same areas account that they are in.

That means they both trust IAM as a service, which means that they both trust the IAM role. for the same account that means that IAM role automatically has access to the source and destination buckets, as long as the roles permission policy grant that access. if your configuring replication between different AWS account though, that's not enough. the destination bucket, because it is in a different AWS account, doesn't trust the source account or the role that's used to replicate that bucket contents.

S3 replication considerations

- Replication is not retroactive. you enable replication on a pair of buckets a source and a destination.
- If you enable replication on a bucket which already has objects, those objects will not be replicated.
- In order to enable replication on a bucket, both the source and destination bucket need to have versioning enabled.
- It is one-way replication process only objects are replicated from the source to the destination.
- It is capable of handling objects which are encrypted using SSE-S3 and SSE-KMS, but this an extra piece of configuration that you will need to enable. (Extra permissions required because KMS is involved)
- Replication also requires the owner of the source bucket needs permissions on the objects which will replicate.
- Another limitation is it will not replicate system events. So, if any changes are made in the source bucket by Lifecycle Management, they will not be replicated to the destination bucket, so only user events are replicated.
- in Addition to that, it can't replicate any objects inside a bucket that are using the Glacier or Glacier Deep Archie Storage classes.
- DELETE are not replicated. if you perform any DELETES in the source bucket, they are not replicated to the destination.

Why use replication?

- Same-region replications specifically, you might this process for Log Aggregation. So, if you have got multiple different S3 buckets, which store logs for different systems, then you could use this to aggregate those logs into a single S3 bucket.
- You might want to use Same-Region Replication to configure some sort of synchronization between Production and Test accounts. maybe you want to replicate data from PROD to TEST periodically, or maybe you want to replicate some testing data into your PROD account.
- Same-region Replication to implement resilience if you have strict sovereignty requirements.
- If you don’t have sovereignty requirements, then you can use Cross-region Replication and use replication to implement global resilience improvements, so you can have backups of your data copied to different AWS regions, to cope with large scale failure, you can also replicate data into different regions to reduce latency.

S3 PreSigned URLs

- PreSigned URLs are a way that you can give another person or application access to an object inside an S3 bucket using your credentials in a safe and secure way.
- All objects by default are private. Only the object owner has permission to access these objects. However, the object owner can optionally share objects with others by creating a presigned URL, using their own security credentials, to grant time-limited permission to download the objects.
- Anyone with valid security credentials can create a presigned URL. However, in order to successfully access an object, the presigned URL must be created by someone who has permission to perform the operation that the presigned URL is based upon.
- The credentials that you can use to create a presigned URL include:
    - IAM instance profile: Valid up to 6 hours
    - AWS Security Token Service: Valid up to 36 hours when signed with permanent credentials, such as the credentials of the AWS account root user or an IAM user
    - IAM user: Valid up to 7 days when using AWS Signature Version 4
    - To create a presigned URL that's valid for up to 7 days, first designate IAM user credentials (the access key and secret access key) to the SDK that you're using. Then, generate a presigned URL using AWS Signature Version 4.
- If you created a presigned URL using a temporary token, then the URL expires when the token expires, even if the URL was created with a later expiration time.
- Since presigned URLs grant access to your Amazon S3 buckets to whoever has the URL, we recommend that you protect them appropriately. For more details about protecting presigned URLs, see Limiting presigned URL capabilities.

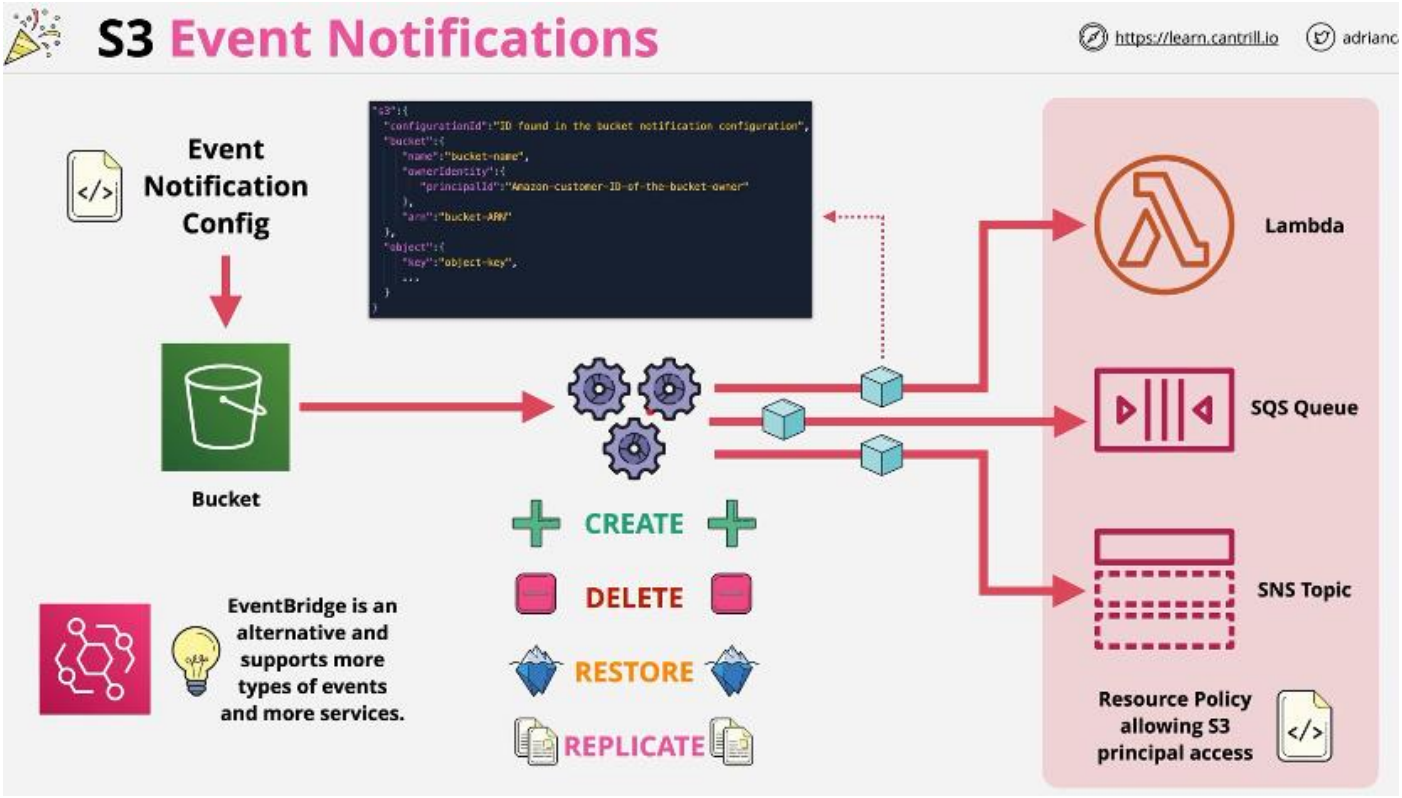
S3 Select and Glacier Select

- **S3** and **Glacier Select** allow you to use a **SQL-Like statement** to retrieve partial objects from S3 and Glacier.
- S3 can store **HUGE** objects (up to **5TB**)
- You often want to retrieve the entire object
- Retrieving a 5TB object will takes time, uses 5TB
- Filtering at the client side doesn't reduce this
- S3/ Glacier select let you use SQL-like statements.
- to select part of the object, pre-filtered by S3.
- S3 Select and Glacier Select allow you to use number of file format such as CSV, JSON, Parquet, BZIP2 compression for CSV and JSON.

S3 Events

S3 event notification

- event notification of S3 allows you to create event notification configurations on a bucket.
- When enabled, a notification is generated when a certain thing occurs within a bucket and these can be delivered to different destinations including SNS topics, SQS queues, or Lambda functions.
- various different types of events are supported. for example, you can generate event notifications when objects are created, which means Put, Post, Copy, and when long MultiparUpload operations complete.
- You can also set event notification to trigger on object deletion so you can match any type of deletion using the wild card.
- You can also have it trigger for object restores. So, if you have objects in S3 glacier or glacier deep archive and you perform a restore operation, you can be notified when it starts and completes.
- You can get notifications relating to replication.



S3 Access Logs

- Server access logging provides detailed records for the requests that are made to a bucket. Server access logs are useful for many applications. For example, access log information can be useful in security and access audits. It can also help you learn about your customer base and understand your Amazon S3 bill.
- By default, Amazon S3 doesn't collect server access logs. When you enable logging, Amazon S3 delivers access logs for a source bucket to a target bucket that you choose. The target bucket must be in the same AWS Region as the source bucket and must not have a default retention period configuration.
- An access log record contains details about the requests that are made to a bucket. This information can include the request type, the resources that are specified in the request, and the time and date that the request was processed.
- **Important**: There is no extra charge for enabling server access logging on an Amazon S3 bucket. However, any log files that the system delivers to you will accrue the usual charges for storage. (You can delete the log files at any time.) We do not assess data transfer charges for log file delivery, but we do charge the normal data transfer rate for accessing the log files.
- You can enable or disable server access logging by using the Amazon S3 console, Amazon S3 API, the AWS Command Line Interface (AWS CLI), or AWS SDKs.
- Before you enable server access logging, consider the following:
  - In Amazon S3, you can grant permission to deliver access logs through bucket access control lists (ACLs), but not through bucket policy.
  - Adding deny conditions to a bucket policy might prevent Amazon S3 from delivering access logs.
  - You can use default bucket encryption on the target bucket only if AES256 (SSE-S3) is selected. SSE-KMS encryption is not supported.
  - You can't enable S3 Object Lock on the target bucket.

MODULE 6 - VIRTUAL PRIVATE CLOUD (VPC) BASICS

Network Refresh

- An IP address (internet protocol address) is a numerical representation that uniquely identifies a specific interface on the network.
- Addresses in IPv4 are 32-bits long. This allows for a maximum of 4,294,967,296 (232) unique addresses. Addresses in IPv6 are 128-bits, which allows for 3.4 x 1038 (2128) unique addresses.
- The total usable address pool of both versions is reduced by various reserved addresses and other considerations.
- IP addresses are binary numbers but are typically expressed in decimal form (IPv4) or hexadecimal form (IPv6) to make reading and using them easier for humans.
- The Internet Protocol (IP) is part of the Internet layer of the Internet protocol suite. In the OSI model, IP would be considered part of the network layer. IP is traditionally used in conjunction with a higher-level protocol, most notably TCP. The IP standard is governed by RFC 791.

Subnet masks

- single IP address identifies both a network, and a unique interface on that network. A subnet mask can also be written in dotted decimal notation and determines where the network part of an IP address ends, and the host portion of the address begins.
- When expressed in binary, any bit set to one means the corresponding bit in the IP address is part of the network address. All the bits set to zero mark the corresponding bits in the IP address as part of the host address.
- The bits marking the subnet mask must be consecutive ones. Most subnet masks start with 255. and continue on until the network mask ends.

Private addresses

Within the address space, certain networks are reserved for private networks. Packets from these networks are not routed across the public internet. This provides a way for private networks to use internal IP addresses without interfering with other networks. The private networks are

- 10.0.0.1 - 10.255.255.255
- 172.16.0.0 - 172.31.255.255
- 192.168.0.0 - 192.168.255.255

Special addresses

Certain IPv4 addresses are set aside for specific uses:

- 127.0.0.0 Loopback address (the host’s own interface)
- 224.0.0.0 IP Multicast
- 255.255.255.255 Broadcast (sent to all interfaces on network)

CIDR (Classless Inter-Domain Routing) -- CIDR (Classless Inter-Domain Routing) also known as supernetting -- is a method of assigning Internet Protocol (IP) addresses that improves the efficiency of address distribution and replaces the previous system based on Class A, Class B and Class C networks. The initial goal of CIDR was to slow the increase of routing tables on routers across the internet and decrease the rapid exhaustion of IPv4 addresses. As a result, the number of available internet addresses has greatly increased.

IP Header Classes:

Class	Address Range	Subnet masking	Example IP	Leading bits	Max number of networks	Application
IP Class A	1 to 126	255.0.0.0	1.1.1.1	8	128	Used for large number of hosts.
IP Class B	128 to 197	255.255.0.0	128.1.1.1	16	16384	Used for medium size network.
IP Class C	192 to 223	255.255.255.0	192.1.11.	24	2097157	Used for local area network.
IP Class D	224 to 239	NA	NA	NA	NA	Reserve for multi-tasking.
IP Class E	240 to 254	NA	NA	NA	NA	This class is reserved for research and Development Purposes.

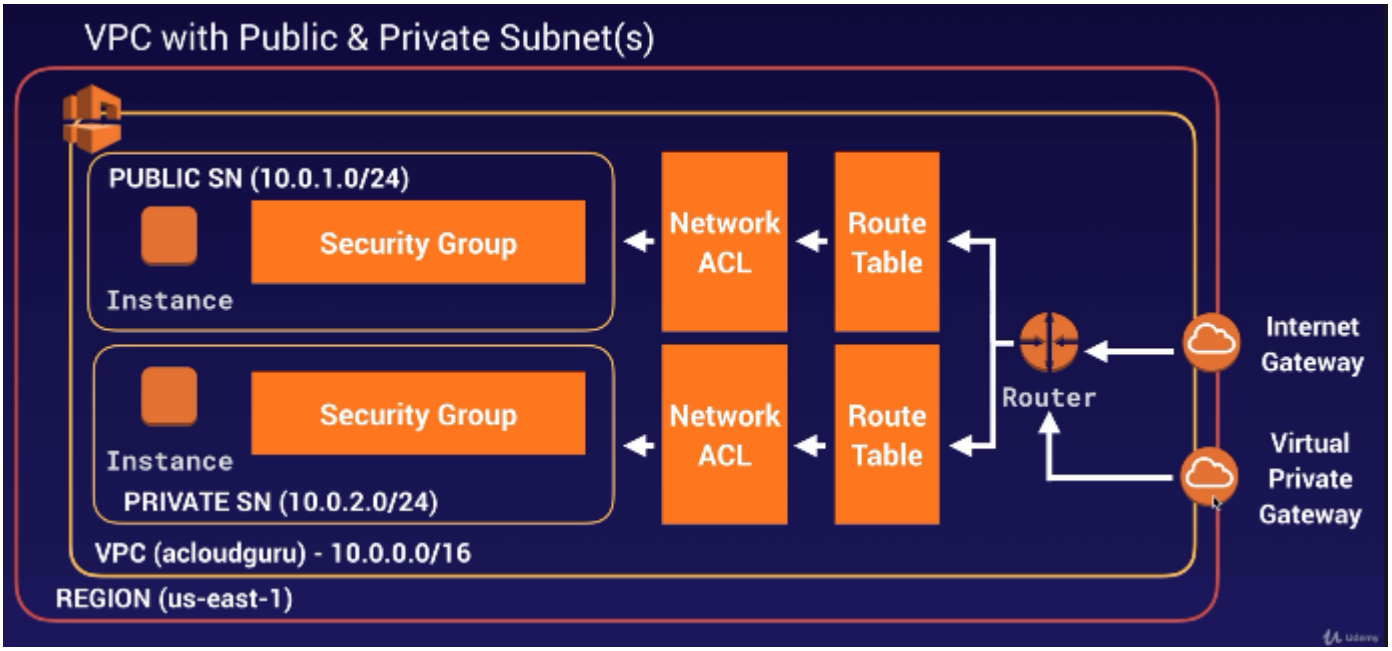
Amazon Virtual Private Cloud VPC

Amazon Virtual Private Cloud (Amazon VPC) enables you to launch AWS resources into a virtual network that you've defined. This virtual network closely resembles a traditional network that you'd operate in your own data center, with the benefits of using the scalable infrastructure of AWS.

Amazon VPC is the networking layer for Amazon EC2.

The following are the key concepts for VPCs:

- **Virtual private cloud (VPC) —** A virtual network dedicated to your AWS account.
- **Subnet —** A range of IP addresses in your VPC.
- **Route table —** A set of rules, called routes, that are used to determine where network traffic is directed.
- **Internet gateway —** A gateway that you attach to your VPC to enable communication between resources in your VPC and the internet.
- **VPC endpoint —** Enables you to privately connect your VPC to supported AWS services and VPC endpoint services powered by PrivateLink without requiring an internet gateway, NAT device, VPN connection, or AWS Direct Connect connection. Instances in your VPC do not require public IP addresses to communicate with resources in the service. Traffic between your VPC and the other service does not leave the Amazon network.
- **CIDR block —**Classless Inter-Domain Routing. An internet protocol address allocation and route aggregation methodology.



VPC considerations

- What size should the VPC be?
- Are there any networks we can't use?
- VPCs, Cloud, on-premises, Partners & Vendors
- Try to predict the future
- VPC structure- Tiers & Resiliency (Available) zones

Access Amazon VPC

You can create, access, and manage your VPCs using any of the following interfaces:

- **AWS Management Console** — Provides a web interface that you can use to access your VPCs.
- **AWS Command Line Interface (AWS CLI)** — Provides commands for a broad set of AWS services, including Amazon VPC, and is supported on Windows, Mac, and Linux.
- **AWS SDKs** — Provides language-specific APIs and takes care of many of the connection details, such as calculating signatures, handling request retries, and error handling.
- **Query API** — Provides low-level API actions that you call using HTTPS requests. Using the Query API is the most direct way to access Amazon VPC, but it requires that your application handle low-level details such as generating the hash to sign the request, and error handling.

VPCs and subnets

- A virtual private cloud (VPC) is a virtual network dedicated to your AWS account. It is logically isolated from other virtual networks in the AWS Cloud. You can launch your AWS resources, such as Amazon EC2 instances, into your VPC. You can specify an IP address range for the VPC, add subnets, associate security groups, and configure route tables.
- A subnet is a range of IP addresses in your VPC. You can launch AWS resources into a specified subnet. Use a public subnet for resources that must be connected to the internet, and a private subnet for resources that won't be connected to the internet.
- To protect the AWS resources in each subnet, you can use multiple layers of security, including security groups and network access control lists (ACL).
- You can optionally associate an IPv6 CIDR block with your VPC, and assign IPv6 addresses to the instances in your VPC.

What can we do with a VPC?

- Launch instances into a subnet of your choice
- Assign custom IP address ranges in each subnet
- Configure router tables between subnets
- Create internet gateway and attach it to our VPC
- Much better security control over your AWS resources
- Instance Security groups
- Subnet network access network list (ACLs)

Default VPC vs Custom VPC

- Default VPC is user friendly, allowing you to immediately deploy instances
- All Subnets in default VPC have a router out to the internet.
- Each EC2 instance has both a public and private IP address

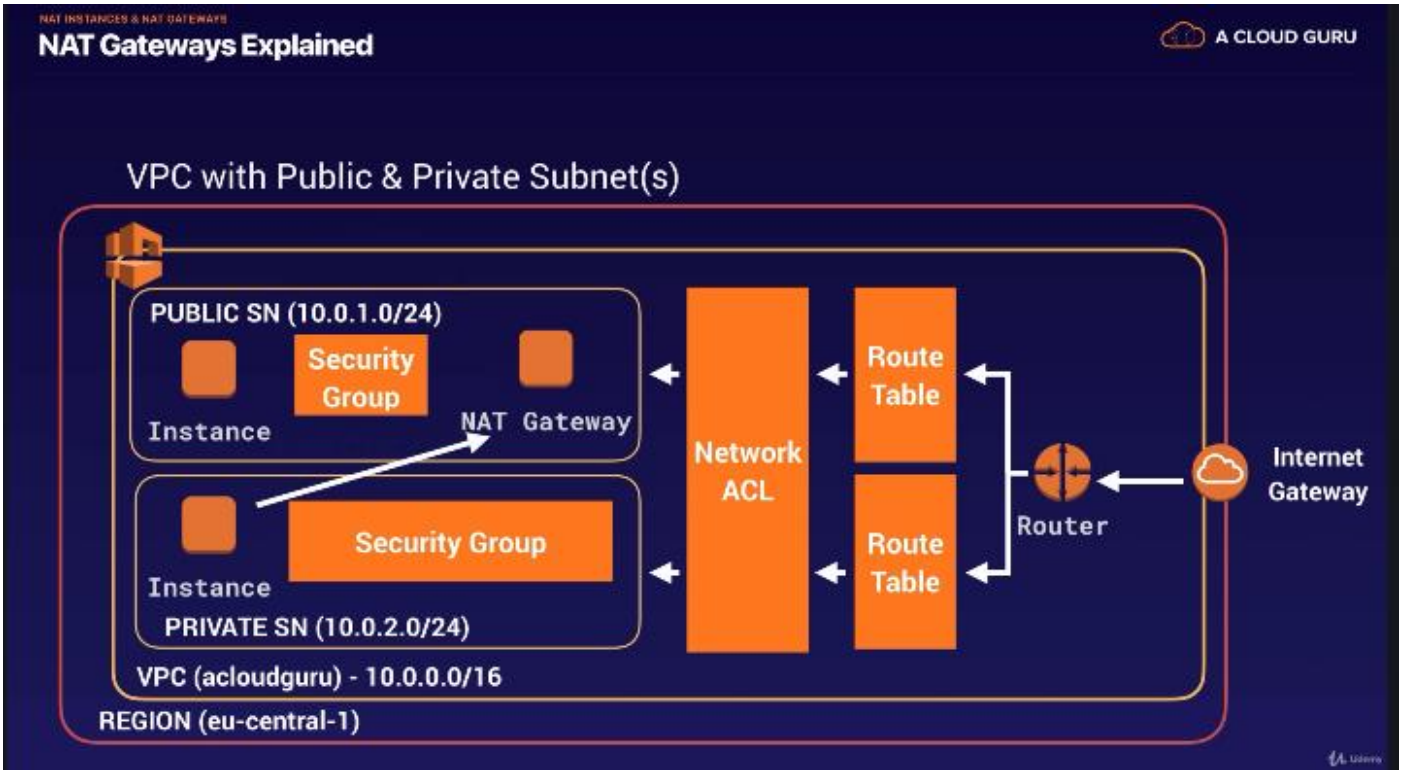
VPC Peering

- Allows you to connect one VPC with another via a direct network route using private IP addresses.
- Instances behave as if they were on the same private network
- You can peer VPCs with other AWS accounts as well as with other VPCs in the same account.
- Peering is in a star configuration: i.e., 1 central VPC peers with 4 others. NO TRANSITIVE PEERING

Network Address Translation (NAT)

You can now use Network Address Translation (NAT) Gateway, a highly available AWS managed service that makes it easy to connect to the Internet from instances within a private subnet in an AWS Virtual Private Cloud (VPC). Previously, you needed to launch a NAT instance to enable NAT for instances in a private subnet.

Amazon VPC NAT Gateway is available in the US East (N. Virginia), US West (Oregon), US West (N. California), EU (Ireland), Asia Pacific (Tokyo), Asia Pacific (Singapore), and Asia Pacific (Sydney) regions.



Remember the following:

- When creating a NAT instance, Disable Source/Destination Check on the instances.
- NAT instances must be in a public subnet.
- There must be a route out of the private subnets to the NAT instance, in order for this to work.
- The amount of traffic that NAT instances can support depends on the instance size. if you are bottlenecking, increase the instance size.
- You can create high availability using Autoscaling Groups, multiple subnets in different AZs, and a script to automate failover.
- Behind a Security Group.
- Redundant inside the Availability Zone
- Preferred by the enterprise.
- No need to patch
- Not associated with Security groups
- Automatically assigned as public IP address
- Remember to update your route tables.

Security groups (SGs)

A security group acts as a virtual firewall for your EC2 instances to control incoming and outgoing traffic. Inbound rules control the incoming traffic to your instance, and outbound rules control the outgoing traffic from your instance. When you launch an instance, you can specify one or more security groups.

- It supports only allow rules, and by default, all the rules are denied. You cannot deny the rule for establishing a connection.
- It is a stateful means that any changes made in the inbound rule will be automatically reflected in the outbound rule. For example, if you are allowing an incoming port 80, then you also have to add the outbound rule explicitly.
- It is associated with an EC2 instance.
- All the rules are evaluated before deciding whether to allow the traffic.
- Security Group is applied to an instance only when you specify a security group while launching an instance.
- It is the first layer of defense.
- Security Groups (SGs) are another security feature of AWS VPC ... only unlike NACLs they are attached to AWS resources, not VPC subnets.
- SGs offer a few advantages vs NACLs in that they can recognize AWS resources and filter based on them, they can reference other SGs and also themselves.
- But, SGs are **not capable of explicitly blocking traffic** - so often require assistance from NACLs

Network Access control Lists (NACL)

A network access control list (NACL) is an optional layer of security for your VPC that acts as a firewall for controlling traffic in and out of one or more subnets. You might set up network ACLs with rules similar to your security groups in order to add an additional layer of security to your VPC.

- Your VPC automatically comes with a default network ACL, and by default it allows all outbound and inbound traffic.
- You can create custom network ACLs. By default, each custom network ACL denies all inbound and outbound traffic until you add rules.
- Each subnet in you VPC must be associated with a network ACL. if you don't explicitly associate a subnet with a network ACL, the subnet is automatically associated with the default network ACL.
- Block IP Addresses using network ACLs not Security Groups.

- You can associate a network ACL with multiple subnets; however, a **subnet can be associated with only one network ACL at a time.** When you associate a network ACL with a subnet, the previous association is removed.
- network ACLs contain a numbered list of rules that is evaluated in order, starting with the lowest numbered rule.
- network ACLs have separate inbound and outbound rules, and each rule can either allow or deny traffic.
- network ACLs are stateless; responses to allowed inbound traffic and subject to the rules for outbound traffic (and vice versa).
- Network Access Control Lists (NACLs) are a type of security filter (like firewalls) which can filter traffic as it enters or leaves a subnet.
- NACLs are attached to subnets and only filter data as it **crosses the subnet boundary**.
- NACLs are stateless and so see initiation and response phases of a connection and 1 inbound and 1 outbound stream requiring two roles (one IN one OUT)

Internet Gateways (IG)

- An internet gateway is a horizontally scaled, redundant, and highly available VPC component that allows communication between your VPC and the internet.
- An internet gateway serves two purposes: to provide a target in your VPC route tables for internet-routable traffic, and to perform network address translation (NAT) for instances that have been assigned public IPv4 addresses.
- An internet gateway supports IPv4 and IPv6 traffic. It does not cause availability risks or bandwidth constraints on your network traffic. There's no additional charge for having an internet gateway in your account.
- If a subnet is associated with a route table that has a route to an internet gateway, it's known as a public subnet.
- If a subnet is associated with a route table that does not have a route to an internet gateway, it's known as a private subnet.
- When you add a new subnet to your VPC, you must set up the routing and security that you want for the subnet.
- Internet Gateway is **region resilient** attached to VPC.
- There's a one-to-one relationship between internet gateways and the VPC. A VPC can have no internet gateways which makes it entirely private, or it can have one internet gateway, those are the two choices. And IGW can be created and not attached to VPC, so it can have zero attachment, but it can only ever be attached to one VPC at a time at which point it's valid in all AZs that the VPC uses.

  1 VPC = 0 or 1 IGW, 1 IGW = 0 or 1 VPC
- Runs from within the AWS public zone.
- Gateways traffic between the VPC and the internet or AWS public zone.
- Managed by - AWS handles Performance

VPC Routing

- A route table contains a set of rules, called routes, that are used to determine where network traffic from your subnet or gateway is directed.
- Every VPC has a VPC Router - its Highly available
- The Router has a network interface in every subnet in your VPC, the network plus (Network+1) one address of the subnet.
- Routes traffic between subnets.
- Controlled by 'route tables' each subnet has one
- A VPC has a Main route table subnet default.

The following are the key concepts for route tables.

Main route table—The route table that automatically comes with your VPC. It controls the routing for all subnets that are not explicitly associated with any other route table.

Custom route table—A route table that you create for your VPC.

Edge association—A route table that you use to route inbound VPC traffic to an appliance. You associate a route table with the internet gateway or virtual private gateway, and specify the network interface of your appliance as the target for VPC traffic.

Route table association—The association between a route table and a subnet, internet gateway, or virtual private gateway.

Subnet route table—A route table that's associated with a subnet.

Gateway route table—A route table that's associated with an internet gateway or virtual private gateway.

Local gateway route table—A route table that's associated with an Outposts local gateway.

Destination—The range of IP addresses where you want traffic to go (destination CIDR). For example, an external corporate network with a 172.16.0.0/12 CIDR.

Propagation—Route propagation allows a virtual private gateway to automatically propagate routes to the route tables. This means that you don't need to manually enter VPN routes to your route tables.

Target—The gateway, network interface, or connection through which to send the destination traffic; for example, an internet gateway.

Local route—A default route for communication within the VPC.

Bastion Host

A bastion host is a special purpose computer on a network specifically designed and configured to withstand attacks. The computer generally hosts a single application, for example a proxy server, and all other services are removed or limited to reduce the threat to the computer. It is hardened in this manner primarily due to its location and purpose, which is either on the outside of a firewall or in a demilitarized zone (DMZ) and usually involved access from untrusted networks or computer.

- A NAT Gateway or NAT Instance is used to provide internet traffic to EC2 instance in a private subnet.
- A Bastion is used to securely administer EC2 Instances (Using SSH or RDP). Bastions are called Jump Boxes in Australia.
- You cannot use a NAT Gateway as a Bastion host.

VPC Summary

- Think of a VPC as a logical datacenter in AWS.
- Consists of IGWs (Or Virtual Private Gateways), Route Tables, Network Access Lists, Subnets and Security Groups
- 1 Subnet = 1 Availability Zone
- Security Groups are Stateful; Network Access Control Lists are Stateless
- NO TRANSITIVE PEERING
- When you create a VPC a default Route Table, Network Access Control List (NACL) and default Security Group.
- It won’t create any subnets, nor will it create a default Internet gateway.
- US-East-1A in your AWS account can be completely different AZ to US-East-1A in another AWS account. The AZs are randomized.

- Amazon always reserve 5IP addresses within your subnets.
- You can only have 1 Internet Gateway per VPC
- Security Groups can't span VPCs.
- You need a minimum of two public subnets to deploy an internet facing load balancer.

MODULE 7- ELASTIC COMPUTE CLOUD (EC2) BASICS

Virtualization 101

EC2 provides virtualization as a service. It's an infrastructure as a service or IS product.

Virtualization is the process of running more than one operating system on a piece of physical hardware, a server.

EC2 Architecture and Resilience

EC2 Architecture

- EC2 instances are virtual machines (OS + resources)
- EC2 instances run on EC2 Hosts
- EC2 provides Shared Hosts or Dedicated Hosts
- Hosts = 1AZ - AZ Fails, Host Fails, Instance Fail

What's EC2 Good for?

- Traditional OS + Application Compute
- Long Running Compute
- Server style application
- burst or steady-state load
- Monolithic application stacks
- Migrated application workloads or disaster recovery

EC2 Instance Types

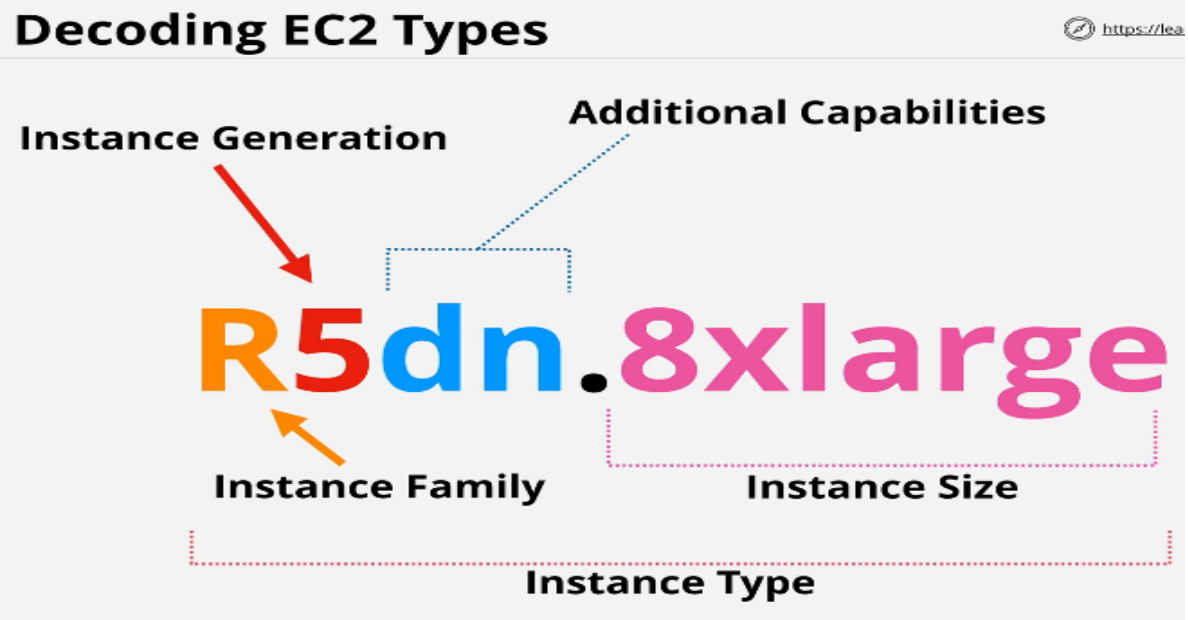
- At a high level, when you choose an EC2 instance type, you are doing so to influence a few different things.
- First, logically, the raw amount of resources that you get. like CPU, Memory, Local Storage Capacity & Type.
- Resources Ratio.
- Storage and Data Network Bandwidth
- System Architecture / Vendor
- Additional Features and Capabilities.

EC2 Categories

- **General Purpose** - Default - Diverse workload, equal resource ratio.
- **Compute Optimized** - Media Processing, HPC, Scientific Modelling, gaming, Machine Learning.
- **Memory Optimized** - Processing large in-memory datasets, some database workloads.
- **Accelerated Computing** - Hardware GPU, field programmable gate arrays (FPCAs).
- **Storage Optimized** - Sequential and Random IO - scale-out transactional databases, data warehousing, Elastic search, analytics workloads.

Decoding EC2 Types

- R5dn.8xlarge this is known as instance type.
- The letter at the start is the instance family. there are lots of example of this. The T family, the M family, the I family and the R family. there's lot more, but each of these are designed for a specific type or types of computing.
- The next part is the generation. So, the number 5, in this case is the generation.
- Generally, with AWS always select the most recent generation. it always provides the best price-to-performance option.
- 8xlarge or eight extra-large is the instance size.



EC2 Instance type

EC2 Instance Types

https://learn.cantrill.io adriancantrill

Categories	Type	Details / Notes
General Purpose	A1, M6g	Graviton (A1) Graviton 2 (M6g) ARM based processors. Efficient.
	T3, T3a	Burst Pool - Cheaper assuming nominal low levels of usage, with occasional Peaks.
	M5, M5a, M5n	Steady state workload alternative to T3/3a - Intel / AMD Architecture
Compute Optimized	C5, C5n	Media encoding, Scientific Modelling, Gaming Servers, General Machine learning
Memory Optimized	R5, R5a	Real time analytics, in-memory caches, certain DB applications (in-memory operations)
	X1, X1e	Large scale in-memory applications .. lowest $ per GiB memory in AWS
	High Memory (u-Xtb1)	Highest memory of all AWS instances
	z1d	Large memory and CPU - with directly attached NVMe
Accelerated Computing	P3	GPU instances (Tesla v100 GPUs) - parallel processing & machine learning
	G4	GPU Instances (NVIDIA T4 Tensor) - Machine learning inference and Graphics Intensive
	F1	Field Programmable Gate Arrays (FPGA) - Genomics, Financial Analysis, Big Data
	Inf1	Machine Learning - recommendation, forecasting, analysis, voice, conversation
Storage Optimized	I3/I3en	Local high performance SSD (NVMe) - NoSQL Databases, warehousing, analytics
	D2	Dense Storage (HDD) - data warehousing, HADOOP, Distributed File Systems, Data processing - lowest price disk throughput
	H1	High Throughput, balance CPU/Mem. HDFS, MAPR-FS, File systems, Apache Kafka, Big data

Storage Refresher

key terms

- **Direct** (local) attached Storage - Storage on the EC2 Host.
- **Network** attached Storage - Volumes delivered over the network (EBS)
- **Ephemeral** Storage - Temporary Storage
- **Persistent** Storage - permanent storage -lives on past the lifetime of the instance.
- **Block** Storage - Volume presented to the OS as a collection of blocks...no structure provided. it Mountable and Bootable.
- **File** Storage - Presented as a file share. it has a structure. Mountable. NOT Bootable.
- **Object** Storage - Collection of objects, flat. Not mountable. Not bootable.

EBS Volume Types & Storage performance

Amazon EBS provides a range of volume types that are divided into two major categories: SSD-backed storage volumes and HDD-backed storage volumes. SSD-backed storage volumes offer great price/performance characteristics for random small block workloads, such as transactional applications, whereas HDD-backed storage volumes offer the best price/performance characteristics for large block sequential workloads. You can attach and stripe data across multiple volumes of any type to increase the I/O performance available to your Amazon EC2 applications. The following table presents the storage characteristics of the current generation volume types.

General Purpose SSD (gp2) volumes offer cost-effective storage that is ideal for a broad range of workloads. These volumes deliver single-digit millisecond latencies, the ability to burst to 3,000 IOPS for extended periods of time, and a baseline performance of 3 IOPS/GiB up to a maximum of 10,000 IOPS (at 3,334 GiB). The gp2 volumes can range in size from 1 GiB to 16 TiB. These volumes have a throughput limit range of 128 MiB/second for volumes less than or equal to 170 GiB; for volumes over 170 GiB, this limit increases at the rate of 768 KiB/second per GiB to a maximum of 160 MiB/second (at 214 GiB and larger). You can see the percentage of I/O credits remaining in the burst buckets for gp2 volumes by viewing the BurstBalance metric in Amazon CloudWatch.

Provisioned IOPS SSD (io1) volumes are designed to deliver predictable high performance for I/O-intensive workloads with small I/O size where the dominant performance attribute is IOPS, such as database workloads that are sensitive to storage performance and consistency in random access I/O throughput. You specify an IOPS rate when creating a volume, and then Amazon EBS delivers within 10 percent of the provisioned IOPS performance 99.9 percent of the time over a given year, when attached to an EBS-optimized instance. The io1 volumes can range in size from 4 GiB to 16 TiB, and you can provision up to 20,000 IOPS per volume. The ratio of IOPS provisioned to the volume size requested can be a maximum of 50. For example, a volume with 5,000 IOPS must be at least 100 GB in size.

Throughput Optimized HDD (st1) volumes are ideal for frequently accessed, throughput-intensive workloads with large datasets and large I/O sizes, where the dominant performance attribute is throughput (MiB/s), such as streaming workloads, big data, data warehouse, log processing, and ETL workloads. These volumes deliver performance in terms of throughput, measured in MiB/s, and include the ability to burst up to 250MiB/s per TiB, with a baseline throughput of 40MiB/s per TiB and a maximum throughput of 500MiB/s per volume. The st1 volumes are designed to deliver the expected throughput performance 99 percent of the time and has enough I/O credits to support a full-volume scan at the burst rate. The st1 volumes can’t be used as boot volumes. You can see the throughput credits remaining in the burst bucket for st1 volumes by viewing the BurstBalance metric in Amazon CloudWatch.

Cold HDD (sc1) volumes provide the lowest cost per GiB of all EBS volume types. These are ideal for infrequently accessed workloads with large, cold datasets with large I/O sizes, where the dominant

performance attribute is throughput (MiB/s). Similarly, to st1, sc1 volumes provide a burst model and can burst up to 80 MiB/s per TiB, with a baseline throughput of 12 MiB/s per TiB and a maximum throughput of 250 MB/s per volume. The sc1 volumes are designed to deliver the expected throughput performance 99 percent of the time and have enough I/O credits to support a full-volume scan at the burst rate. The sc1 volumes can't be used as boot volumes. You can see the throughput credits remaining in the burst bucket for sc1 volumes by viewing the BurstBalance metric in CloudWatch.

Elastic Block Store (EBS) Service

Amazon Elastic Block Store (Amazon EBS) provides block level storage volumes for use with EC2 instances. EBS volumes behave like raw, unformatted block devices. You can mount these volumes as devices on your instances. EBS volumes that are attached to an instance are exposed as storage volumes that persist independently from the life of the instance. You can create a file system on top of these volumes, or use them in any way you would use a block device (such as a hard drive).

- Block Storage - raw disk allocation (volume) - can be encrypted using KMS.
- Instance see block device and create file system, on this device (ext3/4, xfs)
- Storage is provided in ONE AZ (Resilient in that AZ)
- Attached to *one EC2 instance (or other service) over a storage network.
- detached and reattached, not lifecycle linked to one instance...persistent.
- Snapshot (backup) into S3. Create volume from snapshot (migrate between AZs)
- Different physical storage types, different sizes, different performance profiles.
- Billed based on GB-month (and in some cases performance)
- Termination Protection is turned off by default, you must turn it on.
- On an EBS-backed instance, the default action is for the root EBS volume to be deleted when the instance is terminated.
- EBS Root Volumes of your DEFAULT AMI's CAN be encrypted. you can also use a third-party tool (such as bit locker etc) to encrypt the root volume, or this can be done when creating AMI's in the AWS console or using the API.
- Additional volumes can be encrypted.

Instance Store Volumes

An instance store provides temporary block-level storage for your instance. This storage is located on disks that are physically attached to the host computer. Instance store is ideal for temporary storage of information that changes frequently, such as buffers, caches, scratch data, and other temporary content, or for data that is replicated across a fleet of instances, such as a load-balanced pool of web servers.

An instance store consists of one or more instance store volumes exposed as block devices. The size of an instance store as well as the number of devices available varies by instance type.

The virtual devices for instance store volumes are ephemeral [0-23]. Instance types that support one instance store volume have ephemeral0. Instance types that support two instance store volumes have ephemeral0 and ephemeral1, and so on.

- Block Storage Devices
- Physically connected to one EC2 Host
- Instances on that host can access them
- Highest storage performance in AWS
- Add at launch ONLY
- Lost on instance move, resize or hardware failure.
- You pay for it anyway - include in instance price
- TEMPRPRY

Choosing between the EC2 Instance Store and EBS

Instance Store

- Instance store backed instance is an EC2 instance using an Instance store as root device volume created from a template stored in S3.
- An instance store is ephemeral storage that provides temporary block level storage for your instance. Instance store is ideal for temporary storage like buffers, caches, and other temporary content.
- Instance store volumes accesses storage from disks that are physically attached to the host computer.
- When an Instance stored instance is launched, the image that is used to boot the instance is copied to the root volume (typically sda1).
- Instance store provides temporary block-level storage for instances.
- Data on an instance store volume persists only during the life of the associated instance; if an instance is stopped or terminated, any data on instance store volumes is lost.

Key points for Instance store backed Instance

- Boot time is slower then EBS backed volumes and usually less than 5 min
- Can be selected as Root Volume and attached as additional volumes
- Instance store backed Instances can be of maximum 10GiB volume size
- Instance store volume can be attached as additional volumes only when the instance is being launched and cannot be attached once the Instance is up and running
- The data in an instance store persists only during the lifetime of its associated instance. If an instance reboots (intentionally or unintentionally), data in the instance store persists
- Instance store backed Instances cannot be stopped, as when stopped and started AWS does not guarantee the instance would be launched in the same host and hence the data is lost
- AMI creation requires usage on AMI tools and needs to be executed from within the running instance
- Instance store backed Instances cannot be upgraded
- For EC2 instance store-backed instances AWS recommends to:
    1. Distribute the data on the instance stores across multiple AZs
    2. Back up critical data from the instance store volumes to persistent storage on a regular basis
- Data on Instance store volume is LOST in following scenarios:
    1. Underlying disk drive fails
    2. Instance stops
    3. Instance terminates
    4. Instance hibernates

Therefore, do not rely on instance store for valuable, long-term data.

Amazon Elastic Block Store (EBS)

- An “EBS-backed” instance means that the root device for an instance launched from the AMI is an EBS volume created from an EBS snapshot
- An EBS volume behaves like a raw, unformatted, external block device that can be attached to a single instance and are not physically attached to the Instance host computer (more like a network attached storage).
- Volume persists independently from the running life of an instance. After an EBS volume is attached to an instance, you can use it like any other physical hard drive.
- EBS volume can be detached from one instance and attached to another instance.
- EBS volumes can be created as encrypted volumes using the EBS encryption feature.

- EBS is block store which is separately attached to EC2. Also, its design such a way that it will be replicated within its availability zone so it provides high availability and durability.
- And the additional advantage of it is, you can have back-ups for EBS by creating Snapshots which is not possible instance store. So that whenever you want to retrieve the data you can just create the EBS volume from the snapshot.

Key points for EBS backed Instance

- Boot time is very fast usually less than a min.
- Can be selected as Root Volume and attached as additional volumes.
- EBS backed Instances can be of maximum 16TiB volume size depending upon the OS.
- EBS volume can be attached as additional volumes when the Instance is launched and even when the Instance is up and running.
- When EBS-backed instance is in a stopped state, various instance– and volume-related tasks can be done for e.g., you can modify the properties of the instance, you can change the size of your instance or update the kernel it is using, or you can attach your root volume to a different running instance for debugging or any other purpose.
- EBS volumes are AZ scoped and tied to a single AZ in which created.
- EBS volumes are automatically replicated within that zone to prevent data loss due to failure of any single hardware component.
- AMI creation is easy using a Single command.
- EBS backed Instances can be upgraded for instance type, Kernel, RAM disk and user data.
- Data on the EBS volume is LOST:
    1. For EBS Root volume, if delete on termination flag is enabled (enabled, by default).
    2. For attached EBS volumes, if the Delete on termination flag is enabled (disabled, by default).
- Data on EBS volume is NOT LOST in following scenarios:
    1. Reboot on the Instance.
    2. Stopping an EBS-backed instance.
    3. Termination of the Instance for the additional EBS volumes. Additional EBS volumes are detached with their data intact.

Snapshots, Restore & Fast Snapshot Restore (FSR)

Amazon EBS Snapshots provide a simple and secure data protection solution that is designed to protect your block storage data such as EBS volumes, boot volumes, as well as on-premises block data. EBS Snapshots are a point in time copy of your data, and can be used to enable disaster recovery, migrate data across regions and accounts, and improve backup compliance.

EBS Snapshots

- EBS Snapshots are backups of data consumed within EBS Volumes - Stored on S3.
- Snapshots are incremental, the first being a full back up - and any future snapshots being incremental.
- Snapshots can be used to migrate data to different availability zones in a region, or to different regions of AWS.
- Snapshots exist on S3. Think of snapshots as a photograph of the disk.
- Snapshots are **point in time** copies of Volumes.
- **Snapshots are incremental** - this means that only the blocks that have changed since your last Snapshot are moved to S3.
- if this is your first Snapshots, it may take some time to create.
- To create a snapshot for Amazon EBS volumes that serve as root devices, you should stop the instance before taking the snapshot.

- However, you can take snap while the instance is running.
- You can create AMI's from both Volumes and Snapshots.
- You can change EBS volume sizes on the fly, including changing the EC2 size and storage type.
- Volumes will ALWAYS be in the same availability zones as the EC2 instance.

Migrating EBS

- To move an EC2 volume from one AZ to another, take a snapshot of it, create an AMI from the snapshot and then use the AMI to launch the EC2 instance in a new AZ.
- To move an EC2 volume from one region to another, take a snapshot of it, create an AMI from the snapshot and then copy the AMI from one region to the other. Then use the copied AMI to launch the new EC2 instance in the new region.

Amazon EBS fast snapshot restore

- Amazon EBS fast snapshot restore enables you to create a volume from a snapshot that is fully initialized at creation. This eliminates the latency of I/O operations on a block when it is accessed for the first time. Volumes that are created using fast snapshot restore instantly deliver all of their provisioned performance.
- To get started, enable fast snapshot restore for specific snapshots in specific Availability Zones. Each snapshot and Availability Zone pair refers to one fast snapshot restore. When you create a volume from one of these snapshots in one of its enabled Availability Zones, the volume is restored using fast snapshot restore.
- You can enable fast snapshot restore for snapshots that you own and for public and private snapshots that are shared with you.

Fast snapshot restores quotas

- **You can enable up to 50 snapshots for fast snapshot restore per Region**. The quota applies to snapshots that you own and snapshots that are shared with you. If you enable fast snapshot restore for a snapshot that is shared with you, it counts towards your fast snapshot restore quota. It does not count towards the snapshot owner's fast snapshot restore quota.

Fast snapshot restores states

After you enable fast snapshot restore for a snapshot, it can be in one of the following states.

enabling — A request was made to enable fast snapshot restore.

optimizing — Fast snapshot restore is being enabled. It takes 60 minutes per TiB to optimize a snapshot. Snapshots in this state offer some performance benefit when restoring volumes.

enabled — Fast snapshot restore is enabled. Snapshots in this state offer the full performance benefit when restoring volumes.

disabling — A request was made to disable fast snapshot restore, or a request to enable fast snapshot restore failed.

disabled — Fast snapshot restore is disabled. You can enable fast snapshot restore again as needed.

EBS Encryption

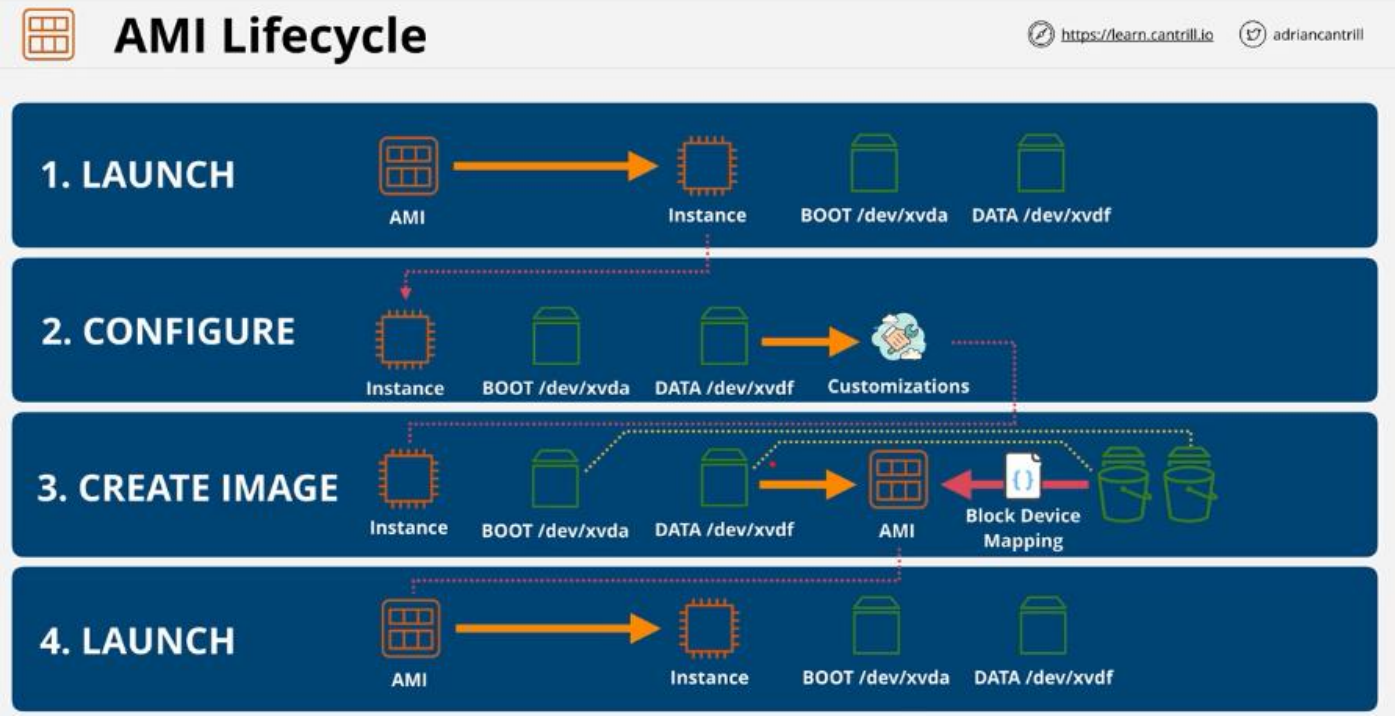
- Snapshots of encrypted volumes are encrypted automatically.
- Volumes restored from encrypted snapshots are encrypted automatically.
- You can share snapshots, but only if they are unencrypted.
- These snapshots can be shared with other AWS accounts or made public,

- Accounts can be set to encrypted by default - default CMK
- Otherwise choose a CMK to use.
- each volume uses 1 unique DEK
- Snapshots & future volumes use the same DEK
- Can’t change a volume to NOT be encrypted
- OS isn’t aware of the encryption... no performance loss
- Root Device Volumes can now be encrypted. if you have an unencrypted root device volume that needs to be encrypted do the following:
    1. Create a Snapshot of the unencrypted root device volume.
    2. Create a copy of Snapshot and select the encrypt option
    3. Create an AMI from the encrypted Snapshot.
    4. Use that AMI to launch new encrypted instances.

Amazon Machine Images (AMI)

Amazon Machine Images (AMI) 's are the images which can create EC2 instances of a certain configuration.

In addition to using AMI's to launch instances, you can customize an EC2 instance to your bespoke business requirements and then generate a template AMI which can be used to create any number of customized EC2 instances.



AMI Exam Tips

- AMI = **One Region**, only works in that one region.
- **AMI Baking** … Creating an AMI from a configured instance + application
- An AMI **can’t be edited** … launch instance, update configuration and make a new AMI
- Can be copies **between regions** (include its snapshots)
- Remember permissions **Default = your account**

Instance Billing Models

Instance pricing Models

On-Demand Instances: With On-Demand instances, you pay for compute capacity by the hour or the second depending on which instances you run. No longer-term commitments or upfront payments are needed. You can increase or decrease your compute capacity depending on the demands of your application and only pay the specified per hourly rates for the instance you use.

On-Demand instances are recommended for:

- Users that prefer the low cost and flexibility of Amazon EC2 without any up-front payment or long-term commitment
- Applications with short-term, spiky, or unpredictable workloads that cannot be interrupted
- Applications being developed or tested on Amazon EC2 for the first time

Spot Instances: Amazon EC2 Spot instances allow you to request spare Amazon EC2 computing capacity for up to 90% off the On-Demand price.

Spot instances are recommended for:

- Applications that have flexible start and end times
- Applications that are only feasible at very low compute prices
- Users with urgent computing needs for large amounts of additional capacity

Reserved Instances: Reserved Instances provide you with a significant discount (up to 75%) compared to On-Demand instance pricing. In addition, when Reserved Instances are assigned to a specific Availability Zone, they provide a capacity reservation, giving you additional confidence in your ability to launch instances when you need them. For applications that have steady state or predictable usage, Reserved Instances can provide significant savings compared to using On-Demand instances.

Reserved Instances are recommended for:

- Applications with steady state usage
- Applications that may require reserved capacity
- Customers that can commit to using EC2 over a 1- or 3-year term to reduce their total computing costs

Dedicated Hosts: A Dedicated Host is a physical EC2 server dedicated for your use. Dedicated Hosts can help you reduce costs by allowing you to use your existing server-bound software licenses, including Windows Server, SQL Server, and SUSE Linux Enterprise Server (subject to your license terms), and can also help you meet compliance requirements.

- Can be purchased On-Demand (hourly).
- Can be purchased as a Reservation for up to 70% off the On-Demand price.

Instance Status Checks & Auto Recovery

- With instance status monitoring, you can quickly determine whether Amazon EC2 has detected any problems that might prevent your instances from running applications. Amazon EC2 performs automated checks on every running EC2 instance to identify hardware and software issues. You can view the results of these status checks to identify specific and detectable problems.
- The event status data augments the information that Amazon EC2 already provides about the state of each instance (such as pending, running, stopping) and the utilization metrics that Amazon CloudWatch monitors (CPU utilization, network traffic, and disk activity).
- Status checks are performed every minute, returning a pass or a fail status. If all checks pass, the overall status of the instance is **OK**. If one or more checks fail, the overall status is **impaired**. Status checks are built into Amazon EC2, so they cannot be disabled or deleted.
- When a status check fails, the corresponding CloudWatch metric for status checks is incremented. For more information. You can use these metrics to create CloudWatch alarms that are triggered based on the result of the status checks. For example, you can create an alarm to warn you if status checks fail on a specific instance.

Types of status checks

There are two types of status checks: system status checks and instance status checks.

- **System status checks:** System status checks monitor the AWS systems on which your instance runs. These checks detect underlying problems with your instance that require AWS involvement to repair. When a system status check fails, you can choose to wait for AWS to fix the issue, or you can resolve it yourself. For instances backed by Amazon EBS, you can stop and start the instance yourself, which in most cases results in the instance being migrated to a new host. For Linux instances backed by instance store, you can terminate and replace the instance. For Windows instances, the root volume must be an Amazon EBS volume; instance store is not supported for the root volume. Note that instance store volumes are ephemeral and all data is lost when the instance is stopped.
  The following are examples of problems that can cause system status checks to fail:
  - Loss of network connectivity
  - Loss of system power
  - Software issues on the physical host
  - Hardware issues on the physical host that impact network reachability
- **Instance status checks:** Instance status checks monitor the software and network configuration of your individual instance. Amazon EC2 checks the health of the instance by sending an address resolution protocol (ARP) request to the network interface (NIC). These checks detect problems that require your involvement to repair. When an instance status check fails, you typically must address the problem yourself (for example, by rebooting the instance or by making instance configuration changes).

  The following are examples of problems that can cause instance status checks to fail:

  - Failed system status checks
  - Incorrect networking or start-up configuration
  - Exhausted memory
  - Corrupted file system
  - Incompatible kernel

Recover your instance:

If your instance fails a system status check, you can use CloudWatch alarm actions to automatically recover your instance. The recover option is available for over 90% of deployed Amazon EC2 instances. However, the recover option works only for system check failures, not for instance status check failures. In addition, if you terminate your instance, then it can't be recovered.

If your instance fails a status check, you might need to reboot the instance or change the configuration.

Requirements

The recover action is supported only on instances with the following characteristics:

- Uses one of the following instance types: A1, C3, C4, C5, C5a, C5n, C6g, C6gn, Inf1, M3, M4, M5, M5a, M5n, M5zn, M6g, P3, P4, R3, R4, R5, R5a, R5b, R5n, R6g, T2, T3, T3a, T4g, high memory (virtualized only), X1, X1e
- Runs in a virtual private cloud (VPC)
- Uses default or dedicated instance tenancy
- Has only EBS volumes (do not configure instance store volumes)

To create an alarm to recover an instance (Amazon EC2 console)

1. Open the Amazon EC2 console at https://console.AWS.amazon.com/ec2/
2. In the navigation pane, choose **Instances**.
3. Select the instance and choose **Actions**, **Monitor and troubleshoot**, **Manage CloudWatch alarms**.

   Alternatively, you can choose the plus sign (+) in the **Alarm status** column.
4. On the **Manage CloudWatch alarms** page, do the following:
   a. Choose **Create an alarm**.
   b. To receive an email when the alarm is triggered, for **Alarm notification**, choose an existing Amazon SNS topic. You first need to create an Amazon SNS topic using the Amazon SNS console.

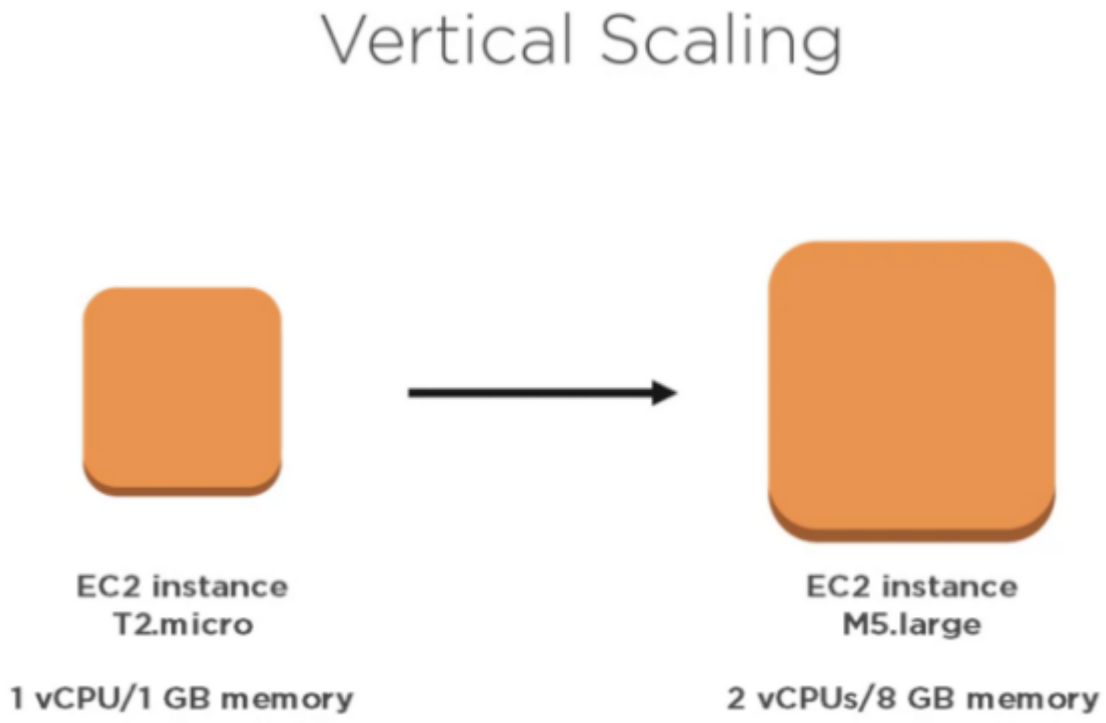
      Note: Users must subscribe to the specified SNS topic to receive email notifications when the alarm is triggered. The AWS account root user always receives email notifications when automatic instance recovery actions occur, even if an SNS topic is not specified or the root user is not subscribed to the specified SNS topic.

   c. Toggle on **Alarm action**, and choose **Recover**.
   d. For **Group samples by** and **Type of data to sample**, choose a statistic and a metric. In this example, choose **Average** and **Status check failed: system**.
   e. For **Consecutive period** and **Period**, specify the evaluation period for the alarm. In this example, enter **2** consecutive periods of **5 Minutes**.
   f. Amazon CloudWatch automatically creates an alarm name for you. To change the name, for **Alarm name**, enter a new name. Alarm names must contain only ASCII characters.
   g. Choose **Create**.

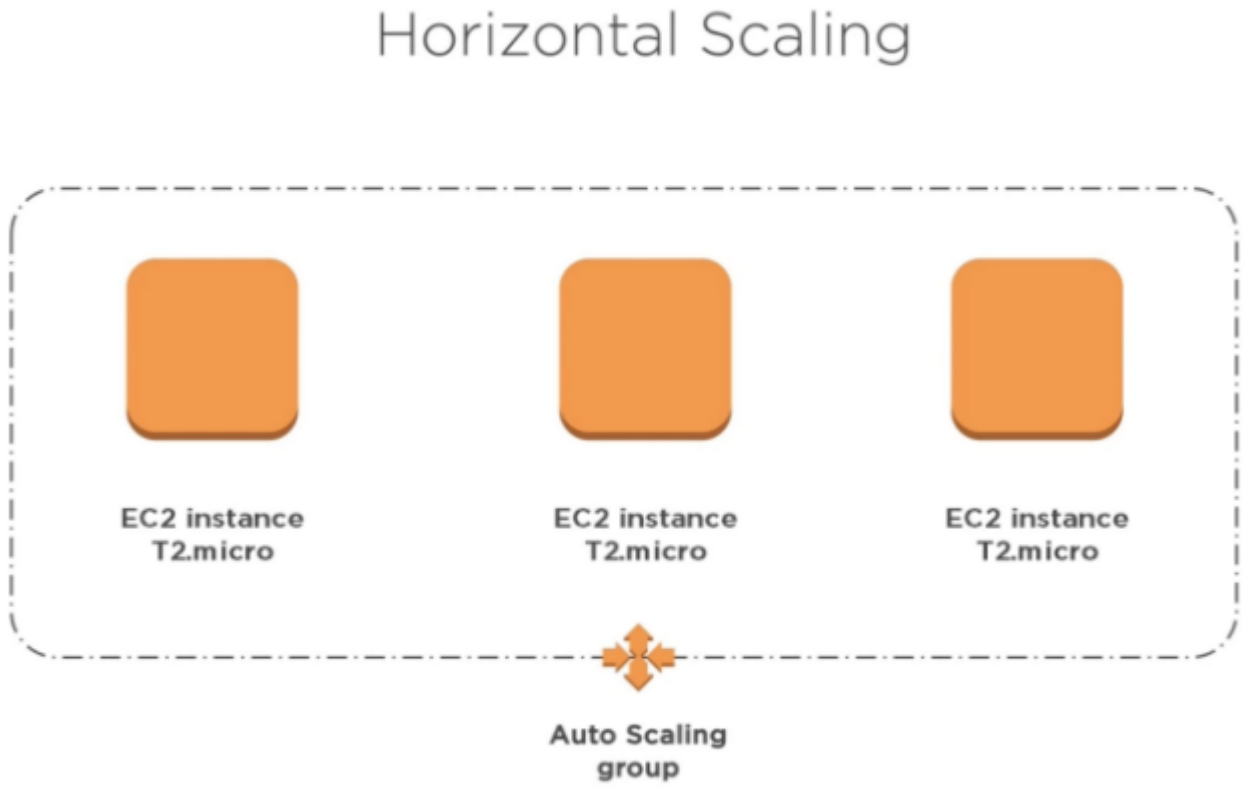
Horizontal & Vertical Scaling

Horizontal scaling means that you scale by adding more ec2 machines into your pool of resources whereas Vertical scaling means that you scale by adding more power (CPU, RAM) to an existing ec2 machine.

Vertical Scaling usually means upgrade of server hardware.



Horizontal Scaling is adding more ec2 machines.



Instance Metadata

- Instance metadata is data about your instance that you can use to configure or manage the running instance. Instance metadata is divided into categories, for example, host name, events, and security groups.
- You can also use instance metadata to access user data that you specified when launching your instance. For example, you can specify parameters for configuring your instance, or include a simple script. You can build generic AMIs and use user data to modify the configuration files supplied at launch time. For example, if you run web servers for various small businesses, they can all use the same generic AMI and retrieve their content from the Amazon S3 bucket that you specify in the user data at launch. To add a new customer at any time, create a bucket for the customer, add their content, and launch your AMI with the unique bucket name provided to your code in the user data.
- If you launch more than one instance at the same time, the user data is available to all instances in that reservation. Each instance that is part of the same reservation has a unique ami-launch-index number, allowing you to write code that controls what to do. For example, the first host might elect itself as the original node in a cluster.
- EC2 instances can also include dynamic data, such as an instance identity document that is generated when the instance is launched.

Important

Although you can only access instance metadata and user data from within the instance itself, the data is not protected by authentication or cryptographic methods. Anyone who has direct access to the instance, and potentially any software running on the instance, can view its metadata. Therefore, you should not store sensitive data, such as passwords or long-lived encryption keys, as user data.

MODULE 8 - CONTAINERS & ECS

Introduction to Containers

Containers are an operating system virtualization technology used to package applications and their dependencies and run them in isolated environments. They provide a lightweight method of packaging and deploying applications in a standardized way across many different types of infrastructure.

- Containers are created from container images: bundles that represent the system, applications, and environment of the container.
- Container images act like templates for creating specific containers, and the same image can be used to spawn any number of running containers.

What is Docker?

Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly. With Docker, you can manage your infrastructure in the same ways you manage your applications. By taking advantage of Docker's methodologies for shipping, testing, and deploying code quickly, you can significantly reduce the delay between writing code and running it in production.

Container Key Concepts

- **Dockerfiles** are used to build images
- It Portable - self-contained, always run as expected.
- Container are lightweight - Parent OS used, fs layer is shared
- Container only runs the application & environment it needs.
- provides much of the isolation VM's do

- Ports are 'exposed' to the host and beyond.
- it's important to understand that some more complex application stacks can consist of multiple containers. you can use multiple containers in a single architecture, either to scale a specific part of the application, or when you're using multiple tiers.

ECS - Concepts

Amazon Elastic Container Service (Amazon ECS) is a fully managed container orchestration service that helps you easily deploy, manage, and scale containerized applications. It deeply integrates with the rest of the AWS platform to provide a secure and easy-to-use solution for running container workloads in the cloud and now on your infrastructure with Amazon ECS Anywhere.

Amazon ECS leverages serverless technology from AWS Fargate to deliver autonomous container operations, which reduces the time spent on configuration, patching, and security. Instead of worrying about managing the control plane, add-ons, and nodes, Amazon ECS enables you to rapidly build applications and grow your business.

The following sections dive into these individual elements of the Amazon ECS architecture in more detail.

Containers and images

To deploy applications on Amazon ECS, your application components must be architected to run in containers. A container is a standardized unit of software development that contains everything that your software application needs to run, including relevant code, runtime, system tools, and system libraries. Containers are created from a read-only template called an image.

Images are typically built from a Dockerfile, which is a plaintext file that specifies all of the components that are included in the container. After being built, these images are stored in a registry where they then can be downloaded and run on your cluster.

Task definitions

To prepare your application to run on Amazon ECS, you must create a task definition. The task definition is a text file (in JSON format) that describes one or more containers (up to a maximum of ten) that form your application. The task definition can be thought of as a blueprint for your application. It specifies various parameters for your application. For example, these parameters can be used to indicate which containers should be used, which ports should be opened for your application, and what data volumes should be used with the containers in the task. The specific parameters available for your task definition depend on the needs of your specific application.

Tasks and scheduling

A task is the instantiation of a task definition within a cluster. After you have created a task definition for your application within Amazon ECS, you can specify the number of tasks to run on your cluster.

The Amazon ECS task scheduler is responsible for placing tasks within your cluster. There are several different scheduling options available. For example, you can define a service that runs and maintains a specified number of tasks simultaneously.

Clusters

An Amazon ECS cluster is a logical grouping of tasks or services. You can register one or more Amazon EC2 instances (also referred to as container instances) with your cluster to run tasks on them. Or, you can use the serverless infrastructure that Fargate provides to run tasks. When your tasks are run on Fargate, your cluster resources are also managed by Fargate.

When you first use Amazon ECS, a default cluster is created for you. You can create additional clusters in an account to keep your resources separate.

Container agent

The container agent runs on each container instance within an Amazon ECS cluster. The agent sends information about the resource's current running tasks and resource utilization to Amazon ECS. It starts and stops tasks whenever it receives a request from Amazon ECS.

Amazon ECS can be used along with the following AWS services:

AWS Identity and Access Management: IAM (Identity and Access Management) is an access management service that helps you securely control access to AWS resources. You can use IAM to control who is authenticated (signed in) and authorized (has permissions) to view or perform specific actions on resources.

Amazon EC2 Auto Scaling: Auto Scaling is a service that enables you to automatically scale out or in your tasks based on user-defined policies, health status checks, and schedules. You can use Auto Scaling with a Fargate task within a service to scale in response to a number of metrics or with an EC2 task to scale the container instances within your cluster.

Elastic Load Balancing: The Elastic Load Balancing service automatically distributes incoming application traffic across the tasks in your Amazon ECS service. It enables you to achieve greater levels of fault tolerance in your applications, seamlessly providing the required amount of load-balancing capacity needed to distribute application traffic. You can use Elastic Load Balancing to create an endpoint that balances traffic across services in a cluster.

Amazon Elastic Container Registry: Amazon ECR is a managed AWS Docker registry service that is secure, scalable, and reliable. Amazon ECR supports private Docker repositories with resource-based permissions using IAM so that specific users or tasks can access repositories and images. Developers can use the Docker CLI to push, pull, and manage images

AWS CloudFormation: AWS CloudFormation gives developers and systems administrators an easy way to create and manage a collection of related AWS resources. More specifically, it makes resource provisioning and updating more orderly and predictable. You can define clusters, task definitions, and services as entities in an AWS CloudFormation script.

ECS - Cluster Mode

- An Amazon ECS cluster is a logical grouping of tasks or services. Your tasks and services are run on infrastructure that is registered to a cluster.
- The infrastructure capacity can be provided by AWS Fargate, which is serverless infrastructure that AWS manages, Amazon EC2 instances that you manage, or an on-premise server or virtual machine (VM) that you manage remotely.
- In most cases, Amazon ECS capacity providers can be used to manage the infrastructure the tasks in your clusters use.
- When you first use Amazon ECS, a default cluster is created for you, but you can create multiple clusters in an account to keep your resources separate.

Cluster concepts

The following are general concepts about Amazon ECS clusters.

- Clusters are Region-specific.
- The following are the possible states that a cluster can be in.
- **ACTIVE:** The cluster is ready to accept tasks and, if applicable, you can register container instances with the cluster.
- **PROVISIONING:** The cluster has capacity providers associated with it and the resources needed for the capacity provider are being created.

- **DEPROVISIONING:** The cluster has capacity providers associated with it and the resources needed for the capacity provider are being deleted.
- **FAILED:** The cluster has capacity providers associated with it and the resources needed for the capacity provider have failed to create.
- **INACTIVE:** The cluster has been deleted. Clusters with an INACTIVE status may remain discoverable in your account for a period of time. However, this behaviour is subject to change in the future, so you should not rely on INACTIVE clusters persisting.
- A cluster may contain a mix of tasks hosted on AWS Fargate, Amazon EC2 instances, or external instances.
- A cluster may contain a mix of both Auto Scaling group capacity providers and Fargate capacity providers, however when specifying a capacity provider strategy, they may only contain one or the other but not both.
- For tasks using the EC2 launch type, clusters can contain multiple different container instance types, but each container instance may only be registered to one cluster at a time.
- Custom IAM policies may be created to allow or restrict user access to specific clusters.

MODULE 9 - ADVANCED EC2

Bootstrapping EC2 using User Data

- EC2 Bootstrapping is the process of configuring an EC2 instance to perform automated install & configuration steps 'post launch' before an instance is brought into service.
- With EC2 this is accomplished by passing a script via the User Data part of the Meta-data service - which is then executed by the EC2 Instance OS.
- When you launch an Amazon ECS container instance, you have the option of passing user data to the instance.
- The data can be used to perform common automated configuration tasks and even run scripts when the instance boots.
- For Amazon ECS, the most common use cases for user data are to pass configuration information to the Docker daemon and the Amazon ECS container agent.
- You can pass multiple types of user data to Amazon EC2, including cloud boothooks, shell scripts, and cloud-init directives.

User Data Key Points

- It's opaque to EC2. it’s just a block of data
- It's NOT secure - don’t use it for passwords or long-term credentials (ideally)
- User data is limited to 16KB in size
- Can be modified when instance stopped
- But ONLY executed once

Enhanced Bootstrapping with CFN-INIT

- CFN-INIT is a powerful desired-state-like configuration engine which is part of the CFN suite of products.
- It allows you to set a state for things like packages, users, groups, sources and files within resources inside a template - and it will make that change happen on the instance, performing whatever actions are required.
- Creation policies create a 'WAIT STATE' on resources .. not allowing the resource to move to CREATE_COMPLETE until signalled using the cfn-signal tool.

EC2 Instance Roles & Profile

- Applications must sign their API requests with AWS credentials. Therefore, if you are an application developer, you need a strategy for managing credentials for your applications that run on EC2 instances.
- For example, you can securely distribute your AWS credentials to the instances, enabling the applications on those instances to use your credentials to sign requests, while protecting your credentials from other users. However, it's challenging to securely distribute credentials to each instance, especially those that AWS creates on your behalf, such as Spot Instances or instances in Auto Scaling groups.
- You must also be able to update the credentials on each instance when you rotate your AWS credentials.
- AWS designed IAM roles so that your applications can securely make API requests from your instances, without requiring you to manage the security credentials that the applications use. Instead of creating and distributing your AWS credentials, you can delegate permission to make API requests using IAM roles as follows:
    1. Create an IAM role.
    2. Define which accounts or AWS services can assume the role.
    3. Define which API actions and resources the application can use after assuming the role.
    4. Specify the role when you launch your instance, or attach the role to an existing instance.
    5. Have the application retrieve a set of temporary credentials and use them.
- You can specify permissions for IAM roles by creating a policy in JSON format. These are similar to the policies that you create for IAM users. If you change a role, the change is propagated to all instances.
- When creating IAM roles, associate least privilege IAM policies that restrict access to the specific API calls the application requires.
- You can only attach one IAM role to an instance, but you can attach the same role to many instances.
- You can apply resource-level permissions to your IAM policies to control the users' ability to attach, replace, or detach IAM roles for an instance.

Instance profiles

- Amazon EC2 uses an instance profile as a container for an IAM role. When you create an IAM role using the IAM console, the console creates an instance profile automatically and gives it the same name as the role to which it corresponds. If you use the Amazon EC2 console to launch an instance with an IAM role or to attach an IAM role to an instance, you choose the role based on a list of instance profile names.
- An instance profile can contain only one IAM role. This limit cannot be increased.
- An application on the instance retrieves the security credentials provided by the role from the instance metadata item **IAM/security-credentials/role-name.**
- The application is granted the permissions for the actions and resources that you've defined for the role through the security credentials associated with the role.
- These security credentials are temporary and we rotate them automatically. We make new credentials available at least five minutes before the expiration of the old credentials.

SSM Parameter Store

- Parameter Store, a capability of AWS Systems Manager, provides secure, hierarchical storage for configuration data management and secrets management.
- You can store data such as passwords, database strings, Amazon Machine Image (AMI) IDs, and license codes as parameter values.
- You can store values as plain text or encrypted data.
- You can reference Systems Manager parameters in your scripts, commands, SSM documents, and configuration and automation workflows by using the unique name that you specified when you created the parameter.
- Parameter Store is also integrated with Secrets Manager.
- You can retrieve Secrets Manager secrets when using other AWS services that already support references to Parameter Store parameters.

Remember the following:

- The SSM Parameter store is a service which is part of Systems Manager which allows the storage and retrieval of parameters - string, stringlist or secure string.
- The service supports encryption which integrates with KMS, versioning and can be secured using IAM.
- The service integrates natively with many AWS services - and can be accessed using the CLI/APIs from anywhere with access to the AWS Public Spare Endpoints.

System and Application Logging on EC2

- CloudWatch monitors certain performance and reliability aspects of EC2, but crucially, only those metrics that are available on the external face of an EC2 instance.
- There are situations when you need to enable monitoring inside an instance, so have access to certain performance counters of the operating system itself.
- CloudWatch is the product responsible for storing and managing metrics within AWS, and you also know that CloudWatch Logs is a subset of that product aimed at storing, managing, and visualizing any logging data.
- CloudWatch neither natively capture data inside an instance.
- It required CloudWatch Agent to do this.
- CloudWatch agent is a piece of software which runs inside an EC2 instance, so running on the OS, it captures OS-visible data and sends it into CloudWatch or CloudWatch Logs so that you can then use it and visualize it within the console of both the products
- CloudWatch Agent needs to have configuration and permission to be able to send that data into both the products.

EC2 Placement Groups

- When you launch a new EC2 instance, the EC2 service attempts to place the instance in such a way that all of your instances are spread out across underlying hardware to minimize correlated failures.
- You can use placement groups to influence the placement of a group of interdependent instances to meet the needs of your workload.
- Depending on the type of workload, you can create a placement group using one of the following placement strategies:
    1. **Cluster** – packs instances close together inside an Availability Zone. This strategy enables workloads to achieve the low-latency network performance necessary for tightly-coupled node-to-node communication that is typical of HPC applications.
    2. **Partition** – spreads your instances across logical partitions such that groups of instances in one partition do not share the underlying hardware with groups of instances in different

partitions. This strategy is typically used by large distributed and replicated workloads, such as Hadoop, Cassandra, and Kafka.

3. **Spread** – strictly places a small group of instances across distinct underlying hardware to reduce correlated failures.

Dedicated Hosts

- Dedicated hosts are EC2 Hosts which support a certain type of instance which are dedicated to your account.
- You can pay an on-demand or reserved price for the hosts and then you have no EC2 instance pricing to pay for instances running on these dedicated hosts.
- Generally dedicated hosts are used for applications which use physical core/socket licensing
- The price for a Dedicated Host varies by instance family, region, and payment option. Regardless of the quantity or the size of instances that you choose to launch on a particular Dedicated Host you only pay for each active Dedicated Host.

Enhanced Networking & EBS Optimized

EC2 optimization topics

Enhanced networking and EBS optimized instances. Both of these are important on their own.

Enhanced Networking

- Enhanced networking is a feature which is designed to improve the overall performance of EC2 networking.
- It's a feature which is required for any high-end performance features such as cluster placement groups.
- Enhanced networking uses a technique called SR-IOV, or Single Route I/O Virtualization. At a high level, it makes it so that a physical network interface inside an EC2 host is aware of virtualization.
- R-IOV is a method of device virtualization that provides higher I/O performance and lower CPU utilization when compared to traditional virtualized network interfaces.
- Enhanced networking provides higher bandwidth, higher packet per second (PPS) performance, and consistently lower inter-instance latencies. There is no additional charge for using enhanced networking.
- All current generation instance types support enhanced networking, except for T2 instances.
- No charge - available on most EC2 Types
- Higher I/O & Lower Host CPU Usage
- More Bandwidth
- Higher packets-per-second (PPS)
- Consistent lower latency

EBS Optimized

- An Amazon EBS–optimized instance uses an optimized configuration stack and provides additional, dedicated capacity for Amazon EBS I/O.
- This optimization provides the best performance for your EBS volumes by minimizing contention between Amazon EBS I/O and other traffic from your instance.
- EBS is block storage for EC2, which is delivered over the network.
- Historically, networking on EC2 instances was actually shared with being used for both data networking and EBS storage networking.
- An instance being EBS optimized means that some stack optimizations have taken place, and dedicated capacity has been provided for that instance for EBS's usage.

- It means that faster speeds are possible with EBS, and the storage side of things doesn't impact the data performance and vice versa.
- Its supported and enabled by default at no extra charge. Disabling it has no effect, because the hardware now comes with the capability built in.
- EBS optimization is something that's required on instance types and sizes which offer higher levels of performance.

MODULE 10 - ROUTE 53 – GLOBAL DNS

R53 Hosted Zones

A hosted zone is a container for records, and records contain information about how you want to route traffic for a specific domain, such as example.com, and its subdomains (acme.example.com, zenith.example.com). A hosted zone and the corresponding domain have the same name. There are two types of hosted zones:

Public hosted zones contain records that specify how you want to route traffic on the internet.

Private hosted zones contain records that specify how you want to route traffic in an Amazon VPC

- A **R53 Hosted Zone** is a DNS DB for a domain e.g., animals4life.org
- It **Globally resilient** (multiple DNS Servers)
- Created with domain registration via R53 - can be created separately
- Host **DNS Records** (e.g., A, AAAA, MX, NS, TXT)
- Hosted Zones are what the DNS system references- **Authoritative** for a domain.

R53 Public Hosted Zones

A public hosted zone is a container that holds information about how you want to route traffic on the internet for a specific domain, such as example.com, and its subdomains (acme.example.com, zenith.example.com). You get a public hosted zone in one of two ways:

- When you register a domain with Route 53, we create a hosted zone for you automatically.
- When you transfer DNS service for an existing domain to Route 53, you start by creating a hosted zone for the domain.

Remember the following:

- DNS Database (**zone file**) hosted by R53 (Public Name Servers)
- Accessible from the **public internet** & **VPCs**
- Hosted on "4" R53 Name servers (**NS**) specific for the zone
- Use "**NS records**" to point at these **NS** (connect to global DNS)
- Resource Records (**RR**) created within the Hosted Zone
- Externally registered domains can point at R53 public zone

R53 Private Hosted Zones

- A private hosted zone is a container that holds information about how you want Amazon Route 53 to respond to DNS queries for a domain and its subdomains within one or more VPCs that you create with the Amazon VPC service. Here's how private hosted zones work:
- You create a private hosted zone, such as example.com, and specify the VPCs that you want to associate with the hosted zone.
- You create records in the hosted zone that determine how Route 53 responds to DNS queries for your domain and subdomains within and among your VPCs. For example, suppose you have a database server that runs on an EC2 instance in one of the VPCs that you associated with your

private hosted zone. You create an A or AAAA record, such as db.example.com, and you specify the IP address of the database server.

- When an application submits a DNS query for db.example.com, Route 53 returns the corresponding IP address. The application must also be running on an EC2 instance in one of the VPCs that you associated with the example.com private hosted zone.
- The application uses the IP address that it got from Route 53 to establish a connection with the database server.

CNAME vs R53 Alias

Amazon Route 53 alias records provide a Route 53–specific extension to DNS functionality. Alias records let you route traffic to selected AWS resources, such as CloudFront distributions and Amazon S3 buckets. They also let you route traffic from one record in a hosted zone to another record.

Unlike a **CNAME record**, you can create an alias record at the top node of a DNS namespace, also known as the zone apex. For example, if you register the DNS name example.com, the zone apex is example.com. You can't create a CNAME record for example.com, but you can create an alias record for example.com that routes traffic to www.example.com.

Alias records are similar to CNAME records, but there are some important differences. The following list compares alias records and CNAME records.

List	Alias record	CNAME
Resources that you can redirect queries to	An alias record can only redirect queries to selected AWS resources, such as the following: ➢ Amazon S3 buckets ➢ CloudFront distributions ➢ Another record in the same Route 53 hosted zone	A CNAME record can redirect DNS queries to any DNS record.
Creating records that have the same name as the domain (records at the zone apex)	In most configurations, you can create an alias record that has the same name as the hosted zone (the zone apex). The one exception is when you want to redirect queries from the zone apex (such as example.com) to a record in the same hosted zone that has a type of CNAME (such as zenith.example.com). The alias record must have the same type as the record you're routing traffic to, and creating a CNAME record for the zone apex isn't supported even for an alias record.	You can't create a CNAME record that has the same name as the hosted zone (the zone apex). This is true both for hosted zones for domain names (example.com) and for hosted zones for subdomains (zenith.example.com).
Pricing for DNS queries	Route 53 doesn't charge for alias queries to AWS resources.	Route 53 charges for CNAME queries.
Record type specified in the DNS query	Route 53 responds to a DNS query only when the name of the alias record (such as acme.example.com) and the type of the alias record (such as A or AAAA) match the name and type in the DNS query.	A CNAME record redirects DNS queries for a record name regardless of the record type specified in the DNS query, such as A or AAAA.
How records are listed in dig or nslookup queries	In the response to a dig or nslookup query, an alias record is listed as the	A CNAME record is listed as a CNAME record in response to dig or nslookup queries.

	record type that you specified when you created the record, such as A or AAAA. The alias property is visible only in the Route 53 console or in the response to a programmatic request, such as an AWS CLI list-resource-record-sets command.	

Simple Routing

Simple routing lets you configure standard DNS records, with no special Route 53 routing such as weighted or latency. With simple routing, you typically route traffic to a single resource, for example, to a web server for your website.

- Simple Routing supports 1 record per name (www)
- Each Record can have multiple values
- All values are returned in a random order
- Simple Routing doesn't support health checks - all values are returned for a record when queried
- Use Simple Routing when you want to route requests towards on service such as a web server.

R53 Health Checks

Amazon Route 53 health checks monitor the health and performance of your web applications, web servers, and other resources. Each health check that you create can monitor one of the following:

- The health of a specified resource, such as a web server
- The status of other health checks
- The status of an Amazon CloudWatch alarm
- Health checkers located globally
- Health checkers check every 30s (every 10s cost extra)
- You can view the current and recent status of your health checks on the Route 53 console. You can also work with health checks programmatically through one of the AWS SDKs, the AWS Command Line Interface, AWS Tools for Windows PowerShell, or the Route 53 API.
- If you want to receive a notification when the status of a health check changes, you can configure an Amazon CloudWatch alarm for each health check.

Failover Routing

Failover routing lets you route traffic to a resource when the resource is healthy or to a different resource when the first resource is unhealthy.

- If the target of the health check is Healthy the Primary record is used.
- If the target of the health check is Unhealthy then any queries return the secondary record of the same name.
- A common architecture is to use failover for an "out of band" failure/ maintenance page for a service (e.g., EC2/S3)
- Use when you want to configure active passive failover.

Multi Value Routing

Multi value answer routing lets you configure Amazon Route 53 to return multiple values, such as IP addresses for your web servers, in response to DNS queries. You can specify multiple values for almost any record, but multi value answer routing also lets you check the health of each resource, so Route 53 returns only values for healthy resources.

- Multi Value Routing supports multiple records with the same name.
- Each record is independent and can have an associated health check.
- Any records which fail health checks won’t be returned when queried.
- Up to 8 'healthy' records are returned. If more exist, 8 are randomly selected.
- Multi Value improve availability. it is NOT a replacement for load balancing.

Latency Routing

If your application is hosted in multiple AWS Regions, you can improve performance for your users by serving their requests from the AWS Region that provides the lowest latency.

- Use Latency-based routing when optimising for performance & user experience.
- AWS maintains a database of latency between the user general location and the regions tagged in records.
- The record returned is the one which offers the lowest estimated latency & is healthy
- Latency-Based routing supports one record with the same name in each AWS Region

Geolocation Routing

Geolocation routing lets you choose the resources that serve your traffic based on the geographic location of your users, meaning the location that DNS queries originate from.

- R53 checks for records
  1) in the state,
  2) country,
  3) the continent and
  4) (optionally) default - it returns the most specific record or "NO ANSWER"
- Can be used for regional restrictions, language specific content or load balancing across regional endpoints
- With Geolocation records are tagged with location. Either "US state", country, continent or default.
- An IP check verifies the location of the user (normally the resolver)

Geoproximity Routing

Geoproximity routing lets Amazon Route 53 route traffic to your resources based on the geographic location of your users and your resources. You can also optionally choose to route more traffic or less to a given resource by specifying a value, known as a bias. A bias expands or shrinks the size of the geographic region from which traffic is routed to a resource.

- "+" or "-" bias can be added to rules. "+" increases a region size and decreases neighbouring regions
- Records can be tagged with an AWS Region or latitude & longitude coordinates
- Routing is distance based (including bias)

R53 Interoperability

- R53 normally has 2 jobs - Domain registrar and Domain Hosting
- R53 can do BOTH, or either Domain Registrar or Domain Hosting
- R53 Accepts your money (Domain Registration fee)
- R53 allocates 4 Name Servers (NS) (Domain Hosting)
- R53 Creates a zone file (Domain Hosting) on the above NS
- R53 communicates with the registry of the TLD (Domain Registrar)
- And Set the NS records for the domain to point at the 4 NS above

MODULE 11 - RELATIONAL DATABASE SERVICE (RDS)

Database Refresher

Relational databases

A relational database, also called Relational Database Management System (RDBMS) or SQL database, stores data in tables and rows also referred to as records. A relational database works by linking information from multiple tables through the use of "keys." A key is a unique identifier which can be assigned to a row of data contained within a table. This unique identifier, called a "primary key," can then be included in a record located in another table when that record has a relationship to the primary record in the main table. When this unique primary key is added to a record in another table, it is called a "foreign key" in the associated table. The connection between the primary and foreign key then creates the "relationship" between records contained across multiple tables.

What you need to know about relational databases:

- They work with structured data.
- Relationships in the system have constraints, which promotes a high level of data integrity.
- There are limitless indexing capabilities, which results in faster query response times.
- They are excellent at keeping data transactions secure.
- They provide the ability to write complex SQL queries for data analysis and reporting.
- Their models can ensure and enforce business rules at the data layer adding a level of data integrity not found in a non-relational database.
- They are table and row oriented.
- They Use SQL (structured query language) for shaping and manipulating data, which is very powerful.
- SQL database examples: MySql, Oracle, Sqlite, Postgres and MS-SQL. NoSQL database examples: MongoDB, BigTable, Redis, RavenDb, Cassandra, Hbase, Neo4j and CouchDb.
- SQL databases are best fit for heavy duty transactional type applications.

Non-relational databases

The non-relational database, or NoSQL database, stores data. However, unlike the relational database, there are no tables, rows, primary keys or foreign keys. Instead, the non-relational database uses a storage model optimized for specific requirements of the type of data being stored.

Some of the more popular NoSQL databases are MongoDB, Apache Cassandra, Redis, Couchbase and Apache HBase. There are four popular non-relational types: document data store, column-oriented database, key-value store and graph database. Often combinations of these types are used for a single application.

- They have the ability to store large amounts of data with little structure.
- They provide scalability and flexibility to meet changing business requirements.
- They provide schema-free or schema-on-read options.
- They have the ability to capture all types of data "Big Data" including unstructured data.
- They are document oriented.
- NoSQL or non-relational databases examples: MongoDB, Apache Cassandra, Redis, Couchbase and Apache HBase.
- They are best for Rapid Application Development. NoSQL is the best selection for flexible data storage with little to no structure limitations.
- They provide flexible data model with the ability to easily store and combine data of any structure without the need to modify a schema.

ACID vs BASE

ACID and BASE are DB transactional models.

The CAP theorem states that it is impossible to achieve both consistency and availability in a partition tolerant distributed system (i.e., a system which continues to work in cases of temporary communication breakdowns).

The fundamental difference between ACID and BASE database models is the way they deal with this limitation.

- The ACID model provides a consistent system.
- The BASE model provides high availability.

ACID stands for:

Atomic – Each transaction is either properly carried out or the process halts and the database reverts back to the state before the transaction started. This ensures that all data in the database is valid.

Consistent – A processed transaction will never endanger the structural integrity of the database.

Isolated – Transactions cannot compromise the integrity of other transactions by interacting with them while they are still in progress.

Durable – The data related to the completed transaction will persist even in the cases of network or power outages. If a transaction fails, it will not impact the manipulated data.

ACID Use Case Example

Financial institutions will almost exclusively use ACID databases. Money transfers depend on the atomic nature of ACID. An interrupted transaction which is not immediately removed from the database can cause a lot of issues. Money could be debited from one account and, due to an error, never credited to another.

BASE stands for:

Basically Available – Rather than enforcing immediate consistency, BASE-modelled NoSQL databases will ensure availability of data by spreading and replicating it across the nodes of the database cluster.

Soft State – Due to the lack of immediate consistency, data values may change over time. The BASE model breaks off with the concept of a database which enforces its own consistency, delegating that responsibility to developers.

Eventually Consistent – The fact that BASE does not enforce immediate consistency does not mean that it never achieves it. However, until it does, data reads are still possible (even though they might not reflect the reality).

BASE Use Case Example

Marketing and customer service companies who deal with sentiment analysis will prefer the elasticity of BASE when conducting their social network research. Social network feeds are not well structured but contain huge amounts of data which a BASE-modelled database can easily store.

Databases on EC2

- The Relational Database Service (RDS) is a Database(server) as a service product from AWS which allows the creation of managed databases instances.
- RDS more accurately described as a Database Server-as-a-service product. RDS is a product which provides Managed Database Instances which can themselves hold one or more DBs.
- with RDS, what you're paying for and consuming, is a database server that you have access to.
- The benefits of RDS are that you don't need to manage the physical hardware, or the server OS, or the database system itself. AWS handle all of that behind the scenes.
- RDS supports the most popular database types that you've probably encountered. MySQL, MariaDB, PostgreSQL, Oracle, and even MS SQL Server. You can pick whatever works for your given application requirements. Each of them comes with features and limitations which you should be aware of for real-world usage, but that you don't need a detailed understanding of for the exam.
- Amazon Aurora is a database engine which AWS have created, and you can select when creating RDS instances.

RDS High-Availability (Multi AZ)

Amazon RDS provides high availability and failover support for DB instances using multi-AZ deployments.

MultiAZ is a feature of RDS which provisions a standby replica which is kept in sync Synchronously with the primary instance.

- In a Multi-AZ deployment, Amazon RDS automatically provisions and maintains a synchronous standby replica in a different Availability Zone.
- The primary DB instance is synchronously replicated across Availability Zones to a standby replica to provide data redundancy, eliminate I/O freezes, and minimize latency spikes during system backups.
- Running a DB instance with high availability can enhance availability during planned system maintenance, and help protect your databases against DB instance failure and Availability Zone disruption.
- The standby replica cannot be used for any performance scaling ... only availability.
- Backups, software updates and restarts can take advantage of MultiAZ to reduce user disruption.

RDS Automatic Backup, RDS Snapshots and Restore

- By default, Amazon RDS creates and saves automated backups of your DB instance securely in Amazon S3 for a user-specified retention period.
- In addition, you can create snapshots, which are user-initiated backups of your instance that are kept until you explicitly delete them.
- You can create a new instance from a database snapshots whenever you desire. Although database snapshots serve operationally as full backups, you are billed only for incremental storage use.
- RDS is capable of performing Manual Snapshots and Automatic backups
- Manual snapshots are performed manually and live past the termination of an RDS instance
- Automatic backups can be taken of an RDS instance with a 0 (Disabled) to 35 Day retention.
- Automatic backups also use S3 for storing transaction logs every 5 minutes - allowing for point in time recovery.
- Snapshots can be restored. but create a new RDS instance.

Automated Backups

- Turned on by default, the automated backup feature of Amazon RDS will backup your databases and transaction logs. Amazon RDS automatically creates a storage volume snapshot of your DB instance, backing up the entire DB instance and not just individual databases.
- This backup occurs during a daily user-configurable 30-minute period known as the backup window. Automated backups are kept for a configurable number of days (called the backup retention period). Your automatic backup retention period can be configured to up to thirty-five days.

Point-in-time Restores

- You can restore your DB instance to any specific time during the backup retention period, creating a new DB instance. To restore your database instance, you can use the AWS Console or Command Line Interface.
- To determine the latest restorable time for a DB instance, use the AWS Console or Command Line Interface to look at the value returned in the LatestRestorableTime field for the DB instance. The latest restorable time for a DB instance is typically within 5 minutes of the current time.

Database Snapshots

- Database snapshots are user-initiated backups of your instance stored in Amazon S3 that are kept until you explicitly delete them. You can create a new instance from a database snapshots whenever you desire. Although database snapshots serve operationally as full backups, you are billed only for incremental storage use.

Snapshot Copies

- With Amazon RDS, you can copy DB snapshots and DB cluster snapshots. You can copy automated or manual snapshots. After you copy a snapshot, the copy is a manual snapshot. You can copy a snapshot within the same AWS Region, you can copy a snapshot across AWS Regions, and you can copy a snapshot across AWS accounts.

Snapshot Sharing

- Using Amazon RDS, you can share a manual DB snapshot or DB cluster snapshot with other AWS accounts. Sharing a manual DB snapshot or DB cluster snapshot, whether encrypted or unencrypted, enables authorized AWS accounts to copy the snapshot.
- Sharing an unencrypted manual DB snapshot enables authorized AWS accounts to directly restore a DB instance from the snapshot instead of taking a copy of it and restoring from that. This isn't supported for encrypted manual DB snapshots.
- Sharing a manual DB cluster snapshot, whether encrypted or unencrypted, enables authorized AWS accounts to directly restore a DB cluster from the snapshot instead of taking a copy of it and restoring from that.

RDS Read-Replicas

- Amazon RDS Read Replicas provide enhanced performance and durability for RDS database (DB) instances.
- They make it easy to elastically scale out beyond the capacity constraints of a single DB instance for read-heavy database workloads.
- You can create one or more replicas of a given source DB Instance and serve high-volume application read traffic from multiple copies of your data, thereby increasing aggregate read throughput.
- Read replicas can also be promoted when needed to become standalone DB instances.
- Read replicas are available in Amazon RDS for MySQL, MariaDB, PostgreSQL, Oracle, and SQL Server as well as Amazon Aurora.

(read) performance Improvements

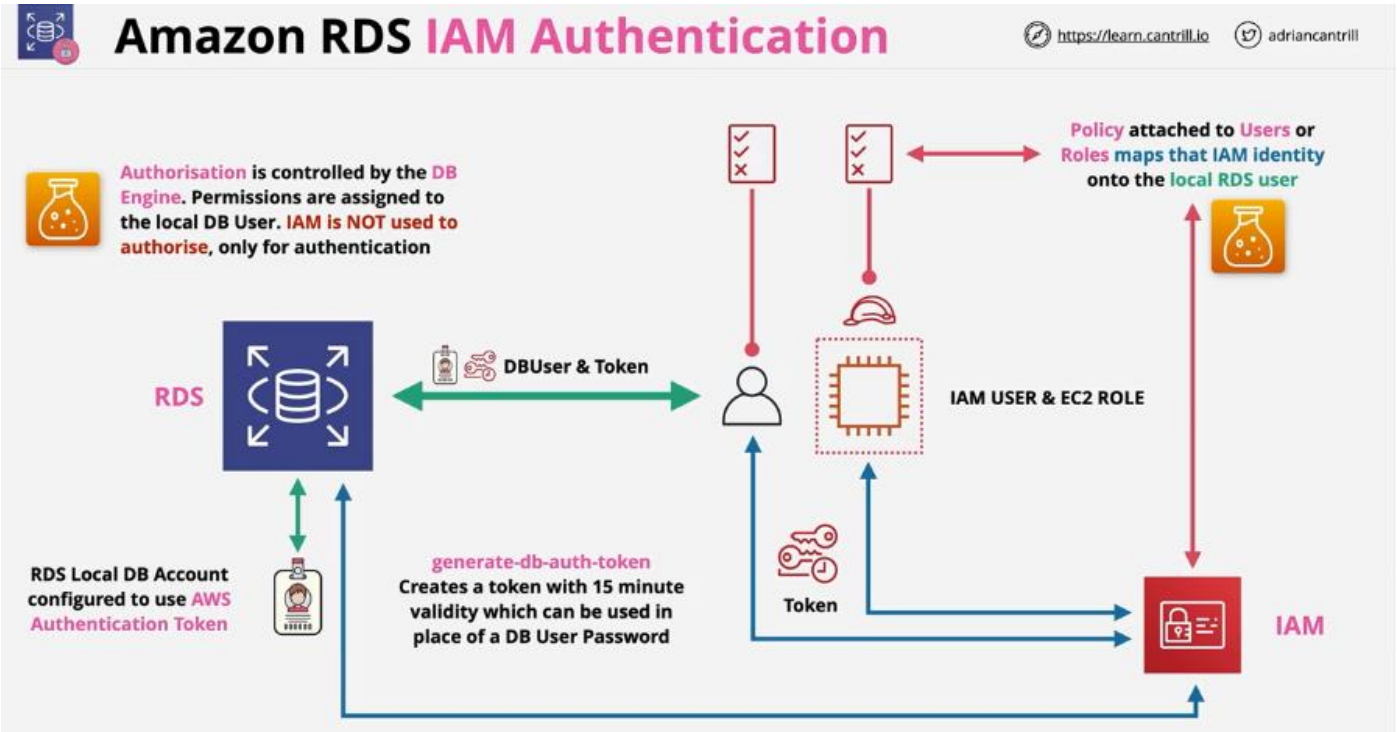
- 5x direct read-replicas per DB instance
- Each providing an additional instance of read performance
- Global performance improvements

RDS Data Security

- SSL/TLS (in transit) is available for RDS, can be mandatory
- RDS supports EBS volume encryption - KMS
- Handled by HOST/EBS
- AWS or Customer Managed CMK generates data keys
- Data Keys used for encryption operations
- Storage, Logs, Snapshots & replicas are encrypted and encryption can't be removed.
- RDS MSSQL and RDS Oracle support TDE (Transparent Data Encryption)
- Encryption handled within the DB engine
- RDS Oracle supports integration with CloudHSM
- Much stronger key controls (even from AWS)

Amazon RDS IAM Authentication

- You can authenticate to your DB instance using AWS Identity and Access Management (IAM) database authentication.
- IAM database authentication works with MySQL and PostgreSQL. With this authentication method, you don't need to use a password when you connect to a DB instance. Instead, you use an authentication token.
- An authentication token is a unique string of characters that Amazon RDS generates on request. Authentication tokens are generated using AWS Signature Version 4. Each token has a lifetime of 15 minutes.
- You don't need to store user credentials in the database, because authentication is managed externally using IAM. You can also still use standard database authentication.

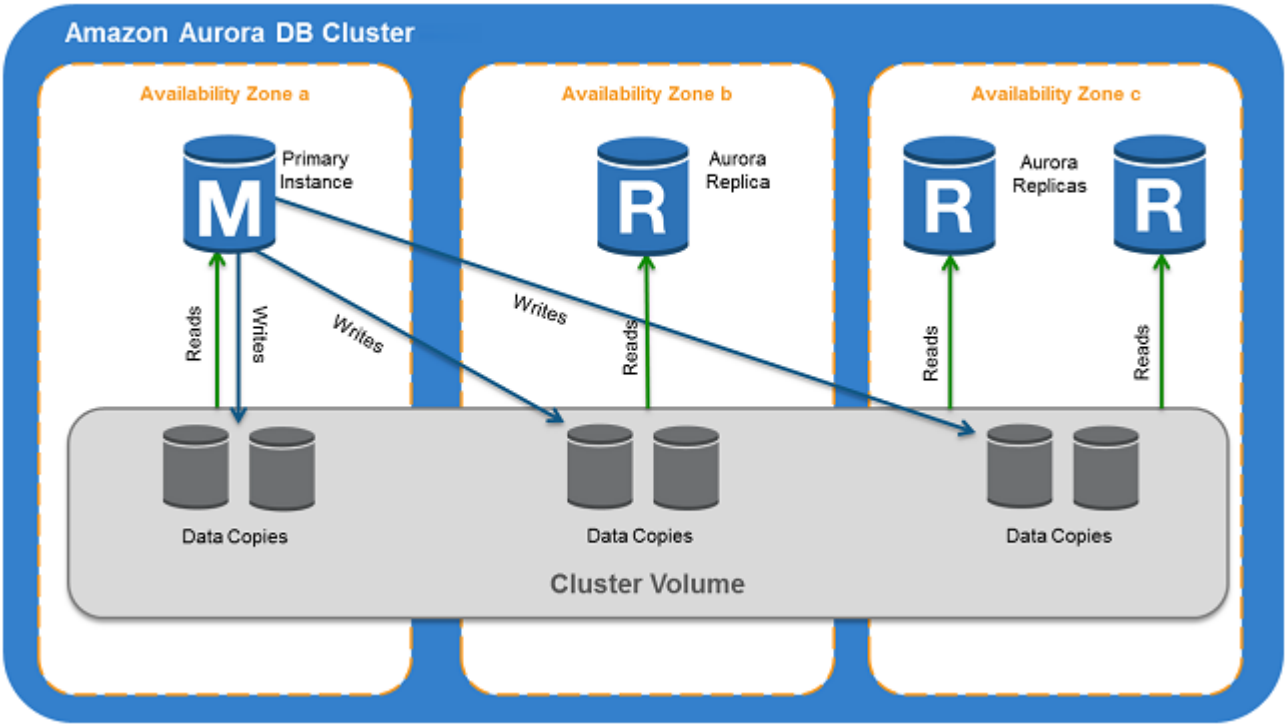


Aurora Architecture

An Amazon Aurora DB cluster consists of one or more DB instances and a cluster volume that manages the data for those DB instances. An Aurora cluster volume is a virtual database storage volume that spans multiple Availability Zones, with each Availability Zone having a copy of the DB cluster data. Two types of DB instances make up an Aurora DB cluster:

1. **Primary DB instance** – Supports read and write operations, and performs all of the data modifications to the cluster volume. Each Aurora DB cluster has one primary DB instance.
2. **Aurora Replica** – Connects to the same storage volume as the primary DB instance and supports only read operations. Each Aurora DB cluster can have up to 15 Aurora Replicas in addition to the primary DB instance. Maintain high availability by locating Aurora Replicas in separate Availability Zones. Aurora automatically fails over to an Aurora Replica in case the primary DB instance becomes unavailable. You can specify the failover priority for Aurora Replicas. Aurora Replicas can also offload read workloads from the primary DB instance.

- Aurora is an AWS designed database engine officially part of RDS
- Aurora implements a number of radical design changes which offer significant performance and feature improvements over other RDS database engines.
- Aurora Architecture is VERY different from RDS. it uses a "Cluster".
- A single primary instance + 0 or more replicas
- No local storage - it uses cluster volume
- Faster provisioning & improved availability & performance.



Aurora storage Architecture

- Aurora data is stored in the cluster volume, which is a single, virtual volume that uses solid state drives (SSDs).
- A cluster volume consists of copies of the data across three Availability Zones in a single AWS Region. Because the data is automatically replicated across Availability Zones, your data is highly durable with less possibility of data loss.

- This replication also ensures that your database is more available during a failover. It does so because the data copies already exist in the other Availability Zones and continue to serve data requests to the DB instances in your DB cluster.
- The amount of replication is independent of the number of DB instances in your cluster.
- All SSD Based - high IOPS, low latency
- Storage is billed based on what's used
- High water mark - billed for the most used
- Storage which is freed up can be re-used
- Replicas can be added and removed without requiring storage provisioning.

Cost

- No free-tier option
- Aurora doesn't support Micro Instances
- Beyond RDS single AZ (micro) Aurora offers better value
- Compute - hourly charge, per second, 10-minute minimum
- Storage - GB-Month consumed, IO cost per request.
- 100% DB size in backups are included

Aurora Restore, Clone & Backtrack

- Backups in Aurora work in the same way as RDS
- Restores create a new cluster
- Backtrack can be used which allow in-place rewinds to a previous point in time
- Fast clones make a new database MUCH faster than copying all the data - copy-on-write.

Aurora Serverless

- Amazon Aurora Serverless is an on-demand, auto-scaling configuration for Amazon Aurora.
- It automatically starts up, shuts down, and scales capacity up or down based on your application's needs. It enables you to run your database in the cloud without managing any database capacity.
- Manually managing database capacity can take up valuable time and can lead to inefficient use of database resources. With Aurora Serverless, you simply create a database endpoint, optionally specify the desired database capacity range, and connect your applications.
- You pay on a per-second basis for the database capacity you use when the database is active, and migrate between standard and serverless configurations with a few clicks in the Amazon RDS Management Console.
- Instead of provisioning and managing database servers, you specify Aurora capacity units (ACUs). Each ACU is a combination of approximately 2 gigabytes (GB) of memory, corresponding CPU, and networking. Database storage automatically scales from 10 gibibytes (GiB) to 128 tebibytes (TiB), the same as storage in a standard Aurora DB cluster.
- You can specify the minimum and maximum ACU. The minimum Aurora capacity unit is the lowest ACU to which the DB cluster can scale down. The maximum Aurora capacity unit is the highest ACU to which the DB cluster can scale up. Based on your settings, Aurora Serverless v1 automatically creates scaling rules for thresholds for CPU utilization, connections, and available memory.
- Cluster adjust based on load
- Can go to 0 and be paused
- Consumption billing per-second basis
- Same resilience as Aurora (6 copies across AZs)

Use cases for Aurora Serverless

Aurora Serverless v1 is designed for the following use cases:

- **Infrequently used applications –** You have an application that is only used for a few minutes several times per day or week, such as a low-volume blog site. With Aurora Serverless v1, you pay for only the database resources that you consume on a per-second basis.
- **New applications –** You're deploying a new application and you're unsure about the instance size you need. By using Aurora Serverless v1, you can create a database endpoint and have the database auto scale to the capacity requirements of your application.
- **Variable workloads –** You're running a lightly used application, with peaks of 30 minutes to several hours a few times each day, or several times per year. Examples are applications for human resources, budgeting, and operational reporting applications. With Aurora Serverless v1, you no longer need to provision for peak or average capacity.
- **Unpredictable workloads –** You're running daily workloads that have sudden and unpredictable increases in activity. An example is a traffic site that sees a surge of activity when it starts raining. With Aurora Serverless v1, your database auto scales capacity to meet the needs of the application's peak load and scales back down when the surge of activity is over.
- **Development and test databases –** Your developers use databases during work hours but don't need them on nights or weekends. With Aurora Serverless v1, your database automatically shuts down when it's not in use.
- **Multi-tenant applications –** With Aurora Serverless v1, you don't have to individually manage database capacity for each application in your fleet. Aurora Serverless v1 manages individual database capacity for you.

Aurora Global Database

- Aurora global databases are a feature of Aurora Provisioned clusters which allow data to be replicated globally providing significant RPO and RTO improvements for BC and DR planning. Additionally, global databases can provide performance improvements for customers. with data being located closer to them, in a read-only form.
- Replication occurs at the storage layer and is generally ~1second between all AWS regions.
- Amazon Aurora global databases span multiple AWS Regions, enabling low latency global reads and providing fast recovery from the rare outage that might affect an entire AWS Region. An Aurora global database has a primary DB cluster in one Region, and up to five secondary DB clusters in different Regions.
- It provides Cross-Region Disaster Recovery. If your primary region suffers a performance degradation or outage, you can promote one of the secondary regions to take read/write responsibilities. An Aurora cluster can recover in less than 1 minute even in the event of a complete regional outage. This provides your application with an effective Recovery Point Objective (RPO) of 1 second and a Recovery Time Objective (RTO) of less than 1 minute, providing a strong foundation for a global business continuity plan.
- Aurora Global Database lets you easily scale database reads across the world and place your applications close to your users. Your applications enjoy quick data access regardless of the number and location of secondary regions, with typical cross-region replication latencies below 1 second. You can achieve further scalability by creating up to 16 database instances in each region, which will all stay continuously up to date.
- Extending your database to additional regions has no impact on performance. Cross-region replication uses dedicated infrastructure in the Aurora storage layer, keeping database resources in the primary and secondary regions fully available to serve application needs.
- With Amazon Aurora Global Database, you pay for replicated write I/Os between the primary region and each secondary region.

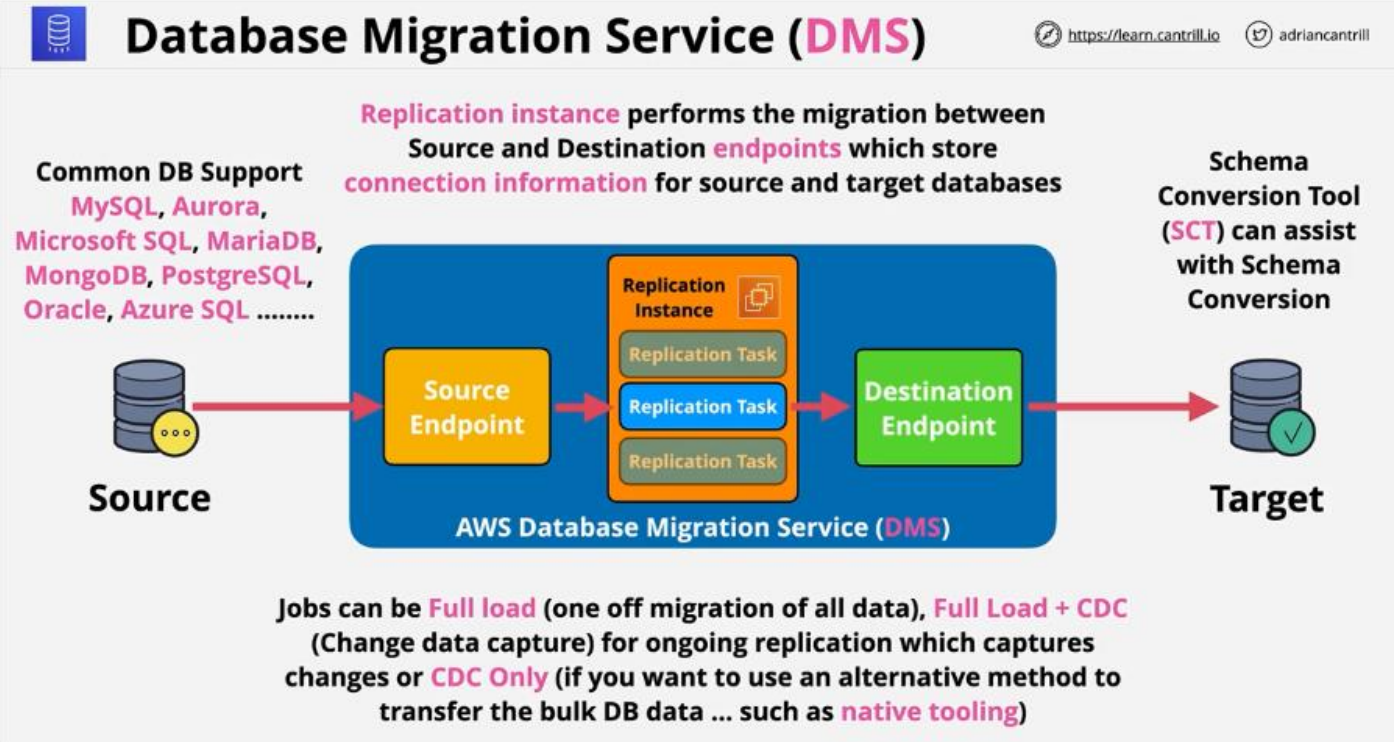
Multi-master writes

- Multi-master write is a mode of Aurora Provisioned Clusters which allows multiple instances to perform reads and writes at the same time - rather than only one primary instance having write capability in a single-master cluster.
- Default Aurora mode is Single-Master
- One R/W and 0+ Read Only Replicas
- Cluster Endpoint is used to write, read endpoint is used for load balanced reads
- Failover takes time - replica promoted to R/W
- In Multi-Master mode all instances are R/W

Database Migration Service (DMS)

The Database Migration Service (DMS) is a managed service which allows for 0 data loss, low or 0 downtime migrations between 2 database endpoints.

- The service is capable of moving databases INTO or OUT of AWS.
- It is a managed database migration service
- Runs using a replication instance
- Source and Destination Endpoints point at Source and Target Databases
- One endpoint MUST be on AWS



MODULE 12 - NETWORK STORAGE

Elastic File System (EFS)

- The Elastic File System (EFS) is an AWS managed implementation of NFS which allows for the creation of shared 'filesystems' which can be mounted within multi EC2 instances.
- EFS can play an essential part in building scalable and resilient systems.
- EFS is an implementation of NFSv4
- EFS Filesystems can be mounted in Linux
- Shared between many EC2 instances
- Private service, via mount targets inside a VPC
- Can be accessed from on-premises - VPN or DX
- To support a wide variety of cloud storage workloads, Amazon EFS offers two performance modes, General Purpose mode and Max I/O mode.
- The two performance modes have no additional costs, so your Amazon EFS file system is billed and metered the same, regardless of your performance mode.
- General Purpose is ideal for latency-sensitive use cases, like web serving environments, content management systems, home directories, and general file serving. If you don't choose a performance mode when you create your file system, Amazon EFS selects the General-Purpose mode for you by default.
- File systems in the Max I/O mode can scale to higher levels of aggregate throughput and operations per second. This scaling is done with a trade-off of slightly higher latencies for file metadata operations. Highly parallelized applications and workloads, such as big data analysis, media processing, and genomic analysis, can benefit from this mode.
- There are two throughput modes to choose from for your file system, Bursting Throughput and Provisioned Throughput.
- With Bursting Throughput mode, throughput on Amazon EFS scales as the size of your file system in the EFS Standard or One Zone storage class grows.
- With Provisioned Throughput mode, you can instantly provision the throughput of your file system (in MiB/s) independent of the amount of data stored.
- Amazon EFS uses a credit system to determine when file systems can burst. Each file system earns credits over time at a baseline rate that is determined by the size of the file system that is stored in the EFS Standard or One Zone storage class.
- Standard and Infrequent Access (IA) Classes
- Lifecycle policies can be used with classes

MODULE 13- HA & SCALING

Load Balancing Fundamentals

- Elastic Load Balancing automatically distributes incoming application traffic across multiple targets, such as Amazon EC2 instances, containers, IP addresses, Lambda functions, and virtual appliances.
- It can handle the varying load of your application traffic in a single Availability Zone or across multiple Availability Zones. Elastic Load Balancing offers four types of load balancers that all feature the high availability, automatic scaling, and robust security necessary to make your applications fault tolerant.

Things to remember.

- Clients connect to the Load Balancer specifically the listener of the LB
- The LB connects on your behalf to 1+ targets(servers)
- There are two connections... listener & Backend
- Client Abstracted from individual servers
- It used for High-Availability, Fault-Tolerance and Scaling.

Application Load balancing (ALB)

- An Application Load Balancer functions at the application layer, the seventh layer of the Open Systems Interconnection (OSI) model.
- After the load balancer receives a request, it evaluates the listener rules in priority order to determine which rule to apply, and then selects a target from the target group for the rule action.
- You can configure listener rules to route requests to different target groups based on the content of the application traffic.
- Routing is performed independently for each target group, even when a target is registered with multiple target groups. You can configure the routing algorithm used at the target group level.
- The default routing algorithm is round robin; alternatively, you can specify the least outstanding requests routing algorithm.

Application Load Balancer components

- A **load balancer** serves as the single point of contact for clients. The load balancer distributes incoming application traffic across multiple targets, such as EC2 instances, in multiple Availability Zones. This increases the availability of your application. You add one or more listeners to your load balancer.
- A **listener** checks for connection requests from clients, using the protocol and port that you configure. The rules that you define for a listener determine how the load balancer routes requests to its registered targets. Each rule consists of a priority, one or more actions, and one or more conditions. When the conditions for a rule are met, then its actions are performed. You must define a default rule for each listener, and you can optionally define additional rules.
- Each **target group** routes requests to one or more registered targets, such as EC2 instances, using the protocol and port number that you specify. You can register a target with multiple target groups. You can configure health checks on a per target group basis. Health checks are performed on all targets registered to a target group that is specified in a listener rule for your load balancer.

Exam Hints

- Targets => Target Groups which are addressed via rules
- Rules are path based or host based
- Support EC2, ECS, EKS, Lambda, HTTPS, HTTP/2 and WebSocket
- ALB can use SNI for multiple SSL Certs - host based rules
- Recommended vs CLB(Legacy)

Launch Configuration and Templates

A **launch configuration** is an instance configuration template that an Auto Scaling group uses to launch EC2 instances. When you create a launch configuration, you specify information for the instance.

Important: AWS strongly recommend that you do not use launch configurations. They do not provide full functionality for Amazon EC2 Auto Scaling or Amazon EC2. We provide information about launch configurations for customers who have not yet migrated from launch configurations to launch templates.

Launch Templates

- A launch template is similar to a launch configuration, in that it specifies instance configuration information. Included are the ID of the Amazon Machine Image (AMI), the instance type, a key pair, security groups, and the other parameters that you use to launch EC2 instances. However, defining a launch template instead of a launch configuration allows you to have multiple versions of a template.
- With versioning, you can create a subset of the full set of parameters and then reuse it to create other templates or template versions. For example, you can create a default template that defines common configuration parameters and allow the other parameters to be specified as part of another version of the same template.
- In addition to the features of Amazon EC2 Auto Scaling that you can configure by using launch templates, launch templates enable you to use newer features of Amazon EC2.
- This includes the current generation of EBS provisioned IOPS volumes (io2), EBS volume tagging, T2 Unlimited instances, elastic inference, and Dedicated Hosts, to name a few.

Auto-Scaling Groups

- An Auto Scaling group contains a collection of Amazon EC2 instances that are treated as a logical grouping for the purposes of automatic scaling and management.
- An Auto Scaling group also enables you to use Amazon EC2 Auto Scaling features such as health check replacements and scaling policies. Both maintaining the number of instances in an Auto Scaling group and automatic scaling are the core functionality of the Amazon EC2 Auto Scaling service.
- It provides Automatic Scaling and Self-Healing for EC2
- Uses Launch templates or Configurations
- It has a Minimum, Desired and Maximum Size (e.g., 1:2:4)
- Provision or Terminate Instances to keep at the Desired level (between Min/Max)
- Scaling policies automate based on metrics

Scaling Policies

1. **Manual Scaling -** Manually adjust the desired capacity. At any time, you can change the size of an existing Auto Scaling group manually. You can either update the desired capacity of the Auto Scaling group, or update the instances that are attached to the Auto Scaling group. Manually scaling your group can be useful when automatic scaling is not needed or when you need to hold capacity at a fixed number of instances.
2. **Scheduled Scaling** - Time based adjustment - e.g., Sales. Scheduled scaling helps you to set up your own scaling schedule according to predictable load changes. For example, let's say that every week the traffic to your web application starts to increase on Wednesday, remains high on Thursday, and starts to decrease on Friday. You can configure a schedule for Amazon EC2 Auto Scaling to increase capacity on Wednesday and decrease capacity on Friday.
3. **Dynamic Scaling-** When you configure dynamic scaling, you define how to scale the capacity of your Auto Scaling group in response to changing demand. For example, let's say that you have a web application that currently runs on two instances, and you want the CPU utilization of the Auto Scaling group to stay at around 50 percent when the load on the application changes. This gives you extra capacity to handle traffic spikes without maintaining an excessive number of idle resources.

Amazon EC2 Auto Scaling supports the following types of dynamic scaling policies:

- **Target tracking scaling—**Increase or decrease the current capacity of the group based on a target value for a specific metric. This is similar to the way that your thermostat maintains the temperature of your home—you select a temperature and the thermostat does the rest.
- **Step scaling—**Increase or decrease the current capacity of the group based on a set of scaling adjustments, known as step adjustments, that vary based on the size of the alarm breach.

- **Simple scaling—**Increase or decrease the current capacity of the group based on a single scaling adjustment.
- A **scaling cooldown** helps you prevent your Auto Scaling group from launching or terminating additional instances before the effects of previous activities are visible.
- When you use simple scaling, after the Auto Scaling group scales using a simple scaling policy, it waits for a cooldown period to complete before any further scaling activities initiated by simple scaling policies can start.

Final points

- Autoscaling Groups are free
- Only the resources created are billed
- Use cool downs to avoid rapid scaling
- Think about more, smaller instances -granularity
- Use with ALB's for elasticity - abstraction
- ASG defined WHEN and WHERE, LT defined WHAT

Network Load Balancing (NLB)

- A Network Load Balancer functions at the fourth layer of the Open Systems Interconnection (OSI) model.
- It can handle millions of requests per second. After the load balancer receives a connection request, it selects a target from the target group for the default rule.
- It attempts to open a TCP connection to the selected target on the port specified in the listener configuration.
- When you enable an Availability Zone for the load balancer, Elastic Load Balancing creates a load balancer node in the Availability Zone. By default, each load balancer node distributes traffic across the registered targets in its Availability Zone only.
- If you enable cross-zone load balancing, each load balancer node distributes traffic across the registered targets in all enabled Availability Zones.
- Can't understand HTTP/S but are faster - ~100ms vs 400ms for application load balancers
- Rapid scaling - millions of requests per second
- Can do SSL Pass through
- Can load balance non-HTTP/S applications - doesn't care about anything above TCP/UDP

SSL Offload & Session Stickiness

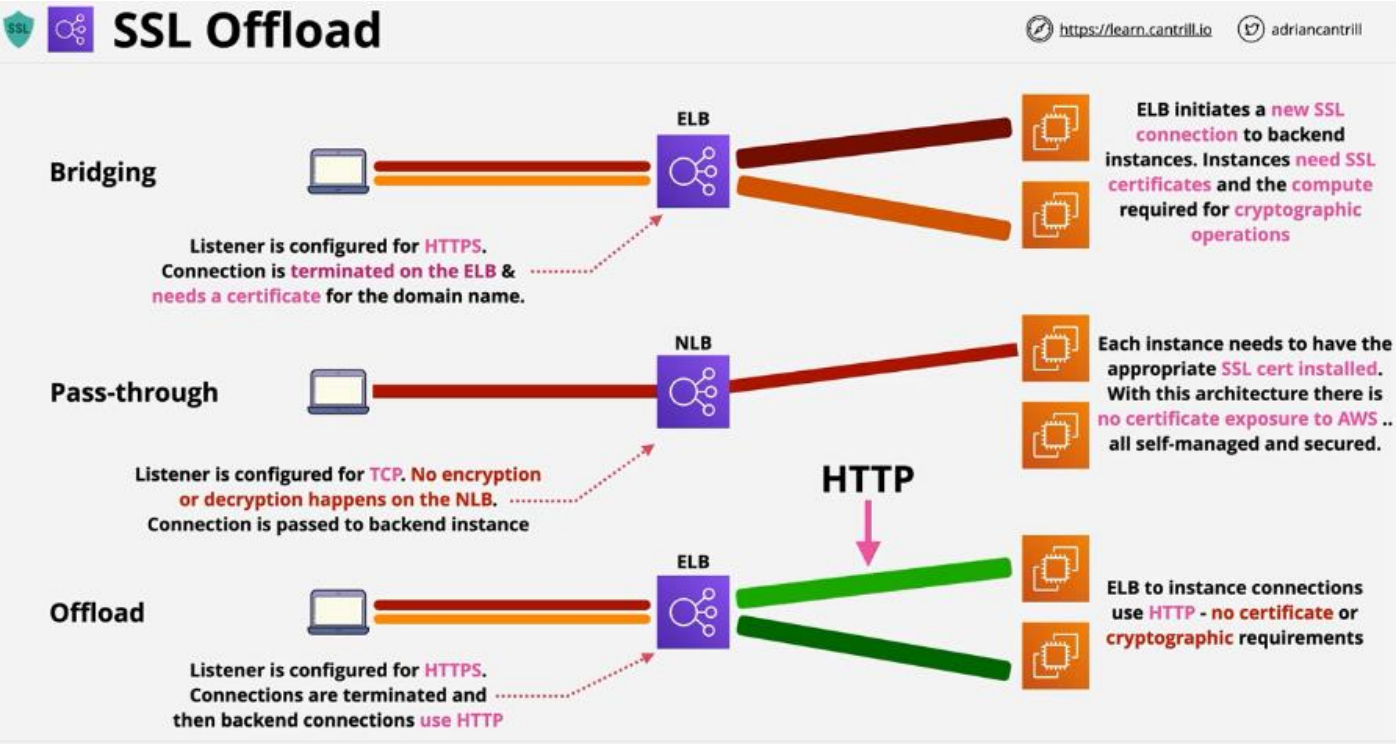
Session Stickiness

- By default, an Application Load Balancer routes each request independently to a registered target based on the chosen load-balancing algorithm. However, you can use the sticky session feature (also known as session affinity) to enable the load balancer to bind a user's session to a specific target.
- This ensures that all requests from the user during the session are sent to the same target. This feature is useful for servers that maintain state information in order to provide a continuous experience to clients. To use sticky sessions, the client must support cookies.
- Application Load Balancers support both duration-based cookies and application-based cookies.
- The key to managing sticky sessions is determining how long your load balancer should consistently route the user's request to the same target. Sticky sessions are enabled at the target group level. You can use a combination of duration-based stickiness, application-based stickiness, and no stickiness across all of your target groups.
- The content of load balancer generated cookies are encrypted using a rotating key. You cannot decrypt or modify load balancer generated cookies.

- For both stickiness types, the Application Load Balancer resets the expiry of the cookies it generates after every request. If a cookie expires, the session is no longer sticky and the client should remove the cookie from its cookie store.

There are 3 different ways that ELB's can handle SSL

1. SSL Bridging
2. SSL Pass Through
3. SSL Offloading



MODULE 14 - SERVERLESS AND APPLICATION SERVICES

Architecture Evolution Monolithic and Tiered

Traditional monolithic architectures are hard to scale. As an application's code base grows, it becomes complex to update and maintain. Introducing new features, languages, frameworks, and technologies becomes very hard, limiting innovation and new ideas.

- You want to practice continuous deployment of the application
- You must run multiple instances of the application on multiple machines in order to satisfy scalability and availability requirements
- You want to take advantage of emerging technologies (frameworks, programming languages, etc)

Queue Based, Microservice & Event-Driven

Event-Driven

- An event-driven architecture uses events to trigger and communicate between decoupled services and is common in modern applications built with microservices. An event is a change in state, or an update, like an item being placed in a shopping cart on an e-commerce website. Events can either carry the state (the item purchased, its price, and a delivery address) or events can be identifiers (a notification that an order was shipped).

- Event-driven architectures have three key components: event producers, event routers, and event consumers. A producer publishes an event to the router, which filters and pushes the events to consumers. Producer services and consumer services are decoupled, which allows them to be scaled, updated, and deployed independently.
- No constant running or waiting for things
- Producers generate events when something happens such as clicks, errors, criteria met, uploads, actions
- Events are delivered to consumers
- actions are taken & the system returns to waiting
- Mature event-driven architecture only consumes resources while handling events.
- AWS offers two queue-based services: Amazon Simple Queue Service (SQS) and Amazon MQ

Microservices

- Microservices are an architectural and organizational approach to software development to speed up deployment cycles, foster innovation and ownership, improve maintainability and scalability of software applications, and scale organizations delivering software and services by using an agile approach that helps teams to work independently from each other. Using a microservices approach, software is composed of small services that communicate over well-defined APIs that can be deployed independently. These services are owned by small autonomous teams. This agile approach is skey to successfully scale your organization.
- There are three common patterns that we observe when our customers build microservices: **API driven**, **event driven**, and **data streaming**.
- Microservices architectures are not a completely new approach to software engineering, but rather a combination of various successful and proven concepts such as:
    - Agile software development
    - Service-oriented architectures
    - API-first design
    - Continuous Integration/Continuous Delivery (CI/CD)

AWS Lambda

- AWS Lambda is a function as a service product which provides the ability to execute small lambda functions and pay only for the execution duration.
- Lambda functions can be invoked manually, on a scheduled or in an event-driven way
- AWS Lambda is a serverless compute service that lets you run code without provisioning or managing servers, creating workload-aware cluster scaling logic, maintaining event integrations, or managing runtimes. With Lambda, you can run code for virtually any type of application or backend service - all with zero administration.
- Event-driven invocation (execution)
- Lambda function = piece of code in one language
- Lambda functions use a runtime (e.g., Python 3.6)
- Runs in a runtime environment
- You are billed only for the duration a function runs
- Key component of serverless architecture.
- Just upload your code as a ZIP file or container image, and Lambda automatically and precisely allocates compute execution power and runs your code based on the incoming request or event, for any scale of traffic.

Key consideration

- Currently – 15-minute execution limit
- New runtime environment every execution – no persistence
- Execution Role provides permissions

- Load data from other services (e.g., S3)
- Store data from other services (e.g., S3)
- (Free tier) 1M free requests per month and 400,000 GB- seconds of computer time month.

CloudWatchEvents and EventBridge

- CloudWatch Events and EventBridge have visibility over events generated by supported AWS services within an account.
- They can monitor the default account event bus - and pattern match events flowing through and deliver these events to multiple targets.
- They are also the source of scheduled events which can perform certain actions at certain times of day, days of the week, or multiple combinations of both - using the Unix CRON time expression format.
- Both services are one way how event driven architectures can be implemented within AWS.

Amazon CloudWatch Events

- Amazon CloudWatch Events delivers a near real-time stream of system events that describe changes in Amazon Web Services (AWS) resources. Using simple rules that you can quickly set up; you can match events and route them to one or more target functions or streams. CloudWatch Events becomes aware of operational changes as they occur. CloudWatch Events responds to these operational changes and takes corrective action as necessary, by sending messages to respond to the environment, activating functions, making changes, and capturing state information.

Amazon EventBridge

- Amazon EventBridge is a serverless event bus service that you can use to connect your applications with data from a variety of sources. EventBridge delivers a stream of real-time data from your applications, software as a service (SaaS) application, and AWS services to targets such as AWS Lambda functions, HTTP invocation endpoints using API destinations, or event buses in other AWS accounts.
- EventBridge was formerly called Amazon CloudWatch Events. The default event bus and the rules you created in CloudWatch Events also display in the EventBridge console. EventBridge uses the same CloudWatch Events API, so your code that uses the CloudWatch Events API stays the same. New features added to EventBridge are not added to CloudWatch Events.
- Your account has a default event bus which receives events from AWS services, and you can create custom event buses to send or receive events from a different account or Region.
- EventBridge receives an **event**, an indicator of a change in environment, and applies a **rule** to route the event to a **target**. Rules match events to targets based on either the structure of the event, called an **event pattern**, or on **a schedule**. For example, when an Amazon EC2 instance changes from pending to running, you can have a rule that sends the event to a Lambda function.
- All events that come to EventBridge are associated with an event bus.

API Gateway

- API Gateway is a managed service from AWS which allows the creation of API Endpoints, Resources & Methods.
- The API gateway integrates with other AWS services - and can even access some without the need for dedicated compute.
- It serves as a core component of many serverless architectures using Lambda as event-driven and on-demand backing for methods.
- It can also connect to legacy monolithic applications and act as a stable API endpoint during an evolution from a monolith to microservices and potentially through to serverless.
- it easy for developers to create, publish, maintain, monitor, and secure APIs at any scale.
- API Gateway is a managed API Endpoint Service

- Create, Publish, Monitor and Secure APIs as a service
- Billed based on Number of API Calls, Data Transfer and additional performance features such as caching.
- Can be used directly for serverless architecture or during a architecture evolution.

Serverless Architecture

- The Serverless architecture is an evolution/combination of other popular architectures such as event-driven and microservices.
- It aims to use 3rd party services where possible and FAAS products for any on-demand computing needs.
- Using a serverless architecture means little to no base costs for an environment - and any cost incurred during operations scale in a way with matches the incoming load.
- By using a serverless architecture, your developers can focus on their core product instead of worrying about managing and operating servers or runtimes, either in the cloud or on-premises. This reduced overhead lets developers reclaim time and energy that can be spent on developing great products which scale and that are reliable.
- Serverless isn't one single thing.
- You manage few, if any servers - low overhead
- Applications are a collection of small & specialised functions
- it is Stateless and Ephemeral environment
- Event-driven... consumption only when being used.
- FaaS is used where possible for compute functionality
- Managed services are used where possible.

Simple Notification Service

Amazon Simple Notification Service (Amazon SNS) is a managed service that provides message delivery from publishers to subscribers (also known as producers and consumers). Publishers communicate asynchronously with subscribers by sending messages to a topic, which is a logical access point and communication channel. Clients can subscribe to the SNS topic and receive published messages using a supported endpoint type, such as Amazon Kinesis Data Firehose, Amazon SQS, AWS Lambda, HTTP, email, mobile push notifications, and mobile text messages (SMS).

- The Simple Notification Service or SNS is a PUB SUB style notification system which is used within AWS products and services but can also form an essential part of serverless, event-driven and traditional application architectures.
- Publishers send messages to TOPICS
- Subscribers receive messages SENT to TOPICS.
- SNS supports a wide variety of subscriber types including other AWS services such as LAMBDA and SQS.
- It is public AWS service - network connectivity with Public Endpoint
- Coordinates the sending and delivery of messages
- Messages are <= 256KB payloads
- SNS Topics are the base entity of SNS - permissions and configuration
- A Publisher sends messages to a TOPIC
- TOPIC have Subscribers which receive messages
- SNS used across AWS for notifications - e.g., CloudWatch & CloudFormation

Step Functions

Some problems with Lambda

- Lambda is FaaS
- 15-minutes max execution time

- Can be chained together
- Gets messy at scale
- Runtime Environments are stateless

Step Functions is a serverless orchestration service that lets you combine AWS Lambda functions and other AWS services to build business-critical applications. Through Step Functions' graphical console, you see your application's workflow as a series of event-driven steps.

Step Functions is based on state machines and tasks. A state machine is a workflow. A task is a state in a workflow that represents a single unit of work that another AWS service performs. Each step in a workflow is a state.

Step functions is a product which lets you build long running serverless workflow-based applications within AWS which integrate with many AWS services.

States Machines

- Serverless workflow: START ->STATES -> END
- When you create a state machine, you can select a Type of either **Standard (default) or Express**. In both cases, you define your state machine using the Amazon States Language (ASL).
- State machines are defined using JSON text that represents a structure containing the following fields.
- IAM used for permissions

States

- Individual states can make decisions based on their input, perform actions, and pass output to other states. In AWS Step Functions you define your workflows in the Amazon States Language. The Step Functions console provides a graphical representation of that state machine to help visualize your application logic.
- States are elements in your state machine. A state is referred to by its name, which can be any string, but which must be unique within the scope of the entire state machine.
- States can perform a variety of functions in your state machine:
    - Do some work in your state machine (a **Task** state)
    - Make a choice between branches of execution (a **Choice** state)
    - Stop an execution with a failure or success (a **Fail** or **Succeed** state)
    - Simply pass its input to its output or inject some fixed data (a **Pass** state)
    - Provide a delay for a certain amount of time or until a specified time/date (a **Wait** state)
    - Begin parallel branches of execution (a **Parallel** state)
    - Dynamically iterate steps (a **Map** state)

Simple Queue Service

- Amazon Simple Queue Service (SQS) is a fully managed message queuing service that enables you to decouple and scale microservices, distributed systems, and serverless applications.
- SQS eliminates the complexity and overhead associated with managing and operating message-oriented middleware, and empowers developers to focus on differentiating work.
- Using SQS, you can send, store, and receive messages between software components at any volume, without losing messages or requiring other services to be available.
- SQS offers two types of message queues. Standard queues offer maximum throughput, best-effort ordering, and at-least-once delivery.
- SQS FIFO queues are designed to guarantee that messages are processed exactly once, in the exact order that they are sent.
- FIFO (performance) 3,000 messages per second with batching, or up to 300 messages per secound without.

- Billed based on 'requests'
- 1 request = 1-10 messages up to 256KB total
- Encryption at rest (KMS) & in-transit

Amazon Kinesis

- Amazon Kinesis makes it easy to collect, process, and analyse real-time, streaming data so you can get timely insights and react quickly to new information.
- Amazon Kinesis offers key capabilities to cost-effectively process streaming data at any scale, along with the flexibility to choose the tools that best suit the requirements of your application.
- With Amazon Kinesis, you can ingest real-time data such as video, audio, application logs, website clickstreams, and IoT telemetry data for machine learning, analytics, and other applications.
- It enables you to process and analyse data as it arrives and respond instantly instead of having to wait until all your data is collected before the processing can begin.
- Kinesis data streams are a streaming service within AWS designed to ingest large quantities of data and allow access to that data for consumers.
- Kinesis is ideal for dashboards and large-scale real-time analytics needs.
- Kinesis data firehose allows the long-term persistent storage of kinesis data onto services like S3.
- Producers send data into a kinesis stream
- Streams can scale from low to near infinite data rates
- Public service and highly available by design
- Streams store a 24-hurs moving window of data
- Multiple consumers access data from that moving window.

Details	Kinesis Data Streams	SQS
Purpose	Streaming service to handle Realtime processing of big data	Messaging service- message queue to store messages between distributed application components
Durability	Up to retention period - delete on expiry	Up to retention period - delete by consumer
Scaling	Manual- Increasing the shards	Fully manged
Message Ordering	Ordered within the Shard	Standard - best effort FIFO - Ordered within the message group
Max Data Retention	7 days	14 days
Message Size	1MB	256KB - Can be extended using SQS Extended Client
Replay Supported	Yes	No
Delivery	Multiple consumers per shard - Consumers can read the same data	Multiple consumers per Queue - but message is consumed by one consumer at a time

Amazon Cognito - User and Identity Pools

Amazon Cognito provides authentication, authorization, and user management for your web and mobile apps. Your users can sign in directly with a user name and password, or through a third party such as Facebook, Amazon, Google or Apple.

The two main components of Amazon Cognito are user pools and identity pools.

User pools are user directories that provide sign-up and sign-in options for your app users.

Identity pools enable you to grant your users access to other AWS services. You can use identity pools and user pools separately or together.

User pools

A user pool is a user directory in Amazon Cognito. With a user pool, your users can sign in to your web or mobile app through Amazon Cognito, or federate through a third-party identity provider (IdP). Whether your users sign in directly or through a third party, all members of the user pool have a directory profile that you can access through an SDK.

User pools provide:

- Sign-up and sign-in services.
- A built-in, customizable web UI to sign in users.
- Social sign-in with Facebook, Google, Login with Amazon, and Sign in with Apple, and through SAML and OIDC identity providers from your user pool.
- User directory management and user profiles.
- Security features such as multi-factor authentication (MFA), checks for compromised credentials, account takeover protection, and phone and email verification.
- Customized workflows and user migration through AWS Lambda triggers.

Identity pools

With an identity pool, your users can obtain temporary AWS credentials to access AWS services, such as Amazon S3 and DynamoDB. Identity pools support anonymous guest users, as well as the following identity providers that you can use to authenticate users for identity pools:

- Amazon Cognito user pools
- Social sign-in with Facebook, Google, Login with Amazon, and Sign in with Apple
- OpenID Connect (OIDC) providers
- SAML identity providers
- Developer authenticated identities
- To save user profile information, your identity pool needs to be integrated with a user pool.

Amazon Cognito is available in multiple AWS Regions worldwide. In each Region, Amazon Cognito is distributed across multiple Availability Zones.

MODULE 15 - GLOBAL CONTENT DELIVERY AND OPTIMIZATION

CloudFront Architecture Basics

- Amazon CloudFront is a fast content delivery network (CDN) service that securely delivers data, videos, applications, and APIs to customers globally with low latency, high transfer speeds, all within a developer-friendly environment.
- Amazon CloudFront is a web service that speeds up distribution of your static and dynamic web content, such as .html, .css, .js, and image files, to your users.
- CloudFront delivers your content through a worldwide network of data centers called edge locations.
- When a user requests content that you're serving with CloudFront, the request is routed to the edge location that provides the lowest latency (time delay), so that content is delivered with the best possible performance.
  - If the content is already in the edge location with the lowest latency, CloudFront delivers it immediately.
  - If the content is not in that edge location, CloudFront retrieves it from an origin that you've defined—such as an Amazon S3 bucket, a MediaPackage channel, or an HTTP server (for example, a web server) that you have identified as the source for the definitive version of your content.
- CloudFront is a global object cache (CDN)
- Content is cached in locations close to customers
- Provides lower latency and higher throughput
- Load on the content server is decreased
- It can handle static and dynamic content.

CloudFront Terms

- **Origin** - The source location of your content
- **Distribution** - The 'configuration' unit of CloudFront
- **Edge Location** - Local infrastructure which hosts a cache of your data.
- **Regional Edge Cache** - Larger version of an edge location. Provides another layer of caching.

AWS Certificate Manager (ACM)

- The AWS certificate Manage is a service which allows the creation, management and renewal of certificates. It allows deployment of certificates onto supported AWS services such as CloudFront and ALB.
- ACM certificates can secure singular domain names, multiple specific domain names, wildcard domains, or combinations of these.
- ACM wildcard certificates can protect an unlimited number of subdomains.
- The validity period for ACM certificates is 13 months (395 days).
- HTTP - Simple and Insecure
- HTTPs - SSL/TLS layer of encryption added to HTTP
- Data is encrypted in-transit
- Certificates Prove identity and Signed by a trusted authority

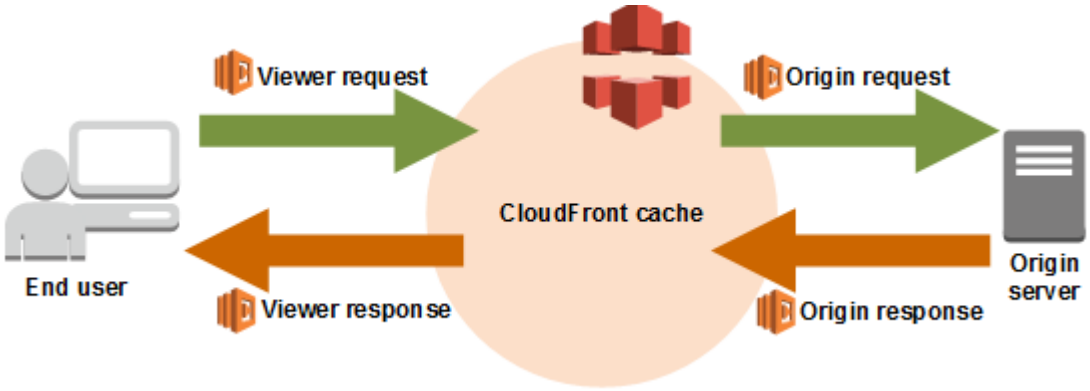
Securing CF and S3 using Origin Access Identity (OAI)

- Origin Access Identities are a feature where virtual identities can be created, associated with a CloudFront Distribution and deployed to edge locations.
- Access to an s3 bucket can be controlled by using these OAI's - allowing access from an OAI, and using an implicit DENY for everything else.
- They are generally used to ensure no direct access to S3 objects is allowed when using private CF Distributions.

For more details check out:
https://docs.AWS.amazon.com/AmazonCloudFront/latest/DeveloperGuide/private-content-restricting-access-to-s3.html

Lambda@Edge

- Lambda@Edge allows CloudFront to run lambda function at CloudFront edge locations to modify traffic between the viewer and edge location and edge locations and origins.
- Lambda@Edge is a feature of Amazon CloudFront that lets you run code closer to users of your application, which improves performance and reduces latency.
- With Lambda@Edge, you don't have to provision or manage infrastructure in multiple locations around the world. You pay only for the compute time you consume - there is no charge when your code is not running.
- With Lambda@Edge, you can enrich your web applications by making them globally distributed and improving their performance — all with zero server administration. Lambda@Edge runs your code in response to events generated by the Amazon CloudFront content delivery network (CDN).
- Just upload your code to AWS Lambda, which takes care of everything required to run and scale your code with high availability at an AWS location closest to your end user.
- You can run lightweight Lambda at edge locations
- Adjust data between the Viewer & Origin
- Currently supported Node.js and Python
- Run in the AWS Public Space (Not VPC)
- Layers are not supported.
- You can use Lambda functions to change CloudFront requests and responses at the following points:
    - After CloudFront receives a request from a viewer (viewer request)
    - Before CloudFront forwards the request to the origin (origin request)
    - After CloudFront receives the response from the origin (origin response)
    - Before CloudFront forwards the response to the viewer (viewer response)



Lambda@Edge Use cases:

- Redirecting Viewer Requests to a Country-Specific URL
- Serving Different Versions of an Object Based on the Device
- Content-Based Dynamic Origin Selection
- Using an Origin-Request Trigger to Change From a Custom Origin to an Amazon S3 Origin
- Using an Origin-Request Trigger to Gradually Transfer Traffic From One Amazon S3 Bucket to Another.

AWS Global Accelerator

- AWS Global Accelerator is designed to improve global network performance by offering entry point onto the global AWS transit network as close to customers as possible using ANycast IP addresses.
- AWS Global Accelerator is a service in which you create accelerators to improve the performance of your applications for local and global users. Depending on the type of accelerator you choose, you can gain additional benefits.
    - By using a standard accelerator, you can improve availability of your internet applications that are used by a global audience. With a standard accelerator, Global Accelerator directs traffic over the AWS global network to endpoints in the nearest Region to the client.
    - By using a custom routing accelerator, you can map one or more users to a specific destination among many destinations.
- Global Accelerator is a global service that supports endpoints in multiple AWS Regions, which are listed in the AWS Region Table.
- By default, Global Accelerator provides you with two static IP addresses that you associate with your accelerator.
- With a standard accelerator, instead of using the IP addresses that Global Accelerator provides, you can configure these entry points to be IPv4 addresses from your own IP address ranges that you bring to Global Accelerator. The static IP addresses are anycast from the AWS edge network.
- Global Accelerator Components
- AWS Global Accelerator include the following components
    - Static IP addresses
    - Accelerator
    - DNS Name
    - Network Zone
    - Listener
    - Endpoint Group

Static IP addresses: By default, Global Accelerator provides you with two Static IP addresses that you associate with your accelerator. OR you can bring your own.

Accelerator: An Accelerator directs traffic to optimal endpoints over the AWS Global network to improve the availability and performance of your internet applications Each Accelerator include one or more listeners.

DNS Name:

- Global Accelerator assigns each accelerator a default Domain Name System (DNS) name - those points to the static IP addresses that Global Accelerator assigns to you.
- Depending on the use case, you can use your accelerator's static IP addresses or DNS name to route traffic to your accelerator, or set up DNS records to route traffic using your own custom domain name.

Network Zone:

- A Network Zone services the static IP addresses for your accelerator from a unique IP subnet. Similar to an AWS AZ, a Network Zone is an isolated unit with its own set of physical infrastructure.
- When you configure an accelerator, by default Global Accelerator allocates two IPv4 addresses for it. if one IP address from a Network Zone becomes unavailable due to IP address bocking by certain client networks, or network disruptions, client applications can retry on the healthy static IP address from the other isolated Network Zone.

Listener:

- A Listener processes inbound connection from clients to Global Accelerators, based on the port (or port range) and protocol that you configure.
- Global Accelerator supports both TCP and UDP protocols. Each Listener has one or more endpoint groups associated with it, and traffic is forwarded to endpoints in one of the groups.
- You associate endpoint groups with listeners by specifying the Regions that you want to distribute traffic to. Traffic is distributed to optimal endpoints within the endpoint group associated with a Listener

Endpoint group

- Each endpoint group is associated with a specific AWS Region.
- Endpoint groups include one or more endpoints in the Region.
- You can increase or reduce the percentage of traffic that would be otherwise directed to an endpoint group by adjusting a setting called a traffic dial.
- The traffic dial lets you easily do performance testing or blue/green deployment testing for new releases across different AWS Regions

Know what a Global Accelerator is and where you would use it.

- AWS Global Accelerator is a service in which you create accelerators to improve availability and performance of your applications for local and global users.
- You can assign two static IP addresses (or alternatively you can bring your own).
- You can control traffic using traffic dials. This is done within the endpoint group.

MODULE 16 - ADVANCED VPC NETWORKING

VPC Flow Logs

- VPC Flow Logs is a feature that enables you to capture information about the IP traffic going to and from network interfaces in your VPC. Flow log data can be published to Amazon CloudWatch Logs or Amazon S3. After you've created a flow log, you can retrieve and view its data in the chosen destination.
- Flow logs can help you with a number of tasks, such as:
    - Diagnosing overly restrictive security group rules
    - Monitoring the traffic that is reaching your instance
    - Determining the direction of the traffic to and from the network interfaces
- Flow log data is collected outside of the path of your network traffic, and therefore does not affect network throughput or latency. You can create or delete flow logs without any risk of impact to network performance.
- You can create a flow log for a VPC, a subnet, or a network interface. If you create a flow log for a subnet or VPC, each network interface in that subnet or VPC is monitored.
- Flow log data for a monitored network interface is recorded as flow log records, which are log events consisting of fields that describe the traffic flow.
- VPC Flow logs is a feature allowing the monitoring of traffic flow to and from interfaces within a VPC.
- VPC Flow logs can be added at a VPC, Subnet or Interface level.
- Flow Logs DON'T monitor packet contents ... that requires a packet sniffer.
- Flow Logs can be stored on S3 or CloudWatch Logs.

Remember the following:

- you cannot enable flow logs for VPC that are peered with your VPC unless the peer is in your account. you can tag flow logs. After you have created a flow log, you cannot change its configuration; for example, you can't associate a different IAM role with the flow log.
- Not all IP traffic is monitored;
- traffic generated by instances when they contact the Amazon DNS server. if you use your own DNS server then all traffic to that DNS Server is logged.
- Traffic generated by a Windows instance for Amazon Windows licence activation.
- Traffic to and from 169.254.169.254 for instance metadata.
- DHCP traffic.
- Traffic to the reserved IP address for the default VPC router.

Egress-Only Internet gateway

- An egress-only internet gateway is a horizontally scaled, redundant, and highly available VPC component that allows outbound communication over IPv6 from instances in your VPC to the internet, and prevents the internet from initiating an IPv6 connection with your instances.
- Egress-Only internet gateways allow outbound (and response) only access to the public AWS services and Public Internet for IPv6 enabled instances or other VPC based services.
- An egress-only internet gateway is stateful: it forwards traffic from the instances in the subnet to the internet or other AWS services, and then sends the response back to the instances.
- An egress-only internet gateway has the following characteristics:
    - You cannot associate a security group with an egress-only internet gateway. You can use security groups for your instances in the private subnet to control the traffic to and from those instances.
    - You can use a network ACL to control the traffic to and from the subnet for which the egress-only internet gateway routes traffic.

- NAT allows private IPs to access public networks. without allowing externally initiated connection (IN)
- With IPv6 all IPs are public
- Internet Gateway (IPv6) allows all IPs IN and OUT
- Egress-Only is outbound-only for IPv6

VPC Endpoints (Gateway)

- A VPC endpoint enables private connections between your VPC and supported AWS services and VPC endpoint services powered by AWS PrivateLink. AWS PrivateLink is a technology that enables you to privately access services by using private IP addresses. Traffic between your VPC and the other service does not leave the Amazon network. A VPC endpoint does not require an internet gateway, virtual private gateway, NAT device, VPN connection, or AWS Direct Connect connection. Instances in your VPC do not require public IP addresses to communicate with resources in the service.
- VPC endpoints are virtual devices. They are horizontally scaled, redundant, and highly available VPC components. They allow communication between instances in your VPC and services without imposing availability risks.
- There are two types of VPC Endpoints:
    - Interface Endpoints
    - Gateway Endpoints
- Gateway Endpoints is an elastic network interface with a private IP address from the IP address range of your subnet. Gateway Load Balancer endpoints are powered by AWS PrivateLink. This type of endpoint serves as an entry point to intercept traffic and route it to a service that you've configured using Gateway Load Balancers, for example, for security inspection. You specify a Gateway Load Balancer endpoint as a target for a route in a route table. Gateway Load Balancer endpoints are supported for endpoint services that are configured for Gateway Load Balancers only.
- Currently Gateway Endpoint support
    - Amazon S3
    - DynamoDB
- It provides private access to S3 and DynamoDB
- Prefix List added to route table => Gateway Endpoint
- Highly Available across all AZs in a region by default
- Endpoint policy is used to control what it can access
- Regional... can’t access cross region services
- Prevent Leaky Buckets - S3 Buckets can be set to private only by allowing access ONLY from a gateway endpoint.

VPC Endpoints (Interface)

- An interface endpoint is an elastic network interface with a private IP address from the IP address range of your subnet. It serves as an entry point for traffic destined to a supported AWS service or a VPC endpoint service. Interface endpoints are powered by AWS PrivateLink.
- Interface endpoints are used to allow private IP addressing to access public AWS services.
- S3 and DynamoDB are handled by gateway endpoints - other supported services are handled by interface endpoints.
- Unlike gateway endpoints - interface endpoints are not highly available by default - they are normal VPC network interfaces and should be placed 1 per AZ to ensure full HA.
- Network access controlled via Security Groups
- Endpoint Policies - restrict what can be done with the endpoint
- TCP and IPv4 ONLY
- Uses PrivateLink

VPC Peering

A VPC peering connection is a networking connection between two VPCs that enables you to route traffic between them using private IPv4 addresses or IPv6 addresses. Instances in either VPC can communicate with each other as if they are within the same network. You can create a VPC peering connection between your own VPCs, or with a VPC in another AWS account. The VPCs can be in different regions (also known as an inter-region VPC peering connection).

- VPC peering is a software define and logical networking connection between two VPC's
- They can be created between VPCs in the same or different accounts and the same or different regions.
- Direct encrypted network link between two VPCs
- Works same/cross-region and same/cross-account
- Same region Security groups can reference peer SGs
- VPC peering does NOT support transitive peering
- Routing configuration is needed, SGs & NACLs can filter.

MODULE 17 - HYBRID ENVIRONMENTS AND MIGRATION

AWS Site-to-Site VPN

- By default, instances that you launch into an Amazon VPC can't communicate with your own (remote) network. You can enable access to your remote network from your VPC by creating an AWS Site-to-Site VPN (Site-to-Site VPN) connection, and configuring routing to pass traffic through the connection.
- Although the term VPN connection is a general term, in this documentation, a VPN connection refers to the connection between your VPC and your own on-premises network. Site-to-Site VPN supports Internet Protocol security (IPsec) VPN connections.
- Your Site-to-Site VPN connection is either an AWS Classic VPN or an AWS VPN.
- A logical connection between a VPC and on-premises network encrypted using IPsec, running over the public internet.
- Full HA - if you design and implement it correctly
- Quick to provision... less than an hour
- The following are the key concepts for Site-to-Site VPN:
    - **VPN connection:** A secure connection between your on-premises equipment and your VPCs.
    - **VPN tunnel:** An encrypted link where data can pass from the customer network to or from AWS. Each VPN connection includes two VPN tunnels which you can simultaneously use for high availability.
    - **Customer gateway:** An AWS resource which provides information to AWS about your customer gateway device.
    - **Customer gateway device:** A physical device or software application on your side of the Site-to-Site VPN connection.
    - **Virtual private gateway:** The VPN concentrator on the Amazon side of the Site-to-Site VPN connection. You use a virtual private gateway or a transit gateway as the gateway for the Amazon side of the Site-to-Site VPN connection.
    - **Transit gateway:** A transit hub that can be used to interconnect your VPCs and on-premises networks. You use a transit gateway or virtual private gateway as the gateway for the Amazon side of the Site-to-Site VPN connection.

VPN Considerations

- Speed Limitations ~ **1.25 Gbps**
- latency Considerations - inconsistent, public internet
- Cost - **AWS hourly cost**, GB out cost, data cap (on-premises)
- Speed of setup - hours... all software configuration
- Can be used as a **backup** for Direct Connect (DX)
- Can be used with Direct Connect (DX)

Direct Connect

- AWS Direct Connect makes it easy to establish a dedicated connection from an on-premises network to one or more VPCs in the same region. Using private VIF on AWS Direct Connect, you can establish private connectivity between AWS and your data center, office, or colocation environment.
- Multiple dynamically routed AWS Direct Connect connections are necessary to support high availability.
- The following are the key components that you use for AWS Direct Connect:

Connections

- Create a connection in an AWS Direct Connect location to establish a network connection from your premises to an AWS Region.

Virtual interfaces

- Create a virtual interface to enable access to AWS services. A public virtual interface enables access to public services, such as Amazon S3. A private virtual interface enables access to your VPC.
- To use AWS Direct Connect in an AWS Direct Connect location, your network must meet one of the following conditions:
    - Your network is colocated with an existing AWS Direct Connect location. For more information about available AWS Direct Connect locations.
    - You are working with an AWS Direct Connect partner who is a member of the AWS Partner Network (APN).
    - You are working with an independent service provider to connect to AWS Direct Connect.
    - Your network must use single-mode fiber with a 1000BASE-LX (1310 nm) transceiver for 1 gigabit Ethernet, a 10GBASE-LR (1310 nm) transceiver for 10 gigabits, or a 100GBASE-LR4 for 100 gigabit Ethernet.
    - Auto-negotiation for the port must be disabled. Auto-negotiation is supported only if the port speed is 1 Gbps. Port speed and full-duplex mode must be configured manually.
    - 802.1Q VLAN encapsulation must be supported across the entire connection, including intermediate devices.
    - Your device must support Border Gateway Protocol (BGP) and BGP MD5 authentication.

DX Considerations

- Takes MUCH longer to provision VS VPC
- DX port provisioning is quick... the cross connect takes longer
- Extension to premises can take weeks/months
- Use VPN first...then replace with DX (or leave as backup)
- Faster... 40 Gbps with Aggregation
- Low consistent latency, doesn’t use business bandwidth
- NO ENCRYPTION

Direct Connect Resilience

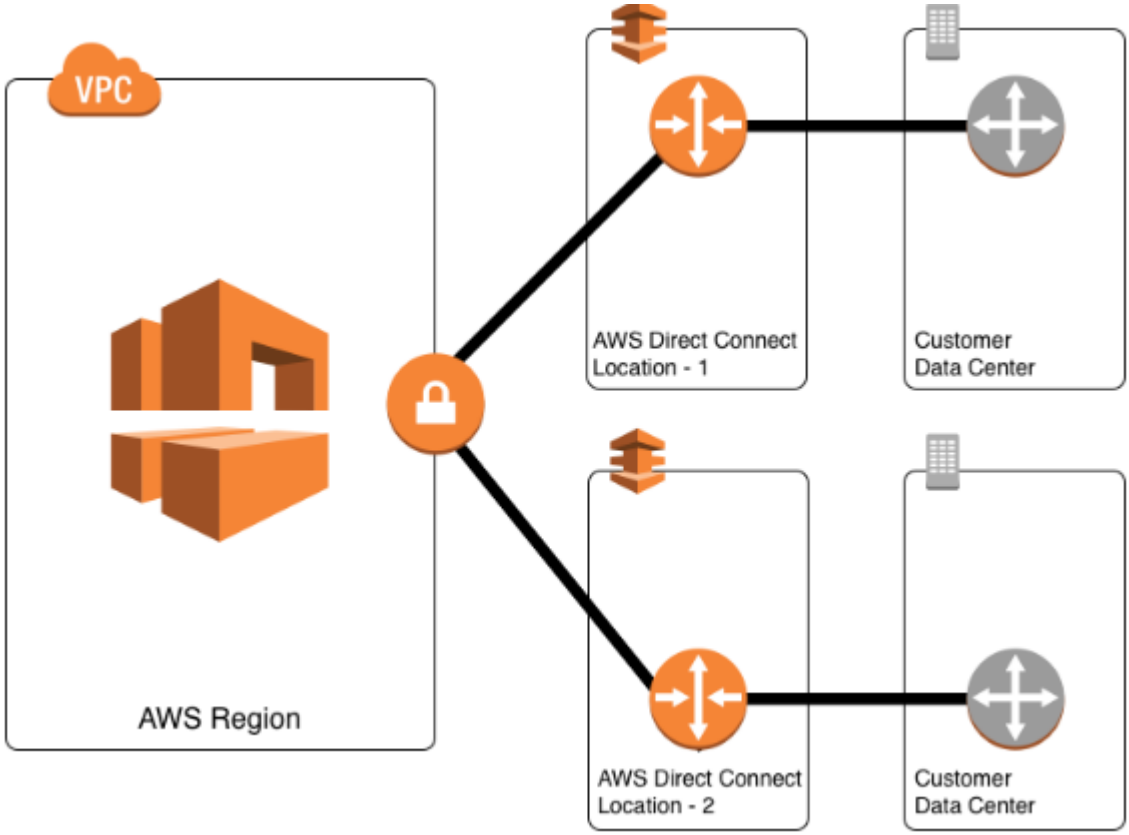
Using AWS Direct Connect for High Resiliency

Amazon Web Services (AWS) offers customers the ability to achieve highly resilient network connections between Amazon Virtual Private Cloud (Amazon VPC) and their on-premises infrastructure. This capability extends customer access to AWS resources in a reliable, scalable, and cost-effective way. This page documents our best practices for ensuring high resiliency with AWS Direct Connect.

Recommended Best Practices

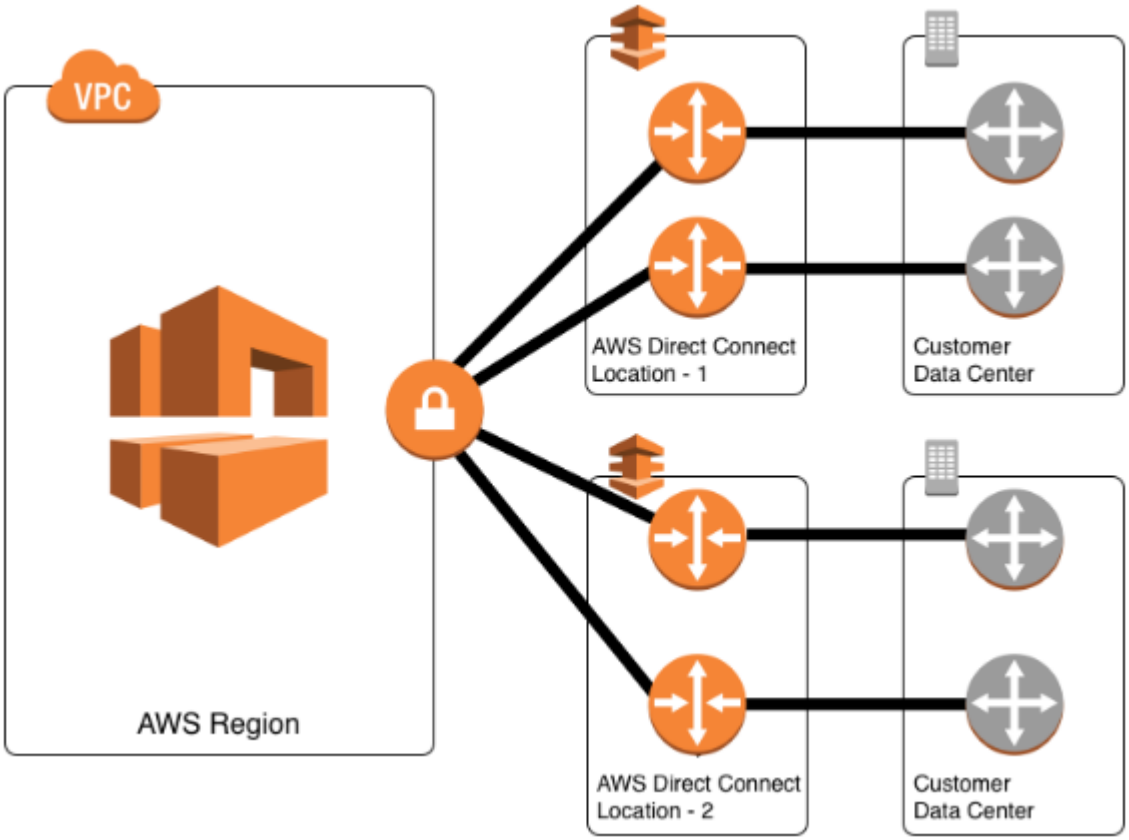
Highly resilient, fault-tolerant network connections are key to a well-architected system. AWS recommends connecting from multiple data centers for physical location redundancy. When designing remote connections, consider using redundant hardware and telecommunications providers. Additionally, it is a best practice to use dynamically routed, active/active connections for automatic load balancing and failover across redundant network connections. Provision sufficient network capacity to ensure that the failure of one network connection does not overwhelm and degrade redundant connections.

High Resiliency for Critical Workloads



For critical production workloads that require high resiliency, it is recommended to have one connection at multiple locations. As shown in the figure above, such a topology ensures resilience to connectivity failure due to a fiber cut or a device failure as well as a complete location failure. You can use Direct Connect Gateway to access any AWS Region (except AWS Regions in China) from any AWS Direct Connect location.

Maximum Resiliency for Critical Workloads



Maximum resilience is achieved by separate connections terminating on separate devices in more than one location. This configuration offers customers maximum resilience to failure. As shown in the figure above, such a topology provides resilience to device failure, connectivity failure, and complete location failure. You can use Direct Connect Gateway to access any AWS Region (except AWS Regions in China) from any AWS Direct Connect locations.

AWS Managed VPN connections as a backup for the Direct Connect

Some AWS customers would like the benefits of one or more AWS Direct Connect connections for their primary connectivity to AWS, coupled with a lower-cost backup connection. To achieve this objective, they can establish AWS Direct Connect connections with a VPN backup.

It is important to understand that AWS Managed VPN supports up to 1.25 Gbps throughput per VPN tunnel and does not support Equal Cost Multi Path (ECMP) for egress data path in the case of multiple AWS Managed VPN tunnels terminating on the same VGW. Thus, we do not recommend customers use AWS Managed VPN as a backup for AWS Direct Connect connections with speeds greater than 1 Gbps.

Transit Gateway

- AWS Transit Gateway provides a hub and spoke design for connecting VPCs and on-premises networks as a fully managed service without requiring you to provision virtual appliances like the Cisco CSRs. No VPN overlay is required, and AWS manages high availability and scalability.
- Transit Gateway enables customers to connect thousands of VPCs. You can attach all your hybrid connectivity (VPN and Direct Connect connections) to a single Transit Gateway— consolidating and controlling your organization's entire AWS routing configuration in one place.
- Transit Gateway controls how traffic is routed among all the connected spoke networks using route tables. This hub and spoke model simplify management and reduces operational costs because VPCs only connect to the Transit Gateway to gain access to the connected networks.
- Transit Gateway is a regional resource and can connect thousands of VPCs within the same AWS Region.

- You can create multiple Transit Gateways per Region, but Transit Gateways within an AWS Region cannot be peered, and you can connect to a maximum of three Transit Gateways over a single Direct Connect Connection for hybrid connectivity.
- For these reasons, you should restrict your architecture to just one Transit Gateway connecting all your VPCs in a given Region, and use Transit Gateway routing tables to isolate them wherever needed. There is a valid case for creating multiple Transit Gateways purely to limit misconfiguration blast radius.
- Use AWS Resource Access Manager (RAM) to share a Transit Gateway for connecting VPCs across multiple accounts in your AWS Organization within the same Region.

Transit Gateway Considerations

- Supports transitive routing
- Can be used to create global networks
- Share between accounts using AWS RAM
- Peer with different regions... same or cross account
- Less complexity

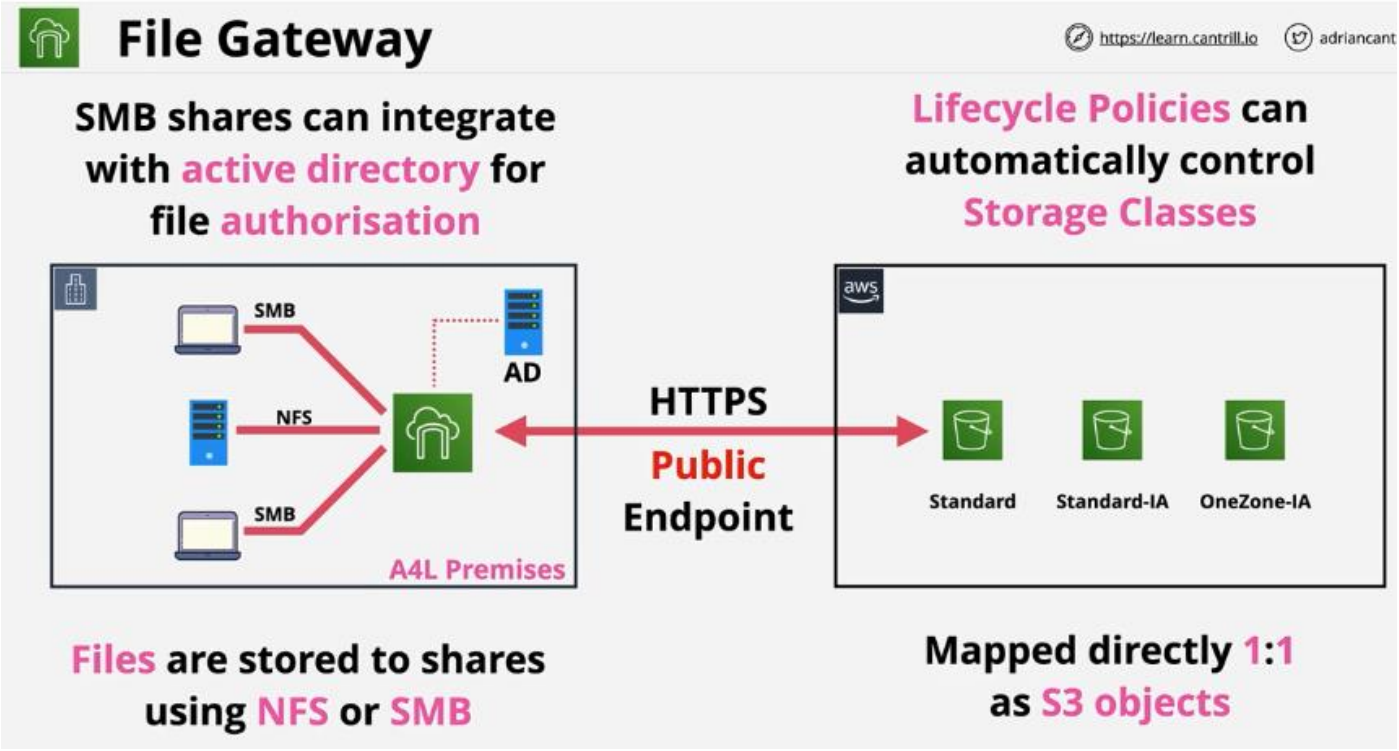
Storage gateway

AWS Storage Gateway connects an on-premises software appliance with cloud-based storage to provide seamless integration with data security features between your on-premises IT environment and the AWS storage infrastructure. You can use the service to store data in the Amazon Web Services Cloud for scalable and cost-effective storage that helps maintain data security.

AWS Storage Gateway offers file-based file gateways (Amazon S3 File and Amazon FSx File), volume-based (Cached and Stored), and tape-based storage solutions:

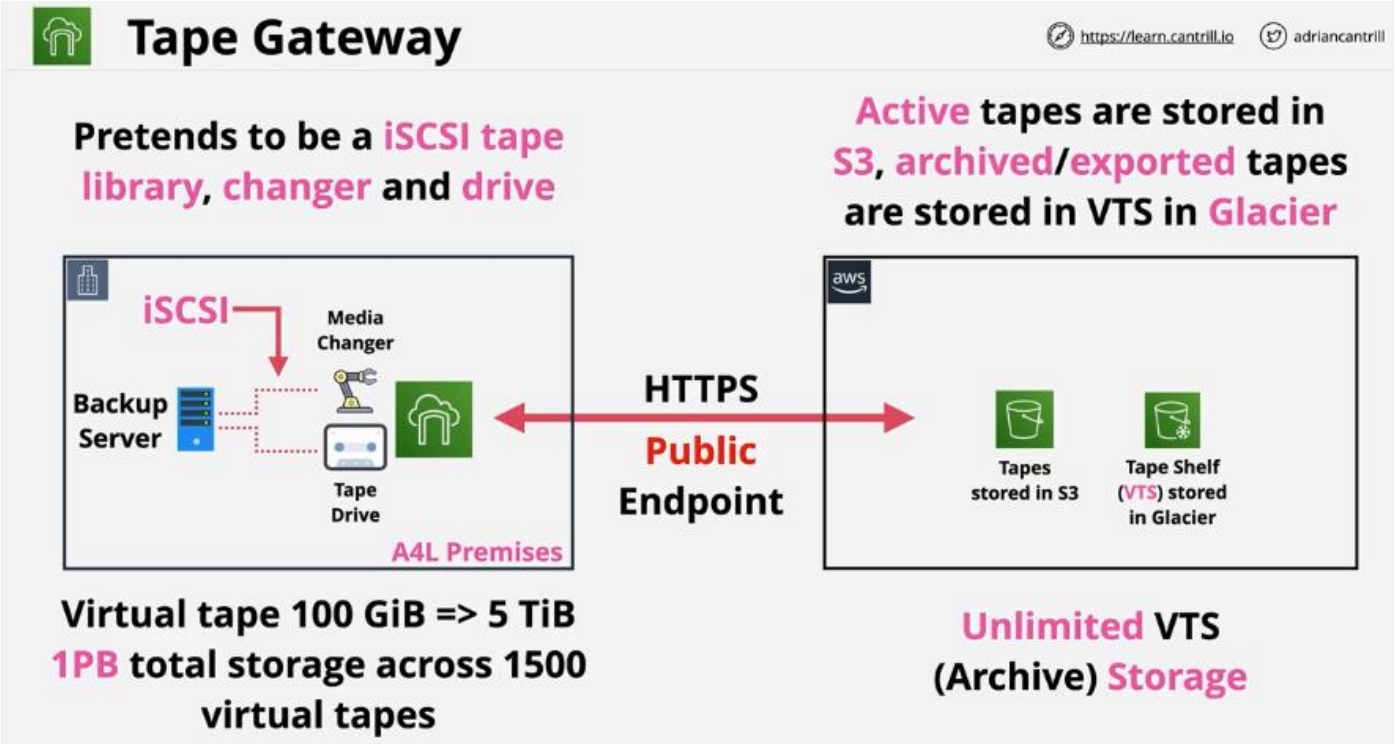
Amazon S3 File Gateway

Amazon S3 File Gateway supports a file interface into Amazon Simple Storage Service (Amazon S3) and combines a service and a virtual software appliance. By using this combination, you can store and retrieve objects in Amazon S3 using industry-standard file protocols such as Network File System (NFS) and Server Message Block (SMB).



Tape Gateway

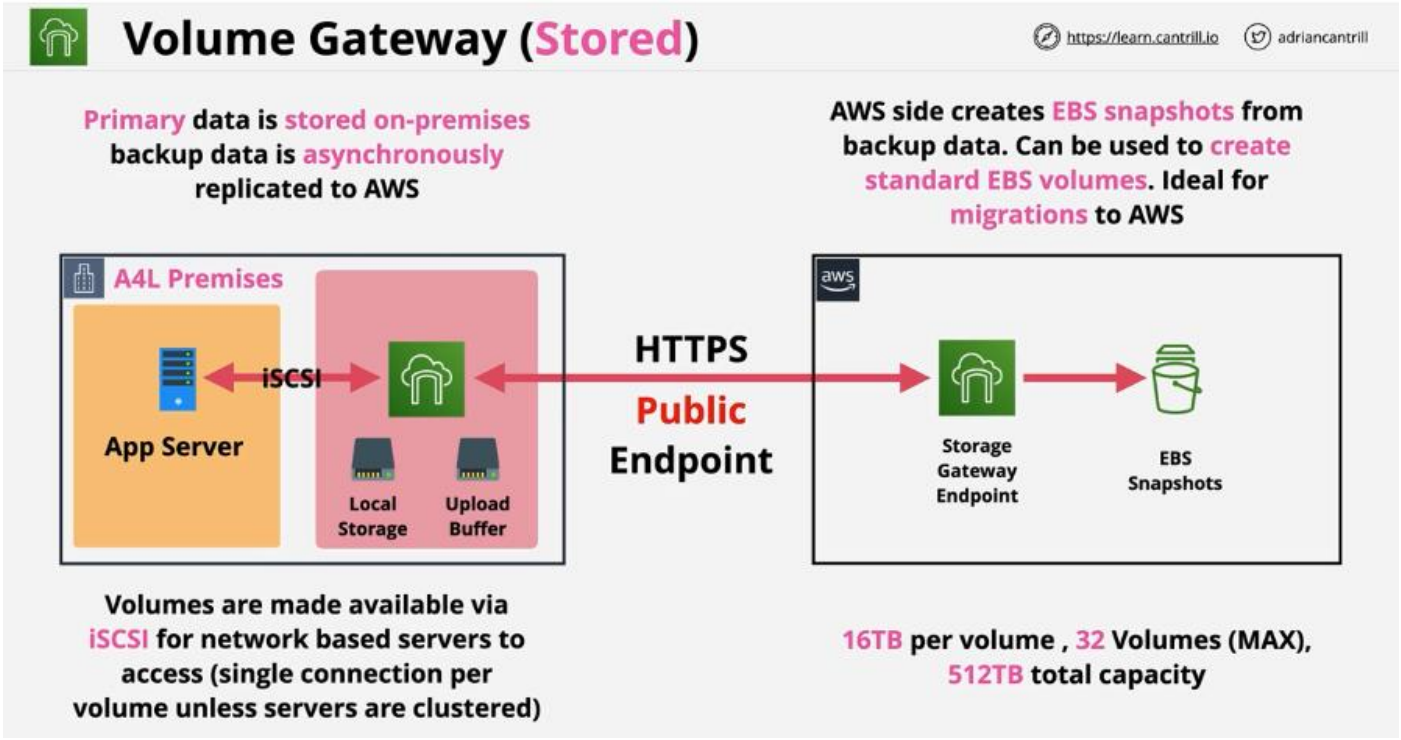
- A tape gateway provides cloud-backed virtual tape storage. The tape gateway is deployed into your on-premises environment as a VM running on VMware ESXi, KVM, or Microsoft Hyper-V hypervisor.
- With a tape gateway, you can cost-effectively and durably archive backup data in GLACIER or DEEP_ARCHIVE. A tape gateway provides a virtual tape infrastructure that scales seamlessly with your business needs and eliminates the operational burden of provisioning, scaling, and maintaining a physical tape infrastructure.



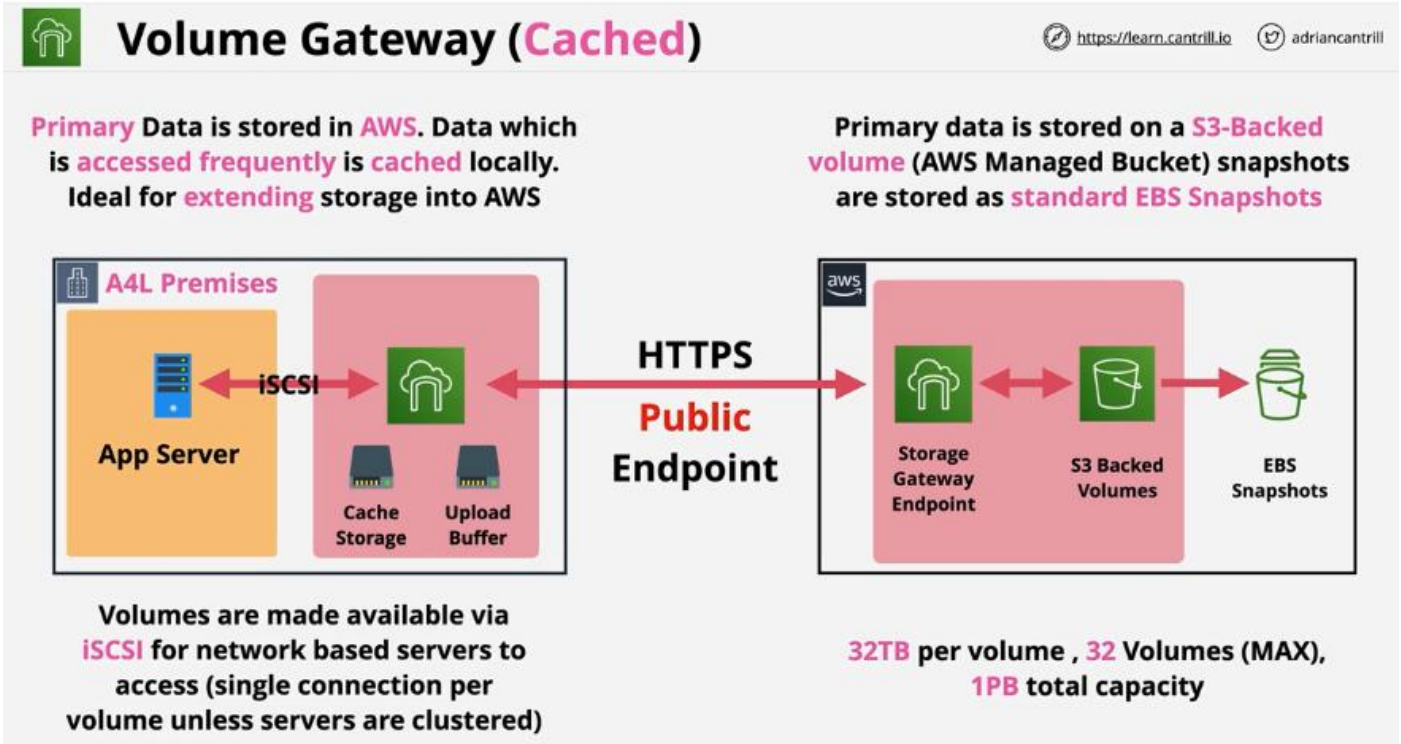
Volume Gateway

- A volume gateway provides cloud-backed storage volumes that you can mount as Internet Small Computer System Interface (iSCSI) devices from your on-premises application servers.
- The volume gateway is deployed into your on-premises environment as a VM running on VMware ESXi, KVM, or Microsoft Hyper-V hypervisor.
- The gateway supports the following volume configurations:

Stored volumes – If you need low-latency access to your entire dataset, first configure your on-premises gateway to store all your data locally. Then asynchronously back up point-in-time snapshots of this data to Amazon S3. This configuration provides durable and inexpensive offsite backups that you can recover to your local data center or Amazon Elastic Compute Cloud (Amazon EC2). For example, if you need replacement capacity for disaster recovery, you can recover the backups to Amazon EC2.



Cached volumes – You store your data in Amazon Simple Storage Service (Amazon S3) and retain a copy of frequently accessed data subsets locally. Cached volumes offer a substantial cost savings on primary storage and minimize the need to scale your storage on-premises. You also retain low-latency access to your frequently accessed data.



Snowball / Edge / Snowmobile

Snowball, Snowball Edge and Snowmobile are three parts of the same product family designed to allow the physical transfer of data between business locations and AWS.

Key Concepts

- Move large amount of data IN and OUT of AWS
- Physical storage such as suitcase or truck
- Ordered from AWS Empty, Load up, Return
- Ordered from AWS with data, empty & Return
- For the exam know which to use.

Snowball

Snowball is a petabyte-scale data transport solution that uses secure appliances to transfer large amounts of data into and out of the AWS cloud. Using Snowball addresses common challenges with large-scale data transfers including high network costs, long transfer times, and security concerns.

- Ordered from AWS, log a Job, Device Delivered (not instant)
- Data Encryption uses **KMS**
- **50TB** or **80TB** Capacity
- Gbps (**RJ45 1GBase-TX**) or **10Gbps (LR/SR)** Network
- **10TB** to **10PT** economical range (multiple devices)
- Multiple devices to multiple premises
- Only storage

Snowball Edge

Snowball Edge Storage Optimized devices provide both block storage and Amazon S3-compatible object storage, and 40 vCPUs. They are well suited for local storage and large scale-data transfer. Snowball Edge Compute Optimized devices provide 52 vCPUs, block and object storage, and an optional GPU for use cases like advanced machine learning and full motion video analysis in disconnected environments.

- Support Storage and Compute
- Larger capacity as compared to Snowball
- **Gbps (RJ45)**, 10/25 (SEP), 45/50/100 Gbps (QSEP+)
- Storage optimized (with EC2) - 80TB, 24 vCPU, 32 Gib RAM, 1 TB SSD
- Compute optimized - 100TB + 7.68 NVME, 52 vCPU and 208 Gib RAM
- Compute with GPU - same as above
- Ideal for remote sites or where data processing on ingestion is needed.

Snowmobile

AWS Snowmobile is an Exabyte-scale data transfer service used to move extremely large amounts of data to AWS. You can transfer up to 100PB per Snowmobile, a 45-foot-long ruggedized shipping container, pulled by a semi-trailer truck. Snowmobile makes it easy to move massive volumes of data to the cloud, including video libraries, image repositories, or even a complete data center migration. Transferring data with Snowmobile is more secure, fast and cost effective.

- Portable DC within a shipping container on a truck
- Special order
- Ideal for single location when 10 PB+ is required
- Up to 100PB per snowmobile
- Not economical for multi-site (Unless huge)
- Literally a Truck

Directory Service

What's a Directory?

- Stores objects (e.g., Users, Groups, Computers, Servers, File Shares) with a structure (domain/tree)
- Multiple trees can be grouped into a forest
- Commonly used in Windows Environments
- Sign in to multiple devices with the same username/password provides centralised management for assets
- Microsoft Active Directory Domain Services (AD DS)
- AD DS most popular, open-source alternatives (SAMBA)

Directory Service

- AWS Directory Service for Microsoft Active Directory, also known as AWS Managed Microsoft Active Directory (AD), enables your directory-aware workloads and AWS resources to use managed Active Directory (AD) in AWS.
- AWS Managed Microsoft AD is built on actual Microsoft AD and does not require you to synchronize or replicate data from your existing Active Directory to the cloud.
- You can use the standard AD administration tools and take advantage of the built-in AD features, such as Group Policy and single sign-on. With AWS Managed Microsoft AD, you can easily join Amazon EC2 and Amazon RDS for SQL Server instances to your domain, and use AWS End User Computing (EUC) services, such as Amazon WorkSpaces, with AD users and groups.
- It runs within a VPC
- To implement HA... deploy into multiple AZs.
- Some AWS services NEED a directory e.g., Amazon Workspaces.
- It can be isolated or integrated with existing on-premises system or act as a 'proxy' back to on-premises.

AWS Directory Service includes several directory types to choose from.

Simple AD

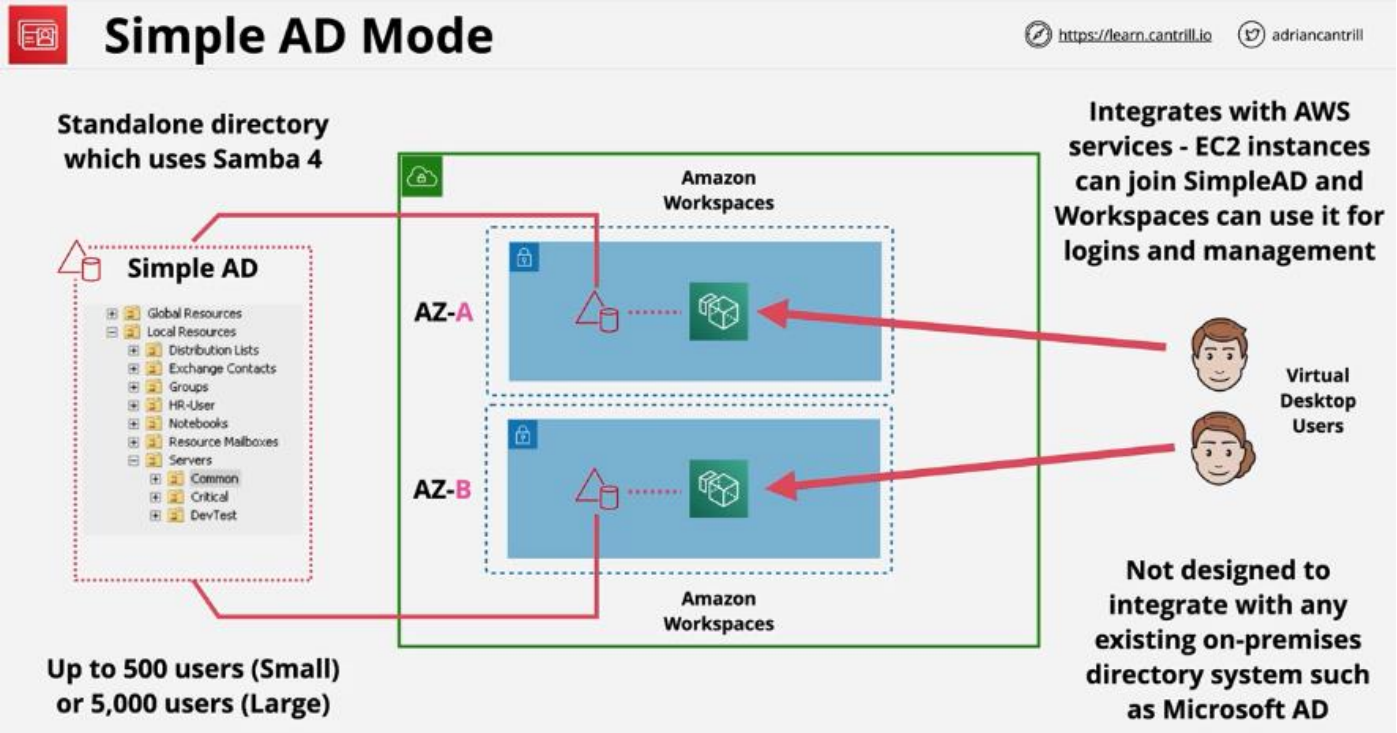
Simple AD is a Microsoft Active Directory–compatible directory from AWS Directory Service that is powered by Samba 4. Simple AD supports basic Active Directory features such as user accounts, group memberships, joining a Linux domain or Windows based EC2 instances, Kerberos-based SSO, and group policies. AWS provides monitoring, daily snap-shots, and recovery as part of the service.

Simple AD is a standalone directory in the cloud, where you create and manage user identities and manage access to applications. You can use many familiar Active Directory–aware applications and tools that require basic Active Directory features.

Simple AD does not support multi-factor authentication (MFA), trust relationships, DNS dynamic update, schema extensions, communication over LDAPS, PowerShell AD cmdlets, or FSMO role transfer.

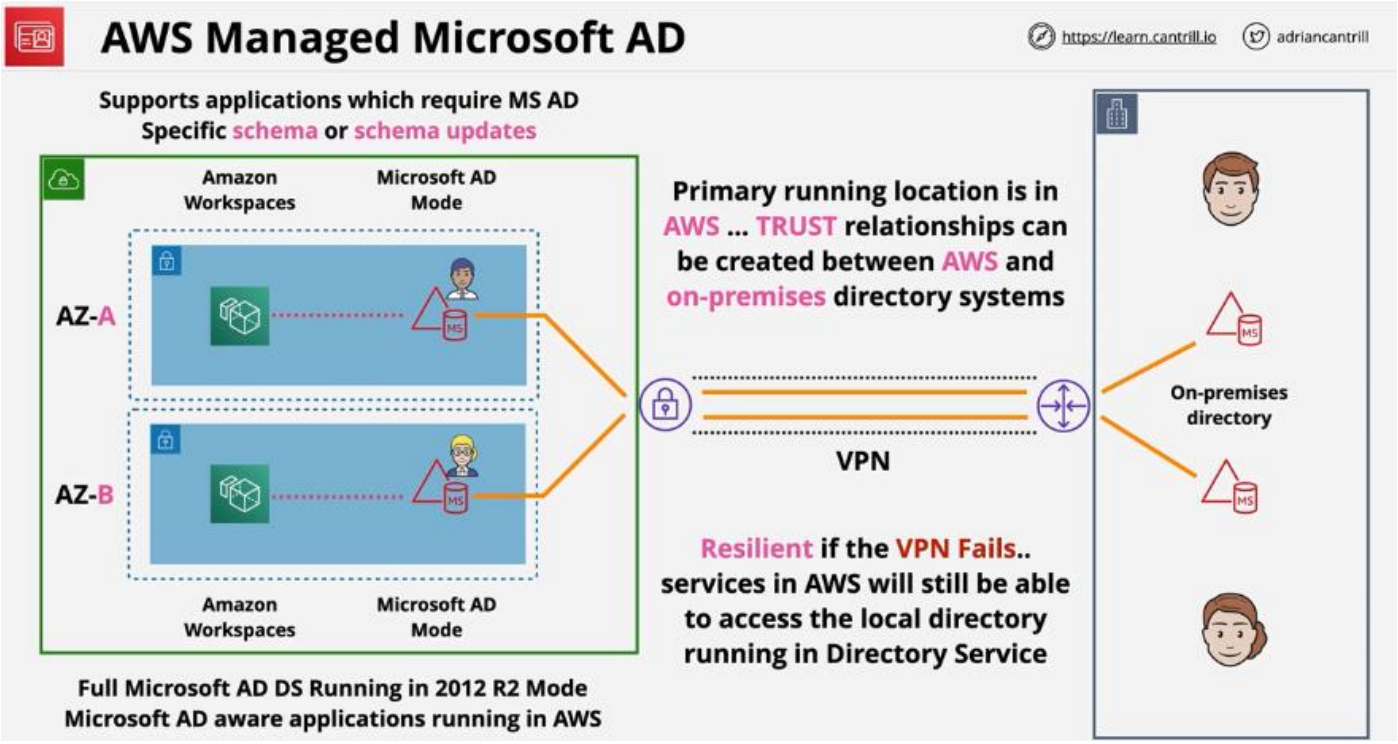
Simple AD is not compatible with RDS SQL Server. Customers who require the features of an actual Microsoft Active Directory, or who envision using their directory with RDS SQL Server should use AWS Managed Microsoft AD instead.

You can use Simple AD as a standalone directory in the cloud to support Windows workloads that need basic AD features, compatible AWS applications, or to support Linux workloads that need LDAP service.



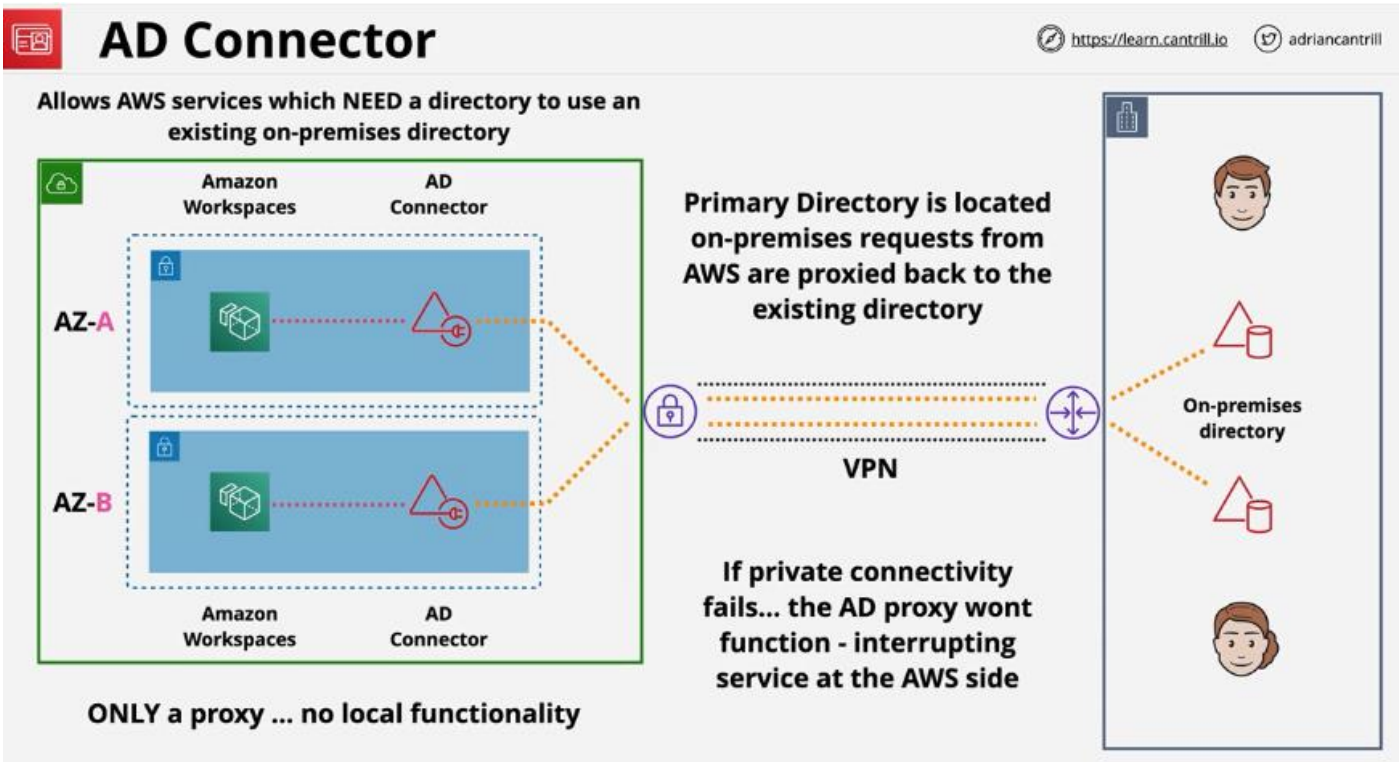
AWS Managed Microsoft AD

- AWS Directory Service lets you run Microsoft Active Directory (AD) as a managed service. AWS Directory Service for Microsoft Active Directory, also referred to as AWS Managed Microsoft AD, is powered by Windows Server 2012 R2. When you select and launch this directory type, it is created as a highly available pair of domain controllers connected to your virtual private cloud (VPC). The domain controllers run in different Availability Zones in a Region of your choice. Host monitoring and recovery, data replication, snapshots, and software updates are automatically configured and managed for you.
- With AWS Managed Microsoft AD, you can run directory-aware workloads in the AWS Cloud, including Microsoft SharePoint and custom .NET and SQL Server-based applications. You can also configure a trust relationship between AWS Managed Microsoft AD in the AWS Cloud and your existing on-premises Microsoft Active Directory, providing users and groups with access to resources in either domain, using single sign-on (SSO).
- AWS Directory Service makes it easy to set up and run directories in the AWS Cloud, or connect your AWS resources with an existing on-premises Microsoft Active Directory. Once your directory is created, you can use it for a variety of tasks:
    - Manage users and groups
    - Provide single sign-on to applications and services
    - Create and apply group policy
    - Simplify the deployment and management of cloud-based Linux and Microsoft Windows workloads
    - You can use AWS Managed Microsoft AD to enable multi-factor authentication by integrating with your existing RADIUS-based MFA infrastructure to provide an additional layer of security when users access AWS applications.
    - Securely connect to Amazon EC2 Linux and Windows instances



AD Connector

- AD Connector is a proxy service that provides an easy way to connect compatible AWS applications, such as Amazon WorkSpaces, Amazon QuickSight, and Amazon EC2 for Windows Server instances, to your existing on-premises Microsoft Active Directory. With AD Connector, you can simply add one service account to your Active Directory. AD Connector also eliminates the need of directory synchronization or the cost and complexity of hosting a federation infrastructure.
- You can also use AD Connector to enable multi-factor authentication (MFA) for your AWS application users by connecting it to your existing RADIUS-based MFA infrastructure. This provides an additional layer of security when users access AWS applications.
- AD Connector is your best choice when you want to use your existing on-premises directory with compatible AWS services.



Picking between Modes

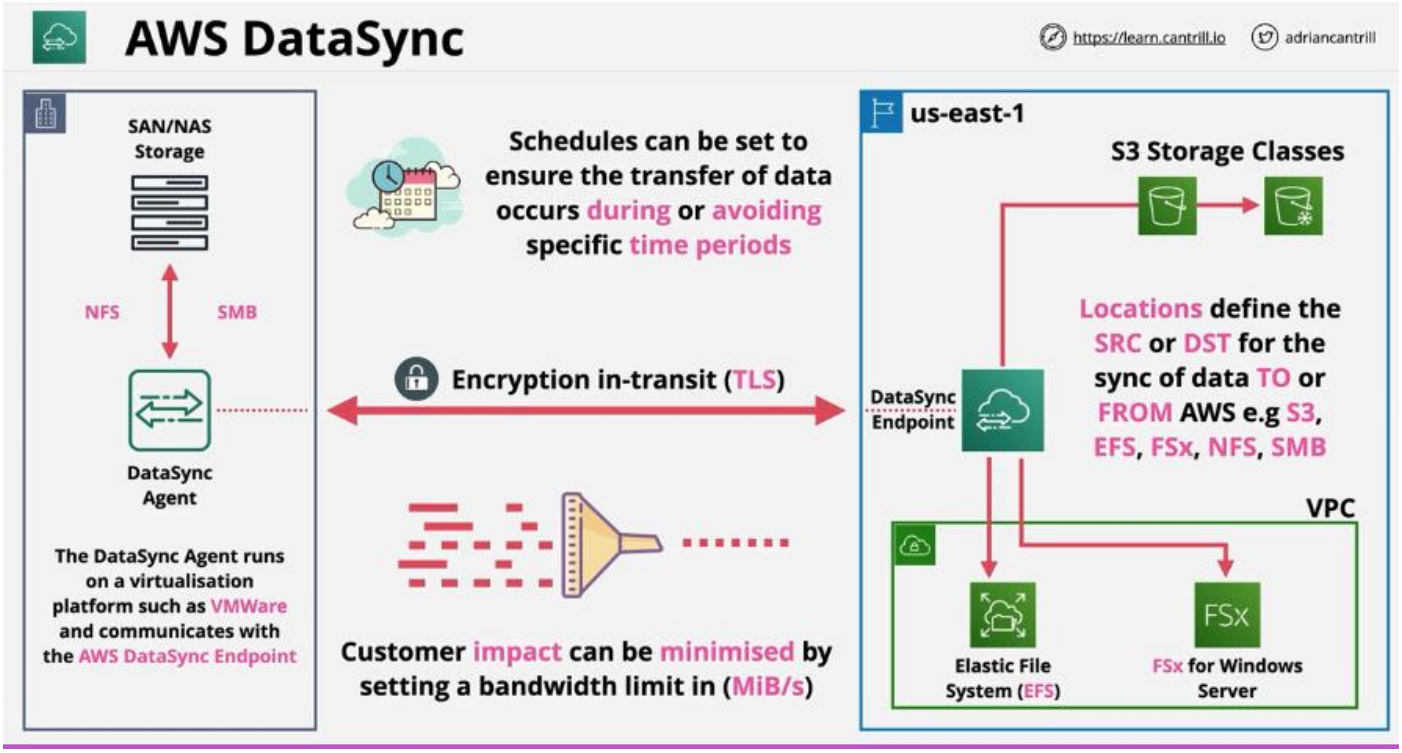
1. **Simple AD** - The default. Simple requirements. A directory in AWS.
2. **Microsoft AD** - Applications in AWS which need MS AD DS, or you need to TRUST AD DS
3. **AD Connector** - Use AWS Services which need a directory without storing any directory info in the cloud... proxy to your on-premises Directory

DataSync

- AWS DataSync is an online data transfer service that simplifies, automates, and accelerates moving data between on-premises storage systems and AWS Storage services, as well as between AWS Storage services. You can use DataSync to migrate active datasets to AWS, archive data to free up on-premises storage capacity, replicate data to AWS for business continuity, or transfer data to the cloud for analysis and processing.
- Writing, maintaining, monitoring, and troubleshooting scripts to move large amounts of data can burden your IT operations and slow migration projects.
- DataSync eliminates or automatically handles this work for you.
- DataSync provides built-in security capabilities such as encryption of data in-transit, and data integrity verification in-transit and at-rest.
- It optimizes use of network bandwidth, and automatically recovers from network connectivity failures.
- In addition, DataSync provides control and monitoring capabilities such as data transfer scheduling and granular visibility into the transfer process through Amazon CloudWatch metrics, logs, and events.
- DataSync can copy data between Network File System (NFS) shares, Server Message Block (SMB) shares, self-managed object storage, AWS Snowcone, Amazon Simple Storage Service (Amazon S3) buckets, Amazon Elastic File System (Amazon EFS) file systems, and Amazon FSx for Windows File Server file systems.

Key Features

- Scalable - **10Gbps** per agent (~ **100TB per day)**
- Bandwidth Limiters (avoid link saturation)
- Incremental and scheduled transfer options
- Compression and encryption
- Automatic recovery from transit errors
- AWS Service integration - S3, EFS, FSx
- Pay as you use... per GB cost for data moved

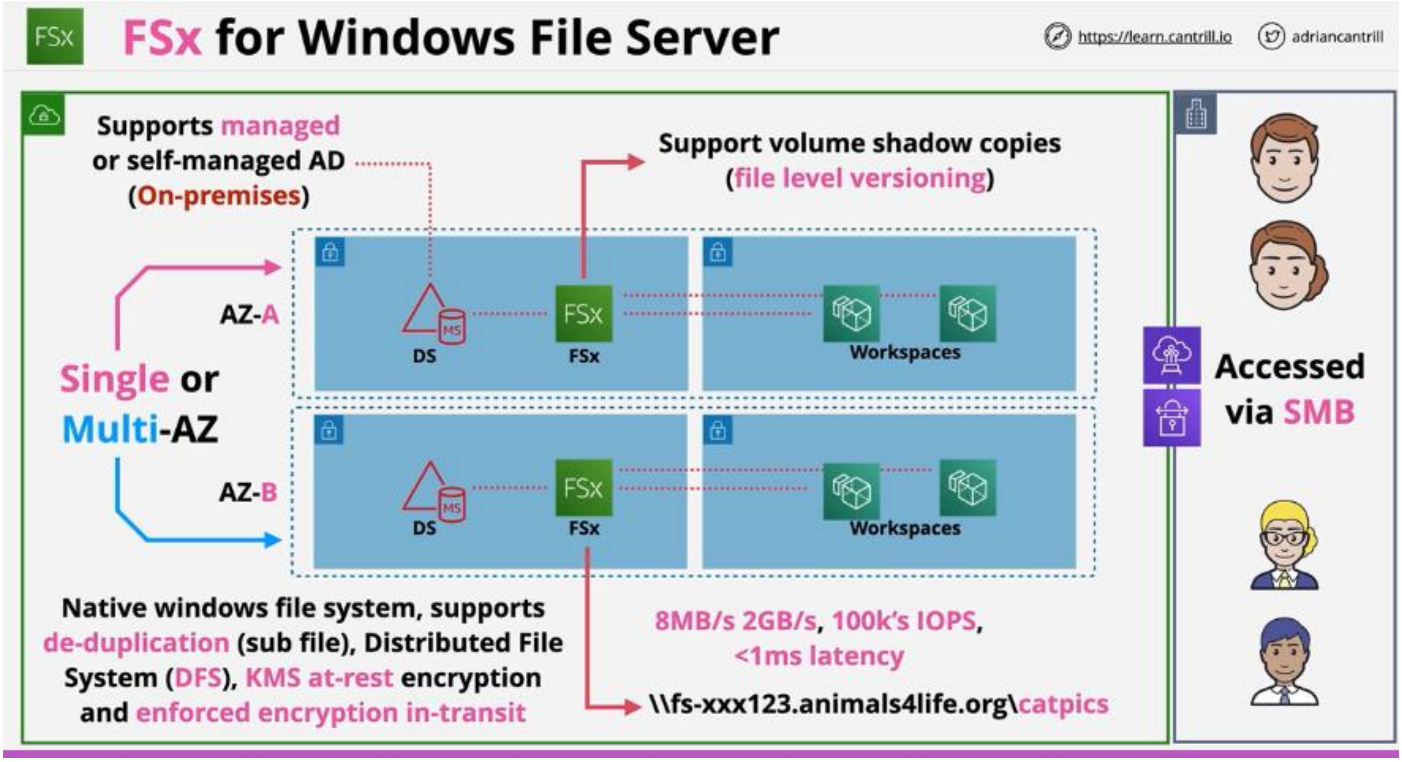


DataSync Components

- **Task** - A 'job' within DataSync defines what is being synced, how quickly, FROM where and TO where
- **Agent** - Software used to read or write to on-premises data stores using NFS or SMB
- **Location** - every task has two locations FROM and TO. E.g., Network File System (NFS), Server Message Block (SMB), Amazon EFS, Amazon FSx and Amazon S3

FSx for Windows Servers

- Amazon FSx for Windows File Server provides fully managed, highly reliable, and scalable file storage that is accessible over the industry-standard Server Message Block (SMB) protocol.
- It is built on Windows Server, delivering a wide range of administrative features such as user quotas, end-user file restores, and Microsoft Active Directory (AD) integration.
- It offers single-AZ and multi-AZ deployment options, fully managed backups, and encryption of data at rest and in transit. You can optimize cost and performance for your workload needs with SSD and HDD storage options; and you can scale storage and change the throughput performance of your file system at any time.
- Amazon FSx file storage is accessible from Windows, Linux, and MacOS compute instances and devices running on AWS or on premises.
- Fully managed native windows file servers/shares
- Designed for integration with windows environment
- Integrates with Directory Service or Self-Managed AD
- Single or Multi-AZ within a VPC
- On-demand and Schedules Backups
- Accessible using VPC, Peering, VPN, Direct Connect

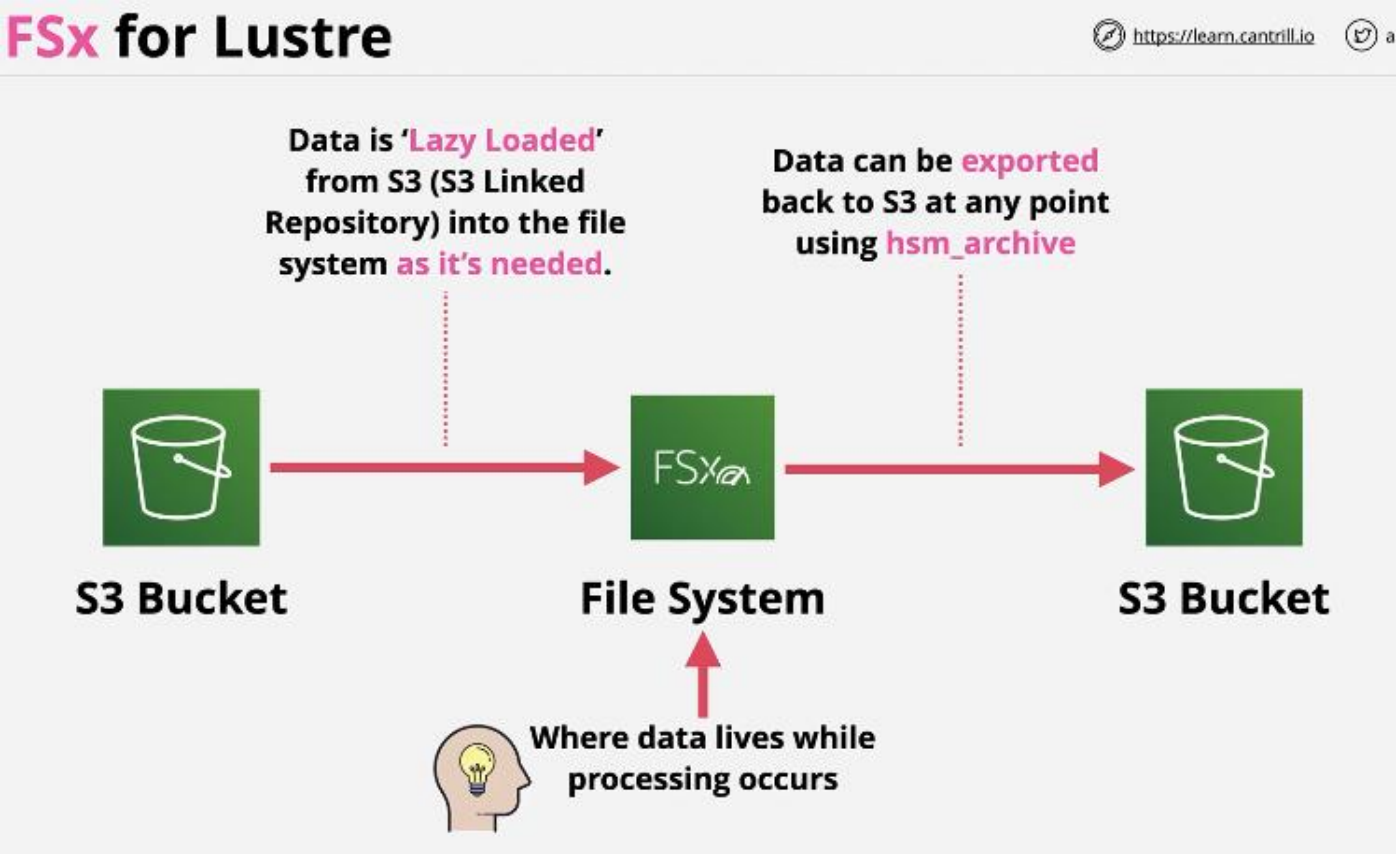


FSx Key Features and Benefits

- VSS - User-Driven Restores
- Native file system accessible over SMB
- Windows permission model
- Supports DFS... Scale-out file shares structure
- Managed - no file server admin
- Integrates with DS and your own directory

FSx For Lustre

- Amazon FSx for Lustre is a fully managed service that provides cost-effective, high-performance, scalable storage for compute workloads. Many workloads such as machine learning, high performance computing (HPC), video rendering, and financial simulations depend on compute instances accessing the same set of data through high-performance shared storage.
- Powered by Lustre, the world's most popular high-performance file system, FSx for Lustre offers sub-millisecond latencies, up to hundreds of gigabytes per second of throughput, and millions of IOPS. It provides multiple deployment options and storage types to optimize cost and performance for your workload requirements.
- FSx for Lustre file systems can also be linked to Amazon S3 buckets, allowing you to access and process data concurrently from both a high-performance file system and from the S3 API.
- Managed Lustre - Designed for HPC - LINUX Clients (POSIX)
- use in Machine Learning, Big Data, Financial Modelling
- 100's GB/s throughput & sub millisecond latency
- Deployment types - Persistent or Scratch
- Scratch - Highly optimised for short term no replication & fast
- Persistent - longer term, HA (in one AZ), self-healing
- Accessible over VPN or Direct Connect



- Metadata stored on Metadata Target (MDTs)
- Objects are stored on called object storage targets (OSTs)(1.17TiB)
- Baseline performance based on size
- Size - min 1.2TiB then increments of 2.4TiB
- For Scratch - Base 200 MB/s per TiB of storage
- Persistent offers 50MB/s, 100MB/s and 200 MB/s per TiB of storage
- Burst up to 1,300 MB/s per TiB (Credit System)
- Scratch is designed for pure performance
- Short term or temp workloads
- No HA. No REPLICATION
- Larger file systems mean more servers, more disks and more chance of failure!
- Persistent has replication within ONE AZ only
- Auto-heals when hardware failure occurs
- You can back up to S3 with both (manual or Automatic 0–35-day retention)

MODULE 18 - SECURITY, DEPLOYMENT & OPERATIONS

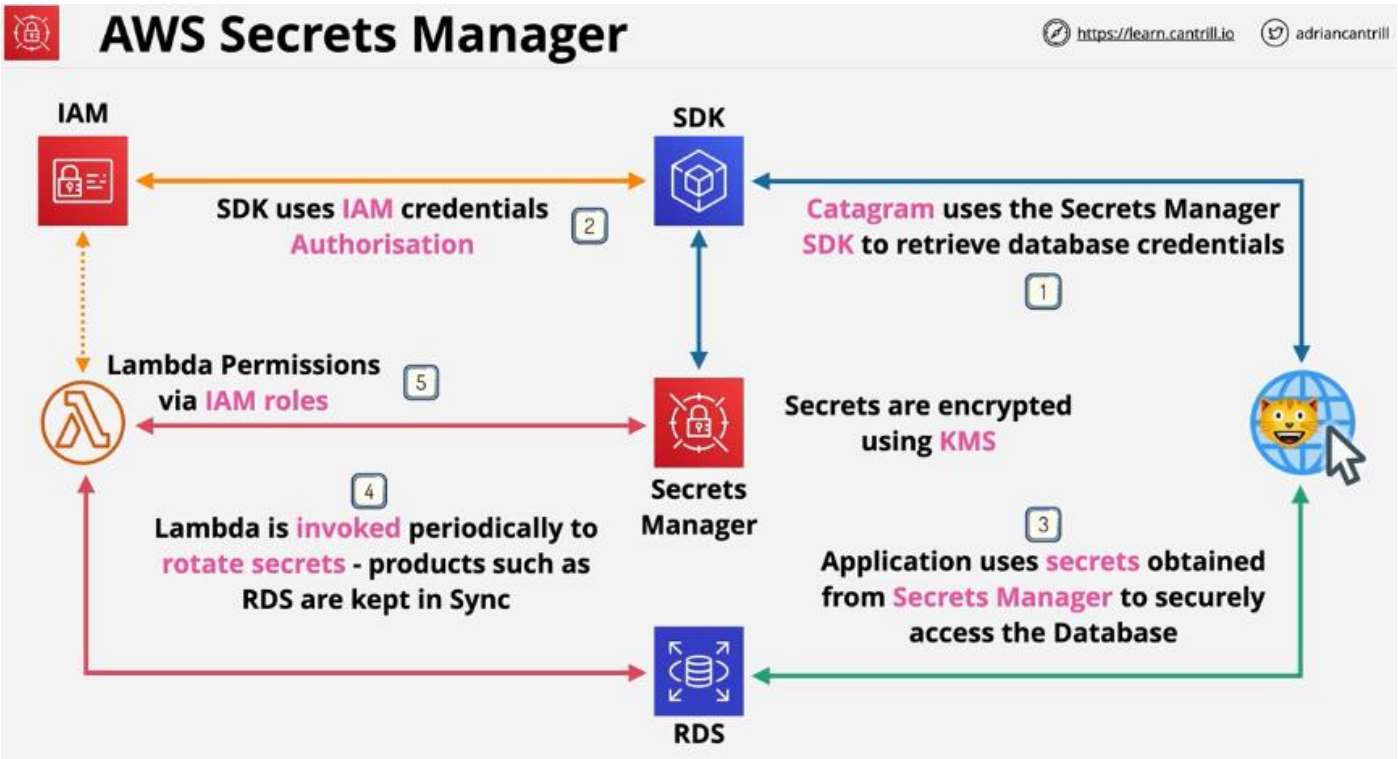
AWS Secrets Manager

- AWS Secrets Manager helps you protect secrets needed to access your applications, services, and IT resources.
- The service enables you to easily rotate, manage, and retrieve database credentials, API keys, and other secrets throughout their lifecycle. Users and applications retrieve secrets with a call to Secrets Manager APIs, eliminating the need to hardcode sensitive information in plain text.
- Secrets Manager offers secret rotation with built-in integration for Amazon RDS, Amazon Redshift, and Amazon DocumentDB. Also, the service is extensible to other types of secrets, including API keys and OAuth tokens.
- In addition, Secrets Manager enables you to control access to secrets using fine-grained permissions and audit secret rotation centrally for resources in the AWS Cloud, third-party services, and on-premises.
- It does share functionality with Parameter Store
- Designed for secrets (Passwords, API KEYS)
- Usable via Console, CLI, API or SDK's (integration)
- Supports automatic rotation and this uses lambda
- Directory Integrates with some AWS products such as RDS

Secrets Manager enables you to replace hardcoded credentials in your code, including passwords, with an API call to Secrets Manager to retrieve the secret programmatically.

Typical users of Secrets Manager have one or more of the following roles:

- **Secrets Manager administrator** – Administers the Secrets Manager service. Grants permissions to individuals who can then perform the other roles listed here.
- **Database or service administrator** – Administers the database or service with secrets stored in Secrets Manager. Determines and configures the rotation and expiration settings for their secrets.
- **Application developer** – Creates the application, and then configures the application to request the appropriate credentials from Secrets Manager.



AWS WAF & Shield

AWS Shield and Web Application Firewall (WAF) are both products which provide perimeter defence for AWS networks.

Shield provides DDOS protection and WAF is a Layer 7 Application Firewall.

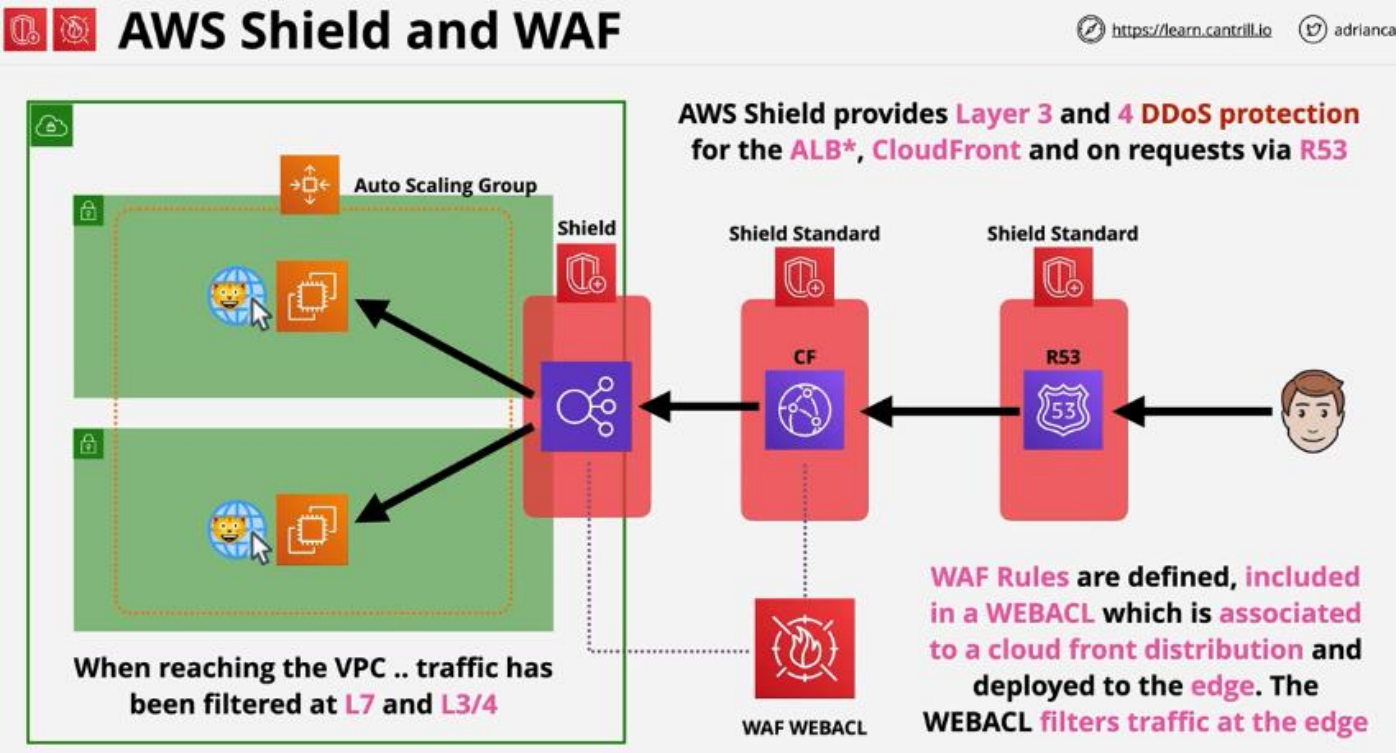
AWS Shield

- You can use AWS WAF web access control lists (web ACLs) to help minimize the effects of a distributed denial of service (DDoS) attack. For additional protection against DDoS attacks, AWS also provides AWS Shield Standard and AWS Shield Advanced.
- AWS Shield Standard is automatically included at no extra cost beyond what you already pay for AWS WAF and your other AWS services.
- AWS Shield Advanced provides expanded DDoS attack protection for your Amazon EC2 instances, Elastic Load Balancing load balancers, CloudFront distributions, Route 53 hosted zones, and AWS Global Accelerator accelerators. AWS Shield Advanced incurs additional charges.
- AWS provides AWS Shield Standard and AWS Shield Advanced for protection against DDoS attacks. AWS Shield Standard is automatically included at no extra cost beyond what you already pay for AWS WAF and your other AWS services. For added protection against DDoS attacks, AWS offers AWS Shield Advanced. AWS Shield Advanced provides expanded DDoS attack protection for your resources.
- You can add protection for any of the following resource types:
    - Amazon CloudFront distributions
    - Amazon Route 53 hosted zones
    - AWS Global Accelerator accelerators
    - Application Load Balancers
    - Elastic Load Balancing (ELB) load balancers
    - Amazon Elastic Compute Cloud (Amazon EC2) Elastic IP addresses
- Shield Standard - free with Route53 and CloudFront
- Protection against Layer 3 and Layer 4 DDoS attacks
- Shield Advanced - $3,000 p/m

AWS Web Application Firewall (WAF)

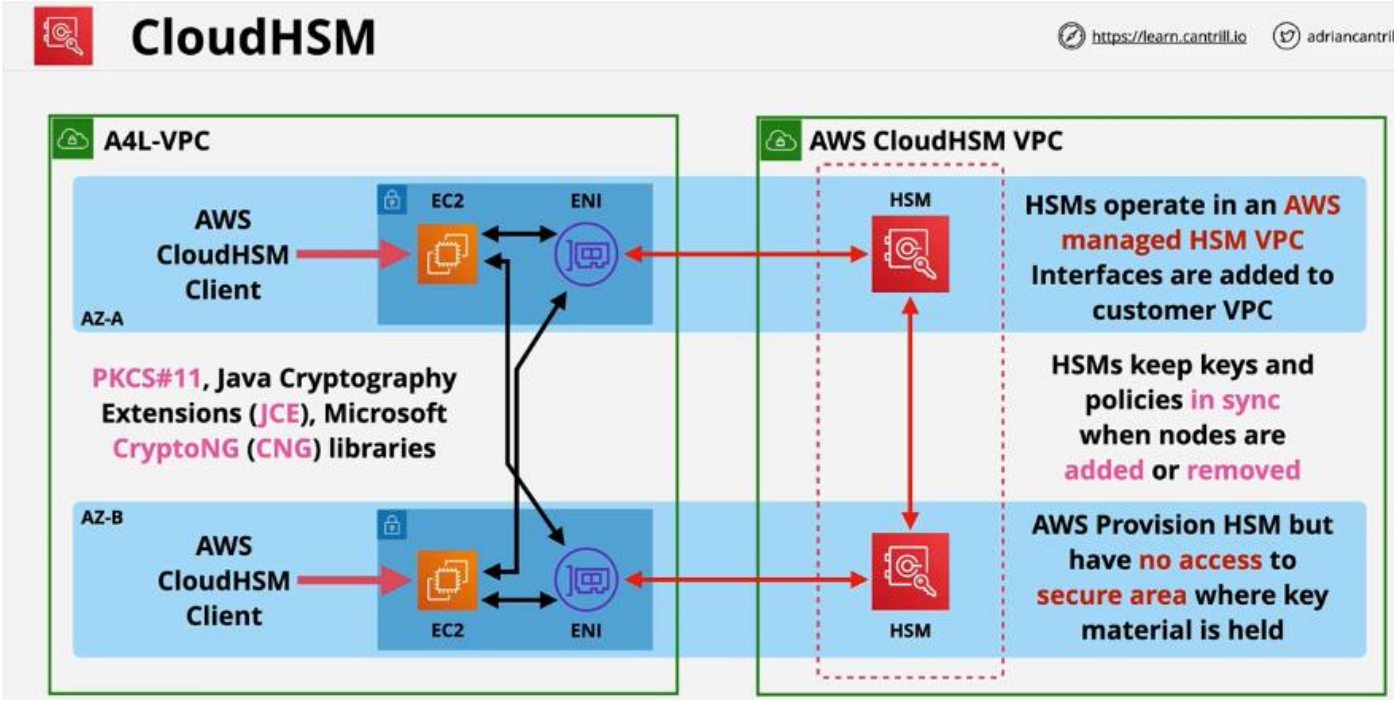
AWS WAF is a web application firewall that lets you monitor the HTTP and HTTPS requests that are forwarded to an Amazon CloudFront distribution, an Amazon API Gateway REST API, an Application Load Balancer, or an AWS AppSync GraphQL API. AWS WAF also lets you control access to your content. Based on conditions that you specify, such as the IP addresses that requests originate from or the values of query strings, Amazon CloudFront, Amazon API Gateway, Application Load Balancer, or AWS AppSync responds to requests either with the requested content or with an HTTP 403 status code (Forbidden). You also can configure CloudFront to return a custom error page when a request is blocked.

- Layer 7 (HTTP/s) Firewall
- Protects against complex Layer 7 attacks/exploits
- SQL Injections, Cross-site Scripting, Geo Blocks, Rate Awareness
- Web Access Control list (WEBACL) integrated with ALB, API Gateway and CloudFront
- Rules are added to a WEBACL and evaluate when traffic arrives



CloudHSM

- AWS CloudHSM is a cryptographic service for creating and maintaining hardware security modules (HSMs) in your AWS environment. HSMs are computing devices that process cryptographic operations and provide secure storage for cryptographic keys. You can use AWS CloudHSM to offload SSL/TLS processing for web servers, protect private keys linked to an issuing certificate authority (CA), or enable Transparent Data Encryption (TDE) for Oracle databases.
- When you use an HSM from AWS CloudHSM, you can perform a variety of cryptographic tasks:
    - Generate, store, import, export, and manage cryptographic keys, including symmetric keys and asymmetric key pairs.
    - Use symmetric and asymmetric algorithms to encrypt and decrypt data.
    - Use cryptographic hash functions to compute message digests and hash-based message authentication codes (HMACs).
    - Cryptographically sign data (including code signing) and verify signatures.
    - Generate cryptographically secure random data.
- AWS CloudHSM organizes HSMs in clusters, which are automatically synchronized collections of HSMs within a given Availability Zone (AZ).
- By adding more HSMs to a cluster and distributing clusters across AZs, you can load balance the cryptographic operations being performed within your cloud environment and provide redundancy and high availability in case of AZ failure. Additionally, AWS CloudHSM periodically generates and stores backups of your clusters, making CloudHSM data recovery secure and simple.
- The keys that you generate in AWS KMS are protected by FIPS 140-2 validated cryptographic modules. If you want a managed service for creating and controlling encryption keys, but do not want or need to operate your own HSM, consider using AWS Key Management Service.



CloudHSM use cases

- Offload the SSL/TLS Processing for Web Servers
- Protect the Private Keys for an Issuing Certificate Authority (CA)
- Enable Transparent Data Encryption (TDE) for Oracle Databases

AWS Config

- AWS Config is a service that enables you to assess, audit, and evaluate the configurations of your AWS resources. Config continuously monitors and records your AWS resource configurations and allows you to automate the evaluation of recorded configurations against desired configurations.
- With Config, you can review changes in configurations and relationships between AWS resources, dive into detailed resource configuration histories, and determine your overall compliance against the configurations specified in your internal guidelines. This enables you to simplify compliance auditing, security analysis, change management, and operational troubleshooting.
- AWS Config also generates configuration items when the configuration of a resource changes, and it maintains historical records of the configuration items of your resources from the time you start the configuration recorder.
- By default, AWS Config creates configuration items for every supported resource in the region.
- AWS Config keeps track of all changes to your resources by invoking the Describe or the List API call for each resource in your account. The service uses those same API calls to capture configuration details for all related resources.

Remember the following:

- Record configuration changes over time on resources
- Does not prevent changes happening ... no protection
- Regional Service and supports cross-region and account aggregation
- Changes can generate SNS notifications and near-real-time events via EventBridge & Lambda
- AWS Config is a service which records the configuration of resources over time (configuration items) into configuration histories.
- All the information is stored regionally in an S3 config bucket.
- AWS Config is capable of checking for compliance and generating notifications and events based on compliance.

Amazon Macie

Amazon Macie is a fully managed data security and data privacy service that uses machine learning and pattern matching to discover and protect your sensitive data in AWS.

Data Security

Amazon Macie Classic enables you to be proactive with security compliance and achieve preventive security as follows:

- Identify and protect various data types, including PII, PHI, regulatory documents, API keys, and secret keys
- Verify compliance with automated logs that allow for instant auditing
- Identify changes to policies and access control lists
- Observe changes in user behaviour and receive actionable alerts
- Receive notifications when data and account credentials leave protected zones
- Detect when large quantities of business-critical documents are shared internally and externally

Remember the following

- It is Data Security and Data Privacy Service
- It Discover, Monitor and Protect Data and stored in S3 Buckets
- Automated discovery of data i.e., PII, PHI, Finance
- managed Data Identifiers -Built in - ML/Patterns
- Custom Data Identifiers - Proprietary - Regex based
- Integrates with Security Hub & 'finding events' to EventBride
- Centrally manage either via AWS ORG or one Macie Account Inviting

Amazon Macie – Identifiers

- It Managed data identifiers and maintained by AWS
- Growing list of common sensitive data types such as Credentials, finance, health, personal Identifies
- **Custom data identifiers** - created by you
- **Regex** - defines a 'pattern' to match in data.
- **Keywords** - optional sequences that need to be in proximity to regex match
- **Maximum Match Distance** - how close keywords are to regex pattern
- **Ignore words** - if regex match contains ignore words, it's ignored

Amazon Macie findings

- Amazon Macie generates two categories of findings: **policy findings** and **sensitive data findings**. A policy finding is a detailed report of a potential policy violation for an Amazon Simple Storage Service (Amazon S3) bucket.
- Macie generates these findings as part of its ongoing monitoring activities for your Amazon S3 data. A sensitive data finding is a detailed report of sensitive data in an S3 object. Macie generates these findings when it discovers sensitive data in S3 objects that you configure it to analyse as part of a sensitive data discovery job.

Policy findings

Macie generates policy findings when the policies or settings for an S3 bucket are changed in a way that reduces the security of the bucket and its objects. Macie does this only if the change occurs after you enable Macie for your Amazon Web Services account.

Macie can generate the following types of policy findings for an S3 bucket.

Policy:IAMUser/S3BlockPublicAccessDisabled

Block public access settings were disabled for the bucket. Access to the bucket is controlled only by access control lists (ACLs) and bucket policies.

Policy:IAMUser/S3BucketEncryptionDisabled

Default encryption was disabled for the bucket. By default, Amazon S3 won't automatically encrypt new objects when they're added to the bucket.

Policy:IAMUser/S3BucketPublic

An ACL or bucket policy for the bucket was changed to allow access by anonymous users or by all authenticated AWS Identity and Access Management (IAM) users or roles.

Policy:IAMUser/S3BucketReplicatedExternally

Data replication was enabled and configured to replicate objects from the bucket to an Amazon Web Services account that isn't part of your organization. An organization is a set of Macie accounts that are centrally managed as a group of related accounts through AWS Organizations or by Macie invitation.

Policy:IAMUser/S3BucketSharedExternally

An ACL or bucket policy for the bucket was changed to allow the bucket to be shared with an Amazon Web Services account that isn't part of your organization. An organization is a set of Macie accounts that are centrally managed as a group of related accounts through AWS Organizations or by Macie invitation.

Sensitive data findings

Macie generates sensitive data findings when it discovers sensitive data in S3 objects that you configure it to analyse as part of a sensitive data discovery job. Macie can generate the following types of sensitive data findings for an object.

SensitiveData: S3Object/Credentials

The object contains credentials, such as private keys or AWS secret keys.

SensitiveData: S3Object/CustomIdentifier

The object contains content that matches one or more custom data identifiers. The object might include more than one type of sensitive data.

SensitiveData:S3Object/Financial

The object contains financial information, such as credit card numbers or bank account numbers.

SensitiveData:S3Object/Multiple

The object contains more than one category of sensitive data—any combination of credentials, financial information, personal information, or content that matches one or more custom data identifiers.

SensitiveData:S3Object/Personal

The object contains personally identifiable information (such as full names or mailing addresses), personal health information (such as health insurance or medical identification numbers), or a combination of the two.

MODULE 19 - NOSQL DATABASE & DYNAMODB

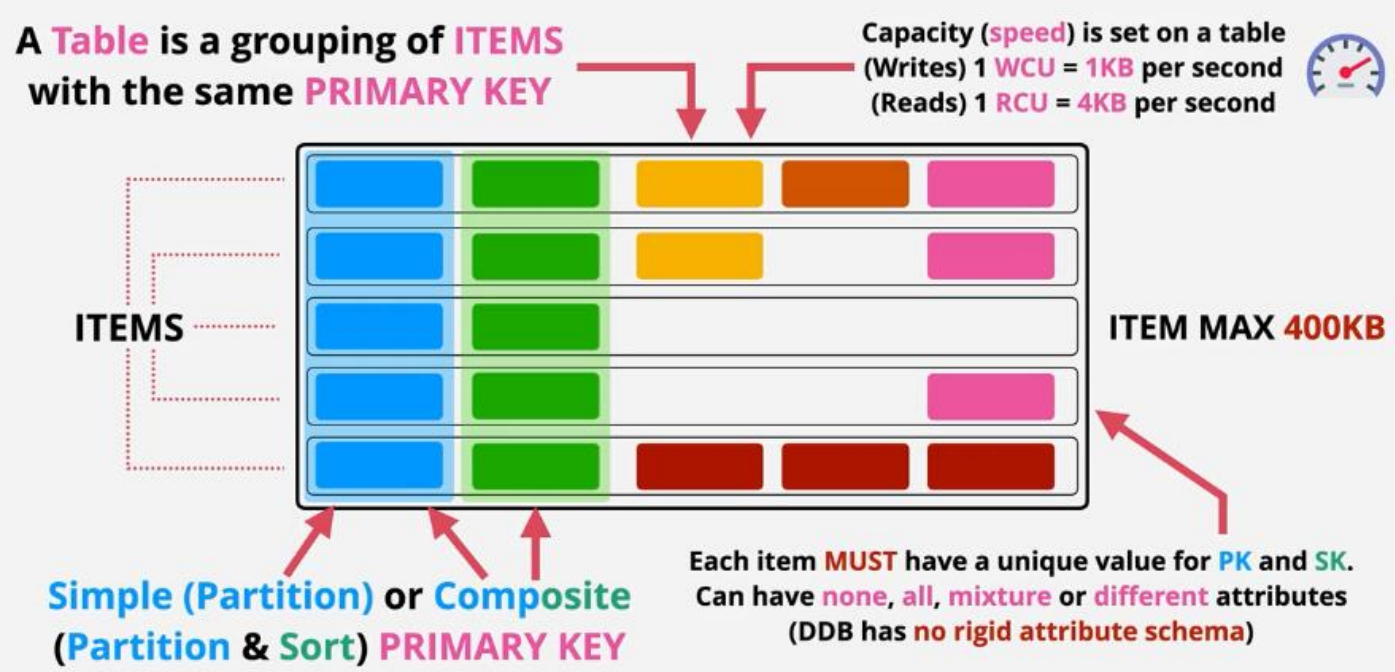
DynamoDB - Architecture

- At a high level, Amazon DynamoDB is designed for high availability, durability, and consistently low latency (typically in the single digit milliseconds) performance.
- Amazon DynamoDB runs on a fleet of AWS managed servers that leverage solid state drives (SSDs) to create an optimized, high-density storage platform. This platform decouples performance from table size and eliminates the need for the working set of data to fit in memory while still returning consistent, low latency responses to queries. As a managed service, Amazon DynamoDB abstracts its underlying architectural details from the user.
- In DynamoDB, tables, items, and attributes are the core components that you work with. A table is a collection of items, and each item is a collection of attributes. DynamoDB uses primary keys to uniquely identify each item in a table and secondary indexes to provide more querying flexibility. You can use DynamoDB Streams to capture data modification events in DynamoDB tables.
- The following are the basic DynamoDB components:
    - **Tables** – Similar to other database systems, DynamoDB stores data in tables. A table is a collection of data. For example, see the example table called People that you could use to store personal contact information about friends, family, or anyone else of interest. You could also have a Cars table to store information about vehicles that people drive.
    - **Items** – Each table contains zero or more items. An item is a group of attributes that is uniquely identifiable among all of the other items. In a People table, each item represents a person. For a Cars table, each item represents one vehicle. Items in DynamoDB are similar in many ways to rows, records, or tuples in other database systems. In DynamoDB, there is no limit to the number of items you can store in a table.
    - **Attributes** – Each item is composed of one or more attributes. An attribute is a fundamental data element, something that does not need to be broken down any further. For example, an item in a People table contains attributes called PersonID, LastName, FirstName, and so on. For a Department table, an item might have attributes such as DepartmentID, Name, Manager, and so on. Attributes in DynamoDB are similar in many ways to fields or columns in other database systems.

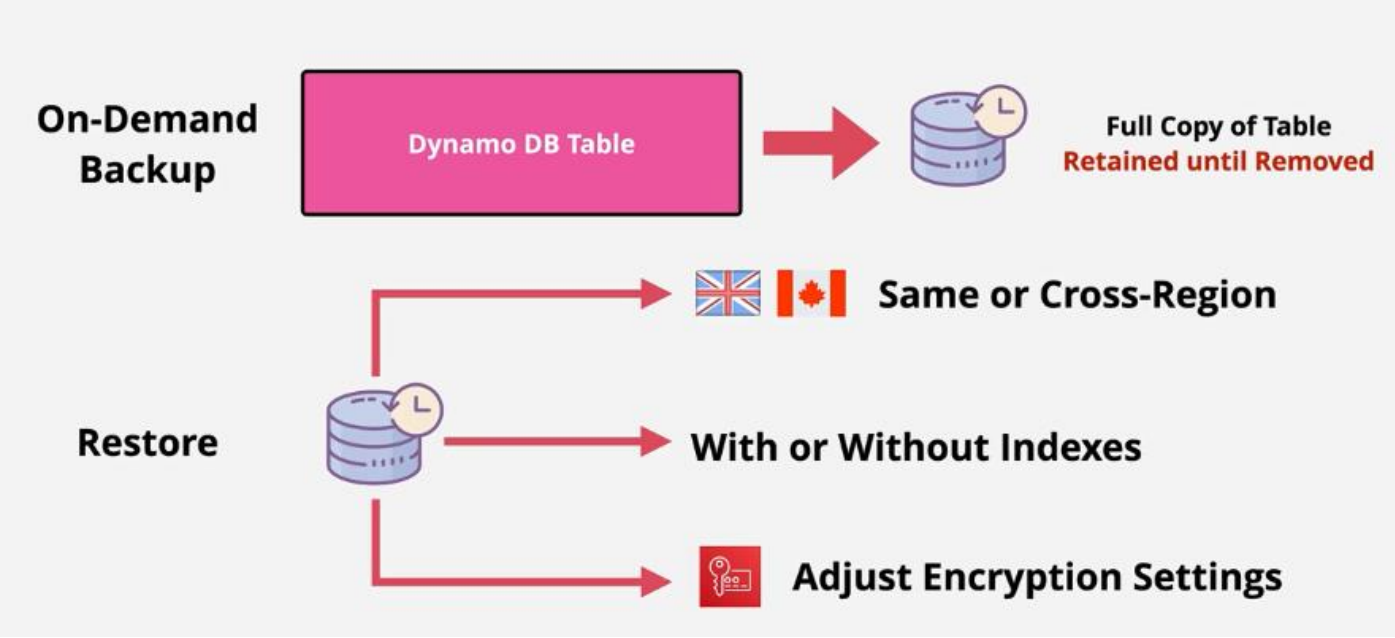
DynamoDB Concepts

- It is NoSQL Public Database-as-a-Service (DBaaS) - Key/Value & Document
- No self-managed servers or infrastructure
- Manual/ Automatic provisioned performance IN/OUT or On-Demand
- Highly Resilient across AZs and optionally global
- Really fast, single-digit milliseconds (SSD based)
- backups, point-in-time recovery, encryption at rest
- Event-Driven integration. do things when data changes

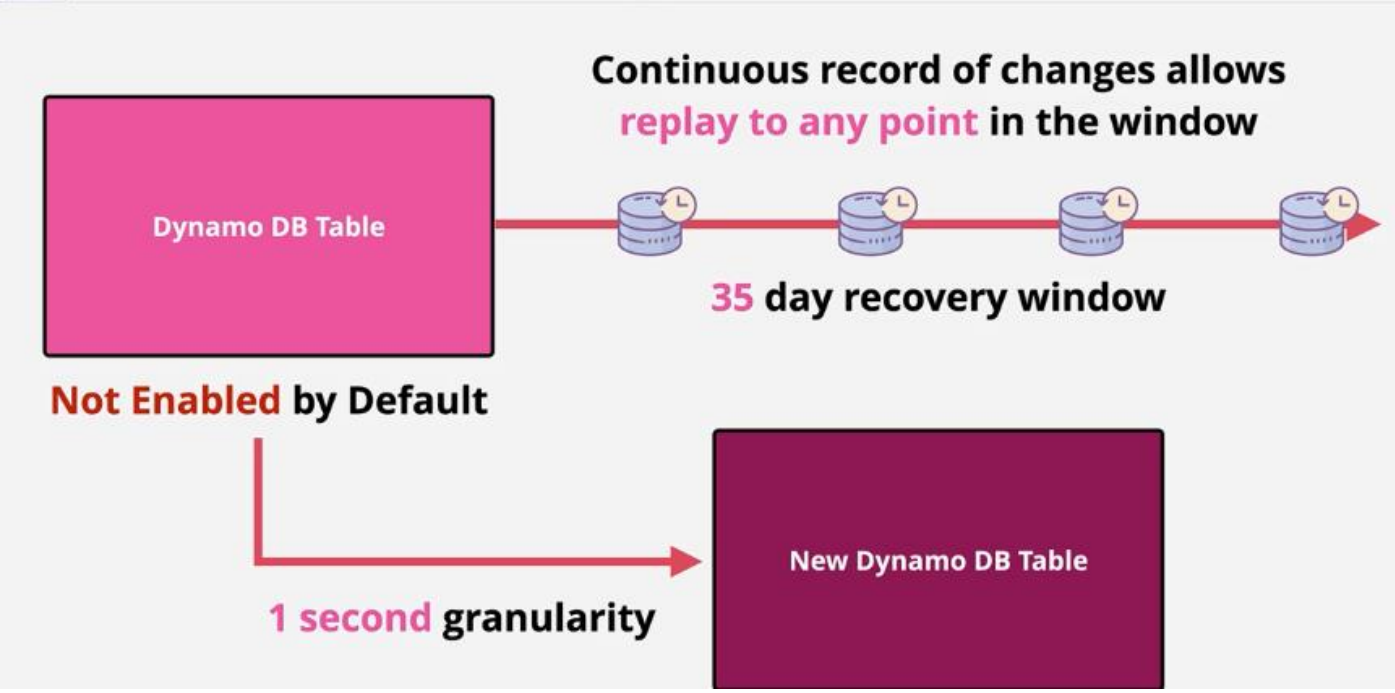
DynamoDB Tables



DynamoDB On-Demand Backups



Point-in-time Recovery (PITR)



DynamoDB - Operations, Consistency and Performance

- Amazon DynamoDB is available in multiple AWS Regions around the world. Each Region is independent and isolated from other AWS Regions. For example, if you have a table called People in the us-east-2 Region and another table named People in the us-west-2 Region, these are considered two entirely separate tables.
- When your application writes data to a DynamoDB table and receives an HTTP 200 response (OK), the write has occurred and is durable. The data is eventually consistent across all storage locations, usually within one second or less.
- DynamoDB supports eventually consistent and strongly consistent reads.

Eventually Consistent Reads

When you read data from a DynamoDB table, the response might not reflect the results of a recently completed write operation. The response might include some stale data. If you repeat your read request after a short time, the response should return the latest data.

Strongly Consistent Reads

When you request a strongly consistent read, DynamoDB returns a response with the most up-to-date data, reflecting the updates from all prior write operations that were successful. However, this consistency comes with some disadvantages:

- A strongly consistent read might not be available if there is a network delay or outage. In this case, DynamoDB may return a server error (HTTP 500).
- Strongly consistent reads may have higher latency than eventually consistent reads.
- Strongly consistent reads are not supported on global secondary indexes.
- Strongly consistent reads use more throughput capacity than eventually consistent reads.

Reading and Writing

Amazon DynamoDB has two read/write capacity modes for processing reads and writes on your tables:

- On-demand
- Provisioned (default, free-tier eligible)

On-Demand Mode

Amazon DynamoDB on-demand is a flexible billing option capable of serving thousands of requests per second without capacity planning. DynamoDB on-demand offers pay-per-request pricing for read and write requests so that you pay only for what you use.

For on-demand mode tables, you don't need to specify how much read and write throughput you expect your application to perform. DynamoDB charges you for the reads and writes that your application performs on your tables in terms of read request units and write request units.

- **One read** request unit represents one strongly consistent read request, or two eventually consistent read requests, for an item up to 4 KB in size. Two read request units represent one transactional read for items up to 4 KB. If you need to read an item that is larger than 4 KB, DynamoDB needs additional read request units. The total number of read request units required depends on the item size, and whether you want an eventually consistent or strongly consistent read.
- For example, if your item size is 8 KB, you require 2 read request units to sustain one strongly consistent read, 1 read request unit if you choose eventually consistent reads, or 4 read request units for a transactional read request.
- **One write** request unit represents one write for an item up to 1 KB in size. If you need to write an item that is larger than 1 KB, DynamoDB needs to consume additional write request units. Transactional write requests require 2 write request units to perform one write for items up to 1 KB. The total number of write request units required depends on the item size. For example, if your item size is 2 KB, you require 2 write request units to sustain one write request or 4 write request units for a transactional write request.

Provisioned Mode

If you choose provisioned mode, you specify the number of reads and writes per second that you require for your application. You can use auto scaling to adjust your table's provisioned capacity automatically in response to traffic changes. This helps you govern your DynamoDB use to stay at or below a defined request rate in order to obtain cost predictability.

Provisioned mode is a good option if any of the following are true:

- You have predictable application traffic.
- You run applications whose traffic is consistent or ramps gradually.
- You can forecast capacity requirements to control costs.

For provisioned mode tables, you specify throughput capacity in terms of read capacity units (RCUs) and write capacity units (WCUs):

- **One read capacity unit represents** one strongly consistent read per second, or two eventually consistent reads per second, for an item up to 4 KB in size. Transactional read requests require two read capacity units to perform one read per second for items up to 4 KB. If you need to read an item that is larger than 4 KB, DynamoDB must consume additional read capacity units. The total number of read capacity units required depends on the item size, and whether you want an eventually consistent or strongly consistent read. For example, if your item size is 8 KB, you require 2 read capacity units to sustain one strongly consistent read per second, 1 read capacity unit if you choose eventually consistent reads, or 4 read capacity units for a transactional read request.

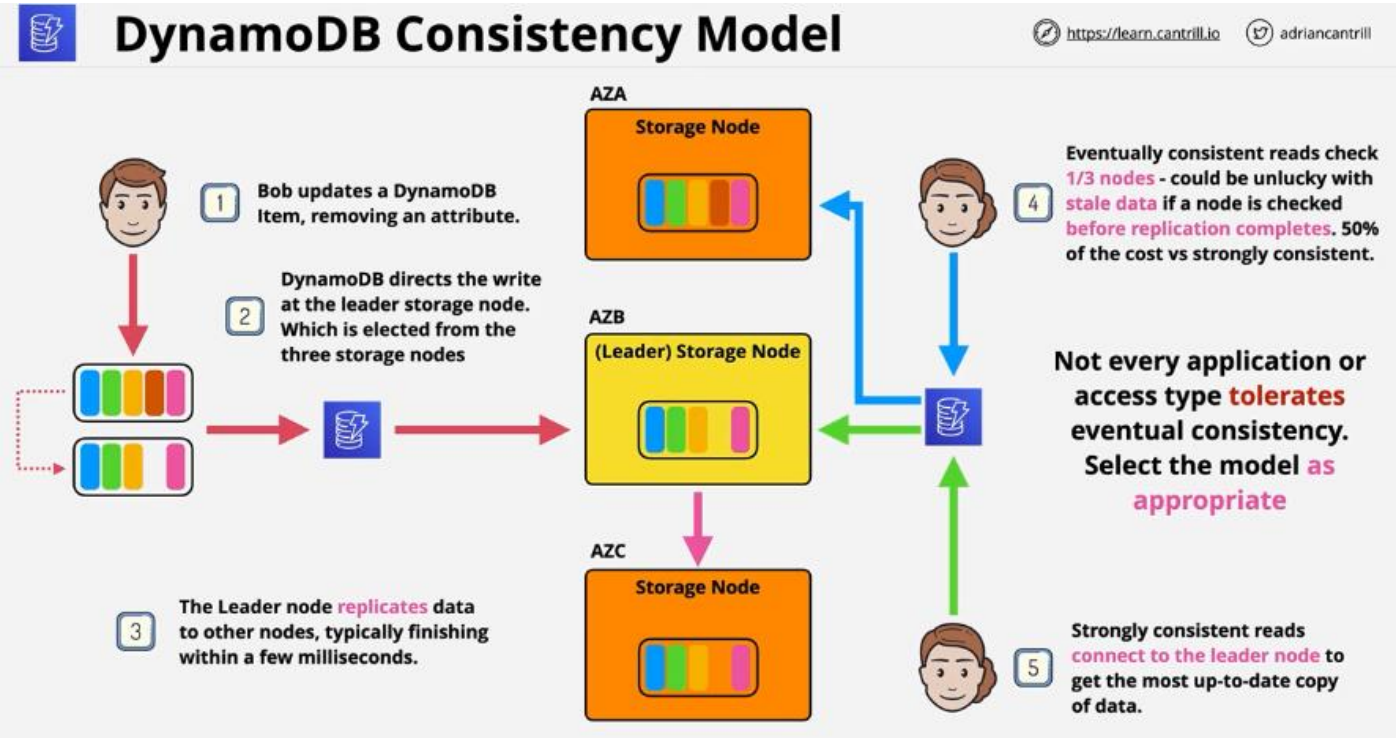
- **One write** capacity unit represents one write per second for an item up to 1 KB in size. If you need to write an item that is larger than 1 KB, DynamoDB must consume additional write capacity units. Transactional write requests require 2 write capacity units to perform one write per second for items up to 1 KB. The total number of write capacity units required depends on the item size. For example, if your item size is 2 KB, you require 2 write capacity units to sustain one write request per second or 4 write capacity units for a transactional write request.

Query

- The Query operation finds items based on primary key values. You can query any table or secondary index that has a composite primary key (a partition key and a sort key). A Query operation will return all of the items from the table or index with the partition key value you provided.
- A Query operation always returns a result set. If no matching items are found, the result set will be empty. Query results are always sorted by the sort key value. If the data type of the sort key is Number, the results are returned in numeric order; otherwise, the results are returned in order of UTF-8 bytes.
- A single Query operation can retrieve items up to a maximum data size of 1MB. You can query a table, a local secondary index, or a global secondary index. For a query on a table or on a local secondary index, you can set the ConsistentRead parameter to true and obtain a strongly consistent result. Global secondary indexes support eventually consistent reads only, so do not specify ConsistentRead when querying a global secondary index.
- For faster response times, design your tables and indexes so that your applications can use Query instead of Scan.

Scan

- The Scan operation returns one or more items and item attributes by accessing every item in a table or a secondary index. The total number of scanned items has a maximum size limit of 1 MB.
- Scan operations proceed sequentially; however, for faster performance on a large table or secondary index, applications can request a parallel Scan operation.
- Scan uses eventually consistent reads when accessing the data in a table; therefore, the result set might not include the changes to data in the table immediately before the operation began. If you need a consistent copy of the data, as of the time that the Scan begins, you can set the ConsistentRead parameter to true when you submit a scan request.
- In general, Scan operations are less efficient than other operations in DynamoDB. If possible, avoid using a Scan operation on a large table or index with a filter that removes many results. Using Scan over large data sets may use up the provisioned throughput for a large table or index in a single operation.
- Instead of using a large Scan operation, you can apply the following techniques to minimize the impact of a scan on a table's provisioned throughput:
    - Reduce page size – because a Scan operation reads an entire page (by default, 1 MB), you can reduce the impact of the scan operation by setting a smaller page size.
    - Isolate scan operations – perform scans on a table that is not taking "mission-critical" traffic. You can configure applications to handle this load by rotating traffic periodically between two tables, whose data is replicated with one another.



DynamoDB - Streams & Lambda Triggers

Stream Concept

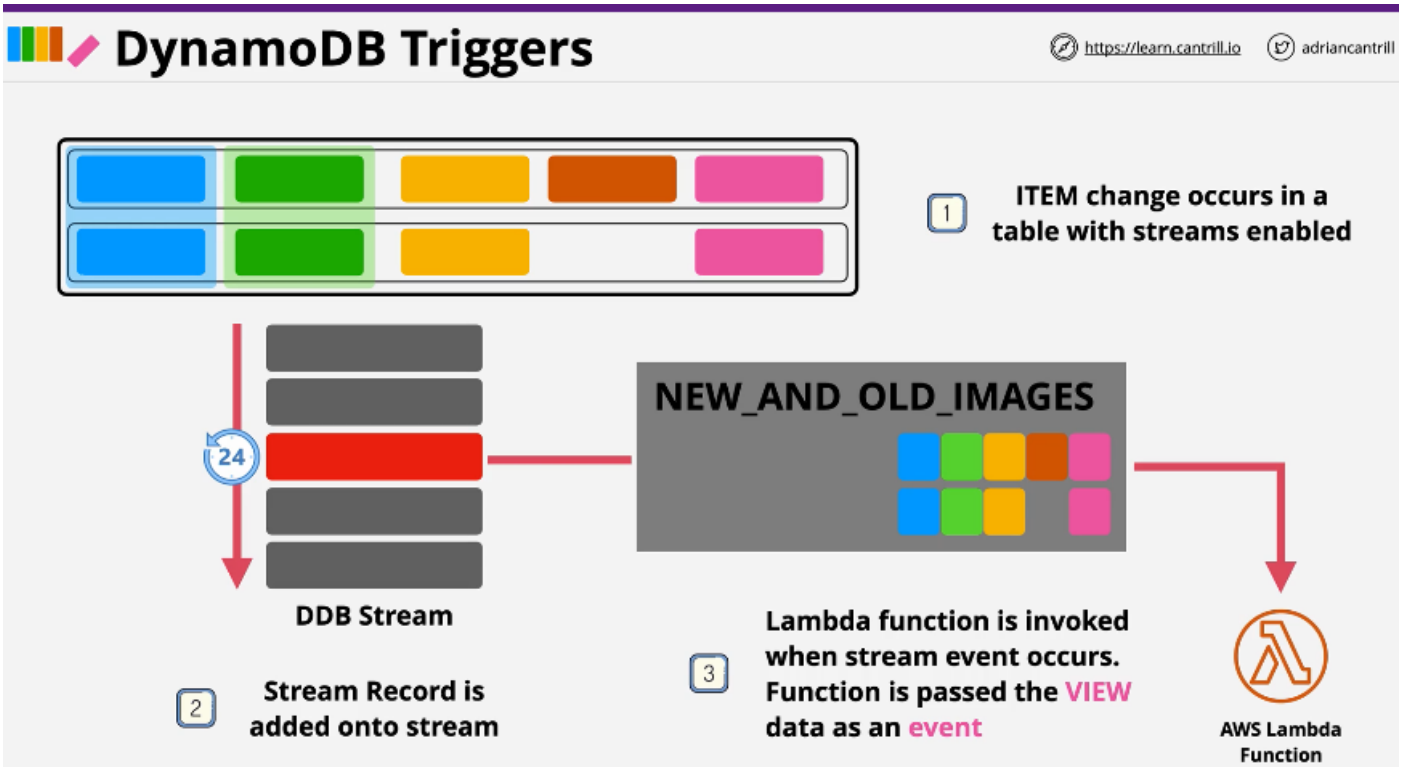
- Time ordered list of ITEM CHANGES in a table
- DynamoDB Streams are a 24-hour rolling window of time ordered changes to ITEMS in a DynamoDB table
- Enabled on a per table basis
- Records INSERT, UPDATES and DELETES
- Different view types influence what is in the stream

Stream View Type — Specifies the information that will be written to the stream whenever data in the table is modified:

- **KEYS_ONLY** — Only the key attributes of the modified item.
- **NEW_IMAGE** — The entire item, as it appears after it was modified.
- **OLD_IMAGE** — The entire item, as it appeared before it was modified.
- **NEW_AND_OLD_IMAGES** — Both the new and the old images of the item.

Trigger Concepts

- ITEM changes generate an event
- That event contains the data which changed and a action is taken using that data
- AWS = Stream + Lambda
- Lambda can be integrated to provide trigger functionality - invoking when new entries are added on the stream.
- It used for Reporting & Analytics, Aggregation, Messaging or Notifications



DynamoDB Local and Global Secondary Indexes

DynamoDB Indexes

Amazon DynamoDB provides fast access to items in a table by specifying primary key values. However, many applications might benefit from having one or more secondary (or alternate) keys available, to allow efficient access to data with attributes other than the primary key. To address this, you can create one or more secondary indexes on a table and issue Query or Scan requests against these indexes.

A secondary index is a data structure that contains a subset of attributes from a table, along with an alternate key to support Query operations. You can retrieve data from the index using a Query, in much the same way as you use Query with a table. A table can have multiple secondary indexes, which give your applications access to many different query patterns.

- Query is the most efficient operation in DDB
- Query can only work on 1 PK (Primary Key) value at a time.
- and optionally a single, or range of SK values
- Indexes are alternative views on table data
- Different SK (LSI) or different PK and SK (GSI)
- Some or all attributes (projection)

Every secondary index is automatically maintained by DynamoDB. When you add, modify, or delete items in the base table, any indexes on that table are also updated to reflect these changes.

DynamoDB supports two types of secondary indexes:

1. **Global secondary index** — An index with a partition key and a sort key that can be different from those on the base table. A global secondary index is considered "global" because queries on the index can span all of the data in the base table, across all partitions. A global secondary index is stored in its own partition space away from the base table and scales separately from the base table.

2. **Local secondary index** — An index that has the same partition key as the base table, but a different sort key. A local secondary index is "local" in the sense that every partition of a local secondary index is scoped to a base table partition that has the same partition key value.

Local Secondary Indexes (LSI)

- Some applications only need to query data using the base table's primary key. However, there might be situations where an alternative sort key would be helpful. To give your application a choice of sort keys, you can create one or more local secondary indexes on an Amazon DynamoDB table and issue Query or Scan requests against these indexes.
- A local secondary index maintains an alternate sort key for a given partition key value. A local secondary index also contains a copy of some or all of the attributes from its base table. You specify which attributes are projected into the local secondary index when you create the table. The data in a local secondary index is organized by the same partition key as the base table, but with a different sort key. This lets you access data items efficiently across this different dimension. For greater query or scan flexibility, you can create up to five local secondary indexes per table.
- Every local secondary index must meet the following conditions:
  - The partition key is the same as that of its base table.
  - The sort key consists of exactly one scalar attribute.
  - The sort key of the base table is projected into the index, where it acts as a non-key attribute.
- LSI is an alternative view for a table
- MUST be created with a table
- 5 LSIs per base table
- Alternatives SK on the table
- Shares the RCU and WCU with the table
- When you create a secondary index, you need to specify the attributes that will be projected into the index. DynamoDB provides three different options for this:
  1. KEYS_ONLY – Each item in the index consists only of the table partition key and sort key values, plus the index key values. The KEYS_ONLY option results in the smallest possible secondary index.
  2. INCLUDE – In addition to the attributes described in KEYS_ONLY, the secondary index will include other non-key attributes that you specify.
  3. ALL – The secondary index includes all of the attributes from the source table. Because all of the table data is duplicated in the index, an ALL projection results in the largest possible secondary index.

Global Secondary Indexes (GSI)

- Some applications might need to perform many kinds of queries, using a variety of different attributes as query criteria. To support these requirements, you can create one or more global secondary indexes and issue Query requests against these indexes in Amazon DynamoDB.
- To speed up queries on non-key attributes, you can create a global secondary index. A global secondary index contains a selection of attributes from the base table, but they are organized by a primary key that is different from that of the table. The index key does not need to have any of the key attributes from the table. It doesn't even need to have the same key schema as a table.
- Every global secondary index must have a partition key, and can have an optional sort key. The index key schema can be different from the base table schema. You could have a table with a simple primary key (partition key), and create a global secondary index with a composite primary key (partition key and sort key)—or vice versa.
- Can be created at any time
- Default limit of 20 per base table
- Alternative PK and SK
- GSI's Have their own RCU and WCU allocations

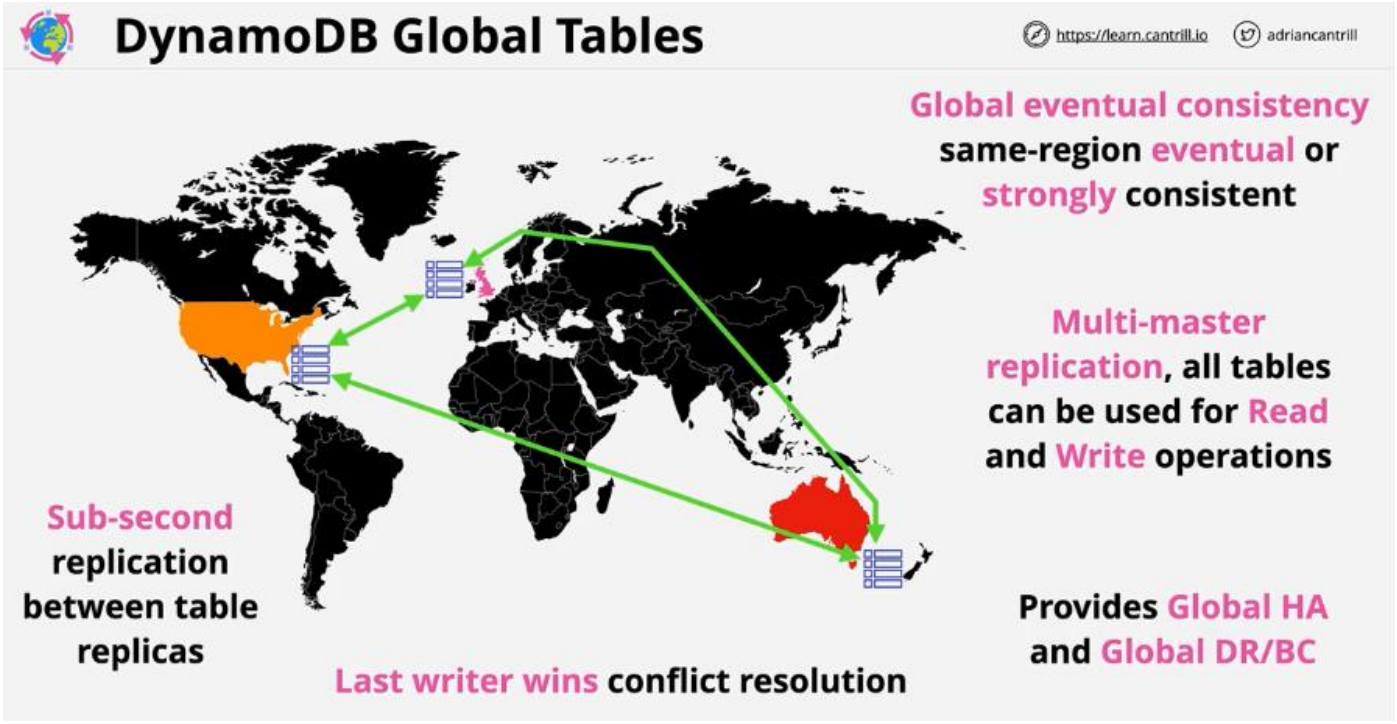
- When you create a secondary index, you need to specify the attributes that will be projected into the index. DynamoDB provides three different options for this:
  1. KEYS_ONLY – Each item in the index consists only of the table partition key and sort key values, plus the index key values. The KEYS_ONLY option results in the smallest possible secondary index.
  2. INCLUDE – In addition to the attributes described in KEYS_ONLY, the secondary index will include other non-key attributes that you specify.
  3. ALL – The secondary index includes all of the attributes from the source table. Because all of the table data is duplicated in the index, an ALL projection results in the largest possible secondary index.

LSI and GSI Considerations

- Careful with Projection (KEYS_ONLY, INCLUDE, ALL)
- Queries on attributes NOT projected are expensive
- Use GSIs as default, LSI only when strong consistency is required
- Use indexes for alternative access patterns

DynamoDB - Global Tables

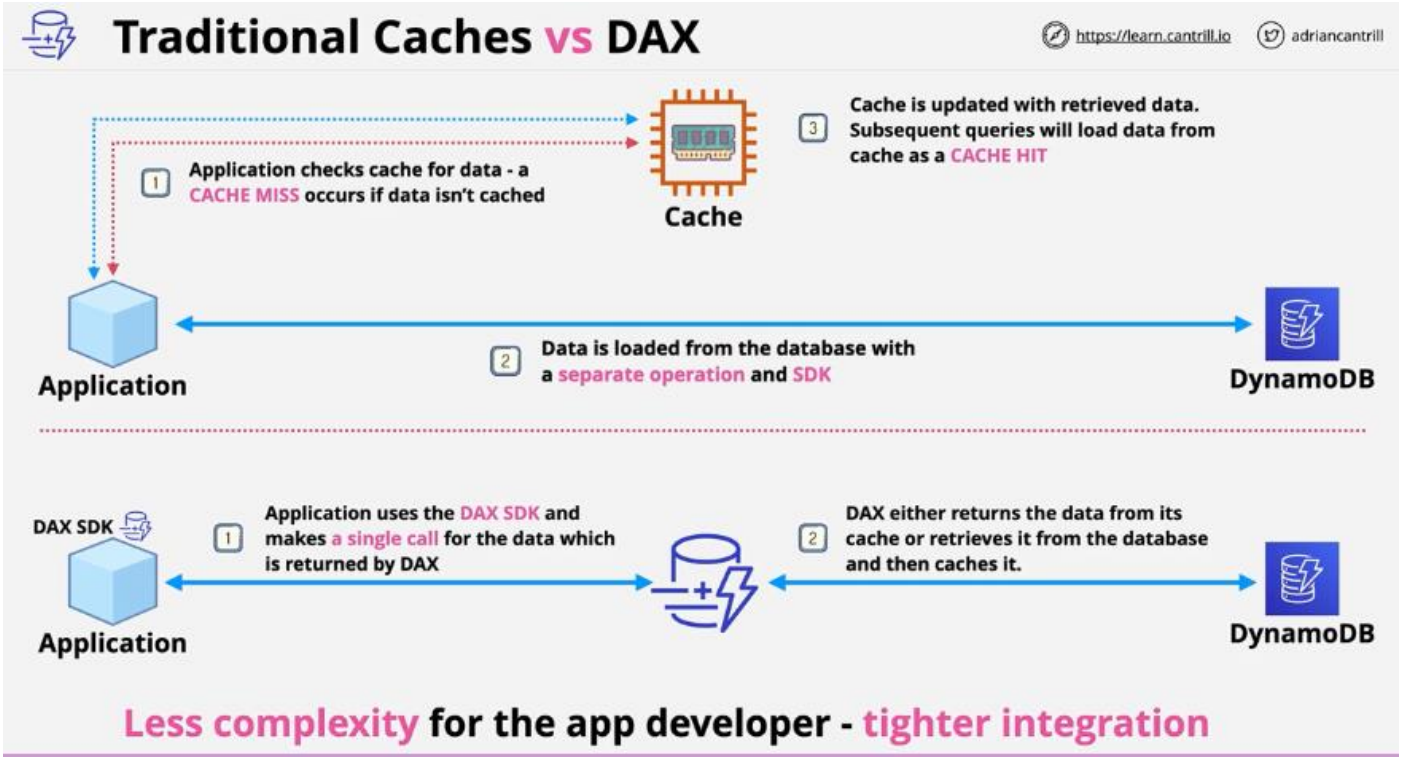
- DynamoDB Global Tables provides multi-master global replication of DynamoDB tables which can be used for performance, HA or DR/BC reasons.
- Global tables build on the global Amazon DynamoDB footprint to provide you with a fully managed, multi-region, and multi-active database that delivers fast, local, read and write performance for massively scaled, global applications. Global tables replicate your DynamoDB tables automatically across your choice of AWS Regions.
- Global tables eliminate the difficult work of replicating data between regions and resolving update conflicts, enabling you to focus on your application's business logic. In addition, global tables enable your applications to stay highly available even in the unlikely event of isolation or degradation of an entire Region.
- You can set up global tables in the AWS Management Console or AWS CLI. No application changes are required because global tables use existing DynamoDB APIs. There are no upfront costs or commitments for using global tables, and you pay only for the resources provisioned.
- Global tables provide multi-master cross-region replication
- Tables are created in multiple regions and added to the same global table (becoming replica tables)
- Last writer wins is used for conflict resolution
- Reads and Writes can occur to any region
- Generally sub-second replication between regions
- Strongly consistent reads ONLY in the same region as writes



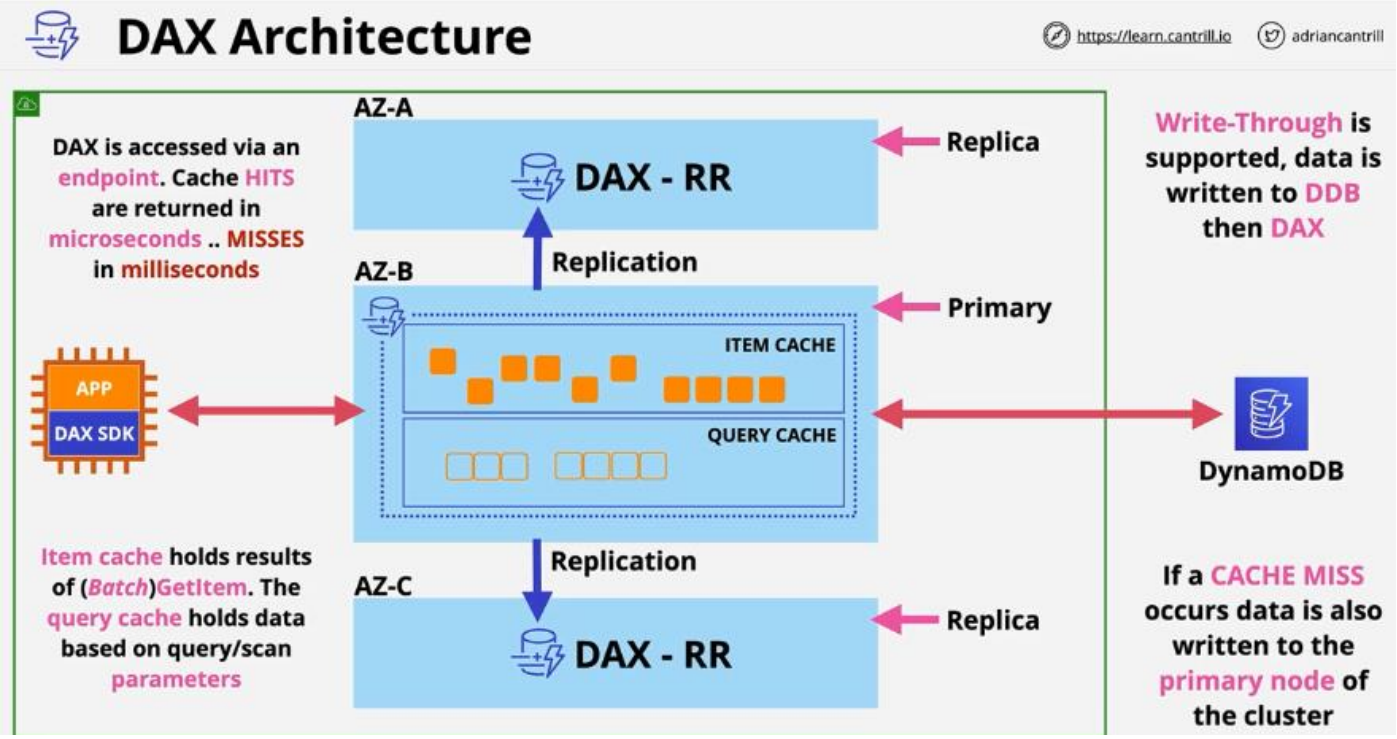
DynamoDB - Accelerator (DAX)

Amazon DynamoDB Accelerator (DAX) is a fully managed, highly available, in-memory cache for Amazon DynamoDB that delivers up to a 10 times performance improvement—from milliseconds to microseconds—even at millions of requests per second.

DAX does all the heavy lifting required to add in-memory acceleration to your DynamoDB tables, without requiring developers to manage cache invalidation, data population, or cluster management.



DAX Architecture



DAX Considerations

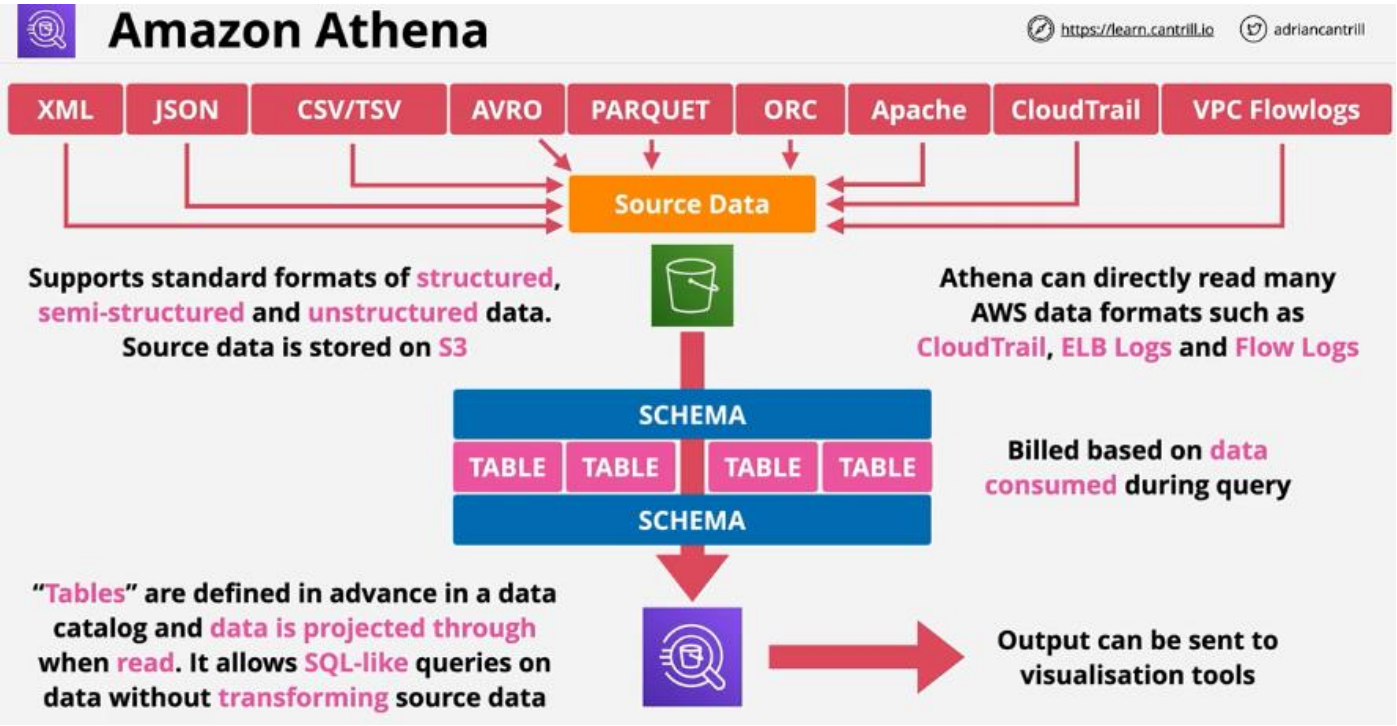
- Primary NODE (Writes) and Replicas (Read)
- Nodes are HA. primary failure = election
- In-Memory cache - Scaling and its much faster reads, reduces costs
- Scale UP and Scale OUT (Bigger or More)
- Supports write-through
- DAX Deployed WITHIN a VPC

Amazon Athena

Amazon Athena is an interactive query service that makes it easy to analyse data directly in Amazon Simple Storage Service (Amazon S3) using standard SQL. With a few actions in the AWS Management Console, you can point Athena at your data stored in Amazon S3 and begin using standard SQL to run ad-hoc queries and get results in seconds.

Athena is serverless, so there is no infrastructure to set up or manage, and you pay only for the queries you run. Athena scales automatically—running queries in parallel—so results are fast, even with large datasets and complex queries.

- Serverless Interactive Querying Service
- Ad-hoc queries on data - pay only **data consumed**
- **Schema-on-read** - table like translation.
- Original data **never changed - remains on S3.**
- Schema translates data => relational-like when read
- Output can be sent to other services
- Athena is an underrated service capable of working with unstructured, semi-structured or structured data.



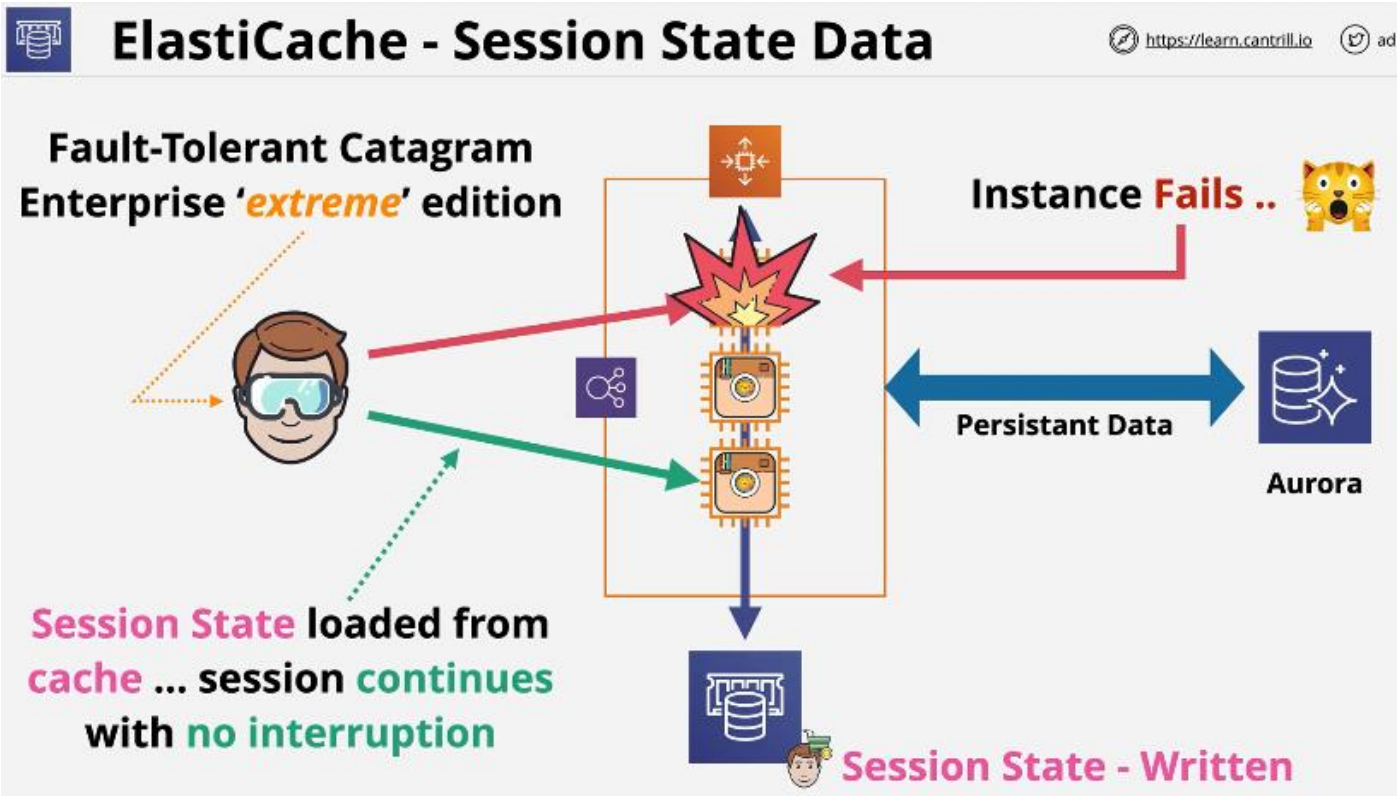
Elasticache

Amazon ElastiCache allows you to seamlessly set up, run, and scale popular open-source compatible in-memory data stores in the cloud. Build data-intensive apps or boost the performance of your existing databases by retrieving data from high throughput and low latency in-memory data stores. Amazon ElastiCache is a popular choice for real-time use cases like Caching, Session Stores, Gaming, Geospatial Services, Real-Time Analytics, and Queuing.

Amazon ElastiCache supports the Memcached and Redis cache engines. Each engine provides some advantages.

- In-memory database... **high performance**
- Managed **Redis** or **Memcached** as a service
- Can be used to **cache data** - for **READ HEAVY** workloads with low latency requirements
- **Reduces database** workloads **(expensive)**
- Can be used to store **Session Data** **(Stateless Servers)**
- **Requires applications code changes!**
- An in-memory cache allows **cost effective scaling** of **read-heavy** workloads - and **Performance improvement** at scale

Memcached	**Redis**
Simple data structures	Advanced Structures
No Replication	Multi-AZ
Multiple Nodes (Sharding)	Replication (Scale Reads)
No backups	Backup & Restore
Multi-threaded	Transactions



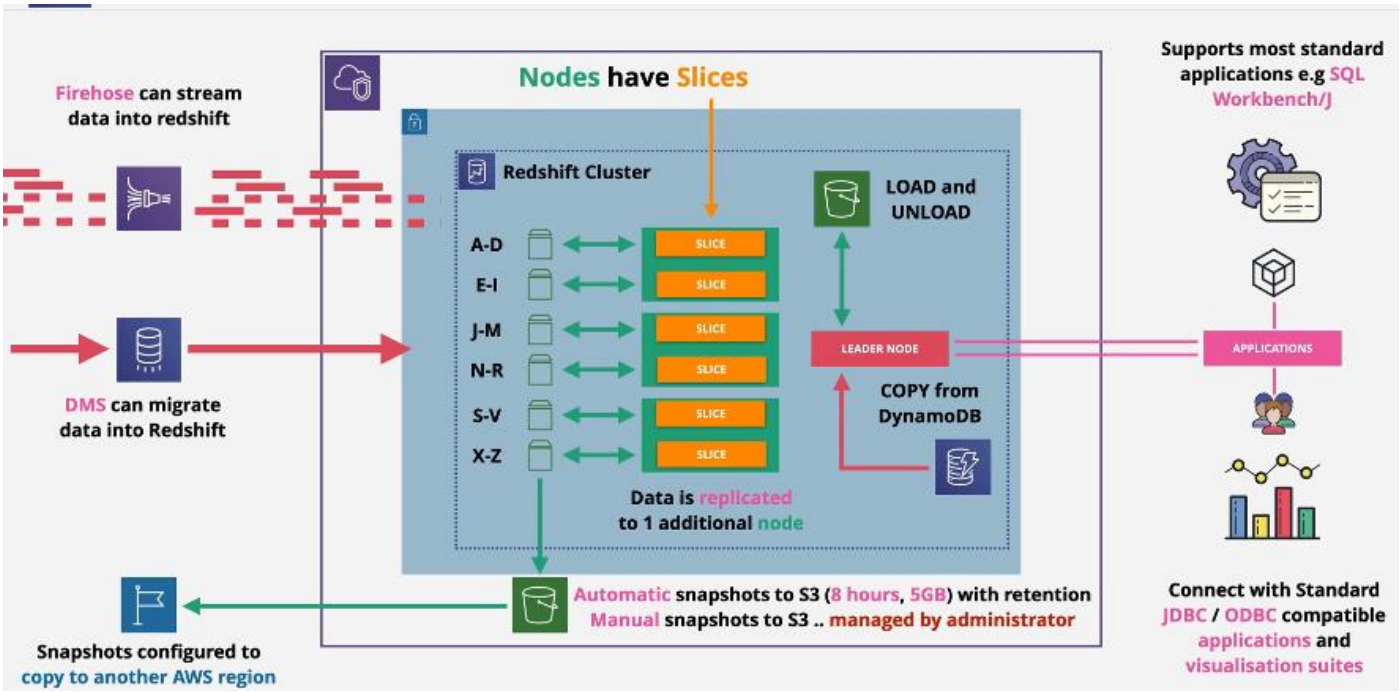
Redshift Architecture

- Amazon Redshift integrates with various data loading and ETL (extract, transform, and load) tools and business intelligence (BI) reporting, data mining, and analytics tools. Amazon Redshift is based on industry-standard PostgreSQL, so most existing SQL client applications will work with only minimal changes. For information about important differences between Amazon Redshift SQL and PostgreSQL.
- Amazon Redshift communicates with client applications by using industry-standard JDBC and ODBC drivers for PostgreSQL.
- The core infrastructure component of an Amazon Redshift data warehouse is a cluster.
- A cluster is composed of one or more compute nodes. If a cluster is provisioned with two or more compute nodes, an additional leader node coordinates the compute nodes and handles external communication. Your client application interacts directly only with the leader node. The compute nodes are transparent to external applications.
- The leader node manages communications with client programs and all communication with compute nodes. It parses and develops execution plans to carry out database operations. The leader node distributes SQL statements to the compute nodes only when a query references tables that are stored on the compute nodes.
- The leader node compiles code for individual elements of the execution plan and assigns the code to individual compute nodes. The compute nodes execute the compiled code and send intermediate results back to the leader node for final aggregation.
- A compute node is partitioned into slices. Each slice is allocated a portion of the node's memory and disk space, where it processes a portion of the workload assigned to the node. The leader node manages distributing data to the slices and apportions the workload for any queries or other database operations to the slices. The slices then work in parallel to complete the operation.

Remember this:

- **Petabyte**-scale **Data warehouse**
- **OLAP** (**Column** based) not OLTP (row/transaction)
- Pay as you use ... similar structure to RDS
- Direct Query S3 using **Redshift Spectrum**

- Direct Query other DBs using **federated query**
- Integrates with AWS tooling such as QuickSight
- SQL-like interface **JDBC/ODBC** connections
- **Server** based (**not serverless**)
- **One AZ** in a VPC - network cost/performance
- **Leader Node** - Query input, planning and aggregation
- **Compute Node** - Performing queries of data
- VPC Security, **IAM** Permissions, **KMS** at rest Encryption, CW Monitoring
- Redshift **Enhanced VPC Routing** - **VPC Networking**!



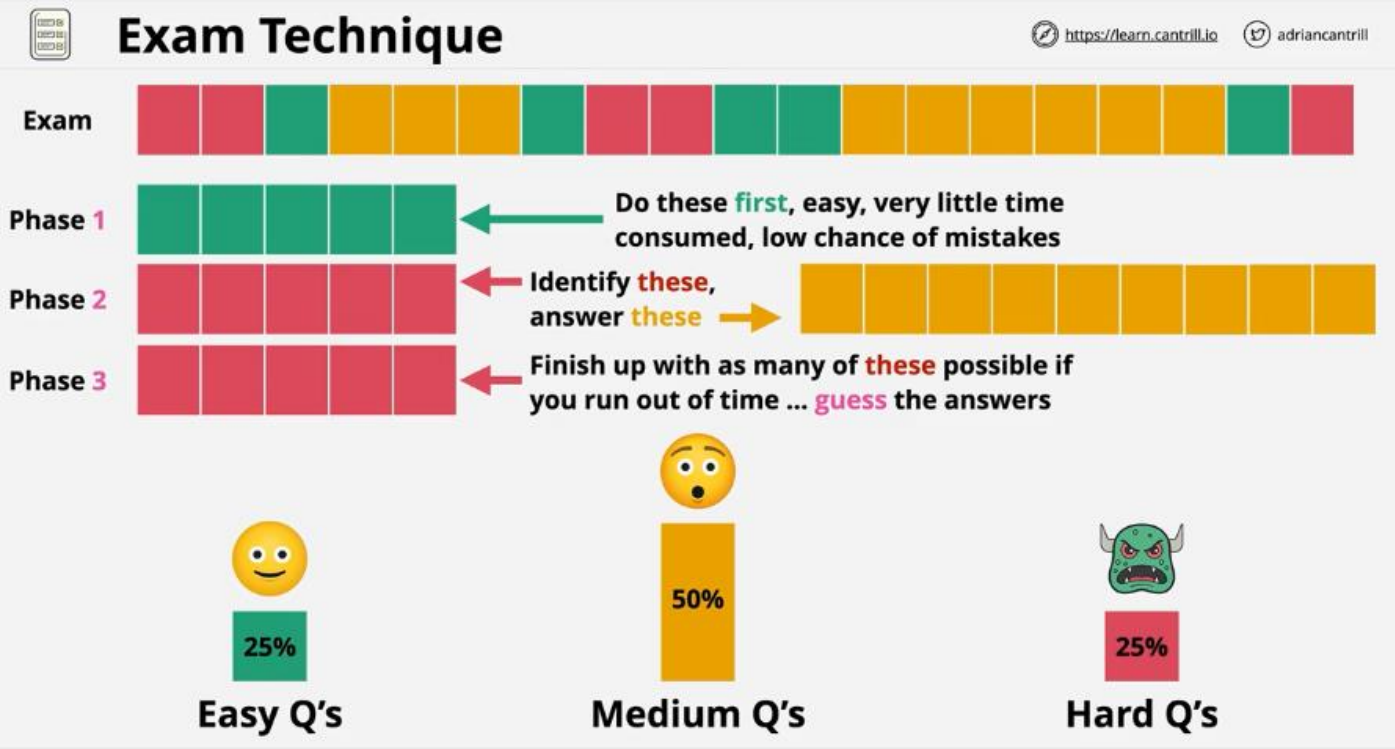
Redshift DR and Resilience

- **Automatic** incremental backups occur every - **8 hours** OR **5GB** of data and by default have **1-day retention** (Configurable up to **35 days**)
- Manual snaps can be taken at any time - and **deleted by an admin as required**
- Redshift backups into S3 protect against AZ failure
- Redshift can be configured to **copy snapshots to another region for DR** - with a **separate configurable retention period.**

MODULE 20 – EXAM

Exam Technique

- Shared Exam room, Kiosk or at home
- **130** Minutes as standard
- ESL (English Second Language) + **30 Minutes**
- **65 Questions** = 2 Minutes Per Question
- **720** /1000 Pass Mark
- **Multiple** Choice (Pick 1 / many)
- **Multi-select** (Pick correct)



- If it's your first exam, **assume you will run out of time**
- The way to succeed is to be **efficient**
- 2 minutes to read Q, Answers and make a decision
- **Don't guess until the end**... later questions may remind you of something **important from earlier**
- Use the **Mark for Review**!
- Take **ALL** the practice tests you can
- Aim for **90%+** before you do the real exam
- Try and **eliminate** any **crazy answers**
- Find **what matters** in the **question**
- Highlight and remove any **question fluff**
- Identify what **matters** in the **answers**
- **DON'T PANIC! mark for review and come back later**