
AWS CERTIFIED
SOLUTIONS ARCHITECT ASSOCIATE
SAA-C03

TABLE OF CONTENTS

INTRODUCTION

As more and more businesses migrate their on-premises workloads to Amazon Web Services (AWS), the demand for highly skilled and certified AWS Professionals will continue to rise over the coming years ahead. Companies are now leveraging on the power of cloud computing to significantly lower their operating costs and dynamically scale their resources based on demand.

Gone are the days of over-provisioning your resources that turn out to be underutilized over time. With AWS, companies can now easily provision the number of resources that they actually need and pay only the computing resources they consume. AWS helps customers to significantly reduce upfront capital investment and replace it with lower variable costs. You can opt to pay your cloud resources using an on-demand pricing option with no long-term contracts or up-front commitments. You can easily discontinue your on-demand cloud resources if you don't need them to stop any recurring operational costs, thereby reducing your operating expenses.

This flexibility isn't available in a traditional on-premises environment where you have to maintain and pay for the resources even if you aren't using them. Moreover, companies can simply launch new AWS resources in seconds to scale and accommodate the surge of incoming requests to their enterprise applications. These are the financial and technical benefits, and the reason why thousands of companies are hiring skilled IT professionals to migrate their workload to the cloud. Conversely, this is also one of the reasons why there is a demand for certified AWS professionals.

The AWS Solutions Architect Associate certification has been consistently regarded as one of the highest-paying certifications in the IT Industry today. This eBook contains essential information about the AWS Certified Solutions Architect Associate exam, as well as the topics you have to review in order to pass it. You will learn the basics of the AWS Global Infrastructure and the relevant AWS services required to build a highly available and fault-tolerant cloud architecture.

Note: We took extra care to come up with these study guides and cheat sheets, however, this is meant to be just a supplementary resource when preparing for the exam. We highly recommend working on **hands-on sessions**, **SAA-C03 video course** and **practice exams** to further expand your knowledge and improve your test taking skills.

AWS CERTIFIED SOLUTIONS ARCHITECT ASSOCIATE EXAM OVERVIEW

In 2013, Amazon Web Services began its Global Certification Program with the primary purpose of validating the technical skills and knowledge of IT Professionals for building secure and reliable cloud-based applications using the AWS platform. The AWS Certified Solutions Architect Associate exam was the first AWS certification that was released in that year. This is followed by two other role-based certification exams: the SysOps Administrator and Developer Associate.

Since then, AWS has continuously expanded the certification program – launching the Professional and Specialty-level certifications that cover various domains such as machine learning, data analytics, advanced networking, and many others. As AWS services continue to evolve, a new and updated version of the AWS certification exam is released on a regular basis. This is to reflect the recent service changes and to include the new knowledge areas.

After almost five years since its initial release, an updated version of the AWS Certified Solutions Architect - Associate certification was launched in February 2018 with an exam code of SAA-C01.

And after two years, AWS released another version of this test with an exam code of SAA-C02 in March 2020. This is followed by yet another new exam version released in August 2022. The following version of this certification test has an exam code of SAA-C03 which contains the new services that were recently released by AWS.

As you can notice, the last digit of the exam version code increments for every new version of the SAA exam. So from SAA-C01, it becomes SAA-C02, SAA-C03, SAA-C04, and so on and so forth. Knowing the latest exam code is important for finding the most up-to-date reviewers for this AWS exam.

Exam Details

The AWS Certified Solutions Architect - Associate certification is intended for IT Professionals who perform a Solutions Architect or DevOps role and have substantial years of hands-on experience designing available, cost-efficient, fault-tolerant, and scalable distributed systems on the AWS platform. It is composed of scenario-based questions that can be either in multiple-choice or multiple response formats. The first question type has one correct answer and three incorrect responses, while the latter has two or more correct responses out of five or more options. You can take the exam from a local testing center or online from the comforts of your home.

Exam Code:	SAA-C03
Release Date:	August 2022
Prerequisites:	None
No. of Questions:	65
Score Range:	100/1000
Cost:	150 USD
Passing Score:	720/1000
Time Limit:	2 hours 10 minutes (130 minutes)
Format:	Scenario-based. Multiple choice/multiple answers.
Delivery Method:	Testing center or online proctored exam

Don't be confused if you see in your Pearson Vue booking that the duration is 140 minutes since they included an additional 10 minutes for reading the Non-Disclosure Agreement (NDA) at the start of the exam and the survey at the end of it. If you booked in PSI, the exam duration time that you will see is 130 minutes.

Exam Domains

The AWS Certified Solutions Architect - Associate exam has 4 different domains, each with corresponding weight and topic coverage. The table below shows the different exam domains for the SAA-C03 exam:

EXAM DOMAIN	EXAM PERCENTAGE
Design Secure Architectures	30%
Design Resilient Architectures	26%
Design High-Performing Architectures	24%
Design Cost-Optimized Architectures	20%
TOTAL	100%

There is no doubt that these domains are really for the Solutions Architect exam. As you can see, these four domains are about designing an architecture focused on security, resiliency, high performance, and cost-optimization.

The list of exam domains can be found on the official Exam Guide for the AWS Certified Solutions Architect - Associate exam. Each exam domain is comprised of several task statements. A task statement is a sub-category of the exam domain that contains the required cloud concepts, knowledge, and skills for you to accomplish a particular task or activity in AWS. Let's look at each of these domains one by one.

The first domain covers the big chunk of the exam at 30%, followed by the second domain, which covers 26%, the third is 24%, while the last one covers 20% of the exam.

The “Design Secure Architectures” exam domain is focused on checking your knowledge in designing resilient architectures in AWS. It has the biggest domain in the exam with 30 percent coverage. Therefore, you must allocate significant time to study the various concepts covered in this domain. The series of scenarios that you will encounter in this section checks your know-how in:

- Designing secure access to AWS resources
- Designing secure workloads and applications
- Determining appropriate data security controls

The second exam domain (“Design Resilient Architectures”) is all about designing resilient architectures in AWS. It is the second biggest domain in the exam, with 26 percent coverage. You must also allocate a lot of time to understand the different concepts covered in this domain.

The questions that you will encounter in this section will challenge your knowledge in:

- Designing scalable and loosely coupled architecture.
- Designing highly available and/or fault-tolerant architectures

The third domain (“Design High-Performing Architectures”) is designing high-performing cloud architectures in AWS. This has an exam coverage of 24 percent and revolves around designing high-performing storage, computing, database, network, data ingestion, and transformation solutions. You should prepare for:

- Determining high-performing and/or scalable storage solutions
- Designing high-performing and elastic compute solutions
- Determining high-performing database solutions
- Determining high-performing and/or scalable network architectures and finally,
- Determining high-performing data ingestion and transformation solutions.

The last exam domain revolves around designing cost-optimized architectures. It comprises 20% of the exam coverage so you have to limit the time you spend reviewing the concepts under this domain. As you might have guessed, this domain is all about the costs of your cloud architecture and the different ways to reduce operational expenditures.

This section tests your knowledge in:

- Designing cost-optimized storage solutions
- Designing cost-optimized compute solutions
- Designing cost-optimized database solutions
- Designing cost-optimized network architectures

These are the four exam domains that you should be familiar with when you start your exam preparations. Again, the SAA exam is primarily focused on security so make sure that you focus on the “Design Secure Architectures” domain and all the related knowledge areas in its task statements.

I highly recommend that you read the official exam guide for the AWS Certified Solutions Architect Associate exam from cover to cover. Pay close attention to the topics included, and don't forget to read the Appendix section, which contains a list of related AWS services that will appear in the exam.

Exam Scoring System

You can get a score from 100 to 1,000 with a minimum passing score of **720** when you take the AWS Certified Solutions Architect - Associate exam. AWS is using a scaled scoring model to equate scores across multiple exam types that may have different difficulty levels. The complete score report will be sent to you by email after a few days.

Individuals who unfortunately do not pass the AWS exam must wait 14 days before they are allowed to retake the exam. Fortunately, there is no hard limit on exam attempts until you pass the exam. Take note that on each attempt, the full registration price of the AWS exam must be paid.

Within 5 business days of completing your exam, your AWS Certification Account will have a record of your complete exam results. The score report contains a table of your performance at each section/domain, which indicates whether you met the competency level required for these domains or not. AWS is using a compensatory scoring model, which means that you do not necessarily need to pass each and every individual section, only the overall examination. Each section has a specific score weighting that translates to the number of questions; hence, some sections have more questions than others. The Score Performance table highlights your strengths and weaknesses that you need to improve on.

Exam Benefits

If you successfully passed any AWS exam, you will be eligible for the following benefits:

- **Exam Discount** - You'll get a 50% discount voucher that you can apply for your recertification or any other exam you plan to pursue. To access your discount voucher code, go to the “Benefits” section of your AWS Certification Account, and apply the voucher when you register for your next exam.

- **Free Practice Exam** - To help you prepare for your next exam, AWS provides another voucher that you can use to take any official AWS practice exam for free. You can access your voucher code from the “Benefits” section of your AWS Certification Account.

- **AWS Certified Store** - All AWS certified professionals will be given access to exclusive AWS Certified merchandise. You can get your store access from the “Benefits” section of your AWS Certification Account.

- **Certification Digital Badges** - You can showcase your achievements to your colleagues and employers with digital badges on your email signatures, Linkedin profile, or on your social media accounts. You can also show your Digital Badge to gain exclusive access to Certification Lounges at AWS re:Invent, regional Appreciation Receptions, and select AWS Summit events. To view your badges, simply go to the “Digital Badges” section of your AWS Certification Account.

- **Eligibility to join AWS IQ -** With the AWS IQ program, you can monetize your AWS skills online by providing hands-on assistance to customers around the globe. AWS IQ will help you stay sharp and be well-versed on various AWS technologies. You can work at the comforts of your home and decide when or where you want to work. Interested individuals must, have an Associate, Professional, or Specialty AWS Certification and be over 18 years of age.

You can visit the official AWS Certification FAQ page to view the frequently asked questions about getting AWS Certified and other information about the AWS Certification: https://aws.amazon.com/certification/faqs/.

AWS CERTIFIED SOLUTIONS ARCHITECT ASSOCIATE EXAM - STUDY GUIDE AND TIPS

The AWS Certified Solutions Architect Associate SAA-C03 exam, or SAA for short, is one of the most sought after certifications in the Cloud industry. This certification attests to your knowledge of the AWS Cloud and building a well-architected infrastructure in AWS.

As a Solutions Architect, it is your responsibility to be familiar with the services that meet your customer requirements. Aside from that, you should also have the knowledge to create an efficient, secure, reliable, fault tolerant, and cost-effective infrastructure out of these services. Your AWS SA Associate exam will be based upon these topics.

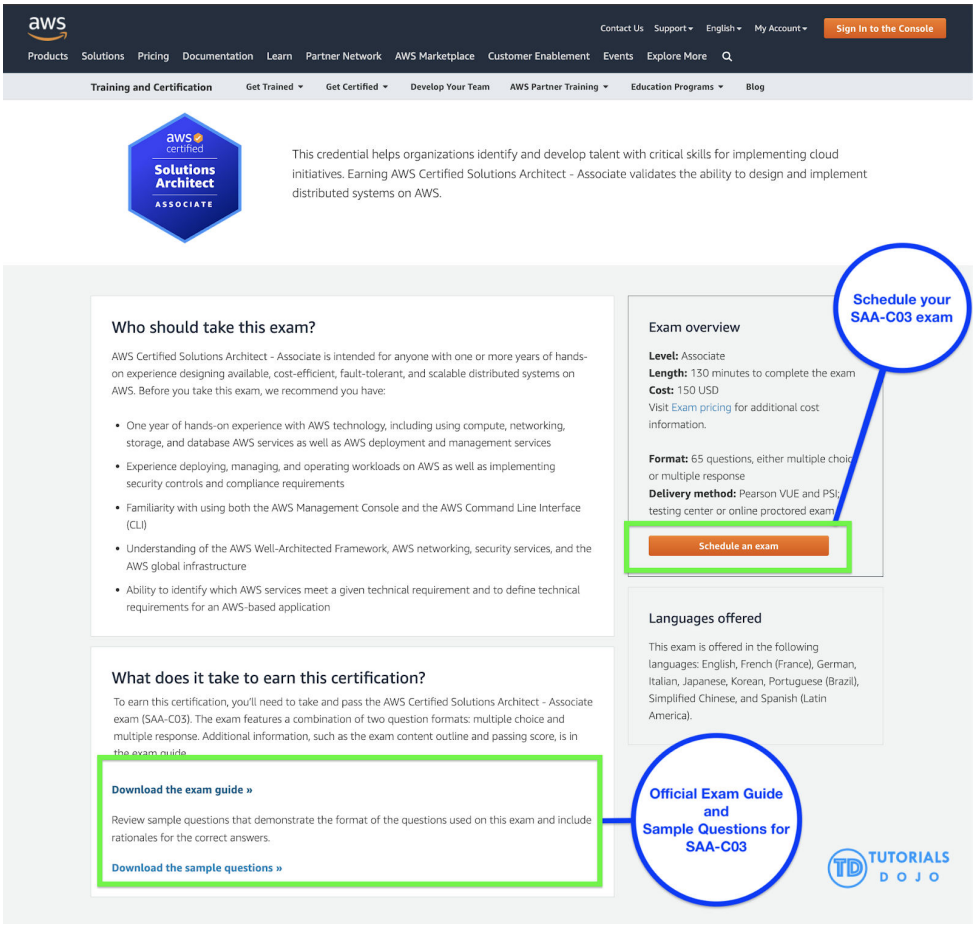
Whitepapers, FAQs, and the AWS Documentation will be your primary study materials for this exam. Experience in building systems will also be helpful, since the exam consists of multiple scenario type questions. You can learn more details on your exam through the official SAA-C03 Exam Guide here. Do a quick read on it to be aware of how to prepare and what to expect on the exam itself.

SAA-C03 Study Materials

As a starting point for your AWS Certified Solutions Architect Associate exam studies, we recommend taking the FREE AWS Certified Cloud Practitioner Essential digital course. This free and highly interactive course aims to improve AWS Cloud knowledge by covering different AWS Cloud concepts, AWS services, security, architecture, pricing, and support plans. If you are quite new to AWS, taking and completing this digital course should be your first step for your SAA-C03 exam prep.

There are a lot of posts on the Internet claiming the “best” course for the AWS Certified Solutions Architect Associate SAA-C03 Exam. However, some of these resources are already obsolete and don't cover the latest topics that were recently introduced in the SAA-C03 test. How can I ensure that you are using the right study materials for your upcoming AWS Certified Solutions Architect Associate test?

The best thing to do is to check the official AWS Certification website for the most up-to-date information. You can also head on to the official AWS Certification page for the **AWS Certified Solutions Architect Associate SAA-C03** exam. This page is where you can find the actual link to schedule your SAA-C03 exam as well as get the official SAA-C03 **Exam Guide** and **Sample Questions** as shown below:



For the exam version (SAA-C03), you should also know the following services:

- AWS Global Accelerator
- Elastic Fabric Adapter (EFA)
- Elastic Network Adapter (ENA)
- AWS ParallelCluster
- Amazon FSx
- AWS DataSync
- AWS Directory Service
- High Performance Computing
- Aurora Serverless

There are more SAA-C03 topics that we have recently added to our:

- **AWS Certified Solutions Architect Associate Practice Exams**
- **AWS Certified Solutions Architect Associate Video Course**

Core AWS Services to Focus On for the SAA-C03 Exam

1. EC2 - As the most fundamental compute service offered by AWS, you should know about EC2 inside out.
2. Lambda - Lambda is the common service used for serverless applications. Study how it is integrated with other AWS services to build a full stack serverless app.
3. Elastic Load Balancer - Load balancing is very important for a highly available system. Study about the different types of ELBs, and the features each of them supports.
4. Auto Scaling - Study what services in AWS can be auto scaled, what triggers scaling, and how auto scaling increases/decreases the number of instances.
5. Elastic Block Store - As the primary storage solution of EC2, study on the types of EBS volumes available. Also study how to secure, backup and restore EBS volumes.
6. S3 / Glacier - AWS offers many types of S3 storage depending on your needs. Study what these types are and what differs between them. Also review on the capabilities of S3 such as hosting a static website, securing access to objects using policies, lifecycle policies, etc. Learn as much about S3 as you can.
7. Storage Gateway - There are occasional questions about Storage Gateway in the exam. You should understand when and which type of Storage Gateway should be used compared to using services like S3 or EBS. You should also know the use cases and differences between DataSync and Storage Gateway.
8. EFS - EFS is a service highly associated with EC2, much like EBS. Understand when to use EFS, compared to using S3, EBS or instance store. Exam questions involving EFS usually ask the trade off between cost and efficiency of the service compared to other storage services.
9. RDS / Aurora - Know how each RDS database differs from one another, and how they are different from Aurora. Determine what makes Aurora unique, and when it should be preferred from other databases (in terms of function, speed, cost, etc). Learn about parameter groups, option groups, and subnet groups.
10. DynamoDB - The exam includes lots of DynamoDB questions, so read as much about this service as you can. Consider how DynamoDB compares to RDS, Elasticache and Redshift. This service is also commonly used for serverless applications along with Lambda.
11. Elasticache - Familiarize yourself with Elasticache redis and its functions. Determine the areas/services where you can place a caching mechanism to improve data throughput, such as managing session state of an ELB, optimizing RDS instances, etc.
12. VPC/NACL/Security Groups - Study every service that is used to create a VPC (subnets, route tables, internet gateways, nat gateways, VPN gateways, etc). Also, review on the differences of network access control lists and security groups, and during which situations they are applied.
13. Route 53 - Study the different types of records in Route 53. Study also the different routing policies. Know what hosted zones and domains are.
14. IAM - Services such as IAM Users, Groups, Policies and Roles are the most important to learn. Study how IAM integrates with other services and how it secures your application through different policies. Also read on the best practices when using IAM.

15. CloudWatch - Study how monitoring is done in AWS and what types of metrics are sent to CloudWatch. Also read upon Cloudwatch Logs, CloudWatch Alarms, and the custom metrics made available with CloudWatch Agent.
16. CloudTrail - Familiarize yourself with how CloudTrail works, and what kinds of logs it stores as compared to CloudWatch Logs.
17. Kinesis - Read about Kinesis sharding and Kinesis Data Streams. Have a high level understanding of how each type of Kinesis Stream works.
18. CloudFront - Study how CloudFront helps speed up websites. Know what content sources CloudFront can serve from. Also check the kinds of certificates CloudFront accepts.
19. SQS - Gather info on why SQS is helpful in decoupling systems. Study how messages in the queues are being managed (standard queues, FIFO queues, dead letter queues). Know the differences between SQS, SNS, SES, and Amazon MQ.
20. SNS - Study the function of SNS and what services can be integrated with it. Also be familiar with the supported recipients of SNS notifications.
21. SWF / CloudFormation / OpsWorks - Study how these services function. Differentiate the capabilities and use cases of each of them. Have a high level understanding of the kinds of scenarios they are usually used in.

Based on our exam experience, you should also know when to use the following:

- AWS DataSync vs Storage Gateway
- FSx (Cold and Hot Storage)
- Cross-Region Read Replicas vs. Multi-Az RDS - which database provides high-availability
- Amazon Object key vs Object Metadata
- Direct Connect vs. Site-to-Site VPN
- AWS Config vs AWS CloudTrail
- Security Group vs NACL
- NAT Gateway vs NAT Instance
- Geolocation routing policy vs. Geoproximity routing policy on Route 53

The AWS Documentation and FAQs will be your primary source of information. You can also visit **Tutorials Dojo's AWS Cheat Sheets** to gain access to a repository of thorough content on the different AWS services mentioned above. Complete our **AWS Certified Solutions Architect Associate Video Course** and aim to get a consistent 90% score on all sets of our **AWS Certified Solutions Architect Associate Practice Exams**.

Lastly, try out these services yourself by signing up in AWS and performing some lab exercises. Experiencing them on your own will help you greatly in remembering what each service is capable of.

Common Exam Scenarios

Scenario	Solution
Domain 1: Design Secure Architectures	
Encrypt EBS volumes restored from the unencrypted EBS snapshots	Copy the snapshot and enable encryption with a new symmetric CMK while creating an EBS volume using the snapshot.
Limit the maximum number of requests from a single IP address.	Create a rate-based rule in AWS WAF and set the rate limit.
Grant the bucket owner full access to all uploaded objects in the S3 bucket.	Create a bucket policy that requires users to set the object's ACL to bucket-owner-full-control.
Protect objects in the S3 bucket from accidental deletion or overwrite.	Enable versioning and MFA delete.
Access resources on both on-premises and AWS using on-premises credentials that are stored in Active Directory.	Set up SAML 2.0-Based Federation by using a Microsoft Active Directory Federation Service.
Secure the sensitive data stored in EBS volumes	Enable EBS Encryption
Ensure that the data-in-transit and data-at-rest of the Amazon S3 bucket is always encrypted	Enable Amazon S3 Server-Side or use Client-Side Encryption
Secure the web application by allowing multiple domains to serve SSL traffic over the same IP address.	Use AWS Certificate Manager to generate an SSL certificate. Associate the certificate to the CloudFront distribution and enable Server Name Indication (SNI).
Control the access for several S3 buckets by using a gateway endpoint to allow access to trusted buckets.	Create an endpoint policy for trusted S3 buckets.
Enforce strict compliance by tracking all the configuration changes made to any AWS services.	Set up a rule in AWS Config to identify compliant and non-compliant services.
Domain 2: Design Resilient Architectures	
Set up asynchronous data replication to another RDS DB instance hosted in another AWS Region	Create a Read Replica
A parallel file system for “hot” (frequently accessed) data	Amazon FSx For Lustre

Implement synchronous data replication across Availability Zones with automatic failover in Amazon RDS.	Enable Multi-AZ deployment in Amazon RDS.
Needs a storage service to host “cold” (infrequently accessed) data	Amazon S3 Glacier
Set up a relational database and a disaster recovery plan with an RPO of 1 second and RTO of less than 1 minute.	Use Amazon Aurora Global Database.
Monitor database metrics and send email notifications if a specific threshold has been breached.	Create an SNS topic and add the topic in the CloudWatch alarm.
Set up a DNS failover to a static website.	Use Route 53 with the failover option to a static S3 website bucket or CloudFront distribution.
Implement an automated backup for all the EBS Volumes.	Use Amazon Data Lifecycle Manager to automate the creation of EBS snapshots.
Monitor the available swap space of your EC2 instances	Install the CloudWatch agent and monitor the SwapUtilizationmetric.
Domain 3: High-Performing Architectures	
Implement a fanout messaging.	Create an SNS topic with a message filtering policy and configure multiple SQS queues to subscribe to the topic.
A database that has a read replication latency of less than 1 second.	Use Amazon Aurora with cross-region replicas.
A specific type of Elastic Load Balancer that uses UDP as the protocol for communication between clients and thousands of game servers around the world.	Use Network Load Balancer for TCP/UDP protocols.
Monitor the memory and disk space utilization of an EC2 instance.	Install Amazon CloudWatch agent on the instance.
Retrieve a subset of data from a large CSV file stored in the S3 bucket.	Perform an S3 Select operation based on the bucket's name and object's key.
Upload 1 TB file to an S3 bucket.	Use Amazon S3 multipart upload API to upload large objects in parts.

Improve the performance of the application by reducing the response times from milliseconds to microseconds.	Use Amazon DynamoDB Accelerator (DAX)
Retrieve the instance ID, public keys, and public IP address of an EC2 instance.	Access the url: http://169.254.169.254/latest/meta-data/ using the EC2 instance.
Route the internet traffic to the resources based on the location of the user.	Use Route 53 Geolocation Routing policy.
Domain 4: Design Cost-Optimized Architectures	
A cost-effective solution for over-provisioning of resources.	Configure a target tracking scaling in ASG.
The application data is stored in a tape backup solution. The backup data must be preserved for up to 10 years.	Use AWS Storage Gateway to backup the data directly to Amazon S3 Glacier Deep Archive.
Accelerate the transfer of historical records from on-premises to AWS over the Internet in a cost-effective manner.	Use AWS DataSync and select Amazon S3 Glacier Deep Archive as the destination.
Globally deliver the static contents and media files to customers around the world with low latency.	Store the files in Amazon S3 and create a CloudFront distribution. Select the S3 bucket as the origin.
An application must be hosted to two EC2 instances and should continuously run for three years. The CPU utilization of the EC2 instances is expected to be stable and predictable.	Deploy the application to a Reserved instance.
Implement a cost-effective solution for S3 objects that are accessed less frequently.	Create an Amazon S3 lifecycle policy to move the objects to Amazon S3 Standard-IA.
Minimize the data transfer costs between two EC2 instances.	Deploy the EC2 instances in the same Region.
Import the SSL/TLS certificate of the application.	Import the certificate into AWS Certificate Manager or upload it to AWS IAM.

Validate Your Knowledge

When you are feeling confident with your review, it is best to validate your knowledge through sample exams. You can take this practice exam from AWS for free as additional material, but do not expect your real exam to be on the same level of difficulty as this practice exam on the AWS website. **Tutorials Dojo** offers a very useful and well-reviewed set of practice tests for AWS Solutions Architect Associate SAA-C03 takers **here**. Each test contains unique questions that will surely help verify if you have missed out on anything important that might appear on your exam. You can pair our practice exams with this study guide eBook to further help in your exam preparations.

If you have scored well on the **Tutorials Dojo AWS Certified Solutions Architect Associate practice tests** and you think you are ready, then go earn your certification with your head held high. If you think you are lacking in certain areas, better go review them again, and take note of any hints in the questions that will help you select the correct answers. If you are not that confident that you'll pass, then it would be best to reschedule your exam to another day, and take your time preparing for it. In the end, the efforts you have put in for this will surely reward you.

Sample SAA-C03 Practice Test Questions

Question 1

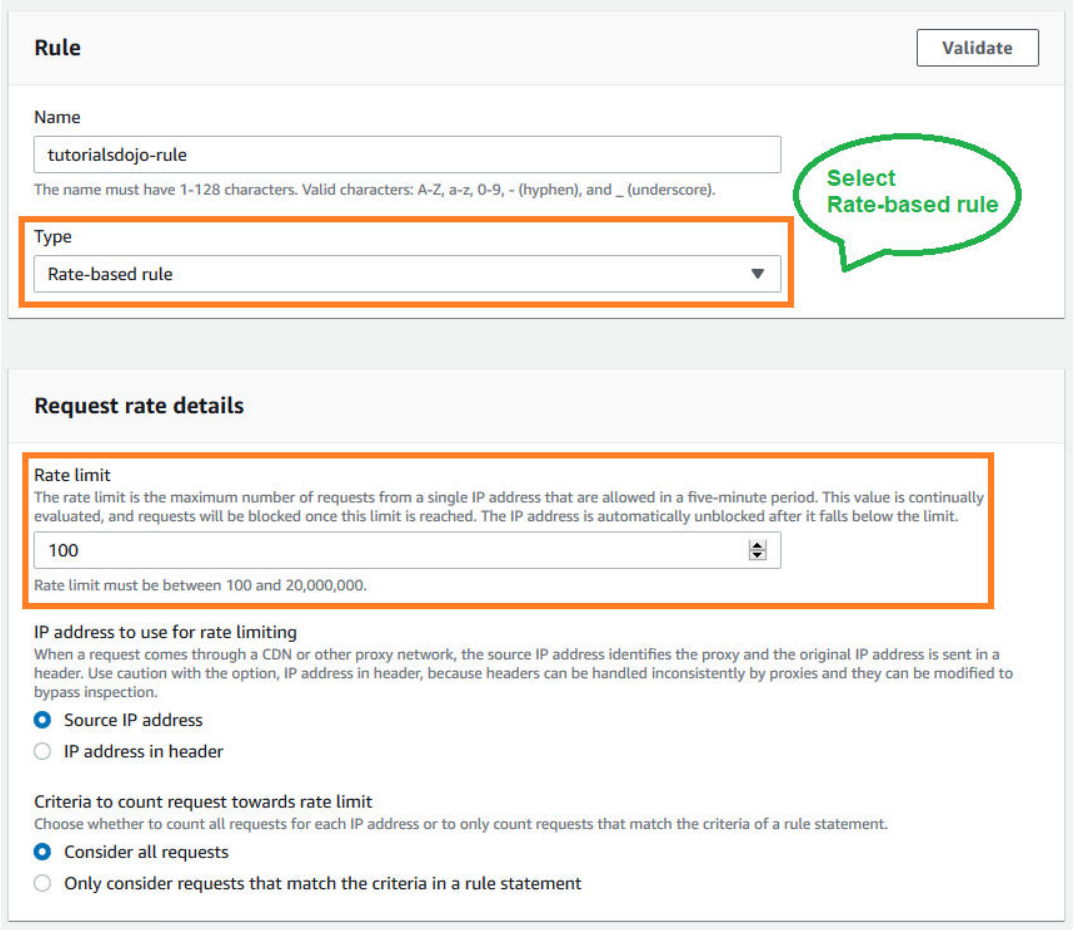
A company hosted an e-commerce website on an Auto Scaling group of EC2 instances behind an Application Load Balancer. The Solutions Architect noticed that the website is receiving a large number of illegitimate external requests from multiple systems with IP addresses that constantly change. To resolve the performance issues, the Solutions Architect must implement a solution that would block the illegitimate requests with minimal impact on legitimate traffic.

Which of the following options fulfills this requirement?

1. Create a regular rule in AWS WAF and associate the web ACL to an Application Load Balancer.
2. Create a custom network ACL and associate it with the subnet of the Application Load Balancer to block the offending requests.
3. Create a rate-based rule in AWS WAF and associate the web ACL to an Application Load Balancer.
4. Create a custom rule in the security group of the Application Load Balancer to block the offending requests.

Correct Answer: 3

AWS WAF is tightly integrated with Amazon CloudFront, the Application Load Balancer (ALB), Amazon API Gateway, and AWS AppSync – services that AWS customers commonly use to deliver content for their websites and applications. When you use AWS WAF on Amazon CloudFront, your rules run in all AWS Edge Locations, located around the world close to your end-users. This means security doesn't come at the expense of performance. Blocked requests are stopped before they reach your web servers. When you use AWS WAF on regional services, such as Application Load Balancer, Amazon API Gateway, and AWS AppSync, your rules run in the region and can be used to protect Internet-facing resources as well as internal resources.



A rate-based rule tracks the rate of requests for each originating IP address and triggers the rule action on IPs with rates that go over a limit. You set the limit as the number of requests per 5-minute time span. You can use this type of rule to put a temporary block on requests from an IP address that's sending excessive requests. Based on the given scenario, the requirement is to limit the number of requests from the illegitimate requests without affecting the genuine requests. To accomplish this requirement, you can use AWS WAF web ACL. There are two types of rules in creating your own web ACL rule: regular and rate-based rules. You need to select the latter to add a rate limit to your web ACL. After creating the web ACL, you can associate it with ALB. When the rule action triggers, AWS WAF applies the action to additional requests from the IP address until the request rate falls below the limit.

Hence, the correct answer is: **Create a rate-based rule in AWS WAF and associate the web ACL to an Application Load Balancer.**

The option that says: **Create a regular rule in AWS WAF and associate the web ACL to an Application Load Balancer** is incorrect because a regular rule only matches the statement defined in the rule. If you need to add a rate limit to your rule, you should create a rate-based rule.

The option that says: **Create a custom network ACL and associate it with the subnet of the Application Load Balancer to block the offending requests** is incorrect. Although NACLs can help you block incoming traffic, this option wouldn't be able to limit the number of requests from a single IP address that is dynamically changing.

The option that says: **Create a custom rule in the security group of the Application Load Balancer to block the offending requests** is incorrect because the security group can only allow incoming traffic. Remember that you can't deny traffic using security groups. In addition, it is not capable of limiting the rate of traffic to your application unlike AWS WAF.

References:
https://docs.aws.amazon.com/waf/latest/developerguide/waf-rule-statement-type-rate-based.html
https://aws.amazon.com/waf/faqs/

Check out this AWS WAF Cheat Sheet:
https://tutorialsdojo.com/aws-waf/

Question 2

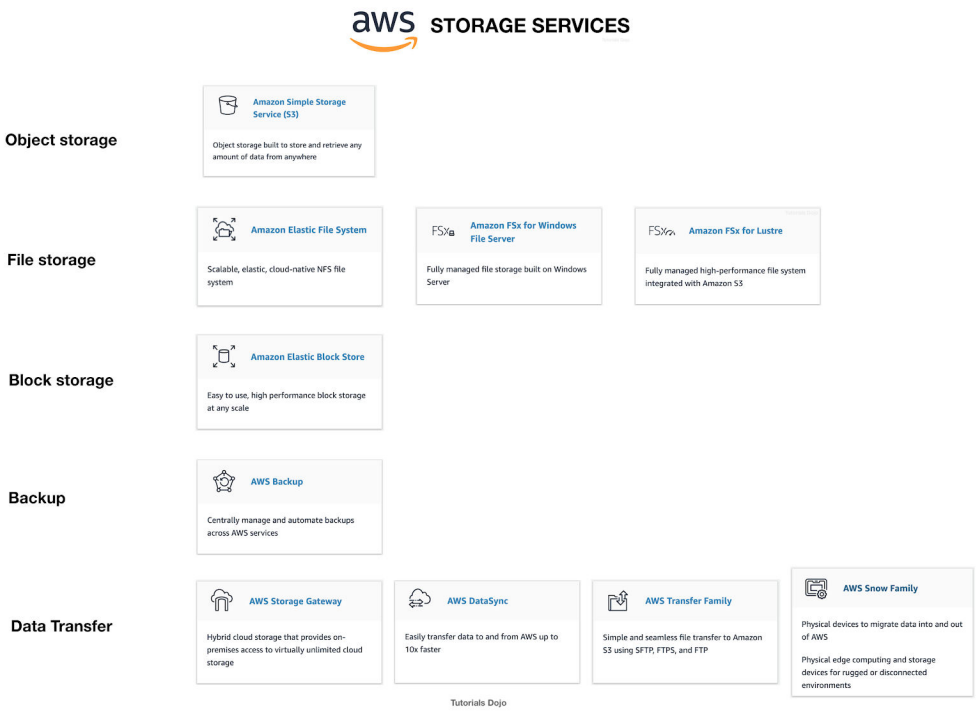
An AI-powered Forex trading application consumes thousands of data sets to train its machine learning model. The application's workload requires a high-performance, parallel hot storage to process the training datasets concurrently. It also needs cost-effective cold storage to archive those datasets that yield low profit.

Which of the following Amazon storage services should the developer use?

1. Use Amazon FSx For Lustre and Amazon EBS Provisioned IOPS SSD (io1) volumes for hot and cold storage respectively.
2. Use Amazon FSx For Lustre and Amazon S3 for hot and cold storage respectively.
3. Use Amazon Elastic File System and Amazon S3 for hot and cold storage respectively.
4. Use Amazon FSx For Windows File Server and Amazon S3 for hot and cold storage respectively.

Correct Answer: 2

Hot storage refers to the storage that keeps frequently accessed data (hot data). **Warm storage** refers to the storage that keeps less frequently accessed data (warm data). **Cold storage** refers to the storage that keeps rarely accessed data (cold data). In terms of pricing, the colder the data, the cheaper it is to store, and the costlier it is to access when needed.



Amazon FSx For Lustre is a high-performance file system for fast processing of workloads. Lustre is a popular open-source **parallel file system** which stores data across multiple network file servers to maximize performance and reduce bottlenecks.

Amazon FSx for Windows File Server is a fully managed Microsoft Windows file system with full support for the SMB protocol, Windows NTFS, Microsoft Active Directory (AD) Integration.

Amazon Elastic File System is a fully-managed file storage service that makes it easy to set up and scale file storage in the Amazon Cloud.

Amazon S3 is an object storage service that offers industry-leading scalability, data availability, security, and performance. S3 offers different storage tiers for different use cases (frequently accessed data, infrequently accessed data, and rarely accessed data).

The question has two requirements:

1. High-performance, parallel hot storage to process the training datasets concurrently.
2. Cost-effective cold storage to keep the archived datasets that are accessed infrequently

In this case, we can use **Amazon FSx For Lustre** for the first requirement, as it provides a high-performance, parallel file system for hot data. On the second requirement, we can use Amazon S3 for storing the cold data. Amazon S3 supports a cold storage system via Amazon S3 Glacier / Glacier Deep Archive.

Hence, the correct answer is: **Use Amazon FSx For Lustre and Amazon S3 for hot and cold storage respectively**.

Using Amazon FSx For Lustre and Amazon EBS Provisioned IOPS SSD (io1) volumes for hot and cold storage respectively is incorrect because the Provisioned IOPS SSD (io1) volumes are designed as a hot storage to meet the needs of I/O-intensive workloads. EBS has a storage option called Cold HDD but it is not used for storing cold data. In addition, EBS Cold HDD is a lot more expensive than using Amazon S3 Glacier / Glacier Deep Archive.

Using Amazon Elastic File System and Amazon S3 for hot and cold storage respectively is incorrect because although EFS supports concurrent access to data, it does not have the high-performance ability that is required for machine learning workloads.

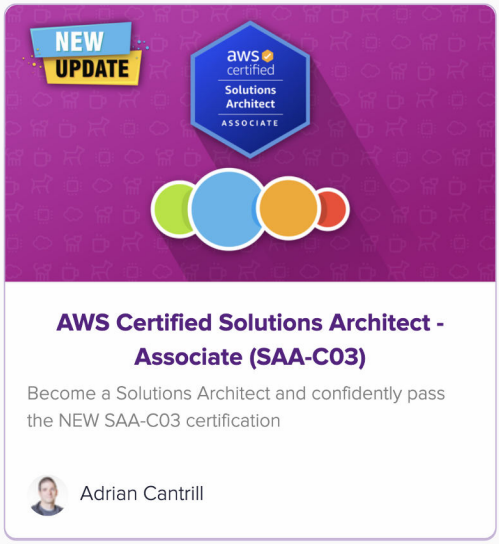
Using Amazon FSx For Windows File Server and Amazon S3 for hot and cold storage respectively is incorrect because Amazon FSx For Windows File Server does not have a parallel file system, unlike Lustre.

References:
https://aws.amazon.com/fsx/
https://docs.aws.amazon.com/whitepapers/latest/cost-optimization-storage-optimization/aws-storage-services.html
https://aws.amazon.com/blogs/startups/picking-the-right-data-store-for-your-workload/

Check out this Amazon FSx Cheat Sheet:
https://tutorialsdojo.com/amazon-fsx/

Additional SAA-C03 Training Materials

There are a few top-rated AWS Certified Solutions Architect Associate SAA-C03 video courses that you can check out as well, which can complement your exam preparations especially if you are the type of person who can learn better through visual courses instead of reading long whitepapers. We recommend the AWS Certified Solutions Architect - Associate video course by Adrian Cantrill which covers both the fundamentals and advanced concepts of the SAA-C03 exam in detail:



We also have a concise **AWS Certified Solutions Architect Associate video training course** that will equip you with the exam-specific knowledge that you need to understand in order to pass the SAA-C03 exam:

Based on the feedback of thousands of our students in our practice test course, the combination of any of these video courses plus our practice tests and this study guide eBook were enough to pass the exam and even get a good score.

Some Notes Regarding Your SAA-C03 Exam

The AWS Solutions Architect Associate (SAA-C03) exam loves to end questions that ask for highly available or cost-effective solutions. Be sure to understand the choices provided to you, and verify that they have correct details. Some choices are very misleading such that it seems it is the most appropriate answer to the question, but contains an incorrect detail of some service.

When unsure of which options are correct in a multi-select question, try to eliminate some of the choices that you believe are false. This will help narrow down the feasible answers to that question. The same goes for multiple choice type questions. Be extra careful as well when selecting the number of answers you submit. Check out the tips mentioned by the recent exam passers in the **r/AWSCertifications** sub in Reddit for more information.

As mentioned in this review, you should be able to differentiate services that belong in one category with one another. Common comparisons include:

- EC2 vs ECS vs Lambda
- S3 vs EBS vs EFS
- CloudFormation vs OpsWorks vs Elastic Beanstalk
- SQS vs SNS vs SES vs MQ
- Security Group vs nACLs
- The different S3 storage types vs Glacier
- RDS vs DynamoDB vs Elasticache
- RDS engines vs Aurora

The **Tutorials Dojo Comparison of AWS Services** contains excellent cheat sheets comparing these seemingly similar services which are crucial to solving the tricky scenario-based questions in the actual exam. By knowing each service's capabilities and use cases, you can consider these types of questions already half-solved.

Lastly, be on the lookout for “key terms” that will help you realize the answer faster. Words such as millisecond latency, serverless, managed, highly available, most cost effective, fault tolerant, mobile, streaming, object storage, archival, polling, push notifications, etc are commonly seen in the exam. Time management is very important when taking AWS certification exams, so be sure to monitor the time you consume for each question.

CLOUD COMPUTING BASICS

Cloud computing is a piece of technology that the industry has embraced to be a strong driver of innovation. Having resources available at your fingertips makes work just way easier and faster to accomplish. With virtually unlimited compute power and storage that one can provision on-demand from anywhere with internet access, companies can shift their focus to delivering their products and services to their customers, and reach their highest potential. Rather than owning these infrastructures, they can rent them as a service and pay only for what they consume.

Cloud computing allows companies and merchants to create a predictable and controllable budget plan that they can allocate and maximize in any way they see fit. Best of all, as more people use the cloud, the more the cost of using cloud services drops, thanks to economies of scale.

The concept of cloud computing has been there for quite a long time already, but it has only gained traction recently when more and more companies began to adopt these cloud providers such as Amazon Web Services. It is not a secret that it was tough to build such large scales of infrastructure and gain the trust of customers to run their applications on these shared spaces. Only in 2006 did Amazon Web Services (AWS) begin offering IT infrastructure services to businesses in the form of web services, which is now known as cloud computing. Even though the cloud provider is still fairly young, AWS has been an initiator and a constant leader in delivering what cloud computing promises to its customers – fast, cheap and reliable infrastructure and software services.

Cloud Computing Service Models

Services in the cloud can be categorized into different models depending on how they work. The most common models include:

1. **IaaS** – which stands for “infrastructure-as-a-service”. These cloud computing services are the counterpart of purchasing your own hardware on-premises, minus the purchasing part. You rent them from the cloud provider and use them as if they were your own compute and storage devices.
2. **PaaS** – which stands for “platform-as-a-service”. These services are a bit similar with IaaS, but offer more utility and convenience for the customer. One example is a web hosting service, where you won't need to worry about the underlying hardware your website is running on, so you can focus on your website deployment and management instead.
3. **SaaS** – which stands for “software-as-a-service”. These services totally remove the infrastructure part from the equation. You use these services according to the features and utility they offer to you. A good example is email.

There are other models that you might encounter here and there, such as DBaaS, which means “database-as-a-service, but for the sake of this study guide, we will be focusing primarily on the three above.

As with every piece of technology, there are pros and cons to using cloud computing. *Cloud computing is not for everyone.* It is not always the case that moving to the cloud lowers your overall expenses, or gives you that competitive edge against your competitors. It takes careful planning for one to commit to the cloud. You might rashly board on to the cloudspace, only to realize later that it is not working out for you financially and functionally. Moving out of the cloud can be as hard *and as expensive* as moving into the cloud. Therefore, you must properly evaluate the benefits that you want to achieve with cloud computing vs having things run on-premises.

History of AWS

Amazon Web Services (AWS) is one of the many cloud service providers that you can choose.

In 2004, Amazon Web Services started out as a department within Amazon. Back then, AWS only refers to a collection of web APIs and tools to access the Amazon.com e-commerce catalog. These web APIs are primarily used by early Amazon customers for their internal processing and integration, rather than a public service. This is how AWS started before it would eventually become the world's leading Infrastructure-As-A-Service platform.

Two years after, in 2006, AWS officially started its operation as a public cloud service provider. From then on, anyone can now access and use their web services. The first product that AWS publicly released is a storage service called Amazon S3. This is followed by a queueing service named: Amazon SQS, which is also released in the same year.

Today, AWS offers hundreds of fully-featured services that are available globally. It provides a highly reliable, scalable, and low-cost infrastructure platform in the cloud that powers hundreds of thousands of businesses around the world. It boasts a broad set of cloud-based products including compute, storage, databases, analytics, networking, security, artificial intelligence, and many more.

AWS is the world's leading cloud platform. It is used by millions of customers to run various workloads, optimize processes, lower costs, and scale their infrastructure in a matter of minutes.

As more and more businesses migrate their on-premises workloads to AWS Cloud, the demand for highly skilled and certified AWS Professionals will continue to rise over the coming years ahead.

CLOUD COMPUTING CONCEPTS

Before we jump into the nitty-gritty of AWS, let's first go through some of the general concepts of cloud computing.

1. **Public Cloud**
   As the name suggests, public cloud is the type of cloud computing that the majority are using right now. This is what you may know as AWS, Azure, Google Cloud and many more. The public cloud offers a lot of benefits to its users given that their infrastructures commonly span multiple locations, which are continuously improved and have dedicated support. The public cloud, therefore, has enough capacity to support a large number of customers simultaneously, and is often the go-to for future companies looking into cloud technology.

2. **Private Cloud**
   Private cloud is a type of cloud computing deployment model that only spans within the network of a company or a corporation. The company manages the hardware and the network that it has, while still enjoying some of the benefits of the cloud. An internal team then decides how to allocate and distribute their resources amongst their developers so that there is less security risk. Companies that have strict compliances against public cloud services use private cloud instead to ensure that their operations can operate with enough capacity and minimal downtime. The catch is that, with this level of infrastructure, the expenses can become much higher and/or it will not be as globally extensive as the public cloud providers.

3. **Hybrid Cloud**
   Hybrid cloud is like a buffet. You take a piece of this and a piece of that, but the whole point of it is you eat happily in the end. Hybrid cloud means you are not committing everything into the public or private cloud. You can have a mix of operations running in the public cloud, while all your data is kept on-premises. Or you can also have different cloud providers handling different projects, depending on the strengths and weaknesses of these cloud providers. There is no rule stating that you should put all your eggs in one basket. By carefully deciding how you want to build your operations, you not only achieve the desired efficiency of your projects, but also gain the best value for your money.

4. **High Availability**
   High availability means having redundant copies of an object or resource to make sure that another can take its place when something happens to it. High availability can apply to almost anything: compute servers, data storage, databases, networks, etc. High availability is one of the main selling points of using the cloud. It might be expensive, but companies that cannot risk having downtime nor data loss should build highly available infrastructures in the cloud to protect their assets. Furthermore, because the data centers in the cloud are geographically distributed and are usually far apart from one another, in case one of these data centers go offline, other data centers are not affected and can continue serving you.

5. **Fault Tolerance**
   Fault tolerance is different from high availability. Fault tolerance means that a system can continue operating even if one or more components begin to degrade and fail. Oftentimes, fault tolerance can be attributed to redundancy as well. When a component begins to fail, the system detects this and replaces the faulty component to restore working operations. Other times, fault tolerance can mean proper error handling. When a component begins to fail, the system detects this and reroutes the operation to somewhere else that is healthy. A properly built infrastructure is capable of withstanding component degradation and eventual failure, and if possible, repair itself as well.

6. **Elasticity**
   Elasticity is the ability to quickly provision resources when you need them, and release them once you don't need them anymore. Unlike traditional infrastructure, in the cloud, you should treat servers and storage as disposable. They should not be kept beyond their usefulness. Compute power and storage space can be easily acquired anyway when you need it, so be cost-effective with your budget, use only what you need and don't keep them idle. Elasticity is another major selling point of the cloud, since you do not have hardware ownership. You don't need to worry about purchasing new hardware to meet your requirements and think about how to get your money back once it is beyond its lifespan.

7. **Scalability**
   Scalability is the concept of provisioning additional resources to increase performance and support high demand, and reducing them once demand is not as high anymore. Scalability is an important practice that you must apply to keep your users happy. Imagine if your website suddenly receives a high number of traffic, and you don't have enough compute power to serve content to all your customers. The negative impact on customer satisfaction will greatly affect your reputation and your profits. When scaling a resource, like a website for example, make sure that it is stateless so that you won't lose any important data once it scales down. You should also use appropriate metrics as a basis of your scaling activity.

8. **Redundancy**
   Redundancy is a mix of all the things above. It is important that you practice redundancy in the cloud, as it can protect you from all sorts of issues that are not as tolerable in an on-premises setup. There are a lot of things in the cloud that you can *and must* apply redundancy. It's not just servers and databases, but also file storages, security applications, networks, monitoring tools and even personnel. By having additional layers of safeguards, you lessen the risk of things going haywire and costing you more than a few bucks of extra servers.

9. **Disaster Recovery**
   Disaster recovery is the practice of ensuring that you have a standardized plan on how to recover your operations in case of total failure. Usually, this means having a copy of your infrastructure running in a different location, so that if your primary experiences a disaster, you can quickly failover to your

secondary. Your disaster recovery plan depends on the amount of time that you have to bring back up your operations (RTO), and the amount of data loss that your business can tolerate (RPO). Having a disaster recovery plan is crucial especially for live production databases. We have a number of DR strategies that meet different RTO and RPO objectives, which we will discuss in more detail later on.

10. **Serverless**
    Serverless is a cloud computing model wherein the cloud provider handles the server and all maintenance, while you just put your code in. The term “Serverless” confuses a bunch of people who think that there are literally no servers involved in this model. That's not true. Serverless is still using servers in the backend, but it takes away from you the responsibility of provisioning and maintaining one, so you can dedicate everything to your code and not have to worry about scalability, patching, etc. Serverless involves a whole new dynamic of writing code and building applications, so it may not fit everyone's bill. The technology can save you a lot of cost due to its lower pricing than those of traditional server models, but it may also introduce additional complexity to your code due to its distributed nature. You also lose a lot of control over your environment if you usually manage your own runtimes, etc. Serverless functions are also event-driven. If you're a Node JS developer, get ready for a lot of callbacks with this one.

AWS BASICS

There is much for us to know about Amazon Web Services. What is their cloud computing model? What advantages do they bring to us users? Are they secure enough for us to trust them with our applications? These are just some of the questions that we will be tackling in this section.

AWS Overview

In 2006, AWS started offering IT infrastructure services to businesses as web services. The intention was to solve common infrastructure troubles that businesses often encounter in a traditional setup. With the cloud, businesses no longer need to plan for and procure servers and other IT infrastructure in advance. In AWS, they can instantly provision hundreds to thousands of servers in a few minutes and deliver results faster. Today, AWS provides a highly reliable, scalable, low-cost infrastructure platform in the cloud that supports multiple businesses around the globe.

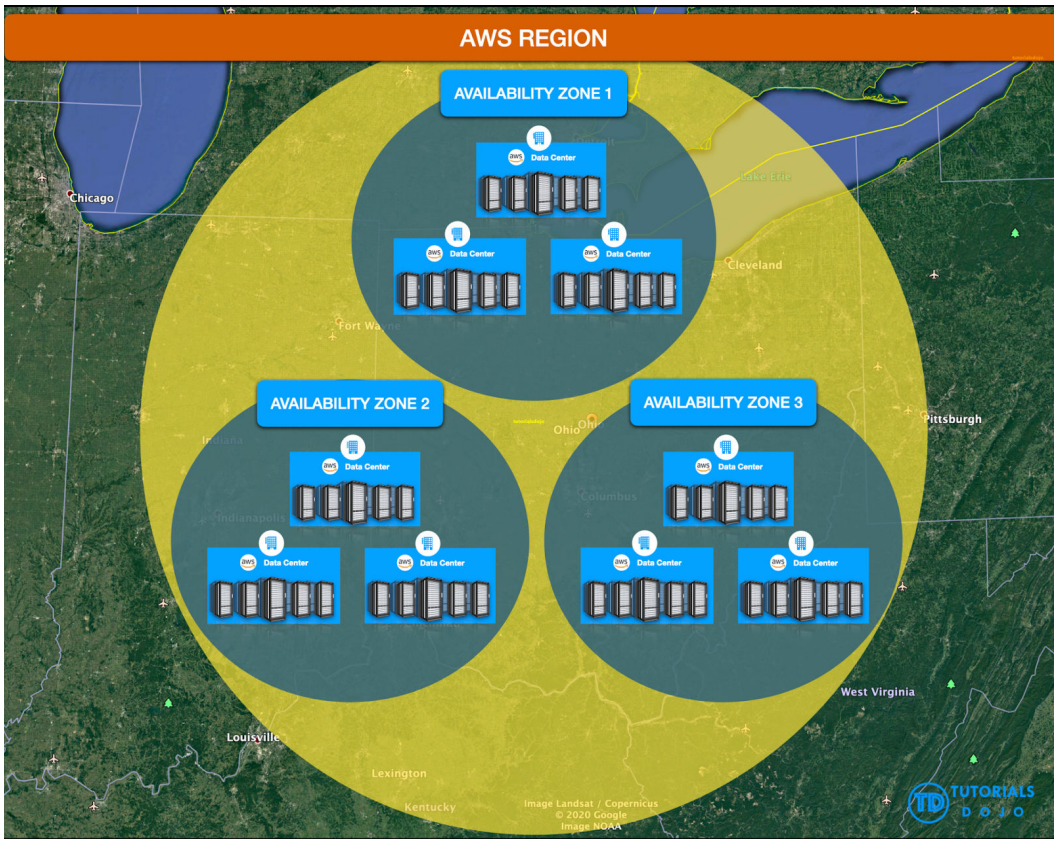
Advantages of AWS Cloud Computing

- **Trade capital expense for variable expense** – The principle of cloud is, pay for what you use, and how much you use it. You don't need to allocate a huge chunk of your capital just so you can purchase additional servers or additional storage *that you think you might need* and leave them idle collecting dust. That's why in the cloud, you should treat resources as something easily attainable, as well as something easily disposable.
- **Benefit from massive economies of scale** – By using cloud computing, you can achieve a lower variable cost than you can get on your own. Many customers adopt AWS as their cloud provider, and the number increases each day. The more customers use AWS, the more AWS can achieve higher economies of scale, which lowers pay as-you-go prices.
- **Stop guessing capacity** – Not knowing how much capacity you need is alright in AWS. AWS can easily scale compute and storage as much as you need it to. That is why it is also a great idea to do some benchmarking in the cloud, since you do not have to worry about running out of resources. Once you have a baseline, you can adjust your scaling metrics and running resources to save on cost.
- **Increase speed and agility** – In a cloud computing environment, new resources can be provisioned in a single click of a button. The cloud brings a lot of convenience to your developers since it reduces the time needed to obtain additional resources. In return, you gain a dramatic increase in agility for the organization, since the cost and time it takes to experiment and innovate is significantly lower.
- **Stop spending money running and maintaining data centers** – Cloud computing lets you focus on your own customers, rather than on the physical maintenance of your servers. Use your time and money on your projects, on your applications and on your people. You can save up on huge capital if you remove the physical aspect from the equation.

- **Go global in minutes** – You can easily deploy your application in multiple regions around the world with just a few clicks thanks to the wide coverage of AWS data centers. By strategically choosing which regions and locations you deploy your applications in, you can provide lower latency and a better experience for your customers at minimal cost.

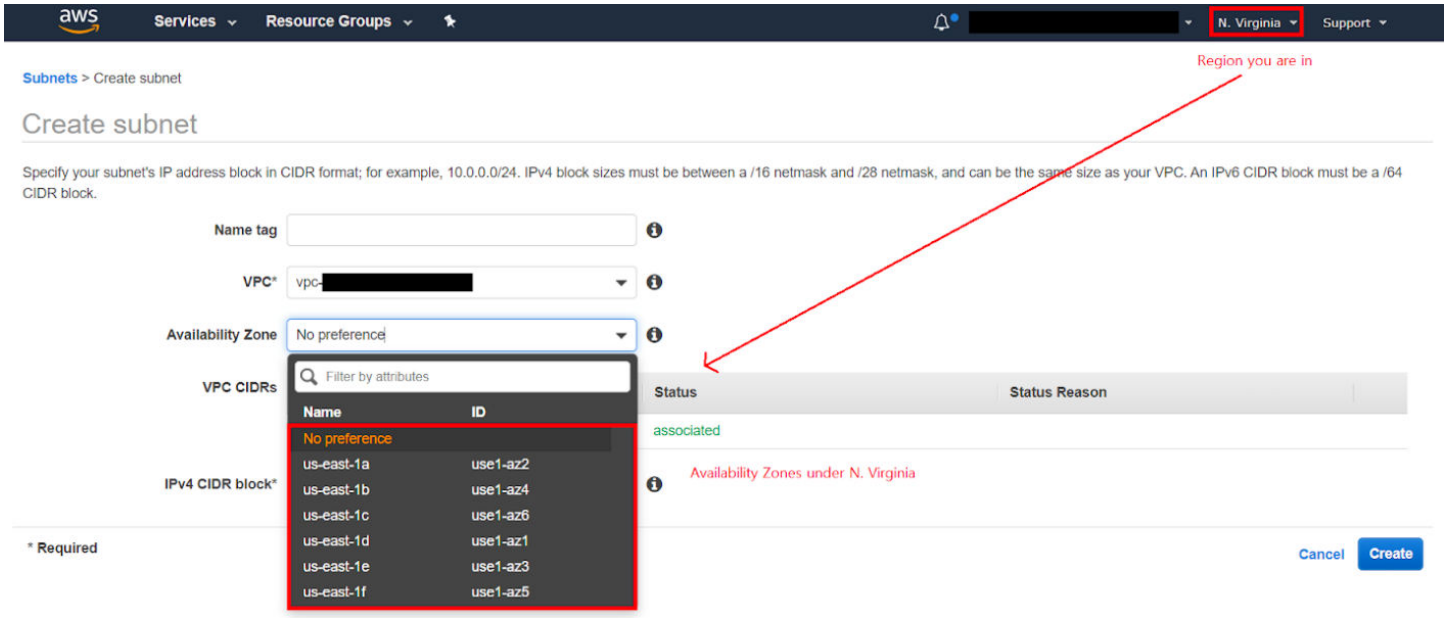
AWS Global Infrastructure

Regions provide multiple, physically separated and isolated **Availability Zones** which are connected with low latency, high throughput, and highly redundant networking.



Availability Zones offer highly availability, fault tolerance, and scalability.

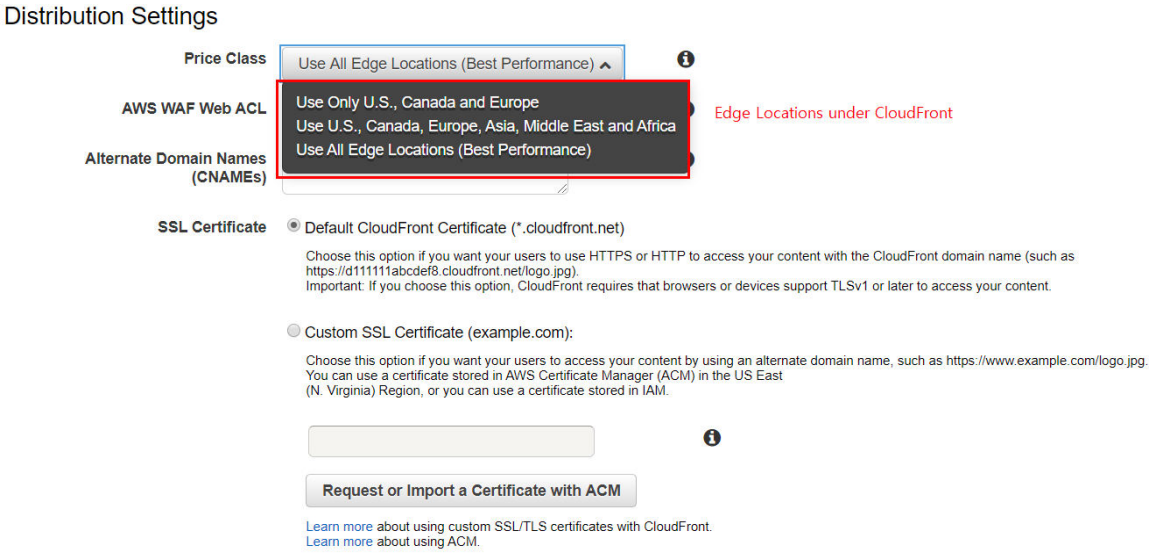
- They consist of one or more discrete data centers, each with redundant power, networking, and connectivity, housed in separate facilities.
- An Availability Zone is represented by a **region code** followed by a **letter identifier**; for example, us-east-1a.
- Availability Zone codes are used almost everywhere, especially if you are interacting with AWS programmatically.



An **AWS Local Region** is a single datacenter designed to complement an existing AWS Region. An **AWS Local Zone** places AWS compute, storage, database, and other select services closer to large population, industry, and IT centers, which makes it ideal for use cases such as content creation, real-time gaming, live video streaming, and more.

To deliver low-latency content to users around the globe, AWS has placed **Points of Presence**, which are either edge locations or edge caches. These points are used by Cloudfront and Lambda@Edge services.

Edge locations are sites that CloudFront uses to cache copies of your content for faster delivery to your users.



AWS Security and Compliance

Since a lot of customers rely on AWS for their infrastructure needs, naturally it is THE PRIORITY of AWS to make sure their security is of the highest level. AWS offers multiple layers of protection to ensure that their hardware is well-protected and their customer data are fully secured. They also make sure to keep everything well-maintained and updated, both hardware and software. Having multiple tenants sharing the same server rack can cause a lot of businesses huge worries over their data privacy and data security. It is only through tight security checks and compliance audits can public cloud providers such as AWS gain the trust of their customers.

As an AWS customer, you inherit all the best practices of AWS policies, architecture, and operational processes built to satisfy the requirements of their most security-sensitive customers. In the cloud, the responsibility of security is a shared one. AWS secures what they can on their end, while you secure what you can on your end. Only this way can everyone protect their valuable data. And therefore, AWS has developed multiple tools and services to help you achieve your security objectives. You can also review the numerous audits and certifications that third-party auditors have conducted on AWS, so that whenever you need to fulfill strict compliance with the use of a service, you can simply verify its status through the catalog.

AWS Pricing

- There are three fundamental drivers of cost with AWS:
    - Compute
    - Storage
    - Outbound data transfer.
- AWS offers pay-as-you-go for pricing.
- For certain services like **Amazon EC2, Amazon EMR,** and **Amazon RDS**, you can invest in reserved capacity. With Reserved Instances, you can save up to 75% over equivalent on-demand capacity. When you buy Reserved Instances, the larger the upfront payment, the greater the discount.
    - With the **All Upfront** option, you pay for the entire Reserved Instance term with one upfront payment. This option provides you with the largest discount compared to On-Demand instance pricing.
    - With the **Partial Upfront** option, you make a low upfront payment and are then charged a discounted hourly rate for the instance for the duration of the Reserved Instance term.
    - The **No Upfront** option does not require any upfront payment and provides a discounted hourly rate for the duration of the term.
- There are also volume-based discounts for services such as **Amazon S3.**
- For new accounts, AWS Free Tier is available.
    - Free Tier offers limited usage of AWS products at no charge for 12 months since the account was created. More details at https://aws.amazon.com/free/.
- You can estimate your monthly AWS bill using **AWS Pricing Calculator**.

AWS Well-Architected Framework Pillars

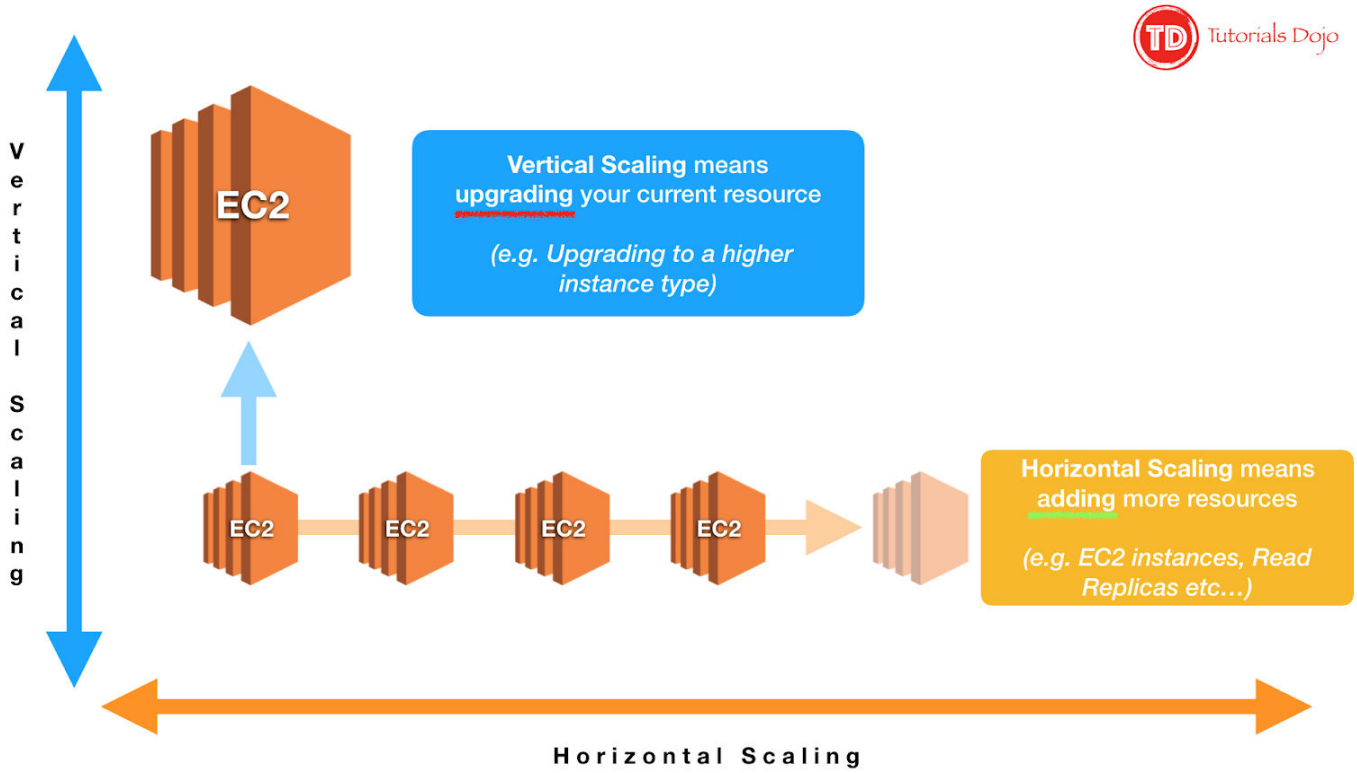
Having well-architected systems greatly increases the plausibility of business success which is why AWS created the AWS Well-Architected Framework. This framework is composed of different pillars that help you understand the pros and cons of decisions you make while building cloud architectures and systems on the AWS platform. You will learn the architectural best practices for designing and operating reliable, efficient, cost-effective and secure systems in the cloud by using the framework. It also provides a way to consistently measure your architectures against best practices and identify areas for improvement.

- Operational Excellence
    - The ability to support development and run workloads effectively, gain insight into their operations, and to continuously improve supporting processes and procedures to deliver business value.
    - Design Principles
        - Perform operations as code
        - Make frequent, small, reversible changes
        - Refine operations procedures frequently
        - Anticipate failure
        - Learn from all operational failures
- Security
    - The ability to protect data, systems, and assets to take advantage of cloud technologies to improve your security.
    - Design Principles
        - Implement a strong identity foundation
        - Enable traceability

        - Apply security at all layers
        - Automate security best practices
        - Protect data in transit and at rest
        - Keep people away from data
        - Prepare for security events
- Reliability
    - The ability of a workload to perform its intended function correctly and consistently when it's expected to. This includes the ability to operate and test the workload through its total lifecycle.
    - Design Principles
        - Automatically recover from failure
        - Test recovery procedures
        - Scale horizontally to increase aggregate workload availability
        - Stop guessing capacity
        - Manage change in automation
- Performance Efficiency
    - The ability to use computing resources efficiently to meet system requirements, and to maintain that efficiency as demand changes and technologies evolve.
    - Design Principles
        - Democratize advanced technologies
        - Go global in minutes
        - Use serverless architectures
        - Experiment more often
        - Consider mechanical sympathy
- Cost Optimization
    - The ability to run systems to deliver business value at the lowest price point.
    - Design Principles
        - Implement Cloud Financial Management
        - Adopt a consumption model
        - Measure overall efficiency
        - Stop spending money on undifferentiated heavy lifting
        - Analyze and attribute expenditure
- Sustainability
    - The ability to increase efficiency across all components of a workload by maximizing the benefits from the provisioned resources.
    - Design Principles:
        - Understand your impact
        - Establish sustainability goals
        - Maximize utilization
        - Anticipate and adopt new, more efficient hardware and software offerings
        - Use managed services
        - Reduce the downstream impact of your cloud workloads

Best Practices when Architecting in the Cloud

- **Focus on scalability**
  - **Scaling Horizontally** - an increase in the number of resources. When scaling horizontally, you want your resources to be stateless and receive a well-distributed load of work.
  - **Scaling Vertically** - an increase in the specifications of an individual resource, such as to a higher instance type for EC2 instances.



- **Disposable Resources Instead of Fixed Servers**
  - **Instantiating Compute Resources** - automate setting up of new resources along with their configuration and code through methods such as bootstrapping, Docker images or golden AMIs.
  - **Infrastructure as Code** - AWS assets are programmable. You can apply techniques, practices, and tools from software development to make your whole infrastructure reusable, maintainable, extensible, and testable.
- **Use Automation**
  - **Serverless Management and Deployment** - being serverless shifts your focus to automation of your code deployment. AWS handles the management tasks for you.
  - **Infrastructure Management and Deployment** - AWS automatically handles details, such as resource provisioning, load balancing, auto scaling, and monitoring, so you can focus on resource deployment.

    - **Alarms and Events** - AWS services will continuously monitor your resources and initiate events when certain metrics or conditions are met.
- **Implement Loose Coupling**
    - **Well-Defined Interfaces** - reduce interdependencies in a system by allowing various components to interact with each other only through specific, technology agnostic interfaces, such as RESTful APIs.
    - **Service Discovery** - applications that are deployed as microservices should be discoverable and usable without prior knowledge of their network topology details. Apart from hiding complexity, this also allows infrastructure details to change at any time.
    - **Asynchronous Integration** - interacting components that do not need an immediate response and where an acknowledgement that a request has been registered will suffice, should integrate through an intermediate durable storage layer.
    - **Distributed Systems Best Practices** - build applications that handle component failure in a graceful manner.
- **Services, Not Servers**
    - **Managed Services** - provide building blocks that developers can consume to power their applications, such as databases, machine learning, analytics, queuing, search, email, notifications, and more.
    - **Serverless Architectures** - allow you to build both event-driven and synchronous services without managing server infrastructure, which can reduce the operational complexity of running applications.
- **Appropriate Use of Databases**
    - Choose the right database technology for each type of workload.
    - **Relational Databases** provide a powerful query language, flexible indexing capabilities, strong integrity controls, and the ability to combine data from multiple tables in a fast and efficient manner.
    - **NoSQL Databases** trade some of the query and transaction capabilities of relational databases for a more flexible data model that seamlessly scales horizontally. It uses a variety of data models, including graphs, key-value pairs, and JSON documents, and are widely recognized for ease of development, scalable performance, high availability, and resilience.
    - **Data Warehouses** are a specialized type of relational database, which is optimized for analysis and reporting of large amounts of data.
    - **Graph Databases** uses graph structures for queries.
        - Search Functionalities
            - Search is often confused with query. A query is a formal database query, which is addressed in formal terms to a specific data set. Search enables datasets to be queried that are not precisely structured.
            - A search service can be used to index and search both structured and free text format and can support functionality that is not available in other databases, such as customizable result ranking, faceting for filtering, synonyms, and stemming.

- **Managing Increasing Volumes of Data**
    - **Data Lake** - an architectural approach that allows you to store massive amounts of data in a central location so that it's readily available to be categorized, processed, analyzed, and consumed by diverse groups within your organization.
- **Removing Single Points of Failure**
    - **Introducing Redundancy**
        - **Standby redundancy** - when a resource fails, functionality is recovered on a secondary resource with the failover process. The failover typically requires some time before it completes, and during this period the resource remains unavailable. This is often used for stateful components such as relational databases.
        - **Active redundancy** - requests are distributed to multiple redundant compute resources. When one of them fails, the rest can simply absorb a larger share of the workload.
    - **Detect Failure** - use health checks and collect logs all the time.
    - **Durable Data Storage**
        - **Synchronous replication** - only acknowledges a transaction after it has been durably stored in both the primary storage and its replicas. It is ideal for protecting the integrity of data from the event of a failure of the primary node.
        - **Asynchronous replication** - decouples the primary node from its replicas at the expense of introducing replication lag. This means that changes on the primary node are not immediately reflected on its replicas.
        - **Quorum-based replication** - combines synchronous and asynchronous replication by defining a minimum number of nodes that must participate in a successful write operation.
    - **Automated Multi-Data Center Resilience** - utilize AWS Regions and Availability Zones (Multi-AZ Principle).
    - **Fault Isolation and Traditional Horizontal Scaling** - apply *Shuffle Sharding*.
- **Optimize for Cost**
    - **Right Sizing** - AWS offers a broad range of resource types and configurations for many use cases.
    - **Elasticity** - save money with AWS by taking advantage of the platform's elasticity.
    - **Take Advantage of the Variety of Purchasing Options** - Reserved Instances vs Spot Instances vs Other Savings Plan options
- **Caching**
    - **Application Data Caching** - store and retrieve information from fast, managed, in-memory caches.
    - **Edge Caching** - serve content by infrastructure that is closer to viewers, which lowers latency and gives high, sustained data transfer rates necessary to deliver large popular objects to end users at scale.
- **Security**
    - **Use AWS Features for Defense in Depth** - secure multiple levels of your infrastructure from network down to application and database.

- **Share Security Responsibility with AWS** - AWS handles security OF the Cloud while customers handle security IN the Cloud.
- **Reduce Privileged Access** - implement Principle of Least Privilege controls.
- **Security as Code** - firewall rules, network access controls, internal/external subnets, and operating system hardening can all be captured in a template that defines a *Golden Environment*.
- **Real-Time Auditing** - implement continuous monitoring and automation of controls on AWS to minimize exposure to security risks.

Sources:
https://docs.aws.amazon.com/whitepapers/latest/aws-overview/introduction.html
https://docs.aws.amazon.com/wellarchitected/latest/framework/welcome.html

Disaster Recovery in AWS

- **RTO or Recovery Time Objective** is the time it takes after a disruption to restore a business process to its service level.
- **RPO or Recovery Point Objective** is the acceptable amount of data loss measured in time.
- Disaster Recovery Methods
    - **Backup and Restore** - as the name implies, you take frequent backups of your most critical systems and data and store them in a secure, durable, and highly available location. Once disaster strikes, you simply restore these backups to recover data quickly and reliably. Backup and restore is usually considered the cheapest option, but also takes the longest RTO. Your RPO will depend on how frequent you take your backups.
    - **Pilot Light** - quicker recovery time than backup and restore because core pieces of the system are already running and are continually kept up to date. Examples are your secondary production databases that are configured with data mirroring or data replication to the primary. Data loss is very minimal in this scenario for the critical parts, but for the others, you have the same RTO and RPO as backup and restore.
    - **Warm Standby** - a scaled-down version of a fully functional environment that is always running. For example, you have a subset of undersized servers and databases that have the same exact configuration as your primary, and are constantly updated also. Once disaster strikes, you only have to make minimal reconfigurations to re-establish the environment back to its primary state. Warm standby is costlier than Pilot Light, but you have better RTO and RPO.
    - **Multi-Site** - run exact replicas of your infrastructure in an active-active configuration. In this scenario, all you should do in case of a disaster is to reroute traffic onto another environment. Multi-site is the most expensive option of all since you are essentially multiplying your expenses with the number of environment replicas. It does give you the best RTO and RPO however.
- A very valuable benefit of the cloud is that it enables you to set up the type of disaster recovery solution that you want, without having to worry about hardware procurement or data center facilities. AWS has a large number of regions, and an even larger set of availability zones for you to choose from. By strategically planning how you construct your disaster recovery operations, you can achieve your target RTOs and RPOs without paying too much.
- AWS also promotes their disaster recovery tool called **AWS Elastic Disaster Recovery** which they are suggesting to their customers as the preferred solution for disaster recovery workloads. Although you can adopt this tool if you wish to, it is still important for you to learn about the different DR solutions available.

Sources:
https://docs.aws.amazon.com/whitepapers/latest/disaster-recovery-workloads-on-aws/disaster-recovery-options-in-the-cloud.html

Deep Dive on AWS Services

The Solutions Architect Associate exam will test your knowledge on choosing the right service for the right situation. There are many cases wherein two services may seem applicable to a situation, but one of them fulfills the requirement better or the other options have incorrect statements. In this deep dive section, we'll be going through different scenarios that you might encounter in the SAA exam. These scenarios can be related to the behavior of a service feature, integration of different services, or how you should use a certain service. We will go as detailed as we can in this section so that you will not only know the service, but also understand what it is capable of. We will also be adding official AWS references and/or diagrams to supplement the scenarios we'll discuss. Without further ado, let's get right into it.

Amazon EC2

Amazon EC2 is a computing service that runs virtual servers in the cloud. It allows you to launch Linux or Windows virtual machines to host your applications and manage them remotely – wherever you are in the globe.

You and AWS have a shared responsibility in managing your Amazon EC2 virtual machines. AWS manages the data centers, physical facilities, the hardware components, the host operating system, and the virtualization layer that powers the entire Amazon EC2 service. On the other hand, you are responsible for your guest operating system, applying OS patches, setting up security access controls, and managing your data.

Amazon EC2 can be integrated into other AWS services to accomplish a certain task or to meet your specifications. It can be used to do a variety of functions – from running applications, hosting a self-managed database, processing batch jobs and so much more.

An Amazon EC2 virtual machine is somewhat similar to your desktop or laptop that you may be using right now. It also has a CPU, a Random Access Memory, a Network Interface, an IP address, and even a system image backup. You can also attach a Solid State Drive, or a Hard Disk Drive (HDD) to your EC2 instance for more storage; You can even connect it to a shared network file system, to allow multiple computers to access the same files.

Just like your computer, you can also integrate a lot of other AWS services with Amazon EC2. You can attach various storage, networking, and security services to an Amazon EC2 instance. There are many options available to purchase your EC2 instance, that can help you lower down your operating costs. Some AWS Services are even using Amazon EC2 as its underlying compute component. These services orchestrate or control a group of EC2 instances to perform a specific function, such as scaling or batch processing. It is also used on AWS-managed databases, containers, serverless computing engines, microservices, and many more! This is why Amazon EC2 is considered as the basic building block in AWS – it is used in almost every service!

For storage, you can use different AWS Storage services with your Amazon EC2 instance to store and process data. You can attach an Instance store for your temporary data or an Amazon EBS volume for persistent storage.

You can also mount a file system to your EC2 instances. You can connect to it to Amazon EFS or Amazon FSx. For your static media files or object data, you can store them in Amazon S3 then retrieve them back to your EC2 instance via an API or through an HTTP and FTP client.

For networking, you launch your EC2 instance on either a public or a private subnet in a Virtual Private Cloud or VPC. You can associate an Elastic IP address to your instance for it to have a static IPv4 address. An elastic network interface can also be used as a virtual network card for your EC2 instance. If you have a group of interdependent instances, you can organize them on a placement group. This placement group can be a cluster, a spread, or a partition type that enables you to minimize correlated failures, lower network latency, and achieve high throughput.

AWS also offers enhanced networking features to provide high-performance networking capabilities by using an Elastic Network Adapter or an Intel 82599 Virtual Function (VF) interface. If you have a High-Performance Computing workload or machine learning applications, you can attach an Elastic Fabric Adapter to your instance to provide a higher network throughput than your regular TCP transport.

For scaling, you can use Amazon EC2 Auto Scaling to automatically add more EC2 instances to process the increasing number of traffic in your application. Auto Scaling can also terminate the underutilized instances if the demand decrease – this can cut down your server expenses in half, or even more!

For system image back up, you can take a snapshot of your EC2 instance by creating an Amazon Machine Image, or AMI.

The AMI is just like a disk image of your Mac, Linux or Windows computer that contains custom data and system configurations that you have set. It enables you to launch a pre-configured Amazon EC2 instance that can be used for auto-scaling, migration and backups. If your EC2 instance crashed, you can easily restore your data using an AMI. It is also helpful if you want to move your server to another Available Zone, another Region or even another AWS account. You can also launch one or more EC2 instances using a single AMI.

There are more AWS services and features that you can integrate with Amazon EC2. We will cover these services in the succeeding chapters of this eBook.

Components of an EC2 Instance

You must know the components of an EC2 instance, since this is one of the core AWS services that you'll be encountering the most in the exam.

1) When creating an EC2 instance, you always start off by choosing a **base AMI or Amazon Machine Image**. An AMI contains the OS, settings, and other applications that you will use in your server. AWS has many pre-built AMIs for you to choose from, and there are also custom AMIs created by other users which are sold on the AWS Marketplace for you to use. If you have created your own AMI before, it will also be available for you to select. AMIs cannot be modified after launch.
2) After you have chosen your AMI, you select the **instance type and size** of your EC2 instance. The type and size will determine the physical properties of your instance, such as CPU, RAM, network speed, and more. There are many instance types and sizes to choose from and the selection will depend on your workload for the instance. You can freely modify your instance type even after you've launched your instance, which is commonly known as “right sizing”.
3) Once you have chosen your AMI and your hardware, you can now configure your instance settings.
    a) If you are working on the console, the first thing you'll indicate is the **number of instances** you'd like to launch with these specifications you made.
    b) You specify whether you'd like to launch **spot instances** or use another instance billing type (on-demand or reserved).
    c) You configure which **VPC and subnet** the instance should be launched in, and whether it should receive a **public IP address** or not.
    d) You choose whether to include the instance in a **placement group** or not.
    e) You indicate if the instance will be joined to one of your **domains/directories**.
    f) Next is the **IAM role** that you'd like to provide to your EC2 instance. The IAM role will provide the instance with permissions to interact with other AWS resources indicated in its permission policy.
    g) **Shutdown behavior** lets you specify if the instance should only be stopped or should be terminated once the instance goes into a stopped state. If the instance supports **hibernation**, you can also enable the hibernation feature.
    h) You can enable the **termination protection** feature to protect your instance from accidental termination.
    i) If you have **EFS file systems** that you'd like to immediately mount to your EC2 instance, you can specify them during launch.
    j) Lastly, you can specify if you have commands you'd like your EC2 instance to execute once it has launched. These commands are written in the **user data** section and submitted to the system.
4) After you have configured your instance settings, you now need to add **storage** to your EC2 instance. A volume is automatically created for you since this volume will contain the OS and other applications of your AMI. You can add more storage as needed and specify the type and size of EBS storage you'd like

to allocate. Other settings include specifying which EBS volumes are to be included for termination when the EC2 instance is terminated, and encryption.

5) When you have allocated the necessary storage for your instances, next is adding **tags** for easier identification and classification.
6) After adding in the tags, you now create or add **security groups** to your EC2 instance, which will serve as firewalls to your servers. Security groups will moderate the inbound and outbound traffic permissions of your EC2 instance. You can also add, remove, and modify your security group settings later on.
7) Lastly, the access to the EC2 instance will need to be secured using one of your **key pairs**. Make sure that you have a copy of this key pair so that you'll be able to connect to your instance when it is launched. There is no way to reassociate another key pair once you've launched the instance. You can also proceed without selecting a key pair, but then you would have no way of directly accessing your instance unless you have enabled some other login method in the AMI or via Systems Manager.
8) Once you are happy with your instance, proceed with the launch. Wait for your EC2 instance to finish preparing itself, and you should be able to connect to it if there aren't any issues.

References:
https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EC2_GetStarted.html
https://tutorialsdojo.com/amazon-elastic-compute-cloud-amazon-ec2/

Types of EC2 Instances

1. **General Purpose** — Provides a balance of compute, memory, and networking resources, and can be used for a variety of diverse workloads. Instances under the T-family have burstable performance capabilities to provide higher CPU performance when CPU is under high load, in exchange for CPU credits. Once the credits run out, your instance will not be able to burst anymore. More credits can be earned at a certain rate per hour depending on the instance size.
2. **Compute Optimized** — Ideal for compute bound applications that benefit from high performance processors. Instances belonging to this family are well suited for batch processing workloads, media transcoding, high performance web servers, high performance computing, scientific modeling, dedicated gaming servers and ad server engines, machine learning inference and other compute intensive applications.
3. **Memory Optimized** — Designed to deliver fast performance for workloads that process large data sets in memory.
4. **Accelerated Computing** — Uses hardware accelerators or co-processors to perform functions such as floating point number calculations, graphics processing, or data pattern matching more efficiently than on CPUs.
5. **Storage Optimized** — Designed for workloads that require high, sequential read and write access to very large data sets on local storage. They are optimized to deliver tens of thousands of low-latency, random I/O operations per second (IOPS) to applications.

6. **Nitro-based** — The Nitro System provides bare metal capabilities that eliminate virtualization overhead and support workloads that require full access to host hardware. When you mount EBS Provisioned IOPS volumes on Nitro-based instances, you can provision from 100 IOPS up to 64,000 IOPS per volume compared to just up to 32,000 on other instances.

References:
https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/instance-types.html
https://tutorialsdojo.com/amazon-elastic-compute-cloud-amazon-ec2/

Storage with Highest IOPS for EC2 Instance

When talking about storage and IOPS in EC2 instances, the first thing that pops into the minds of people is Amazon EBS Provisioned IOPS. Amazon EBS Provisioned IOPS volumes are the highest performing EBS volumes designed for your critical, I/O intensive applications. These volumes are ideal for both IOPS-intensive and throughput-intensive workloads that require extremely low latency. And since they are EBS volumes, your data will also persist even after shutdowns or reboots. You can create snapshots of these volumes and copy them over to your other instances, and much more.

But what if you require really high IOPS, low latency performance, and the data doesn't necessarily have to persist on the volume? If you have this requirement then the instance store volumes on specific instance types might be more preferable than EBS Provisioned IOPS volumes. EBS volumes are attached to EC2 instances virtually, so there is still some latency in there. Instance store volumes are physically attached to the EC2 instances themselves, so your instances are able to access the data much faster. Instance store volumes can come in HDD, SSD or NVME SSD, depending on the instance type you choose. Available storage space will depend on the instance type as well.

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

Instance Purchasing Options

AWS offers multiple options for you to purchase compute capacity that will best suit your needs. Aside from pricing on different instance types and instance sizes, you can also specify how you'd like to pay for the compute capacity. With EC2 instances, you have the following purchase options:

1) **On-Demand Instances** – You pay by the hour or the second depending on which instances you run for each running instance. If your instances are in a stopped state, then you do not incur instance charges. No long term commitments.
2) **Savings Plans** – Receive discounts on your EC2 costs by committing to a consistent amount of usage, in USD per hour, for a term of 1 or 3 years. You can achieve higher discount rates by paying a portion of the total bill upfront, or paying full upfront. There are two types of Savings Plans available:

a) **Compute Savings Plans** provide the most flexibility since it automatically applies your discount regardless of instance family, size, AZ, region, OS or tenancy, and also applies to Fargate and Lambda usage.
b) **EC2 Instance Savings Plans** provide the lowest prices but you are committed to usage of individual instance families in a region only. The plan reduces your cost on the selected instance family in that region regardless of AZ, size, OS, or tenancy. You can freely modify your instance sizes within the instance family in that region without losing your discount.

3) **Reserved Instances (RI)** – Similar to Saving Plans but less flexible since you are making a commitment to a consistent instance configuration, including instance type and Region, for a term of 1 or 3 years. You can also pay partial upfront or full upfront for higher discount rates. A Reserved Instance has four instance attributes that determine its price:
   a) Instance type
   b) Region
   c) Tenancy - shared (default) or single-tenant (dedicated) hardware.
   d) Platform or OS

   Reserved Instances are automatically applied to running On-Demand Instances provided that the specifications match. A benefit of Reserved Instances is that you can sell unused Standard Reserved Instances in the AWS Marketplace. There are also different types of RIs for you to choose from:
   a) Standard RIs - Provide the most significant discount rates and are best suited for steady-state usage.
   b) Convertible RIs - Provide a discount and the capability to change the attributes of the RI as long as the resulting RI is of equal or greater value.
   c) Scheduled RIs - These are available to launch within the time windows you reserve. This option allows you to match your capacity reservation to a predictable recurring schedule that only requires a fraction of a day, a week, or a month.

	Standard RI	**Convertible RI**
Applies to usage across all Availability Zones in an AWS region	Yes	Yes
Can be shared between multiple accounts within a consolidated billing family.	Yes	Yes
Change Availability Zone, instance size (for Linux OS), networking type	Yes	Yes
Change instance families, operating system, tenancy, and payment option	No	Yes
Benefit from Price Reductions	No	Yes
Can be bought/sold in Marketplace	Yes	No

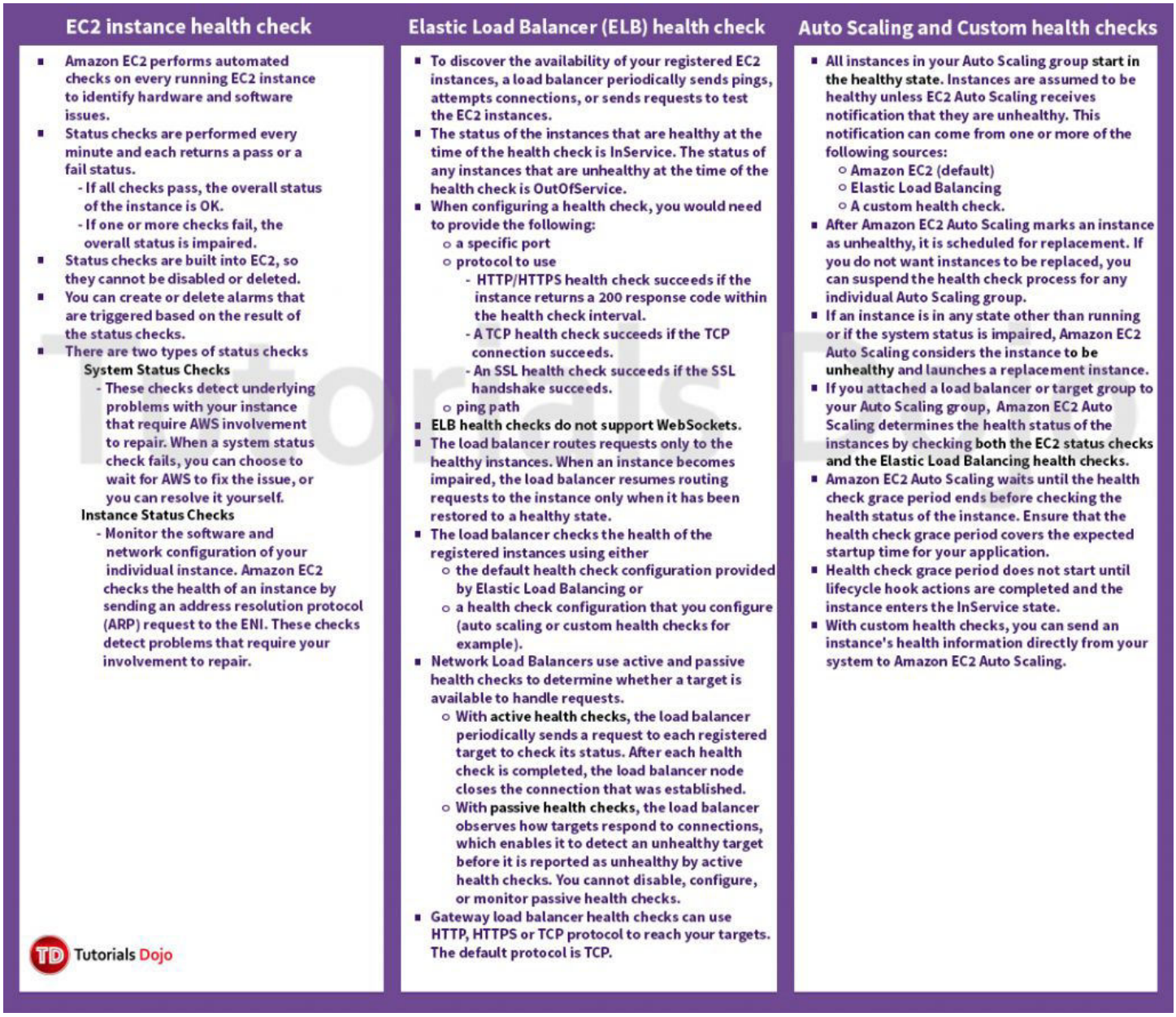
4) **Spot Instances** – Unused EC2 instances that are available for a cheap price, which can reduce your costs significantly. The hourly price for a Spot Instance is called a Spot price. The Spot price of each instance type in each Availability Zone is set by Amazon EC2, and is adjusted gradually based on the long-term supply of and demand for Spot Instances. Your Spot Instance runs whenever capacity is available and the maximum price per hour that you've placed for your request exceeds the Spot price. When the Spot price goes higher than your specified price, your Spot Instance will be stopped or terminated after a two minute warning. Use Spot Instances only when your workloads can be interrupted
5) **Dedicated Hosts** – You pay for a physical host that is fully dedicated to running your instances, and bring your existing per-socket, per-core, or per-VM software licenses to reduce costs. Support for multiple instance sizes on the same Dedicated Host is available for the following instance families: c5, m5, r5, c5n, r5n, and m5n. Dedicated Hosts also offers options for upfront payment for higher discounts.
6) **Dedicated Instances** – Pay by the hour for instances that run on single-tenant hardware. Dedicated Instances that belong to different AWS accounts are physically isolated at a hardware level. Only your compute nodes run in single-tenant hardware; EBS volumes do not.

	Dedicated Hosts	**Dedicated Instances**
Billing	Per-host billing	Per-instance billing
Visibility of sockets, cores, and host ID	Provides visibility on the number of sockets and physical cores	No visibility
Host and instance affinity	Allows you to consistently deploy your instances to the same physical server over time	Not supported
Targeted instance placement	Provides additional visibility and control over how instances are placed on a physical server	Not supported
Automatic instance recovery	Supported	Supported
Bring Your Own License (BYOL)	Supported	Not supported
Instances must run within a VPC	Yes	Yes
Can be combined with other billing options	On-demand Dedicated Hosts, Reserved Dedicated Hosts, Savings Plans	On-demand Instances, Reserved Dedicated Instances, Dedicated Spot Instances

7) **Capacity Reservations** – Allows you to reserve capacity for your EC2 instances in a specific Availability Zone for any duration. No commitment required.

References:
https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/instance-purchasing-options.html
https://aws.amazon.com/ec2/pricing/
https://tutorialsdojo.com/amazon-elastic-compute-cloud-amazon-ec2/

Comparison of Different Types of EC2 Health Checks



Reference:

https://tutorialsdojo.com/ec2-instance-health-check-vs-elb-health-check-vs-auto-scaling-and-custom-health-check/

EC2 Placement Groups

Launching EC2 instances in a placement group influences how they are placed in underlying AWS hardware. Depending on your type of workload, you can create a placement group using one of the following placement strategies:

- **Cluster** – your instances are placed close together inside an Availability Zone. A cluster placement group can span peered VPCs that belong in the same AWS Region. This strategy enables workloads to achieve low-latency, high network throughput network performance.
- **Partition** – spreads your instances across logical partitions, called partitions, such that groups of instances in one partition do not share the underlying hardware with groups of instances in different partitions. A partition placement group can have partitions in multiple Availability Zones in the same Region, with a maximum of seven partitions per AZ. This strategy reduces the likelihood of correlated hardware failures for your application.
- **Spread** – strictly places each of your instances across distinct underlying hardware racks to reduce correlated failures. Each rack has its own network and power source. A spread placement group can have partitions in multiple Availability Zones in the same Region, with a maximum of seven running EC2 instances per AZ per group.

If you try to add more instances to your placement group after you create it, or if you try to launch more than one instance type in the placement group, you might get an insufficient capacity error. If you stop an instance in a placement group and then start it again, it still runs in the placement group. However, the start fails if there isn't enough capacity for the instance. To remedy the capacity issue, simply retry the launch until you succeed.

Some limitations you need to remember:

- You can't merge placement groups.
- An instance cannot span multiple placement groups.
- You cannot launch Dedicated Hosts in placement groups.
- A cluster placement group can't span multiple Availability Zones.

References:
https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/placement-groups.html
https://tutorialsdojo.com/amazon-elastic-compute-cloud-amazon-ec2/

Security Groups And Network Access Control Lists

Security groups and network ACLs are your main lines of defense in protecting your VPC network. These services act as firewalls for your VPCs and control inbound and outbound traffic based on the rules you set. Although both of them are used for VPC network security, they serve two different functions and operate in a different manner.

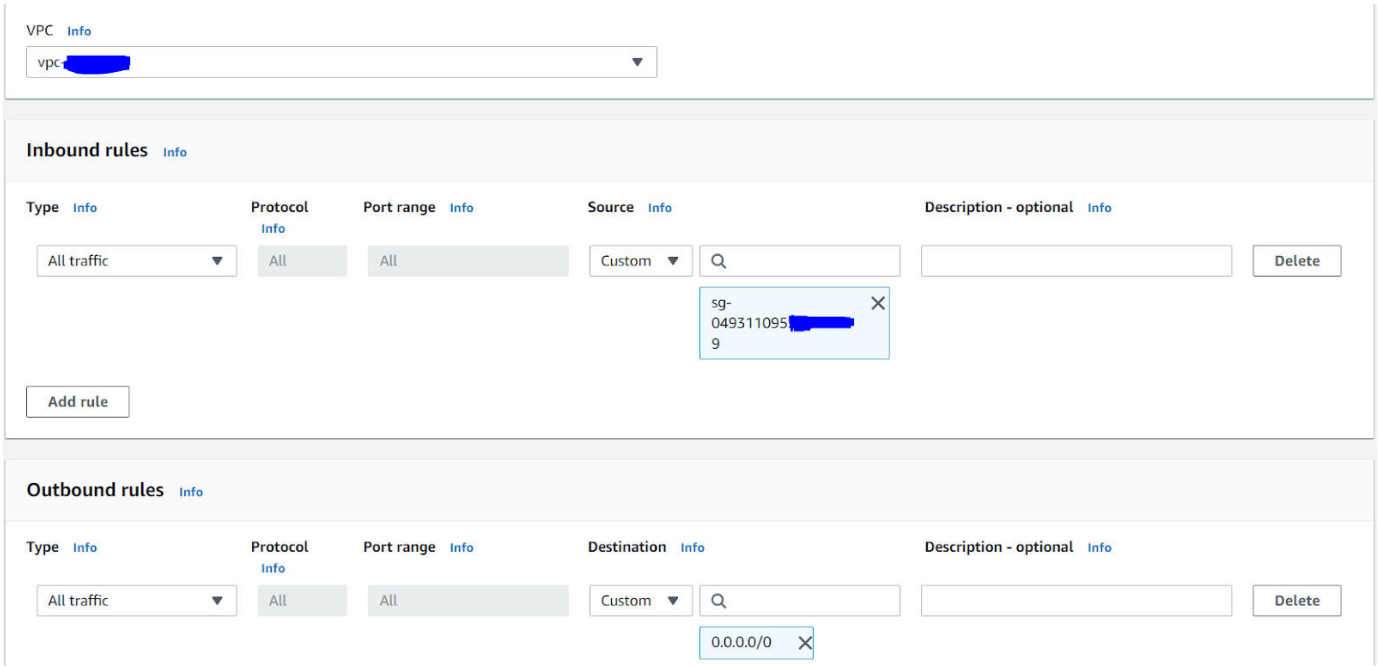
Security groups operate on the instance layer. They serve as virtual firewalls that control inbound and outbound traffic to your VPC resources. Not all AWS services support security groups, but the general idea is that if the service involves servers or EC2 instances then it should also support security groups. Examples of these services are:

1. Amazon EC2
2. AWS Elastic Beanstalk
3. Amazon Elastic Load Balancing
4. Amazon RDS
5. Amazon EFS
6. Amazon EMR
7. Amazon Redshift
8. Amazon Elasticache

To control the flow of traffic to your VPC resources, you define rules in your security group which specify the types of traffic that are allowed. A security group rule is composed of traffic type (SSH, RDP, etc), internet protocol (tcp or udp), port range, origin of the traffic for inbound rules or destination of the traffic for outbound rules, and an optional description for the rule. Origins and destinations can be defined as definite IP addresses, IP address ranges, or a security group ID. If you reference a security group ID in your rule then all resources that are associated with the security group ID are counted in the rule. This saves you the trouble of entering their IP addresses one by one.

You can only create rules that allow traffic to pass through. Traffic parameters that do not match any of your security group rules are automatically denied. By default, newly created security groups do not allow any inbound traffic while allowing all types of outbound traffic to pass through. Security groups are also stateful, meaning if you send a request from your instance, the response traffic for that request is allowed to flow in regardless of inbound rules. Responses to allowed inbound traffic are allowed to flow out, regardless of outbound rules. One thing to remember is, when you are adding rules to allow communication between two VPC instances, you should enter the private IP address of those instances and not their public IP or Elastic IP address.

Security groups are associated with network interfaces, and not the instances themselves. When you change the security groups of an instance, you are changing the security groups associated with its network interface. By default, when you create a network interface, it's associated with the default security group for the VPC, unless you specify a different security group. Network interfaces and security groups are bound to the VPC they are launched in, so you cannot use them for other VPCs. However, security groups belonging to a different VPC can be referenced as the origin and destination of a security group rule of peered VPCs.

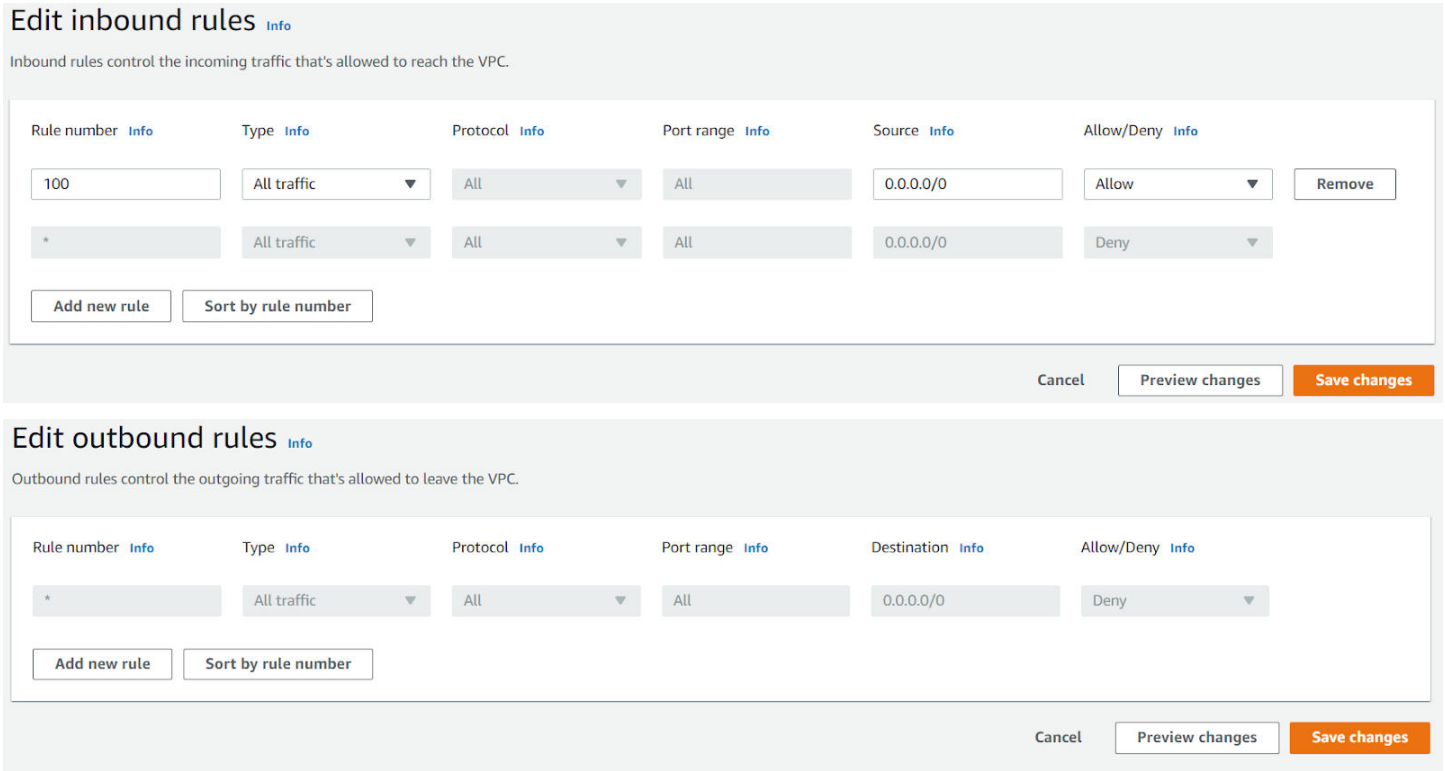


Network ACLs operate on the subnet layer, which means they protect your whole subnet rather than individual instances. Similar to security groups, traffic is managed through the use of rules. A network ACL rule consists of a rule number, traffic type, protocol, port range, source of the traffic for inbound rules or destination of the traffic for outbound rules, and an allow or deny setting.

In network ACL, rules are evaluated starting with the lowest numbered rule. As soon as a rule matches traffic, it's applied regardless of any higher-numbered rule that might contradict it. And unlike security groups, you can create allow rules and deny permissions in NACL for both inbound and outbound rules. Perhaps you want to allow public users to have HTTP access to your subnet, except for a few IP addresses that you found to be malicious. You can create an inbound HTTP allow rule that allows 0.0.0.0/0 and create another inbound HTTP deny rule that blocks these specific IPs. If no rule matches a traffic request or response then it is automatically denied. Network ACLs are also stateless, so sources and destinations need to be allowed on both inbound and outbound for them to freely communicate with the resources in your subnet.

Every VPC comes with a default network ACL, which allows all inbound and outbound traffic. You can create your own custom network ACL and associate it with a subnet. By default, each custom network ACL denies all inbound and outbound traffic until you add rules. Note that every subnet 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. A network ACL can be associated with multiple subnets. However, a subnet can be associated with only one network ACL at a time.

One last thing to note is, for subnets that handle public network connections, you might encounter some issues if you do not add an allow rule for your ephemeral ports. The range varies depending on the client's operating system. A NAT gateway uses ports 1024-65535 for example.



References:

https://docs.aws.amazon.com/vpc/latest/userguide/VPC_SecurityGroups.html
https://docs.aws.amazon.com/vpc/latest/userguide/vpc-network-acls.html
https://tutorialsdojo.com/security-group-vs-nacl/

Amazon EC2 Auto Scaling

The Amazon EC2 Auto Scaling service helps you ensure that you have the right number of EC2 instances available to handle the load for your web applications. This is a type of horizontal scaling, where you scale out or scale in your applications by dynamically launching or terminating EC2 instances.

Auto Scaling has three major components:

- The Auto Scaling Group
- The configuration templates
- Scaling Options

The Auto Scaling service works by organizing your EC2 instances into groups. An Auto Scaling group is treated as a logical unit for scaling and management purposes. A group must have a minimum, maximum, and desired number of EC2 instances.

A configuration template can either be a launch template, or a launch configuration. This acts as a template for your Auto Scaling Group, containing the AMI ID, the instance type, the key pair, the security groups, block device mapping, et cetera. All of this information is used to launch and configure the new EC2 instances. It is also recommended to use a launch template, rather than a launch configuration, as the latter only offers limited features.

The Scaling Option allows you to choose the suitable scaling behavior of your Auto Scaling Group. You can configure an Auto Scaling Group to scale based on certain conditions, such as the CPU Utilization of your EC2 instances or based on a particular date and time. The scaling option can be set to be dynamic, predictive, or scheduled.

You should also be aware that the process of launching or terminating your instances is not done in an instant. There's a certain lead time when you are launching brand new EC2 instances since AWS has to fetch the AMI, do the required configuration, run the user data that you included and install the custom applications that you specify. All of this must be completed before the instance can accept live incoming requests. This duration is also called "instance warm-up".

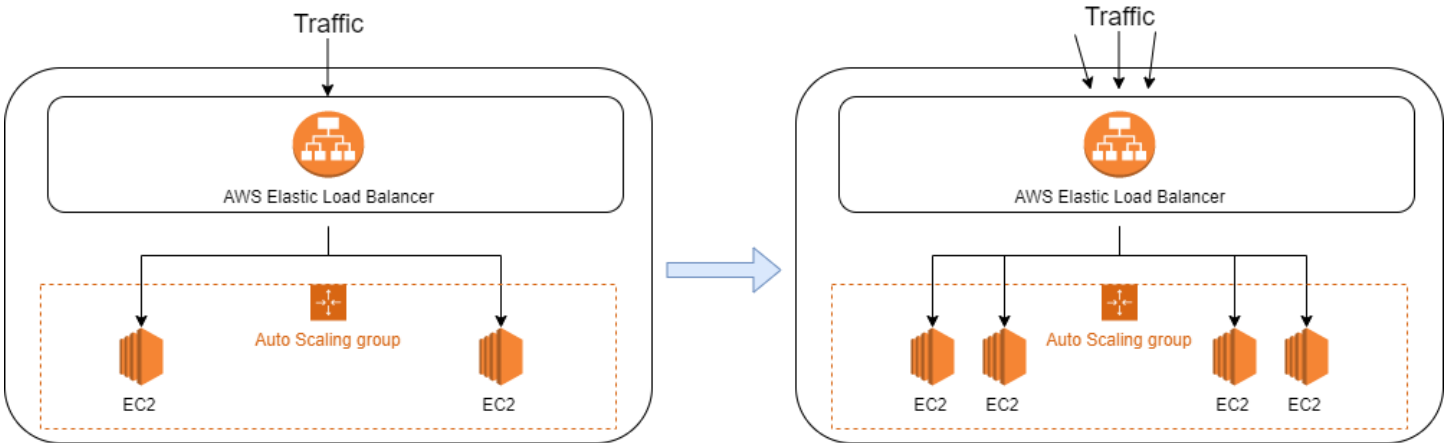
An Auto Scaling group also has a setting called "cool down", which is technically the interval between two scaling actions. This is the number of seconds that must pass before another scaling activity can be executed. This prevents conflicts in your Auto Scaling Group, where one activity is adding an instance while the other one is terminating your resources. You can also set up a termination policy to control which EC2 instances will be terminated first when a scale-in event occurs.

This feature also allows you to do certain actions while the scale-in or scale-out action is being done. The Amazon EC2 Auto Scaling service provides the ability to add lifecycle hooks to your Auto Scaling groups, which allows you to specify the amount of time to complete the lifecycle action before the EC2 instance transitions to the next state. A lifecycle hook enables you to suspend or resume your scaling process to do a variety of tasks, such as sending application logs, doing a system health check, or executing a custom shell script, before an instance gets launched or terminated.

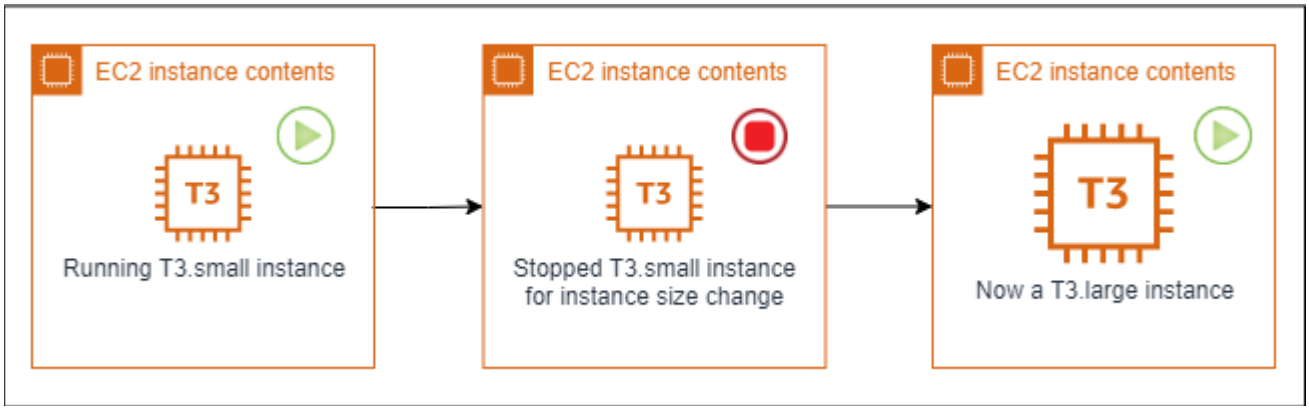
Horizontal Scaling and Vertical Scaling

When you have insufficient capacity for a workload, let's say for example serving a website, there are two ways to scale your resources to accommodate the increasing demand: scale horizontally or scale vertically.

When scaling horizontally, you are adding more servers to the system. More servers mean that workload is distributed to a greater number of workers, which thereby reduces the burden on each server. When you scale horizontally, you need a service such as EC2 auto scaling to manage the number of servers running at a time. You also need an Elastic Load Balancer to intercept and distribute the total incoming requests to your fleet of auto scaling servers. Horizontal scaling is a great way for stateless servers, such as public web servers, to meet varying levels of workloads.



Compared to scaling horizontally, scaling vertically refers to increasing or decreasing the resources of a single server, instead of adding new servers to the system. Vertical scaling is suited for resources that are stateful or have operations difficult to manage in a distributed manner, such as write queries to databases and IOPS sizing in storage volumes. For example, if your EC2 instance is performing slowly, then you can scale up its instance size to obtain more compute and memory capacity. Or when your EBS volumes are not hitting the required IOPS, you can increase their size or IOPS capacity by modifying the EBS volume. Note that for some services such as EC2 and RDS, the instance needs to be stopped before modifying the instance size.



Components of an AWS EC2 Auto Scaling Group

An EC2 Auto Scaling Group has two parts to it: a launch configuration or template that will define your auto scaling instances, and the auto scaling service that performs scaling and monitoring actions.

Creating a launch configuration is similar to launching an EC2 instance. Each launch configuration has a name that uniquely identifies it from your other launch configurations. You provide the AMI that it will use to launch your instances. You also get to choose the instance type and size for your auto scaling instances. You can request spot instances or just use the standard on-demand instances. You can also include an instance profile that will provide your auto scaling instances with permissions to interact with your other services.

If you need Cloudwatch detailed monitoring, you can enable the option for a cost. Aside from that, you can include user data which will be executed every time an auto scaling instance is launched. You can also choose whether to assign public IP addresses to your instances or not. Lastly, you select which security groups you'd like to apply to your auto scaling instances, and configure EBS storage volumes for each of them. You also specify the key pair to be used to encrypt access.

A launch template is similar to a launch configuration, except that you can have multiple versions of a template. Also, with launch templates, you can create Auto Scaling Groups with multiple instance types and purchase options.

Instance purchase options Info

Use the launch template to create a uniform configuration among all of the instances in the group. Or define options to accommodate a wide variety of requirements, such as launching Spot and On-Demand Instances.

- Adhere to launch template
  The launch template determines the purchase option (On-Demand or Spot) and instance type.
- (selected) Combine purchase options and instance types
  Specify how much On-Demand and Spot capacity to launch and multiple instance types (optional). This choice is most helpful for optimizing the scale and cost for a fleet of instances.

Instances distribution

On-Demand base capacity - *optional*
Specify how much On-Demand capacity the Auto Scaling group should have for its base portion. The maximum group size will be increased (but not decreased) to this value.

0 On-Demand Instances

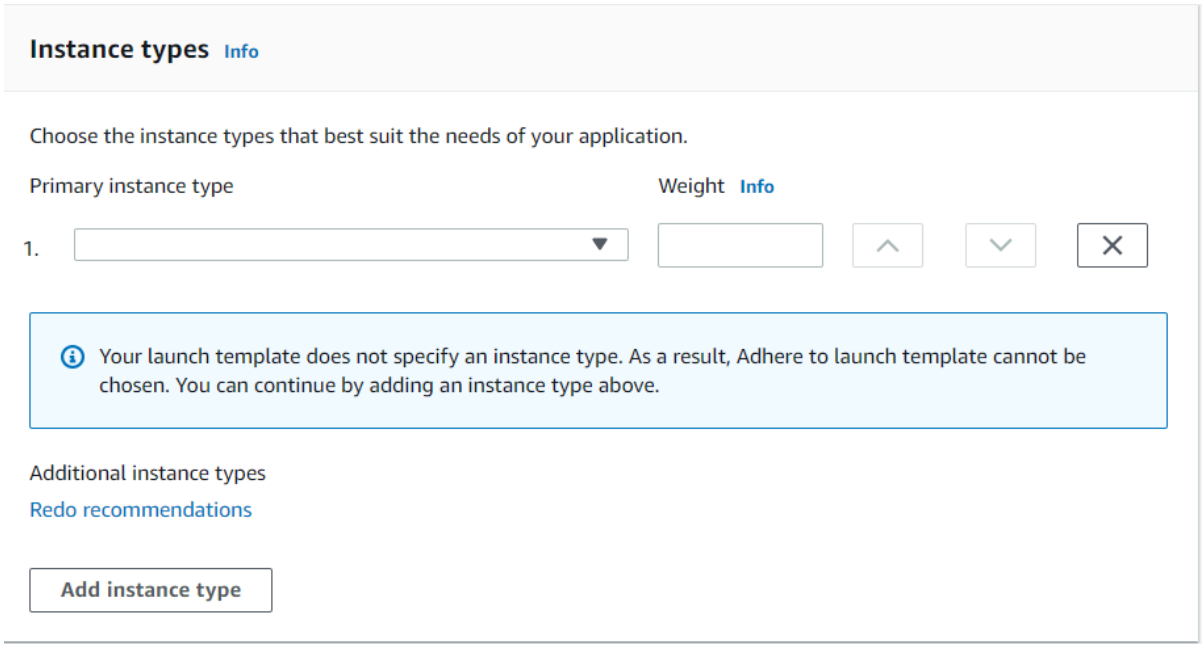
On-Demand percentage above base
Define the percentage split of On-Demand Instances and Spot Instances for your additional capacity beyond the base portion.

70 % On-Demand

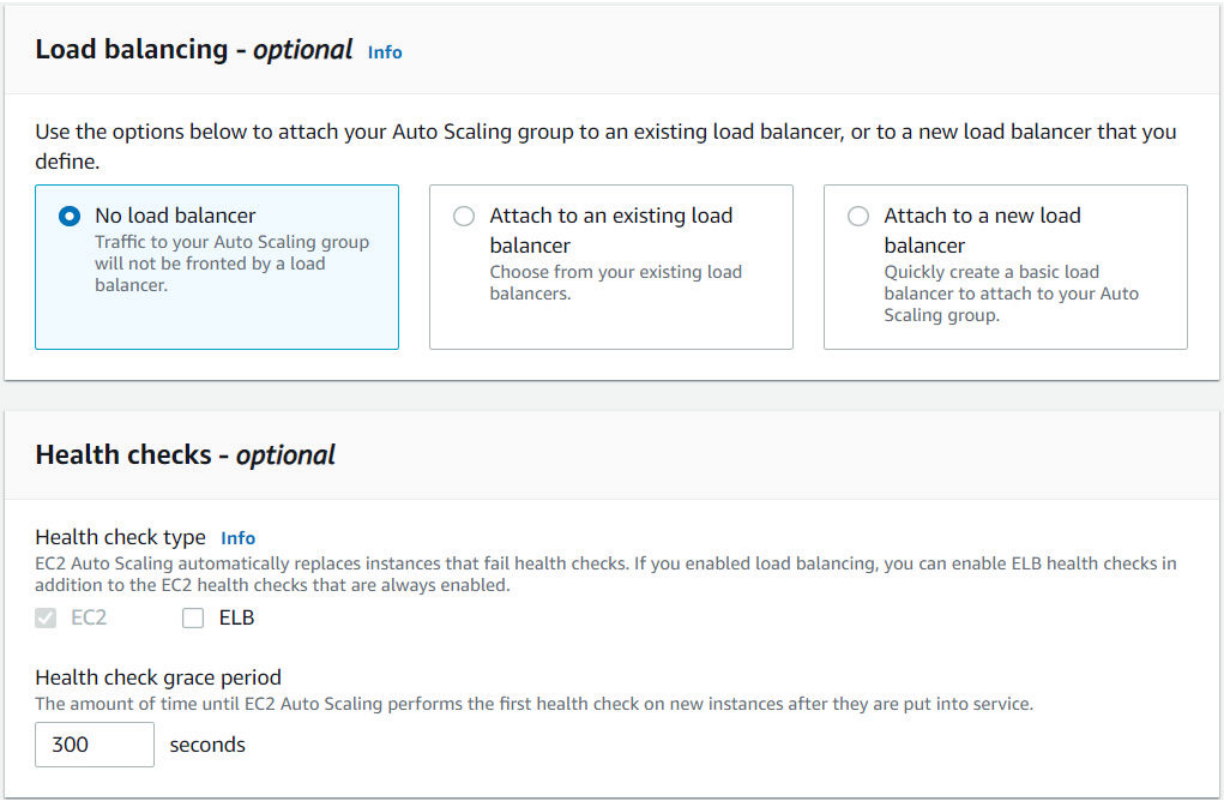
30 % Spot

Spot allocation strategy per Availability Zone

- (selected) Capacity optimized (recommended)
  Launch Spot Instances optimally based on the available Spot capacity.
- Lowest price
  Launch Spot Instances from the lowest priced instance pools.



Once you have created your launch configuration or launch template, you can proceed with creating your auto scaling group. To start off, select the launch configuration/template you'd like to use. Next, you define the VPC and subnets in which the auto scaling group will launch your instances in. You can use multiple Availability Zones and let EC2 Auto Scaling balance your instances across the zones. You can optionally associate a load balancer to the auto scaling group, and the service will handle attaching and detaching instances from the load balancer as it scales. Note that when you do associate a load balancer, you should use the load balancer's health check for instance health monitoring so that when an instance is deemed unhealthy by the load balancer's health check, the load balancer will initiate a scaling event to replace the faulty instance.



Next, you define the size of the auto scaling group — the minimum, desired and the maximum number of instances that your auto scaling group should manage. Specifying a minimum size ensures that the number of running instances do not fall below this count at any time, and the maximum size prevents your auto scaling group from exploding in number. Desired size just tells the auto scaling group to launch this number of instances after you create it. Since the purpose of an auto scaling group *is to auto scale*, you can add cloudwatch monitoring rules that will trigger scaling events once a scaling metric passes a certain threshold. Lastly, you can optionally configure Amazon SNS notifications whenever a scaling event occurs, and add tags to your auto scaling group.

References:
https://docs.aws.amazon.com/autoscaling/ec2/userguide/what-is-amazon-ec2-auto-scaling.html
https://tutorialsdojo.com/aws-auto-scaling/

Types of EC2 Auto Scaling Policies

Amazon's EC2 Auto Scaling provides an effective way to ensure that your infrastructure is able to dynamically respond to changing user demands. For example, to accommodate a sudden traffic increase on your web application, you can set your Auto Scaling group to automatically add more instances. And when traffic is low, have it automatically reduce the number of instances. This is a cost-effective solution since it only provisions

EC2 instances when you need them. EC2 Auto Scaling provides you with several dynamic scaling policies to control the scale-in and scale-out events.

In this article, we'll discuss the differences between a simple scaling policy, a step scaling policy and a target tracking policy. And we'll show you how to create an Auto Scaling group with step scaling policy applied.

Simple Scaling

Simple scaling relies on a metric as a basis for scaling. For example, you can set a CloudWatch alarm to have a CPU Utilization threshold of 80%, and then set the scaling policy to add 20% more capacity to your Auto Scaling group by launching new instances. Accordingly, you can also set a CloudWatch alarm to have a CPU utilization threshold of 30%. When the threshold is met, the Auto Scaling group will remove 20% of its capacity by terminating EC2 instances.

When EC2 Auto Scaling was first introduced, this was the only scaling policy supported. It does not provide any fine-grained control to scaling in and scaling out.

Target Tracking

Target tracking policy lets you specify a scaling metric and metric value that your auto scaling group should maintain at all times. Let's say for example your scaling metric is the average CPU utilization of your EC2 auto scaling instances, and that their average should always be 80%. When CloudWatch detects that the average CPU utilization is beyond 80%, it will trigger your target tracking policy to scale out the auto scaling group to meet this target utilization. Once everything is settled and the average CPU utilization has gone below 80%, another scale in action will kick in and reduce the number of auto scaling instances in your auto scaling group. With target tracking policies, your auto scaling group will always be running in a capacity that is defined by your scaling metric and metric value.

A limitation though – this type of policy assumes that it should scale out your Auto Scaling group when the specified metric is above the target value. You cannot use a target tracking scaling policy to scale out your Auto Scaling group when the specified metric is below the target value. Furthermore, the Auto Scaling group scales out proportionally to the metric as fast as it can, but scales in more gradually. Lastly, you can use AWS predefined metrics for your target tracking policy, or you can use other available CloudWatch metrics (native and custom). Predefined metrics include the following:

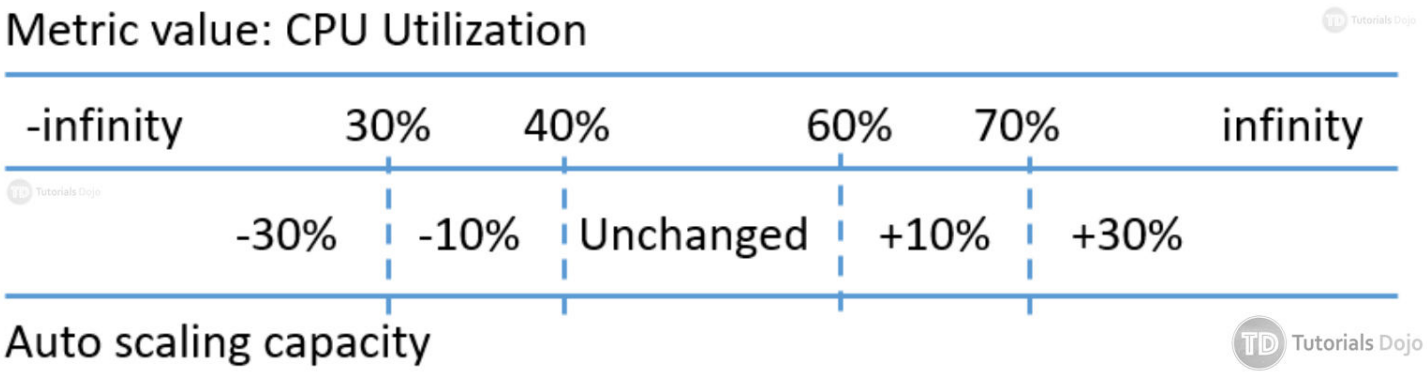
- **ASGAverageCPUUtilization** – Average CPU utilization of the Auto Scaling group.
- **ASGAverageNetworkIn** – Average number of bytes received on all network interfaces by the Auto Scaling group.
- **ASGAverageNetworkOut** – Average number of bytes sent out on all network interfaces by the Auto Scaling group.
- **ALBRequestCountPerTarget** – If the auto scaling group is associated with an ALB target group, this is the number of requests completed per target in the target group.

Step Scaling

Step Scaling further improves the features of simple scaling. Step scaling applies “step adjustments” which means you can set multiple actions to vary the scaling depending on the size of the alarm breach.

When a scaling event happens on simple scaling, the policy must wait for the health checks to complete and the cooldown to expire before responding to an additional alarm. This causes a delay in increasing capacity especially when there is a sudden surge of traffic on your application. With step scaling, the policy can continue to respond to additional alarms even in the middle of the scaling event.

Here is an example that shows how step scaling works:



In this example, the Auto Scaling group maintains its size when the CPU utilization is between 40% and 60%. When the CPU utilization is greater than or equal to 60% but less than 70%, the Auto Scaling group increases its capacity by an additional 10%. When the utilization is greater than 70%, another step in scaling is done and the capacity is increased by an additional 30%. On the other hand, when the overall CPU utilization is less than or equal to 40% but greater than 30%, the Auto Scaling group decreases the capacity by 10%. And if utilization further dips below 30%, the Auto Scaling group removes 30% of the current capacity.

This effectively provides multiple steps in scaling policies that can be used to fine-tune your Auto Scaling group response to dynamically changing workload.

Creating a Step Scaling Policy for an Auto Scaling Group

Based on the step scaling policy described above, the following guide will walk you through the process of applying this policy when creating your Auto Scaling group.

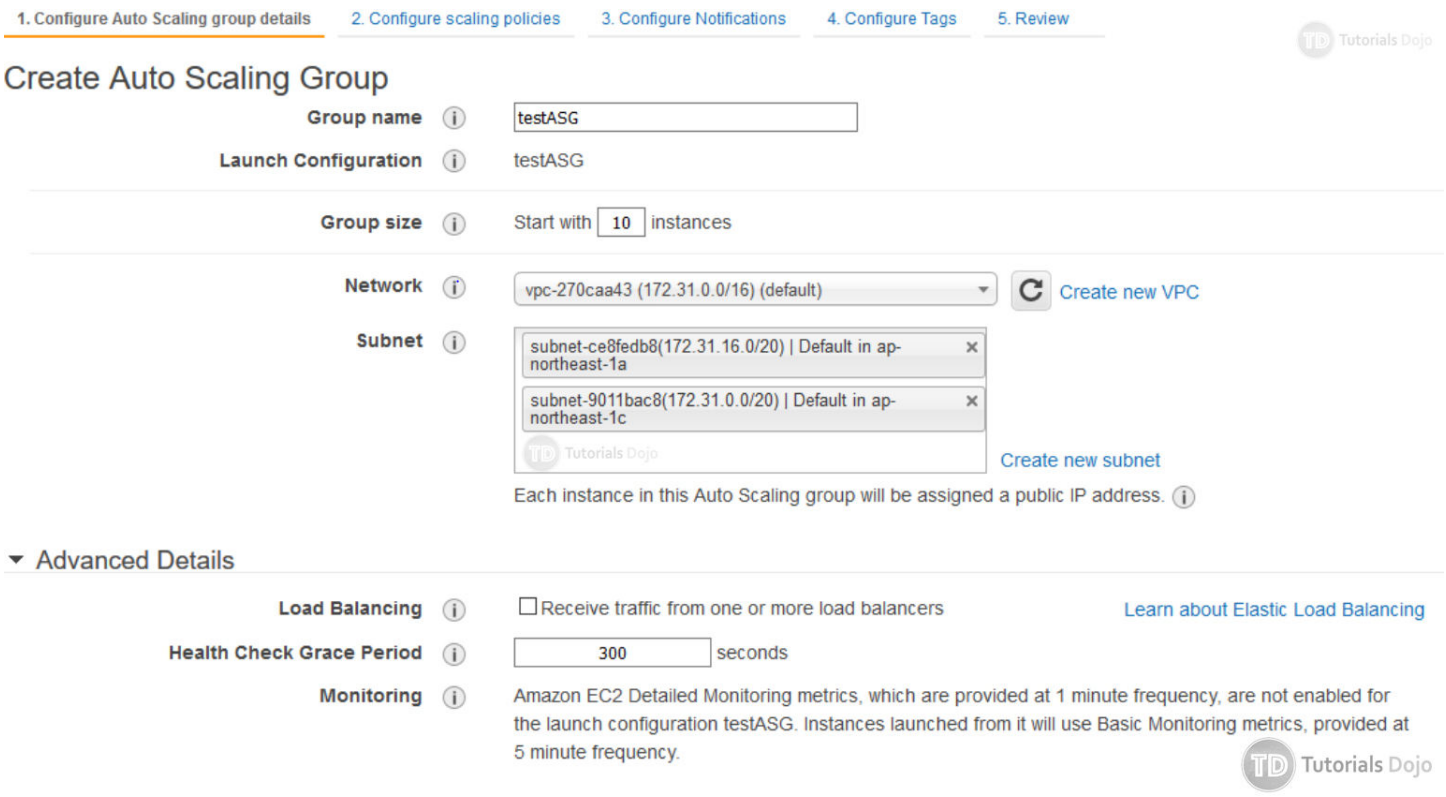
1. First, create your Launch Configuration for your EC2 instances. Check this guide if you haven't created one yet.

2. Go to **EC2** > **Auto Scaling Groups** > **Create Auto Scaling group**

3. Select your **Launch Configuration** and click **Next Step**.

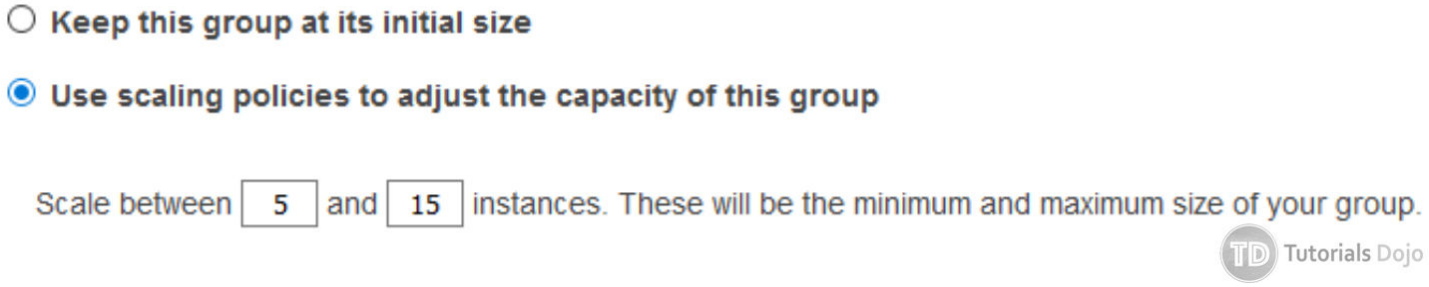
4. Configure details for your Auto Scaling group.

a. **Group name** – descriptive name for this ASG.
b. **Group size** – the initial size of your ASG. Let's set this to 10 for this example.
c. **Network** – the VPC to use for your ASG.
d. **Subnet** – the subnets in the VPC on where to place the EC2 instances. It's recommended to select subnets in multiple availability zones to improve the fault tolerance of your ASG.
e. **Advanced Details** – in this section, you can check the **Load Balancing** option to select which load balancer to use for your ASG. (We won't configure a load balancer for this example). You can also set the **Health Check Grace Period** in this section. This is the length of time that Auto Scaling waits before checking the instance's health status. We'll leave the default to 300 seconds but you can adjust this if you know your EC2 instances need more or less than 5 minutes before they become healthy.

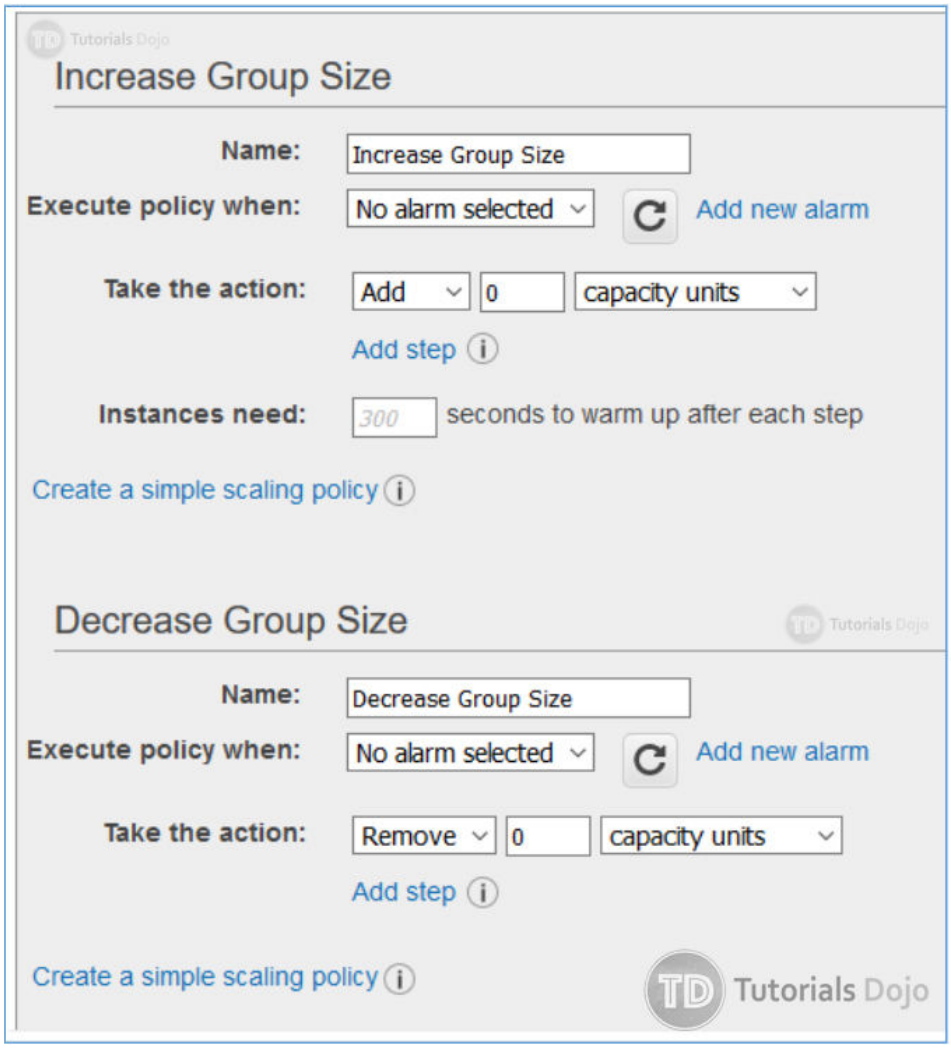


5. Click **Next: Configure scaling policies** to proceed.

6. Here, we'll configure the step scaling policy. Select the "**Use scaling policies to adjust the capacity of this group**" option and this will show an additional section for defining scaling policy. For this example, let's set 5 and 15 as the minimum and maximum size for this Auto Scaling group.

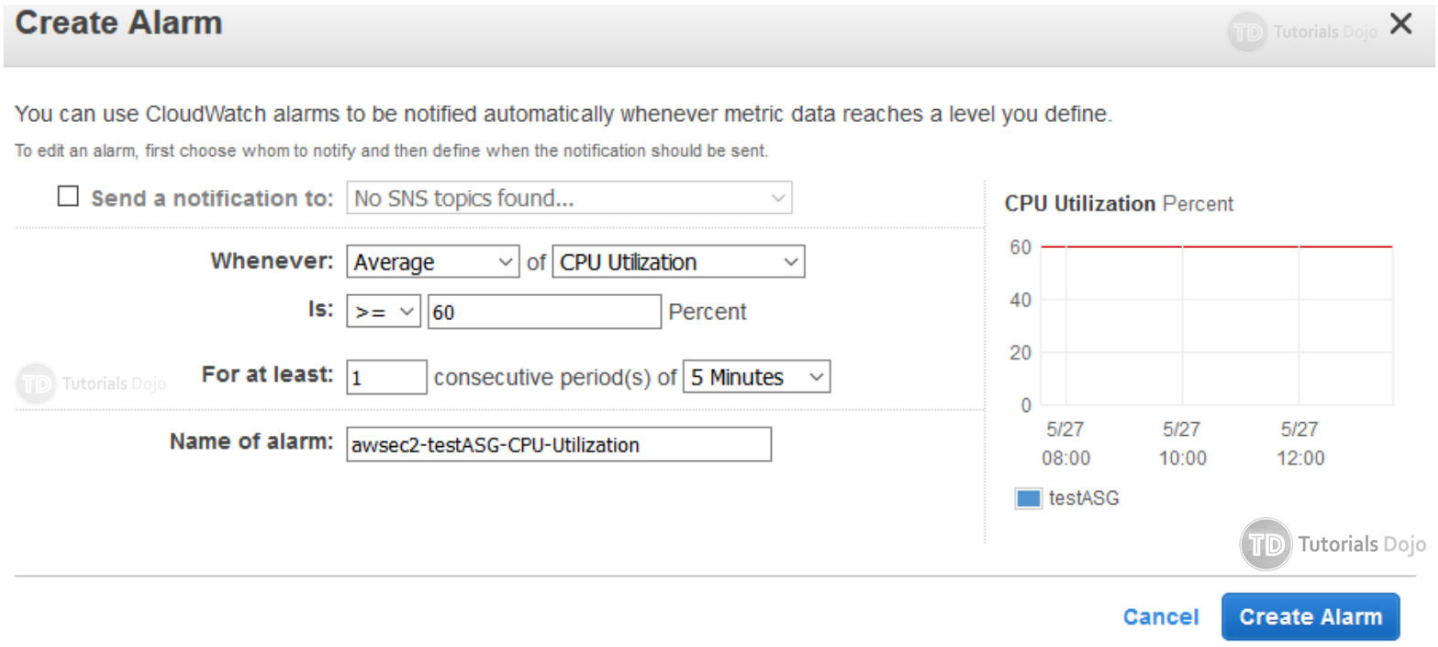


7. In the Scale Group Size section, you will be able to set the scaling policy for the group. But this is only for simple scaling so you have to click the **"Scale the Auto Scaling group using step or simple scaling policies"** link to show more advanced options for step scaling. You should see the **Increase Group Size** and **Decrease Group Size** section after clicking it.

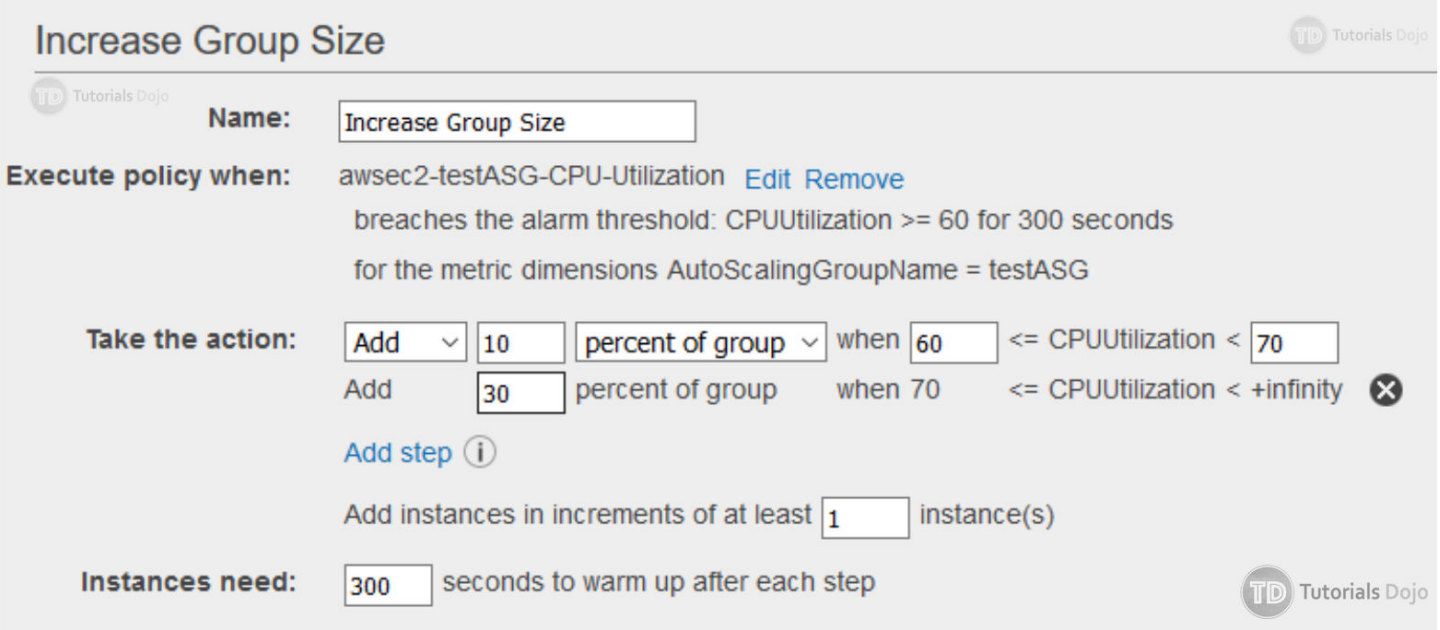


8. Now, we can set the step scaling policy for scaling out.

a. Set a name for your "**Increase Group Size**" policy. Click "**Add a new alarm**" to add a CloudWatch rule on when to execute the policy.
b. b. On the **Create Alarm** box, you can set an SNS notification. (We won't add it for this example).
c. c. Create a rule for whenever the **Average CPU Utilization** is greater than or equal to 60 percent for at least 1 consecutive period of 5 minutes. Set a name for your alarm. Click **Create Alarm**.



d. For the "**Take the action**" setting, we'll **Add 10 percent** of the group when CPU Utilization is greater than or equal to **60 and less than 70 percent**.
e. Click "Add Step" to add another action, we'll **Add 30 percent** of the group when CPU Utilization is **greater than or equal to 70 percent**.



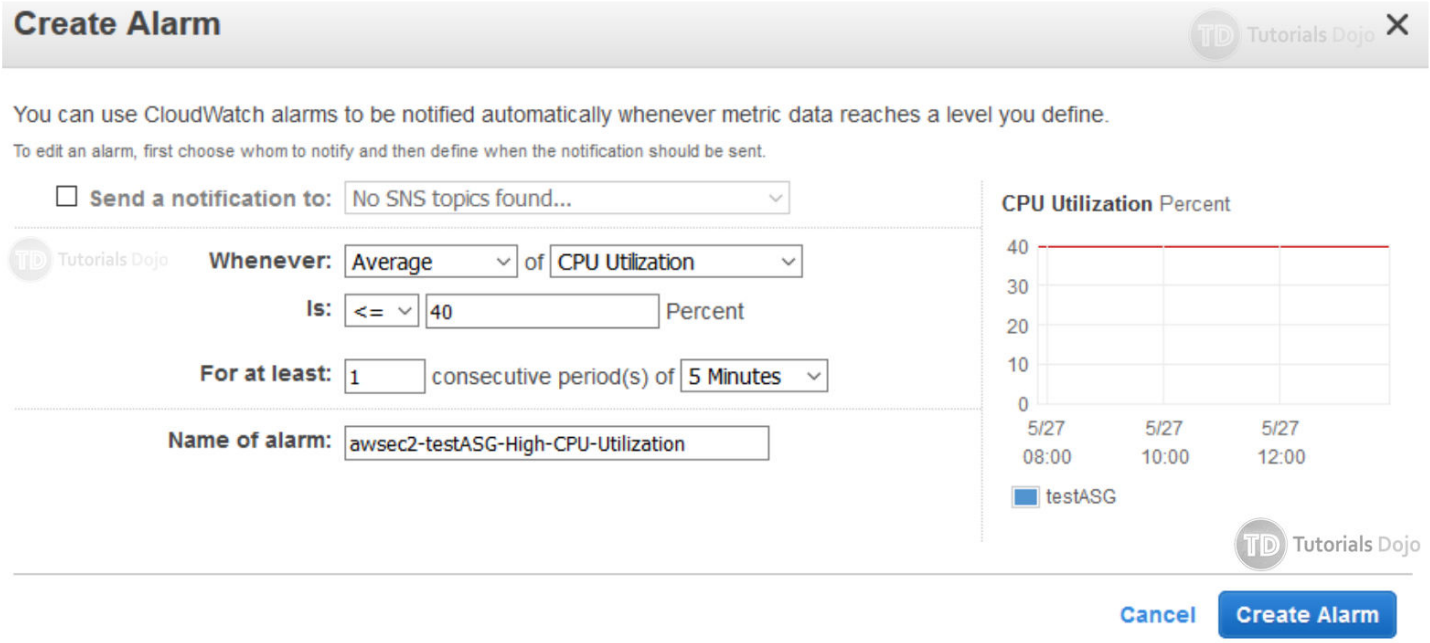
f. Set 1 for "**Add instances in increments of at least**". This will ensure that at least 1 instance is added when the threshold is reached.
g. Set instances need **300 seconds to warm up** after each step.

Instance warmup – this specifies the timeout before the instance's own metric can be added to the group. Until the warmup time expires, the instance metric (CPU utilization in this case) is not counted toward the aggregated metric of the whole Auto Scaling group.

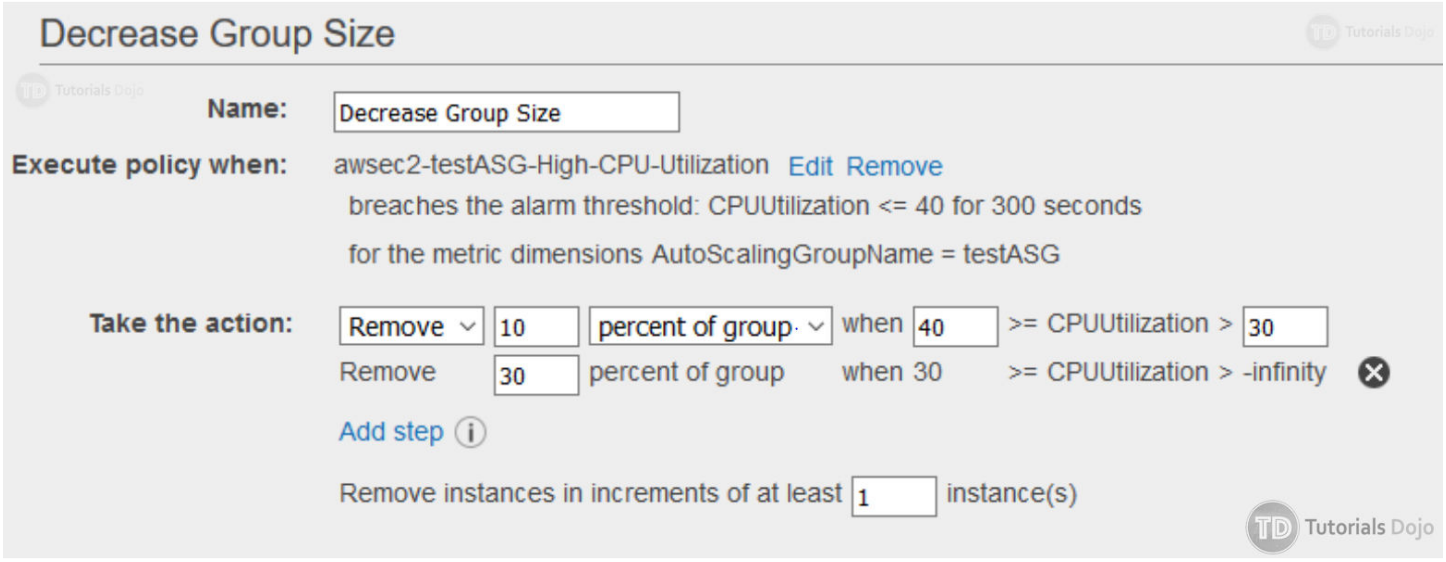
While scaling in, instances that are terminating are considered as part of the current capacity of the group. Therefore, it won't remove more instances from the Auto Scaling group than necessary.

9. Next, we can set the step scaling policy for the scaling in.

a. Set a name for your "**Decrease Group Size**" policy. Click "**Add a new alarm**" to add a CloudWatch rule on when to execute the policy.
b. On the **Create Alarm** box, you can set an SNS notification. (We won't add it for this example).
c. Create a rule for whenever the **Average CPU Utilization** is less than or equal to 40 percent for at least 1 consecutive period of 5 minutes. Set a name for your alarm. Click **Create Alarm**.

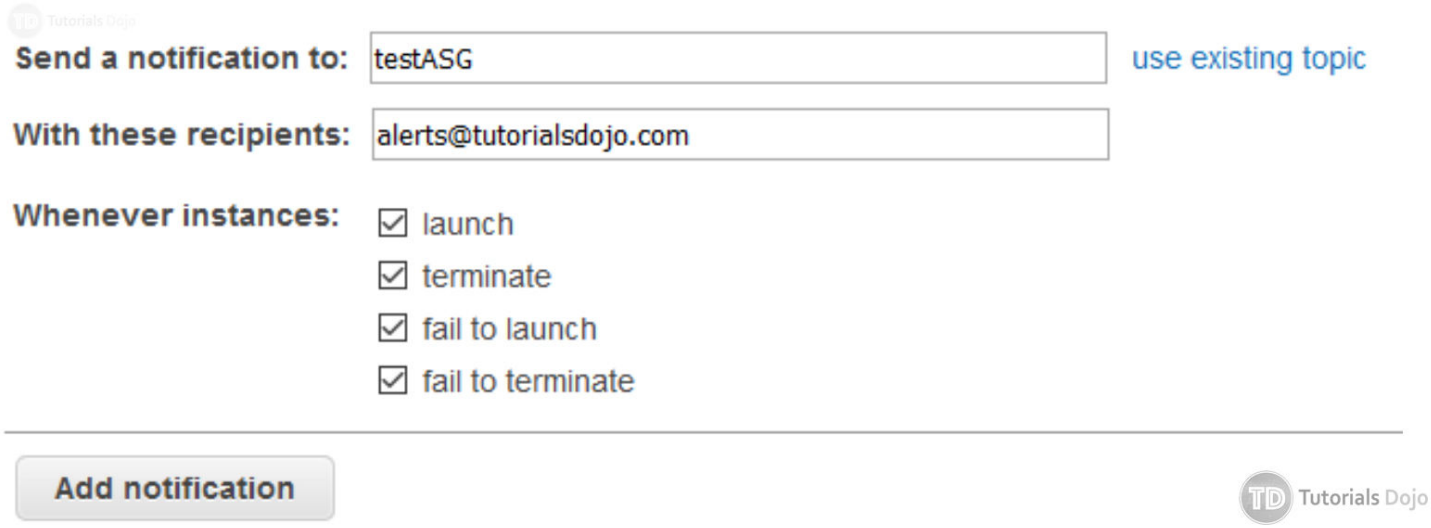


d. For the "**Take the action**" setting, we'll **remove 10 percent** of the group when CPU Utilization is less than or equal to **40 and greater than 30**.
e. Click "Add Step" to add another action, we'll **remove 30 percent** of the group when CPU Utilization is **less than or equal to 30 percent**.



f. Set 1 for "**Remove instances in increments of at least**". This will ensure that at least 1 instance is removed when the threshold is reached.

10. Click **Next: Configure Notifications** to proceed. On this part, you can click "**Add notification**" so that you can receive an email whenever a specific event occurs. Here's an example:



11.Click **Next: Configure Tags**. Create tags for instances in your Auto Scaling group.

12. Click **Review** to get to the review page.

13. After reviewing the details, click **Create Auto Scaling group**.

Your Auto Scaling group with step scaling policies should now be created. Remember, the initial desired size is 10, with a minimum of 5 and a maximum of 15.

The scale-out rule will have a step scaling policy, a 10% increase if CPU utilization is 60 – 70%, and will add 30% more instances if utilization is more than 70%.

The scale-in rule will have a step scaling policy, a 10% decrease if CPU utilization is 30 – 40%, and will remove 30% more instances if the utilization is less than 30%.

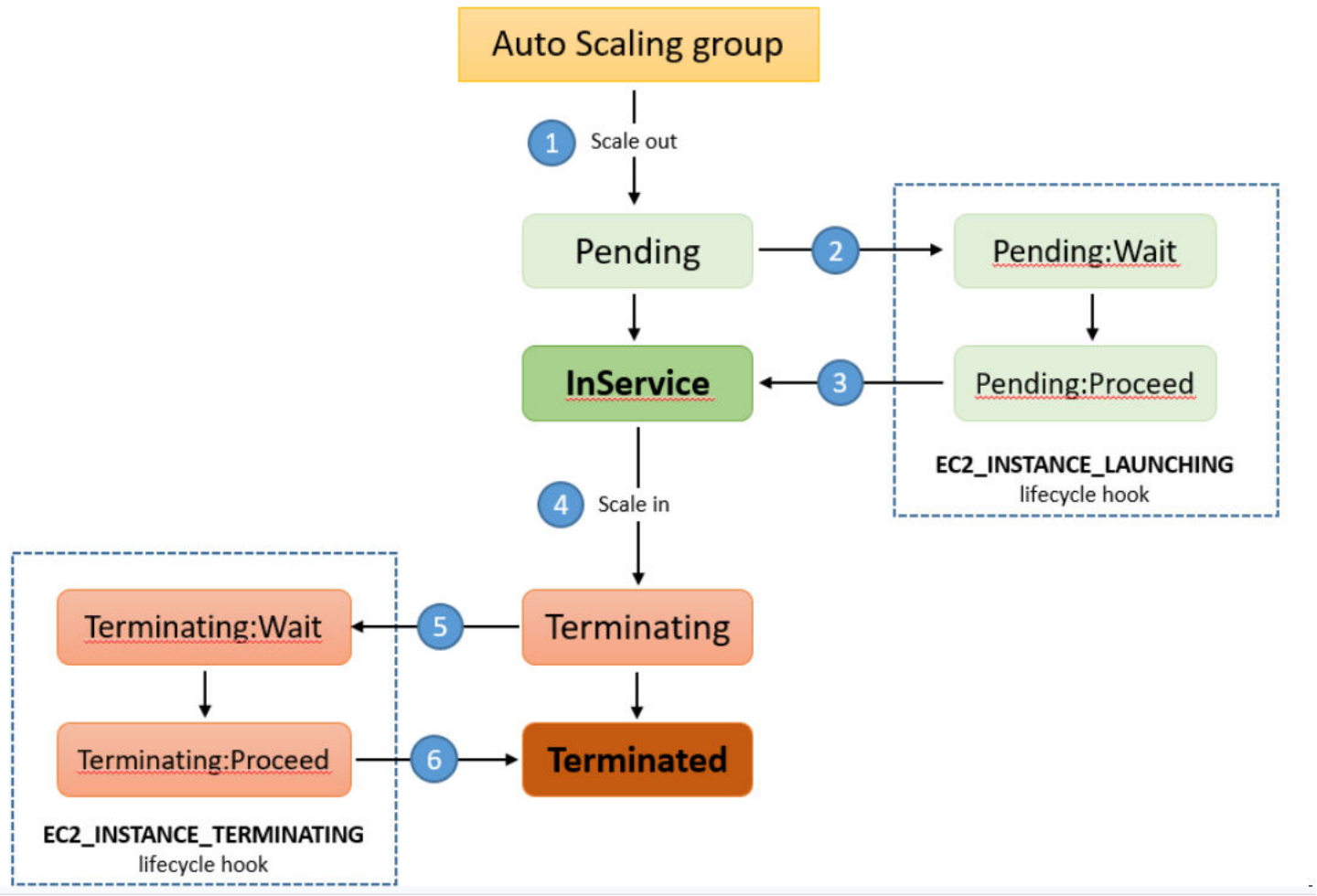
References:
https://docs.aws.amazon.com/autoscaling/ec2/userguide/as-scaling-simple-step.html
https://docs.aws.amazon.com/autoscaling/ec2/userguide/Cooldown.html
https://docs.aws.amazon.com/autoscaling/ec2/userguide/GettingStartedTutorial.html

EC2 Auto Scaling Lifecycle Hooks

As your Auto Scaling group scale-out or scale-in your EC2 instances, you may want to perform custom actions before they start accepting traffic or before they get terminated. Auto Scaling Lifecycle Hooks allow you to perform custom actions during these stages.

For example, during the scale-out event of your ASG, you want to make sure that new EC2 instances download the latest code base from the repository and that your EC2 user data has completed before it starts accepting traffic. This way, the new instances will be fully ready and will quickly pass the load balancer health check when they are added as targets. Another example is this – during the scale-in event of your ASG, suppose your instances upload data logs to S3 every minute. You may want to pause the instance termination for a certain amount of time to allow the EC2 to upload all data logs before it gets completely terminated.

Lifecycle Hooks give you greater control of your EC2 during the launch and terminate events. The following diagram shows the transitions between the EC2 instance states with lifecycle hooks.



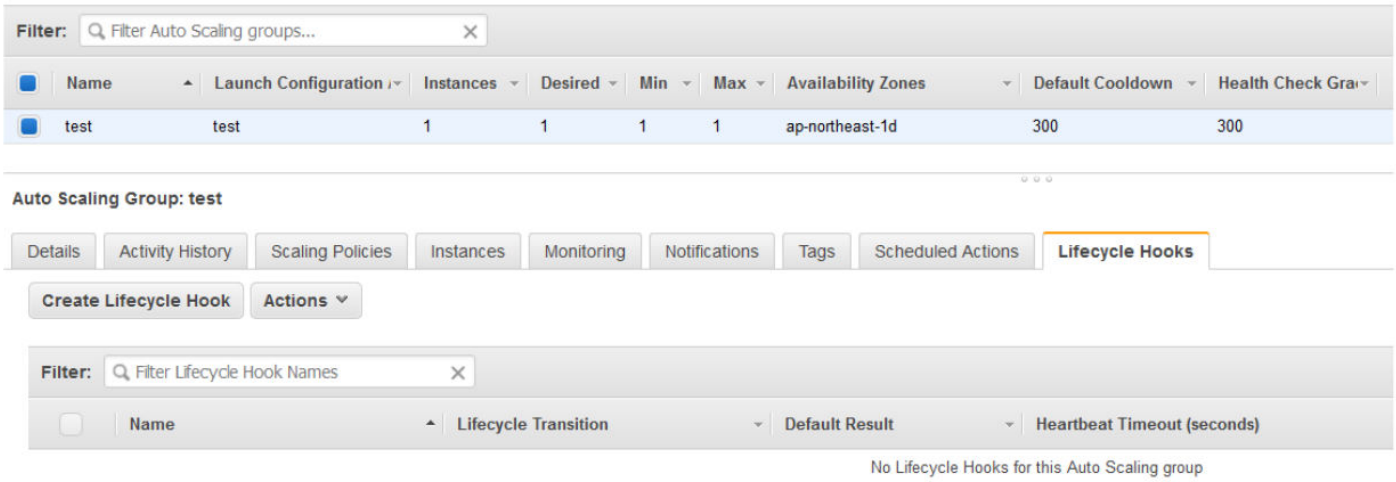
1. The Auto Scaling group responds to a scale-out event and provisions a new EC2 instance.
2. The lifecycle hook puts the new instance on *Pending:Wait* state. The instance stays in this paused state until you continue with the "*CompleteLifecycleAction*" operation or the default wait time of 3600 seconds is finished. For example, you can create a script that runs during the creation of the instance to download and install the needed packages for your application. Then the script can call the "*CompleteLifecycleAction*" operation to move the instance to the *InService* state. Or you can just wait for your configured timeout and the instance will be moved to the *InService* state automatically.
3. The instance is put to *InService* state. If you configured a load balancer for this Auto Scaling group, the instance will be added as targets and the load balancer will begin the health check. After passing the health checks, the instance will receive traffic.
4. The Auto Scaling group responds to a scale-in event and begins terminating an instance.
5. The instance is taken out of the load balancer target. The lifecycle hook puts the instance on *Terminating:Wait* state. For example, you can set a timeout of 2 minutes on this section to allow your instance to upload any data files inside it to S3. After the timeout, the instance is moved to the next state.
6. Auto scaling group completes the termination of the instance.

During the paused state (either launch or terminate), you can do more than just run custom scripts or wait for timeouts. CloudWatch Events (Amazon EventBridge) receives the scaling action and you can define a CloudWatch Events (Amazon EventBridge) Target to invoke a Lambda function that can perform a pre-configured task. You can also configure a notification target for the lifecycle hook so that you will receive a message when the scaling event occurs.

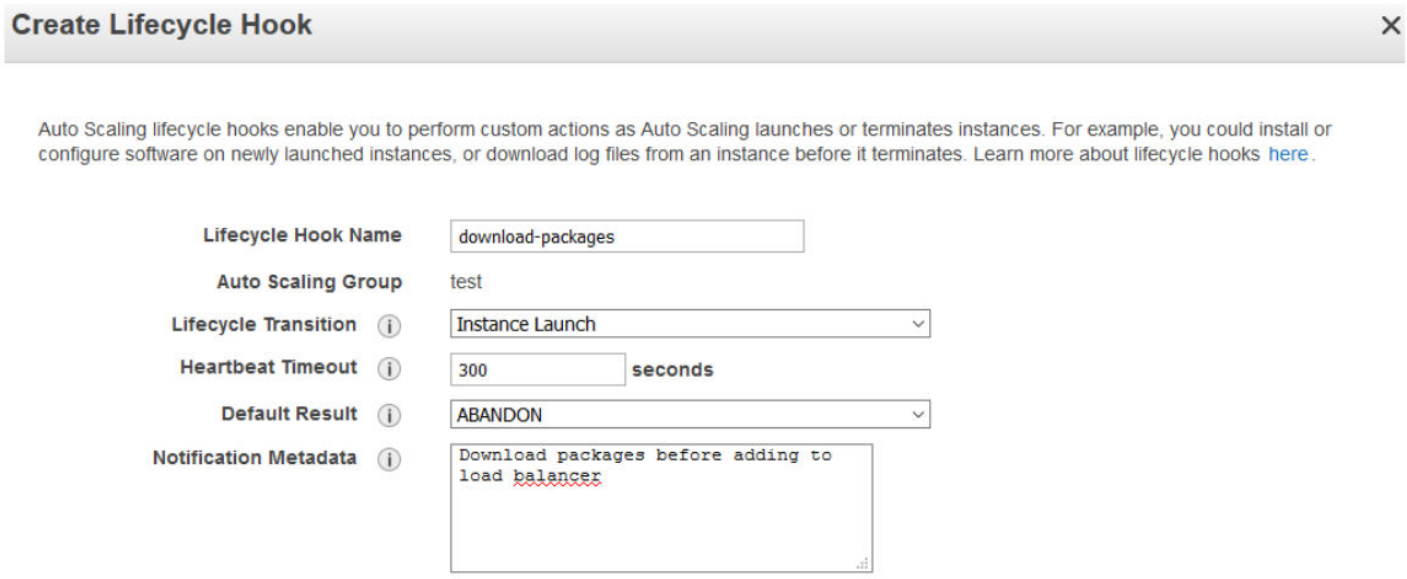
Configure Lifecycle Hooks on your Auto Scaling Groups

The following steps will show you how to configure lifecycle hooks for your Auto Scaling group.

1. On the Amazon EC2 Console, under Auto Scaling, choose Auto Scaling Group.
2. Select your Auto Scaling group.
3. Click the Lifecycle hooks tab then click the Create Lifecycle Hook button.



4. In the Create Lifecycle Hook box, do the following:



- Lifecycle Hook Name – then name for this lifecycle hook
- Lifecycle Transition – choose whether this lifecycle hook is for “Instance Launch” or “Instance Terminate” event. If you need a lifecycle hook for both events, you need to add another lifecycle hook.
- Heartbeat timeout – the amount of time (in seconds) for the instance to remain in the wait state. The range is between 30 seconds to 7200 seconds.
- Default Result – the action the Auto Scaling group takes when the lifecycle hook timeout elapses or if an unexpected error occurs.

        - If you choose CONTINUE and the instance is launching, the Auto Scaling group assumes that the actions are successful and proceeds to put the instance to InService state. If you choose CONTINUE and the instance is terminating, the Auto Scaling group will proceed with other lifecycle hooks before termination.
        - Choosing ABANDON on either state will terminate the instance immediately.
    - Notification Metadata – additional information to include in messages to the notification target.
5. Click Create to apply the lifecycle hook for this Auto Scaling group.

References:
https://docs.aws.amazon.com/autoscaling/ec2/userguide/lifecycle-hooks.html
https://docs.aws.amazon.com/cli/latest/reference/autoscaling/put-lifecycle-hook.html

Configuring Notifications for Lifecycle Hooks

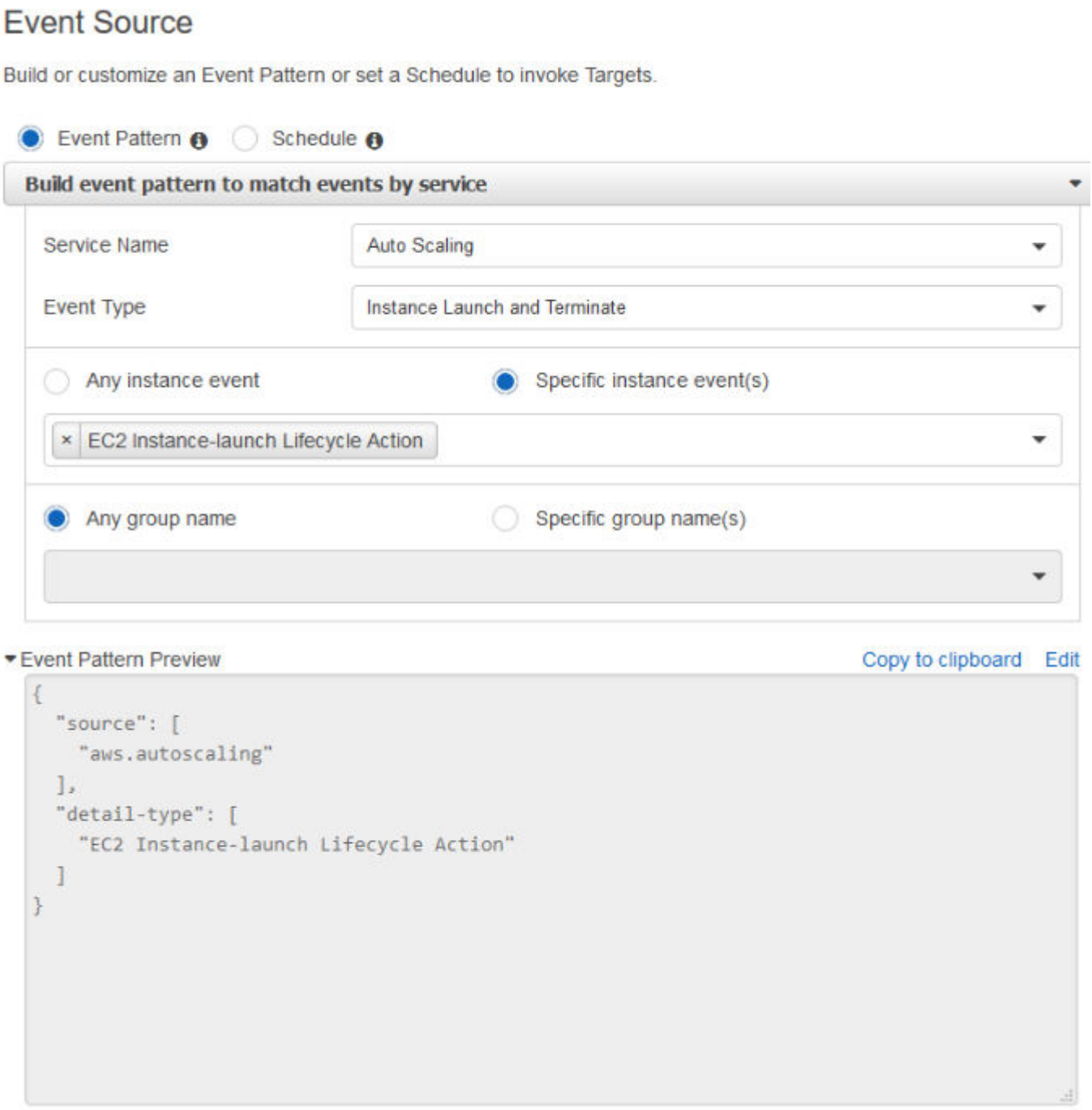
When a lifecycle hook occurs on an Auto Scaling group, it sends event logs to AWS CloudWatch Events (Amazon EventBridge), which in turn can be used to set up a rule and target to invoke a Lambda function.

The following steps assume that you have configured your Auto Scaling Lifecycle hook on the AWS Console.

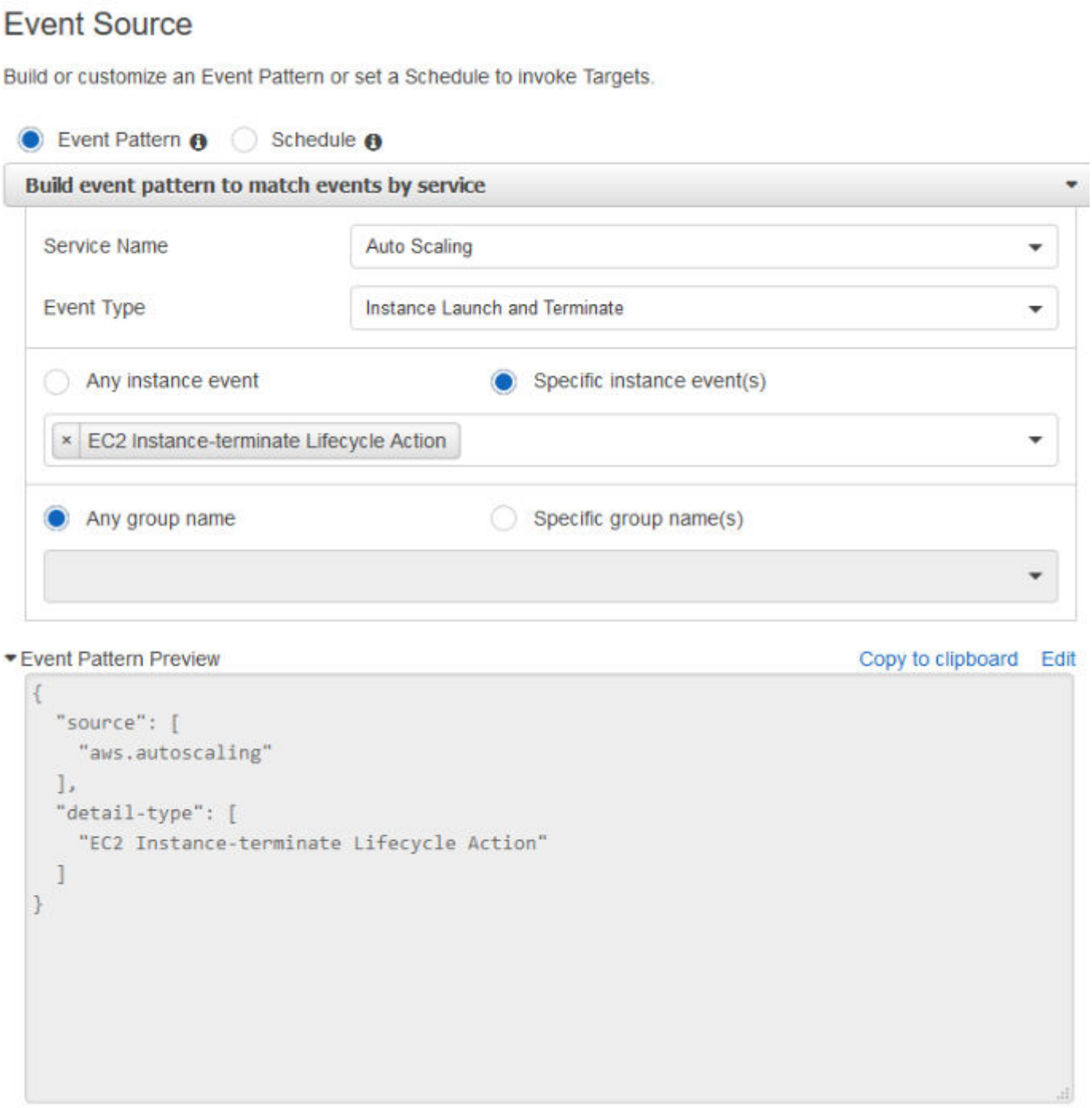
Route Notifications to Lambda using CloudWatch Events (Amazon EventBridge)

1. Create your Lambda function and take note of the ARN. To create your Lambda function, see this link.
2. Go to AWS **CloudWatch** > **Events** > **Rules** and click **Create rule**.
3. Choose the following options:
    a. **Event Pattern** – since you want this rule to filter AWS events
    b. **Service Name: Auto Scaling** – to filter from Auto Scaling service
    c. **Event type: Instance Launch and Terminate** – since the lifecycle hook happens on scale-out and scale-in event
    d. ***Specific Instance events*** *– Select this and you can choose whether you want this rule to trigger for the "Instance-launch Lifecycle Action" or the "Instance-terminate Lifecycle Action"*

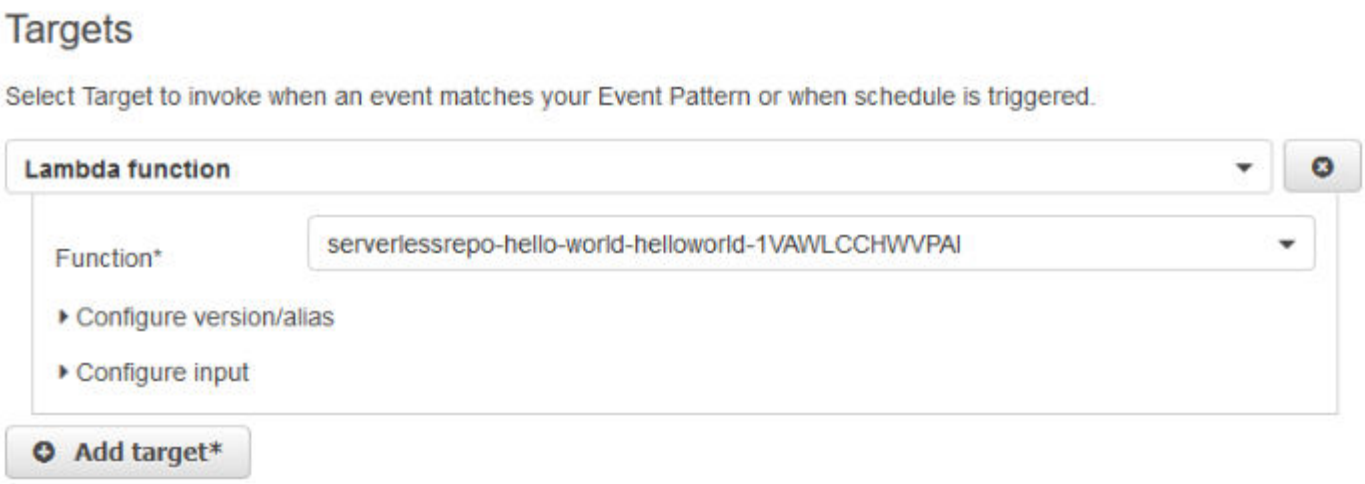
Your rule should be like the screenshot below for the *"Instance-launch Lifecycle Action"*.



Your rule should be like the screenshot below for the *"Instance-terminate Lifecycle Action"*.



4. Click on **"Add target"** on the right side of the page to add a target for this Rule.
5. Select **"Lambda function"** as target and select your Lambda function on the **"Function"** field. You can also add other targets here if you need to. Here's a screenshot for reference:



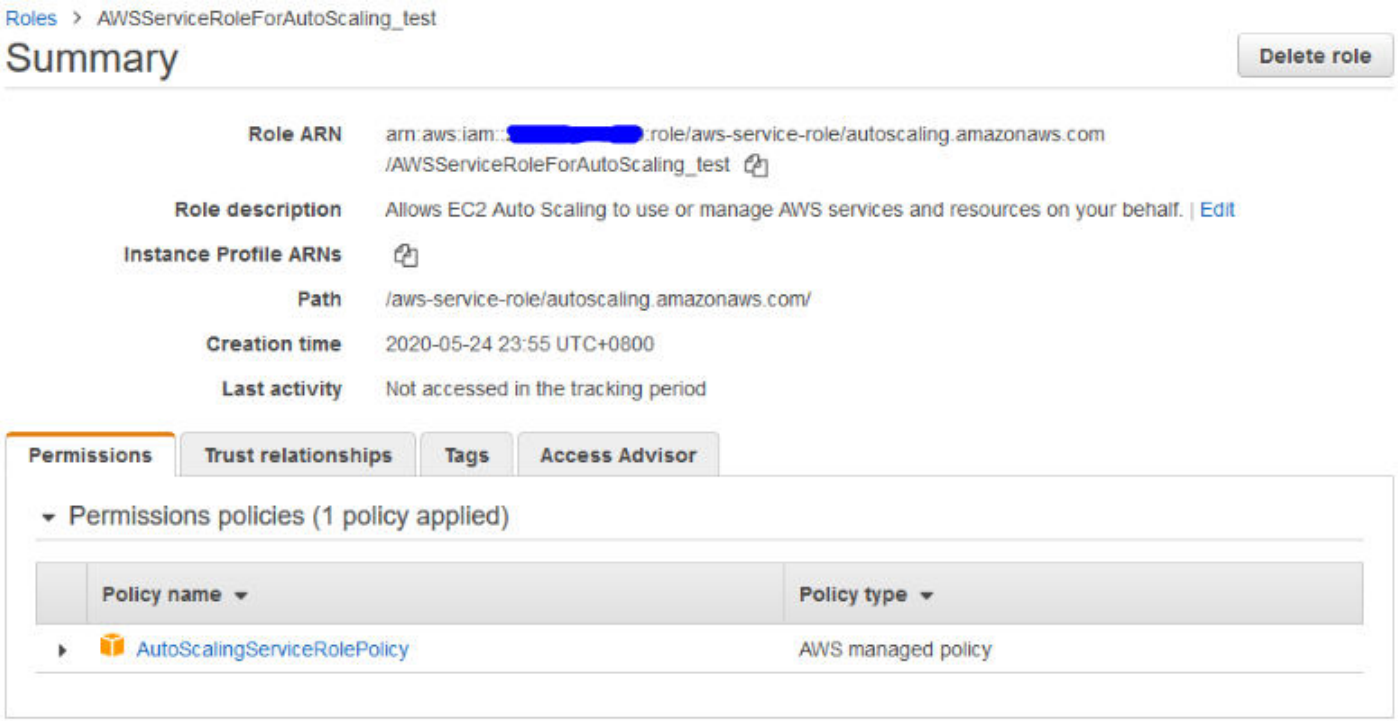
6. Click “Configure details” to proceed to the next step.
7. Add a name to your rule and add a description. You want to make sure the “State Enabled” is checked. Click **Create rule** to finally create your CloudWatch Events (Amazon EventBridge) rule.

That's it, the CloudWatch permission to trigger the Lambda function is automatically taken care of. Now, when the Auto Scaling group scales-out or scales-in with a lifecycle hook, the Lambda function is triggered.

Receive Notification using Amazon SNS

To receive lifecycle hook notifications with Amazon SNS, you can use the AWS CLI to add a lifecycle hook. The key point here is that you need an SNS topic and an IAM role to allow publishing to that topic.

1. Create your SNS topic. Let's assume the SNS topic ARN is arn:aws:sns:ap-northeast-1:1234457689123:test-topic. Make sure that your email is subscribed to this topic.
2. Create an IAM Role that you will associate to the lifecycle hook.
    a. Go to **IAM** > **Role** > **Create role**
    b. Select **AWS Service** under the **Select type of trusted entity**.
    c. Click **EC2 Auto Scaling** from the list under the **Choose a use case section**.
    d. Choose **EC2 Auto Scaling** on the **Select your use case** section.
    e. Click **Next: Permissions** to add permissions to this role. The **AutoScalingServiceRolePolicy** should already be added.
    f. Click **Next: Tags** to add tags to this role.
    g. Click **Next: Review** to add a name to this role
    h. Click **Create role**.



3. Get the ARN of this role. Let's assume the ARN is

   arn:aws:iam::123456789123:role/aws-service role/autoscaling.amazonaws.com/AWSServiceRoleForAutoScaling_test

4. Now we need to add a lifecycle hook and a notification to your Auto Scaling group. Change the values inside the brackets for the correct values.

For the scale-out action lifecycle hook, use the following **put-lifecycle-hook** command.

```
aws autoscaling put-lifecycle-hook --lifecycle-hook-name [lifecycle hook name]
--auto-scaling-group-name [auto scaling group name]  --lifecycle-transition
autoscaling:EC2_INSTANCE_LAUNCHING --notification-target-arn [put sns topic arn here] --role-arn [put iam role arn here]
```

For the scale-in action lifecycle hook, use the following **put-lifecycle-hook** command.

```
aws autoscaling put-lifecycle-hook --lifecycle-hook-name [lifecycle hook name]
--auto-scaling-group-name [auto scaling group name]  --lifecycle-transition
autoscaling:EC2_INSTANCE_TERMINATING --notification-target-arn [put sns topic arn here] --role-arn
[put iam role arn here]
```

Once configured, the SNS topic receives a test notification with the following key-value pair:

"Event": "autoscaling:TEST_NOTIFICATION"

That's it. Your Auto Scaling lifecycle hook is configured with an SNS notification that will send out an email to you once the scale-out or scale-in event lifecycle hook puts the instance on the “wait” state.

References:
https://docs.aws.amazon.com/autoscaling/ec2/userguide/configuring-lifecycle-hook-notifications.html

Suspending and Resuming Scaling Processes

Amazon EC2 Auto Scaling has two primary process types. It will either Launch or Terminate an EC2 instance. Other process types are related to specific scaling features :

- **AddToLoadBalancer** — Adds instances to the attached load balancer or target group when they are launched.
- **AlarmNotification** — Notifications from CloudWatch alarms that are associated with the group's scaling policies.
- **AZRebalance** — Balances the number of EC2 instances in the group evenly across all of the specified Availability Zones when the group becomes unbalanced.
- **HealthCheck** — Monitors the health of the instances and marks an instance as unhealthy if Amazon EC2 or AWS Elastic Load Balancing tells Amazon EC2 Auto Scaling that the instance is unhealthy.
- **ReplaceUnhealthy** — Terminates instances that are marked as unhealthy and then launches new instances to replace them.
- **ScheduledActions** — Performs scheduled scaling actions that you create or that are created by predictive scaling.

You can suspend/resume any of the process types above if you do not want them active in your auto scaling group. You would usually perform this if you are troubleshooting a scaling event and you don't want to impact system performance. When you suspend a primary process type, other process types may cease to function properly.

Reference:
https://docs.aws.amazon.com/autoscaling/ec2/userguide/as-suspend-resume-processes.html

Some Limitations to Remember for Amazon EC2 Auto Scaling Group

Keep in mind that auto scaling groups are regional services and do not span multiple AWS Regions. You can configure them to span multiple Availability Zones, since they were designed in the first place to help you

achieve high availability and fault tolerance. However, if you need to use multiple Regions for scaling horizontally, you will need to implement a different solution to achieve this result. The same goes for launch configurations and launch templates you create. They only exist within the Region you created them in. If you need to copy over your launch configurations and templates to another Region, simply recreate them in the desired target Region. Another thing to remember is when you've configured your EC2 Auto Scaling Group to spread your instances across multiple Availability Zones, you cannot use cluster placement groups in conjunction with this setup, since cluster placement groups cannot span multiple Availability Zones.

Amazon Elastic Container Service

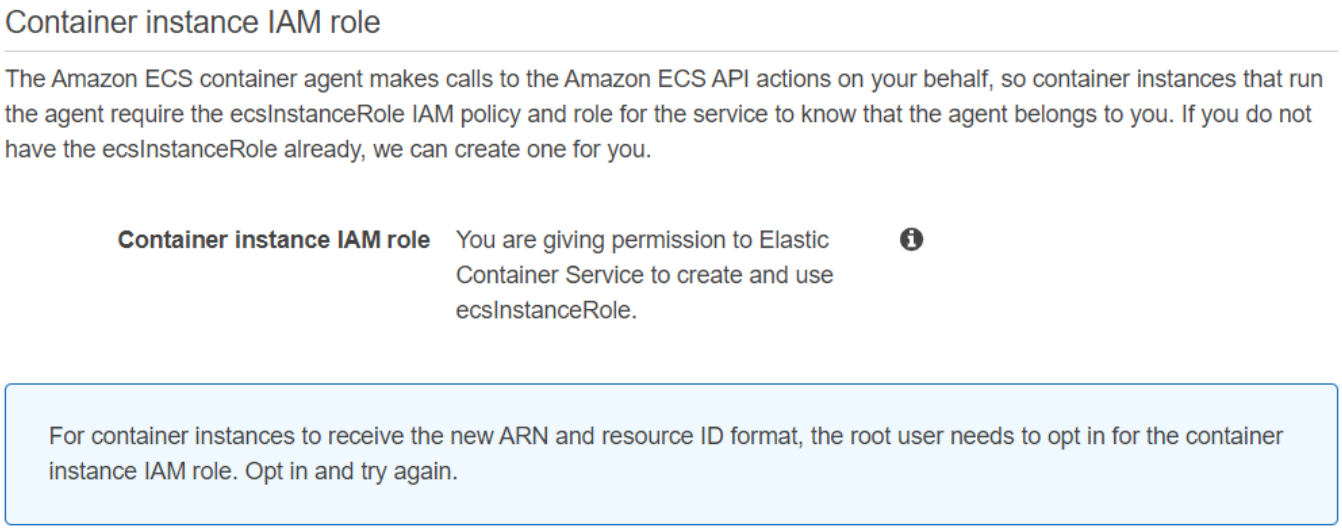
Amazon Elastic Container Service, or Amazon ECS for short, is a highly scalable, high-performance container orchestration service that supports Docker containers. It allows you to easily install, operate, and scale your cluster management infrastructure in AWS.

Amazon ECS eliminates the need for you to install and operate your own container orchestration software. It helps you manage the cluster of servers that power your containers. These containers are defined in a task definition which you use to run individual tasks or are grouped together as a service. You can run your ECS tasks and services using a serverless compute engine called AWS Fargate or with Amazon EC2 instances that you can fully access and manage. For security, you can specify an IAM role that can be used by the containers in a task. You can build docker images and store them in Amazon Elastic Container Registry or ECR.

Just like Amazon EC2, you can integrate Amazon ECS to a wide range of AWS services. For storage, you can use Amazon EFS or Amazon FSx to store your data. You can design your ECS tasks to talk to each other using an Amazon SQS queue, Amazon Kinesis Data Stream, or any other integration service. You can also use Amazon ECS Service Auto Scaling to increase or decrease the number of your ECS tasks automatically.

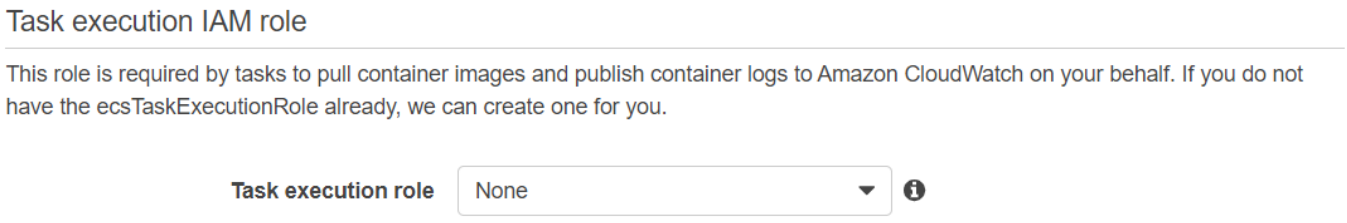
Amazon ECS Container Instance Role vs Task Execution Role vs Task Role

An ECS cluster is the very first resource you create in Amazon ECS. You define your cluster's underlying infrastructure, instance provisioning model (on-demand or spot), instance configuration (AMI, type, size, volumes, key pair, number of instances to launch), cluster network and container instance role. The container instance role allows the Amazon ECS container agent running in your container instances to call ECS API actions on your behalf. This role attaches the *ecsInstanceRole* IAM policy.



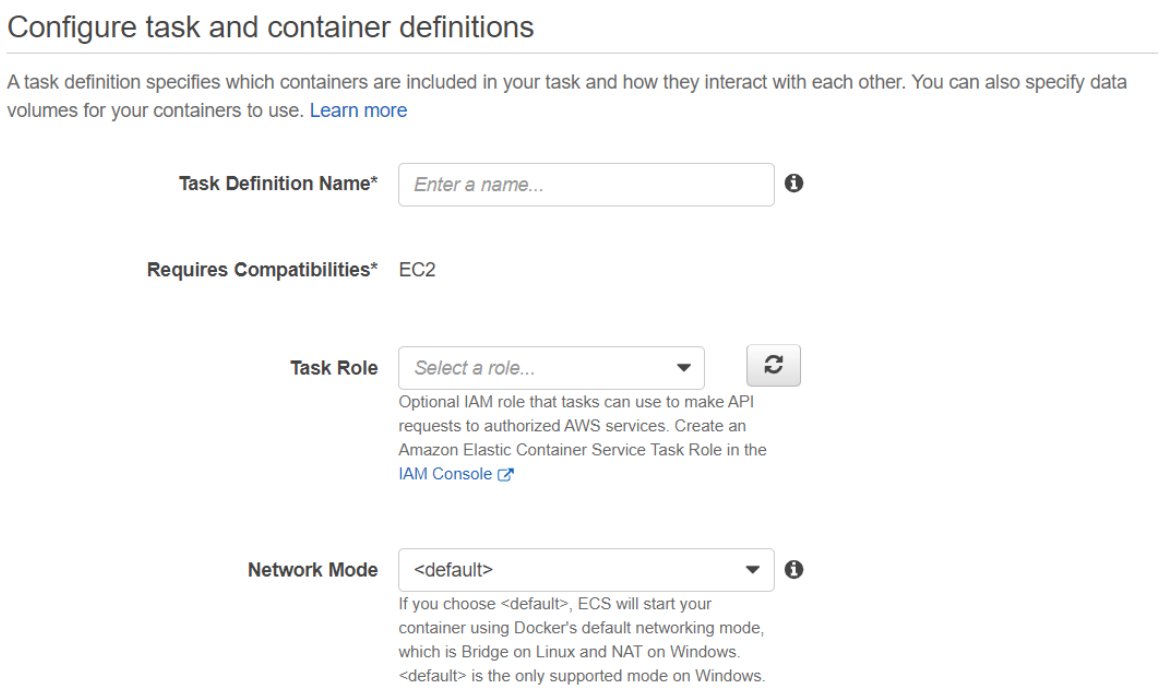
After creating your ECS cluster, one of the very first things you'll do next is create your task definition. A task definition is like a spec sheet for the Docker containers that will be running in your ECS instances or *tasks*. The following are the parameters that are defined in a task definition:

- The Docker image to use with each container in your task
- CPU and memory allocation for each task or each container within a task
- The launch type to use (EC2 or Fargate)
- The Docker networking mode to use for the containers in your task
- The logging configuration to use (bridge, host, awsvpc, or none)
- Whether the task should continue to run if the container finishes or fails
- The command the container executes when it is started
- Volumes that should be mounted on the containers in a task
- The Task Execution IAM role that provides your tasks permissions to pull Docker images and publish container logs.



Lastly, since the containers running in your ECS tasks might need to make some AWS API calls themselves, they will need the appropriate permissions to do so. The task role provides your containers permissions to make API requests to authorized AWS services. In addition to the standard ECS permissions required to run

tasks and services, IAM users also require iam:PassRole permissions to use IAM roles for tasks. Assigning a task role is optional.



References:
https://docs.aws.amazon.com/AmazonECS/latest/developerguide/task_execution_IAM_role.html
https://docs.aws.amazon.com/AmazonECS/latest/developerguide/task-iam-roles.html
https://tutorialsdojo.com/amazon-elastic-container-service-amazon-ecs/

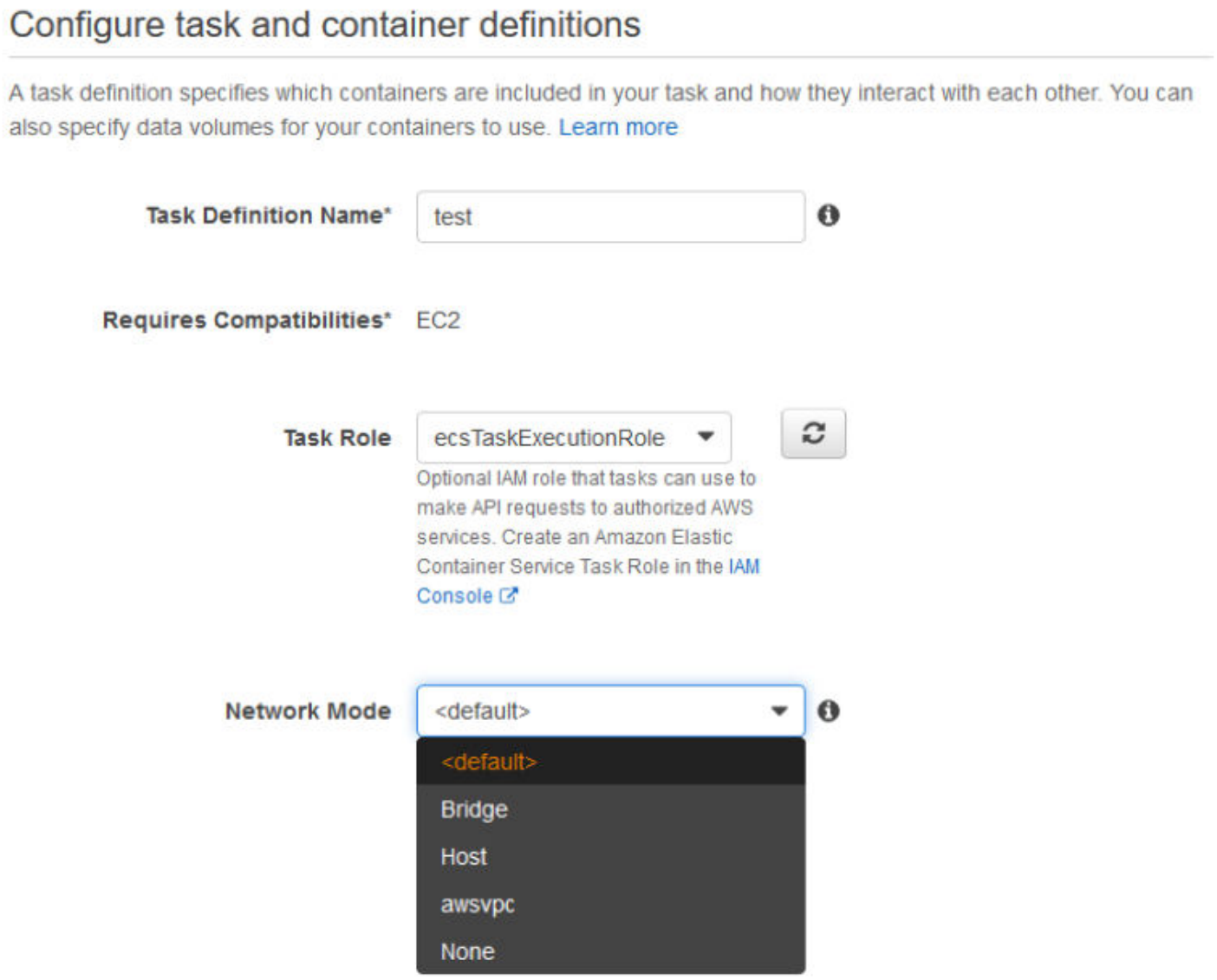
ECS Network Mode Comparison

Amazon Elastic Container Service (ECS) allows you to run Docker-based containers on the cloud. Amazon ECS has two launch types for operation: EC2 and Fargate. The EC2 launch type provides EC2 instances as hosts for your Docker containers. For the Fargate launch type, AWS manages the underlying hosts so you can focus on managing your containers instead. The details and configuration on how you want to run your containers are defined on the ECS Task Definition which includes options on networking mode.

In this post, we'll talk about the different networking modes supported by Amazon ECS and determine which mode to use for your given requirements.

ECS Network Modes

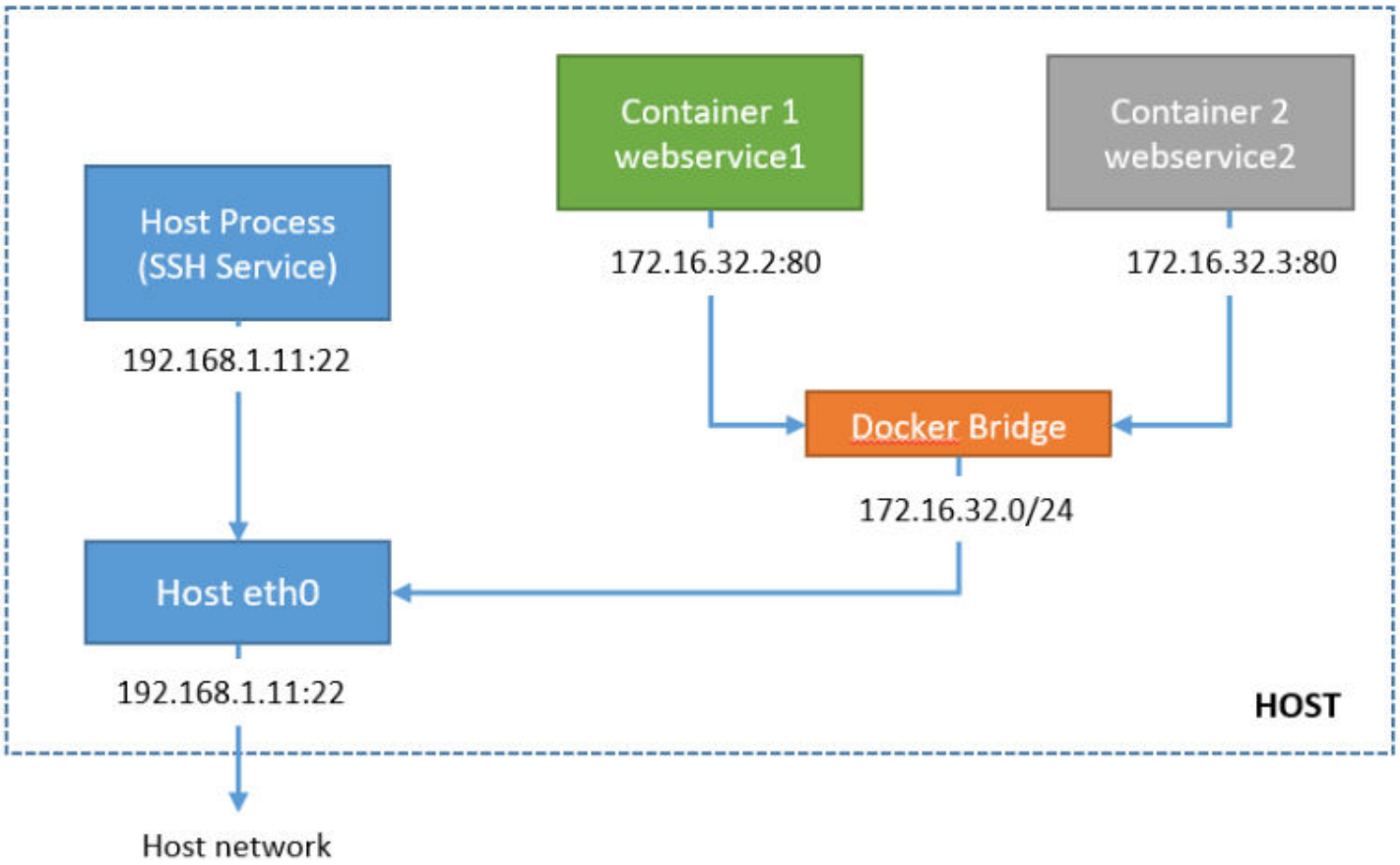
Amazon Elastic Container Service supports four networking modes: **Bridge**, **Host**, **awsvpc**, and **None**. This selection will be set as the Docker networking mode used by the containers on your ECS tasks.



Bridge network mode – Default

When you select the **<default>** network mode, you are selecting the **Bridge** network mode. This is the default mode for Linux containers. For Windows Docker containers, the **<default>** network mode is **NAT**. You must select **<default>** if you are going to register task definitions with Windows containers.

Bridge network mode utilizes Docker's built-in virtual network which runs inside each container. A bridge network is an internal network namespace in the host that allows all containers connected on the same bridge network to communicate. It provides isolation from other containers not connected to that bridge network. The Docker driver handles this isolation on the host machine so that containers on different bridge networks cannot communicate with each other.

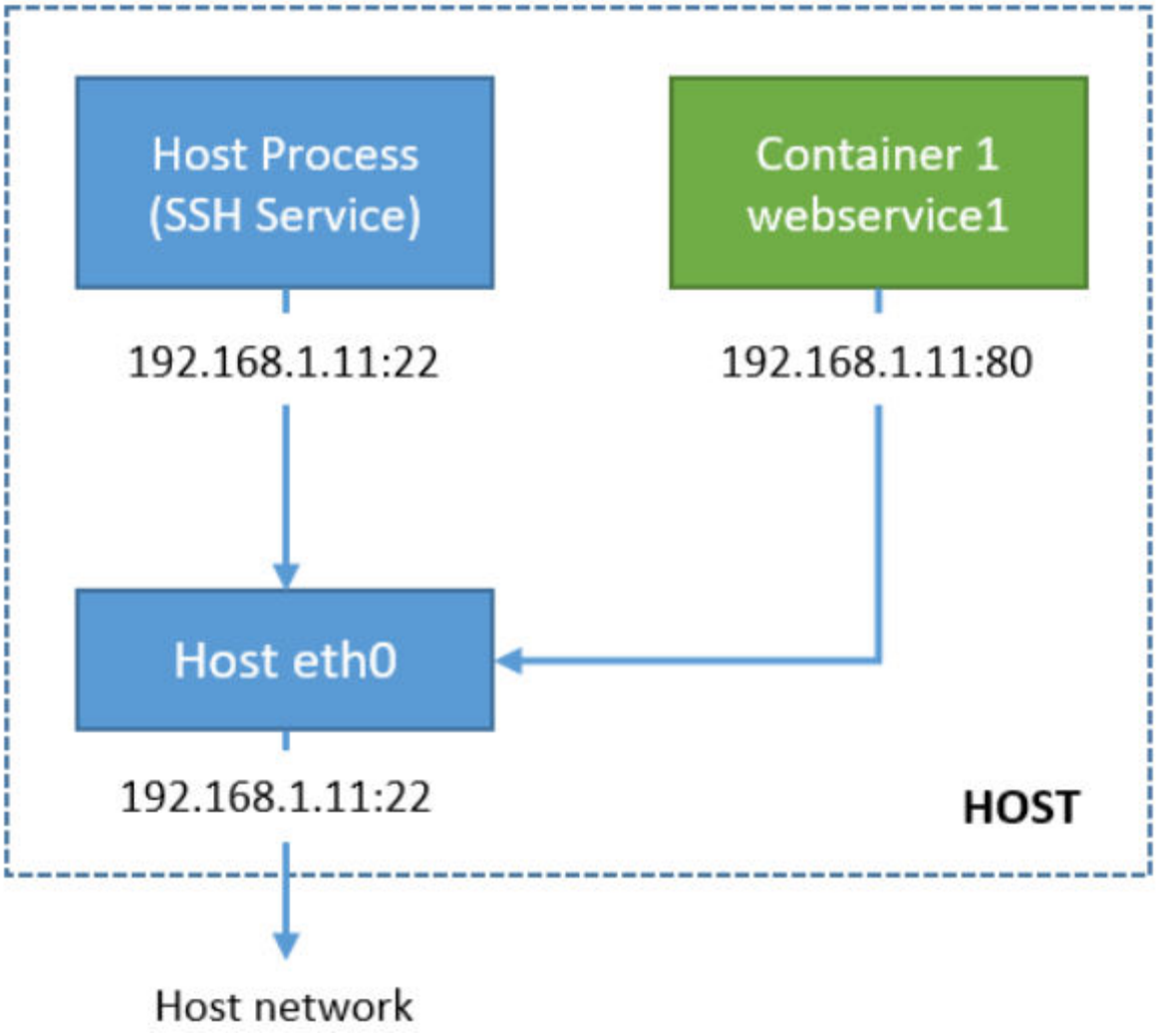


This mode can take advantage of dynamic host port mappings as it allows you to run the same port (ex: port 80) on each container, and then map each container port to a different port on the host. However, this mode does not provide the best networking performance because the bridge network is virtualized and Docker software handles the traffic translations on traffic going in and out of the host.

Host network mode

Host network mode bypasses the Docker's built-in virtual network and maps container ports directly to your EC2 instance's network interface. This mode shares the same network namespace of the host EC2 instance so your containers share the same IP with your host IP address. This also means that you can't have multiple

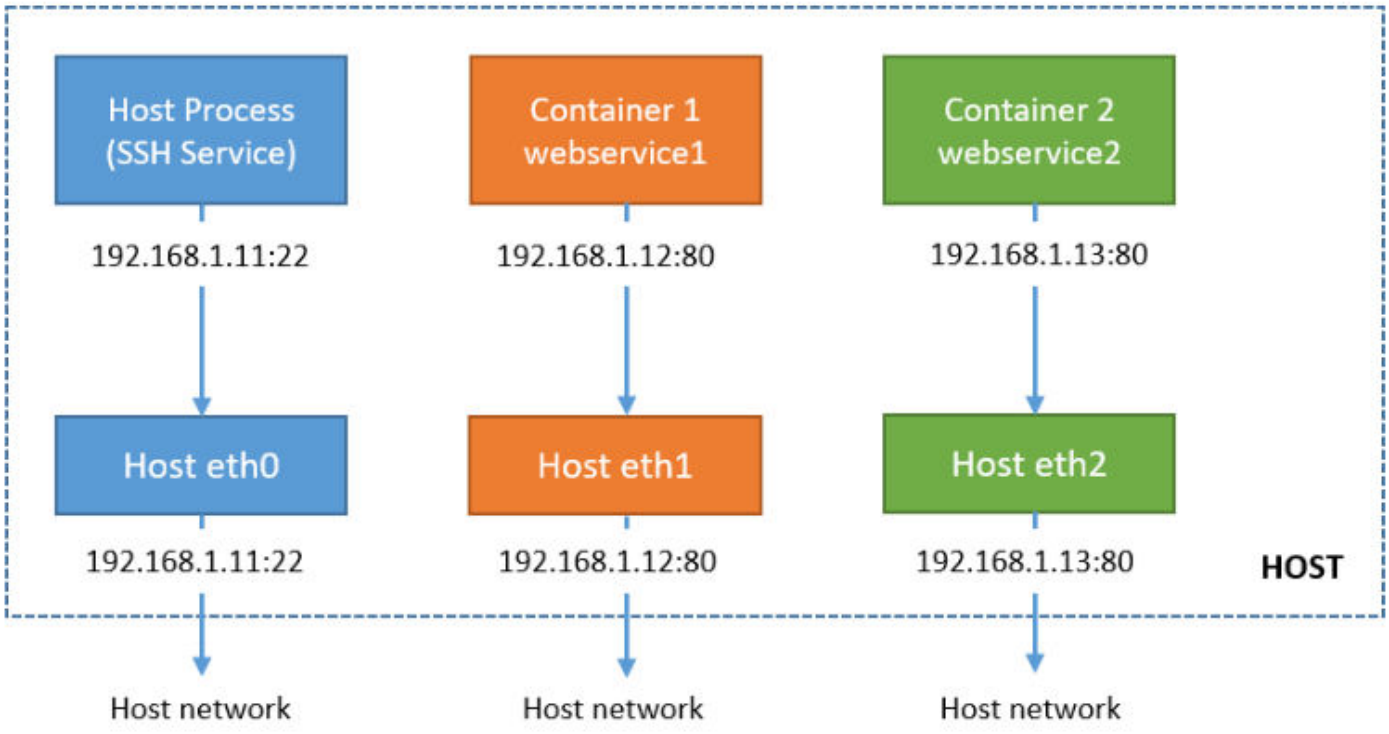
containers on the host using the same port. A port used by one container on the host cannot be used by another container as this will cause conflict.



This mode offers faster performance than the bridge network mode since it uses the EC2 network stack instead of the virtual Docker network.

awsvpc mode

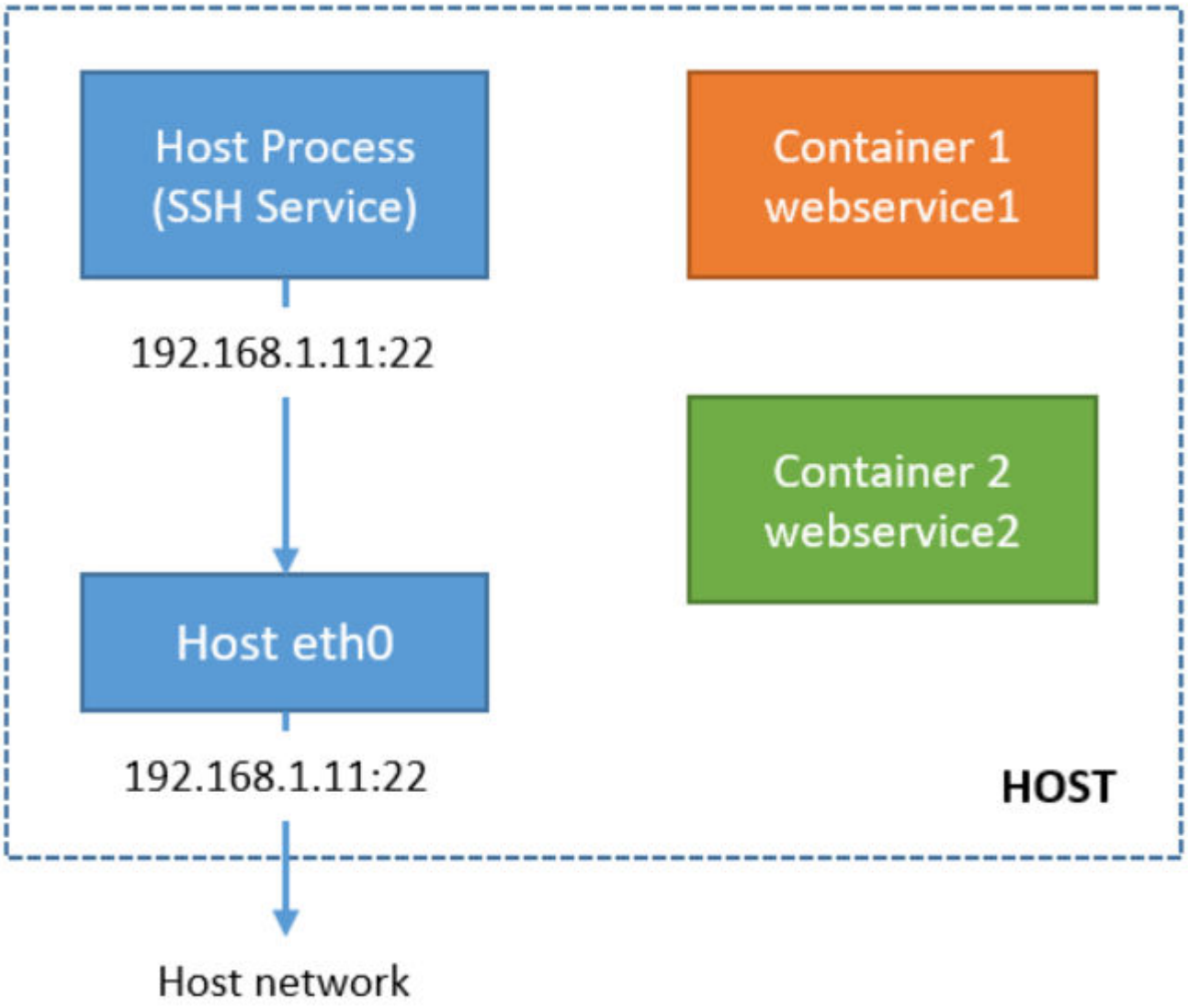
The **awsvpc** mode provides an elastic network interface for each task definition. If you have one container per task definition, each container will have its own elastic network interface and will get its own IP address from your VPC subnet IP address pool. This offers faster performance than the bridge network since it uses the EC2 network stack, too. This essentially makes each task act like their own EC2 instance within the VPC with their own ENI, even though the tasks actually reside on an EC2 host.



Awsvpc mode is recommended if your cluster will contain several tasks and containers as each can communicate with their own network interface. This is the only supported mode by the ECS Fargate service. Since you don't manage any EC2 hosts on ECS Fargate, you can only use awsvpc network mode so that each task gets its own network interface and IP address.

None network mode

This mode completely disables the networking stack inside the ECS task. The loopback network interface is the only one present inside each container since the loopback interface is essential for Linux operations. You can't specify port mappings on this mode as the containers do not have external connectivity.

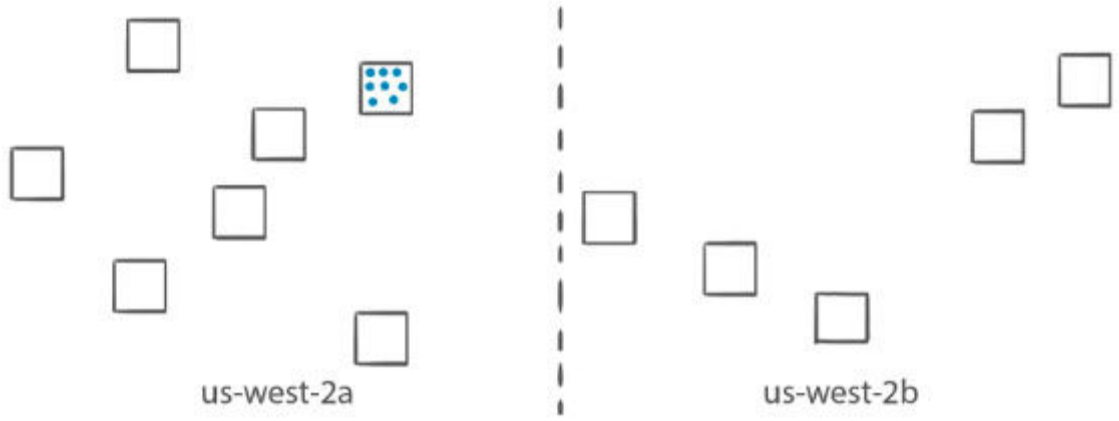


You can use this mode if you don't want your containers to access the host network, or if you want to use a custom network driver other than the built-in driver from Docker. You can only access the container from inside the EC2 host with the Docker command.

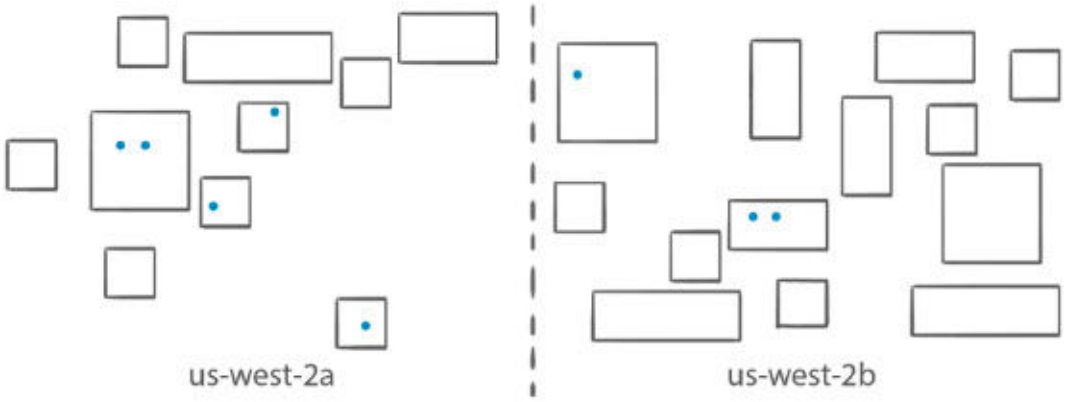
References:
https://docs.aws.amazon.com/AmazonECS/latest/developerguide/task_definition_parameters.html#network_mode
https://docs.aws.amazon.com/AmazonECS/latest/developerguide/task-networking.html

ECS Task Placement Strategies

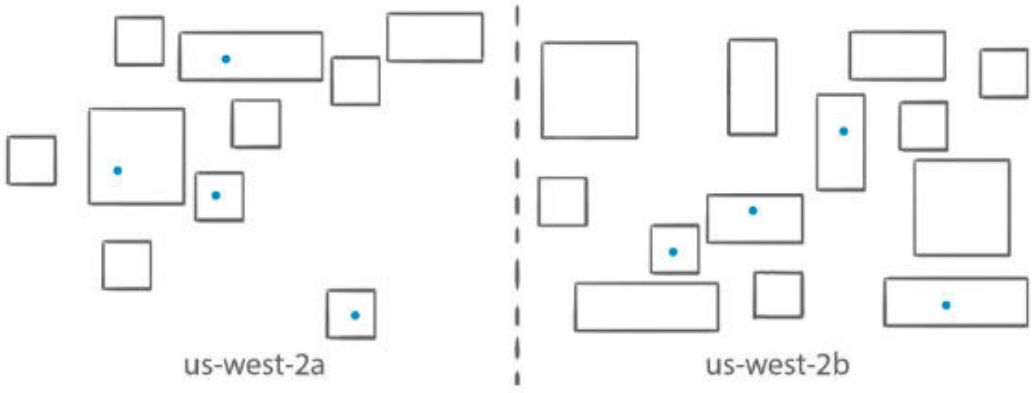
- A *task placement strategy* is an algorithm for selecting instances for task placement or tasks for termination. When a task that uses the EC2 launch type is launched, Amazon ECS must determine where to place the task based on the requirements specified in the task definition, such as CPU and memory. Similarly, when you scale down the task count, Amazon ECS must determine which tasks to terminate.
- A *task placement constraint* is a rule that is considered during task placement.
    - You can use constraints to place tasks based on Availability Zone or instance type.
    - You can also associate attributes, which are name/value pairs, with your container instances and then use a constraint to place tasks based on attribute.
- Task placement strategy types:
    - **Binpack** – Place tasks based on the least available amount of CPU or memory. This minimizes the number of instances in use and allows you to be cost-efficient. For example, you have running tasks in c5.2xlarge instances that are known to be CPU intensive but are not memory consuming. You can maximize your instances' memory allocation by launching tasks in them instead of spawning a new instance.



    - **Random** – Place tasks randomly. You use this strategy when task placement or termination does not matter.



- **Spread** – Place tasks evenly based on the specified value. Accepted values are attribute key-value pairs, instanceId, or host. Spread is typically used to achieve high availability by making sure that multiple copies of a task are scheduled across multiple instances. **Spread across Availability Zones** is the default placement strategy used for services.



- You can combine different strategy types to suit your application needs.
- Task placement strategies are a best effort.
- By default, Fargate tasks are spread across Availability Zones.
- By default, ECS uses the following placement strategies:
  - When you run tasks with the RunTask API action, tasks are placed randomly in a cluster.
  - When you launch and terminate tasks with the CreateService API action, the service scheduler spreads the tasks across the Availability Zones (and the instances within the zones) in a cluster.

References:
https://docs.aws.amazon.com/AmazonECS/latest/developerguide/task-placement.html
https://aws.amazon.com/blogs/compute/amazon-ecs-task-placement/

Amazon Elastic Kubernetes Service

Amazon Elastic Kubernetes Service, or EKS, is a fully-managed Kubernetes service. It is a portable, extensible, and open-source platform for managing containerized workloads and services. Kubernetes lets you orchestrate a cluster of virtual machines. It schedules containers to run on those VMs based on their available resources and the specified requirements of each container.

These containers are grouped into pods, which is the basic operational unit for Kubernetes. Amazon EKS allows you to run these pods on either AWS Fargate or Amazon EC2. Unlike Docker, you can easily move your workloads to your on-premises network or to other cloud service providers like Microsoft Azure, or the Google Cloud Platform (GCP). This is why Kubernetes is considered as cloud-agnostic.

Remain Cloud Agnostic with Kubernetes

Amazon EKS lets you easily run and scale Kubernetes applications in the AWS cloud or on-premises. Kubernetes is not an AWS native service. Kubernetes is an open-source container-orchestration tool used for deployment and management of containerized applications. Amazon EKS just builds additional features on top of this platform so you can run Kubernetes in AWS much easier. If you have containerized applications running on-premises that you would like to move into AWS, but you wish to keep your applications as cloud agnostic as possible then EKS is a great choice for your workload. All the Kubernetes-supported tools and plugins you use on-premises will also work in EKS. You do not need to make any code changes when replatforming your applications.

An EKS cluster consists of two components:

- The Amazon EKS control plane
- And the Amazon EKS nodes that are registered with the control plane

The Amazon EKS control plane consists of control plane nodes that run the Kubernetes software, such as *etcd* and the Kubernetes API server. The control plane runs in an account managed by AWS, and the Kubernetes API is exposed via the cluster's EKS endpoint. Amazon EKS nodes run in your AWS account and connect to your cluster's control plane via the API server endpoint and a certificate file that is created for your cluster.

To join worker nodes to your Amazon EKS cluster, you must complete the following:

1. Enable DNS support for your cluster's VPC
2. Provide sufficient IAM permissions for your instance profile's worker nodes
3. Configure the user data for your worker nodes
4. Launch your worker nodes in a subnet belonging to your cluster's VPC
5. Update the *aws-auth* ConfigMap with the *NodeInstanceRole* of your worker nodes
6. Add in the required security group rules of your worker nodes

7. Set the tags for your worker nodes
8. Verify that your worker nodes can reach the API server endpoint for your EKS cluster
9. Connect to a worker node's EC2 instance via SSH and review the kubelet agent logs for any errors

References:
https://docs.aws.amazon.com/eks/latest/userguide/clusters.html
https://aws.amazon.com/premiumsupport/knowledge-center/eks-worker-nodes-cluster/

AWS Lambda

AWS Lambda is a serverless computing service that allows you to run your code, function, or an entire application, without managing any servers. You only pay for the compute time that you actually consumed. Again, serverless computing still uses servers but all the server management, scaling, and troubleshooting are all done by AWS.

An uploaded code in AWS Lambda is also called a “Lambda function”. These functions can perform a custom task that you can control or program. A Lambda function is also highly scalable. It can instantly support thousands of requests in a matter of seconds! It provides high availability without additional effort on your part.

AWS Lambda is a fully managed service, so unlike Amazon EC2, you can neither access nor manage the underlying server that powers your Lambda function. It supports multiple programming languages such as Java, Go, Ruby, Node.js, Python, and others, through the use of runtimes. These runtimes are powered by Amazon Linux EC2 and other servers that are managed by AWS. Again, you can't access these servers directly unlike your Amazon EC2 instance.

Concurrency Limits

AWS Lambda is a blessing for developers who do not want to maintain any infrastructure. You don't need to worry about things like sizing, scaling, patching, and other management operations that you would normally have on servers such as EC2 instances. In Lambda, you just need to choose a runtime environment, provide your code, and configure other basic settings like the memory size available for each function call, the timeout of each function run, function triggers if applicable, etc. Although AWS Lambda is serverless, this doesn't mean that you don't have anything to manage on your end. If left unchecked, you'll be surprised how each function execution can add to your monthly bill. Your other Lambda functions might not even execute properly if one of your functions is hogging all the compute resources available to you. As with everything that scales automatically, you should be placing hard limits on the scalability so it will not explode all over the place. In AWS Lambda, this limit is known as **concurrency limit**.

Concurrency is the number of requests that your function is serving at any given time. When your function is invoked, Lambda allocates an instance of it to process the event. By default, your AWS account has a default quota of 1000 concurrent Lambda executions per Region. All your Lambda functions count against this limit. By setting a concurrency limit for your Lambda function, you reserve a portion of your concurrency limit for that given function. This allows you to throttle the given function once it reaches the maximum number of concurrent executions you've set for it.

There are two types of concurrency:

- **Reserved concurrency** – A pool of requests that can only be used by the function that reserved the capacity, and also prevents the function from using unreserved concurrency. A function cannot utilize another function's reserved concurrency, so other functions can't prevent your function from scaling.
- **Provisioned concurrency** – Initializes a requested number of execution environments so that they are prepared to respond to your function's invocations without any fluctuations.

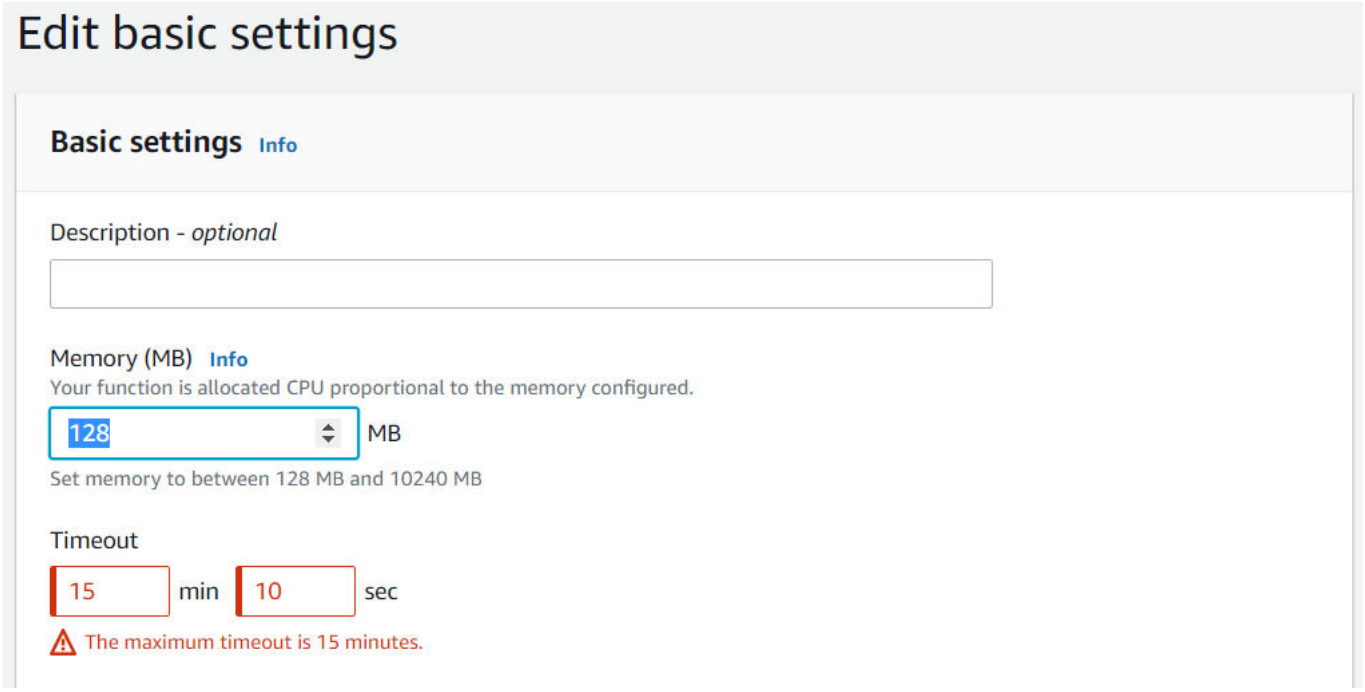
Both of these concurrency plans can be used together, but your provisioned concurrency cannot exceed your maximum reserved concurrency. Furthermore, Lambda integrates with Application Auto Scaling which lets you manage provisioned concurrency for your functions based on a schedule or on utilization. Managing your concurrency limits makes sure that your Lambda functions will run properly, and that they don't scale out of control.

References:
https://docs.aws.amazon.com/lambda/latest/dg/configuration-concurrency.html
https://aws.amazon.com/about-aws/whats-new/2017/11/set-concurrency-limits-on-individual-aws-lambda-functions/
https://tutorialsdojo.com/aws-lambda/

Maximum Memory Allocation and Timeout Duration

AWS Lambda allocates CPU power in proportion to the amount of memory you configure for a single function. And each function also has a timeout setting, which is the amount of time a single function execution is allowed to complete before a timeout is returned. For every Lambda function, you can indicate the maximum memory you'd like to allocate for a single execution as well as the execution duration of the function before timing out. The amount of memory you can allocate for a function is between 128 MB and 10,240 MB in 1-MB increments. At 1,769 MB, a function has the equivalent of one vCPU. For the timeout, the default is three seconds, and the maximum allowed value is 900 seconds or 15 mins.

Knowing this, some might think "*Why not just allocate the maximum memory and timeout for all Lambda functions*?" Well, first of all, allocating large amounts of memory when you don't need it will result in an increase in cost. You are charged an amount corresponding to your memory allocation for every 1ms that your function runs per execution. Same goes with your timeout settings. Aside from being billed for the duration of your function executions, there are cases where an application should fail fast. Choosing the optimal memory and timeout settings can be difficult to gauge for a new function, but with a few test runs and metric data in CloudWatch, you should be able to determine what works best for you.



References:
https://docs.aws.amazon.com/lambda/latest/dg/configuration-console.html
https://docs.aws.amazon.com/lambda/latest/dg/configuration-memory.html
https://docs.aws.amazon.com/whitepapers/latest/serverless-architectures-lambda/timeout.html

Lambda@Edge Computing

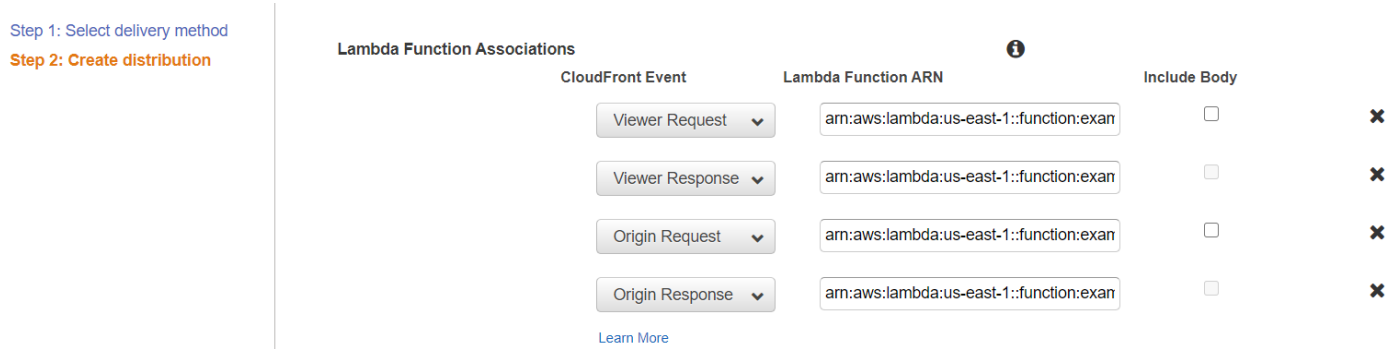
Lambda@Edge is a feature of Amazon CloudFront that lets you run Lambda code at edge locations around the world. Since this is a feature powered by both Lambda and CloudFront, there is no infrastructure to maintain or deploy. You only need to provide your Node JS or Python code and configure the type of CloudFront requests that your function will respond to, and AWS handles the provisioning and scaling of everything else needed by your code.

Your Lambda@Edge functions can be triggered in response to certain types of CloudFront requests:

- After CloudFront receives a request from an end user or device (**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 an end user or device (**viewer response**)

A CloudFront distribution can have multiple Lambda functions associated with it. Lambda@Edge simplifies and speeds up a lot of basic tasks since the code execution does not need to be routed all the way to your application's location before it can send back a response. Associating a Lambda function to your CloudFront

distribution is fairly straightforward. You just need to choose the type of trigger for your Lambda function, and input the corresponding Lambda function ARN. You can associate your Lambda functions during the creation of your CloudFront distribution, or modify an existing distribution.



A few examples on how you can use Lambda@Edge include:

1) Send different objects to your users based on the User-Agent header, which contains information about the device that submitted the request.
2) Inspect headers or authorized tokens, inserting a corresponding header and allowing access control before forwarding a request to the origin.
3) Add, delete, and modify headers, and rewrite the URL path to direct users to different objects in the cache.
4) Generate new HTTP responses to do things like redirect unauthenticated users to login pages, or create and deliver static web pages.

The difference between Lambda@Edge and Lambda with an API Gateway solution is that API Gateway and Lambda are regional services. Using Lambda@Edge and Amazon CloudFront allows you to execute logic across multiple AWS locations based on where your end viewers are located.

References:
https://docs.aws.amazon.com/lambda/latest/dg/lambda-edge.html
https://aws.amazon.com/lambda/edge/
https://tutorialsdojo.com/aws-lambda/

Connecting Your Lambda Function To Your VPC

There are some cases when your Lambda functions need to interact with your AWS resources. This is fairly easy to do if they are accessible via the public internet such as an Amazon S3 bucket or a public EC2 instance. But for private resources, you need to take some extra steps. By default, AWS Lambda is not able to access resources in a VPC. A Lambda function cannot properly resolve network traffic to your private subnets. This is especially frustrating when you need your Lambda function to connect to an RDS database for example. To

grant VPC connectivity to your Lambda functions, you must join them to your VPC, choose the subnets that your functions should have access to, and specify the necessary security groups that will allow communication between your VPC resources.

When you connect a function to a VPC, Lambda creates an elastic network interface for each subnet you included in your function's VPC configuration. Multiple functions connected to the same subnets share network interfaces. Lambda uses your function's permissions to create and manage network interfaces. Therefore, your function's execution role must have the same permissions under the **AWSLambdaVPCAccessExecutionRole** IAM Role. Once you've connected your functions to a VPC, your functions will cease to have public internet access unless your VPC has an internet gateway and/or a NAT (depending on which subnets you link your functions). You can also utilize VPC endpoints to connect to certain AWS services if NAT is an expensive option.

You can configure a Lambda function to be part of a VPC immediately at creation, or edit the VPC settings of an existing function. AWS recommends that you choose at least two subnets for high availability. If the AZ of a subnet becomes unavailable, and your Lambda function is running in this subnet, then your function cannot be invoked.

References:
https://docs.aws.amazon.com/lambda/latest/dg/configuration-vpc.html
https://aws.amazon.com/blogs/compute/announcing-improved-vpc-networking-for-aws-lambda-functions/

Amazon Simple Storage Service (S3)

Amazon S3 is an object storage service in AWS. S3 is actually an acronym for Simple Storage Service. It is a highly durable, highly available, and highly scalable service with many available features that you can use. This storage type is perfect for storing static data that does not change frequently. It makes your data publicly available via the Internet which can be accessed anywhere and not just within your virtual private cloud.

In Amazon S3, you will mainly work with objects and buckets. An object is just a regular file with corresponding metadata while the bucket is simply a resource that acts as a container for your objects. The metadata of the object is basically a set of name-value pairs that describes what the file is. Once the object has been uploaded to the bucket, its metadata would be permanent and cannot be modified. This object can be downloaded by multiple EC2 instances or by millions of users worldwide.

You can store virtually unlimited amounts of files to your bucket, such as text files, office documents, videos, database backups, snapshots, and many other data types. You can upload different objects to an S3 bucket that you own or are owned by others.

There is also a certain convention on how you can name your S3 bucket. An Amazon S3 bucket name is globally unique and its namespace is shared by each and every AWS account around the world! For example, if you already created an S3 bucket named “tutorialsdojo”, no other person on the planet can create a bucket with that same name! So if another person tries to create a new bucket called “tutorialsdojo” then that request will fail.

You can organize your objects by placing them inside a folder. However, this “folder” in S3 is different from a traditional file system storage. Amazon S3 has a flat structure, which is quite different from a hierarchical structure that you would see in a file system like in Amazon EFS. This folder is technically a prefix that is shared by your objects. If an object has a trailing forward-slash in its key name, then it is considered as a folder in S3. Again, this is done just for the sake of organizational simplicity. The folder concept in S3 is only meant for grouping objects and not for implementing file hierarchy.

For example, you can create a folder named “tutorialsdojo” and store a new object called “`aws.jpeg`” in it. The result will be an object with a key name of “`tutorialsdojo/aws.jpeg`”, where “ `tutorialsdojo/` ” is the prefix.

Take note that Amazon S3 does not support Portable Operating System Interface or POSIX. This means that it doesn't provide concurrent access to the same file or directory from thousands of compute instances. It's missing certain capabilities such as file system access semantics or file locking. If you need this functionality, you have to use the Amazon EFS or Amazon FSx for Lustre instead.

Your S3 bucket is a regional resource that uses all available data centers of the AWS region that you choose. The bucket is not hosted on your Amazon VPC or in a single Availability Zone. By default, Amazon S3 automatically replicates your object to all Availability Zones of an AWS region to ensure high data durability and availability. So if one region has 100 data centers, then your files are also replicated over 100 times! That level of replication provides high data durability for your files against unwanted data loss. This is the underlying physical architecture of how Amazon S3 can provide data durability and high availability.

It is designed to provide 99.99% availability over a given year and eleven 9s of durability. Amazon S3 provides 99.99999999999 percent or eleven 9s of durability. Okay, I know I sound funny saying a lot of 9s here, but what this percentage basically refers to is the probability of data loss in a given year.

In data storage terms, durability is the probability that an object remains intact and accessible after a period of one year or so. This is different from the concept of “data availability” which refers to the accessibility of your data at any given point in time. A 100% durability means that there is a 0% probability for the object to be lost in a given year. So if you have 100 objects, none of them will be lost. If it has 99% durability, then it would mean there is a 1% chance of data loss which affects one out of 100 objects. Conversely, if a storage system has 99.99% durability, then you only have a 0.01% data loss probability.

To put it in perspective, if you store 10,000 objects in a bucket, then the possibility of losing a single object will only happen every 10 million years or so. I know that it sounds exaggerated, but that's really how durable your data would be in Amazon S3.

There are different storage classes that you can choose from in Amazon S3 to place your objects. You can use the S3 Standard storage class to store your frequently accessed data; the S3 Intelligent-Tiering class for storing data with changing or unknown access patterns; S3 Standard-Infrequent Access and S3 One Zone-Infrequent Access for storing long-lived, yet less frequently accessed data; and for low-cost long-term storage and data archiving, you can use Amazon S3 Glacier or Amazon S3 Glacier Deep Archive. It also has a feature called “Lifecycle Policy” that automatically transitions or moves your data from one storage class to another.

Amazon S3 has a website hosting feature that you can use. You can host a static website by uploading HTML pages, downloadable packages, images, media files, or other client-side scripts to your bucket. This is perfect if you want a cost-effective and serverless solution to launch your website. However, you cannot run server-side scripts in Amazon S3 such as PHP, JSP, or ASP.NET since it does not support server-side scripting.

What's unique about this storage is its way of accessing its data. Unlike block or file storage systems, the objects in Amazon S3 are retrieved from your bucket via a REST API call. This is somehow similar to the process of downloading a file from an FTP server or a content repository.

You do not have to attach any S3 storage device to your EC2 instance. This is because Amazon S3 is actually a web service that is not confined to a single storage device. It does not reside in your Amazon VPC or in any

Availability Zone, which is quite different from how an EBS volume or an EFS file system works. Amazon S3 is a regional resource that runs on the AWS Cloud network. By default, the traffic between your EC2 instance and an S3 bucket passes through the public Internet, and not just within your VPC.

Amazon S3 has a different storage architecture than block storage and file storage types. It is a unique data storage system that manages your files or data as objects in a flat structure. Every object usually includes a globally unique identifier, the actual data, and its custom metadata. Your S3 storage can be scaled and replicated more extensively because it is not restricted within a hierarchical file structure.

In addition, it is not dependent on the operating system of the compute instance since you don't have to manually attach the S3 bucket to your Amazon EC2, unlike an EBS volume. You just upload or fetch objects using RESTful web APIs, such as the PUT HTTP method for uploading data and the GET HTTP method for downloading objects.

Amazon S3 also offers different security, versioning, and replication capabilities to manage your objects effectively. You can set up S3 Versioning and Multi-Factor Authentication to prevent accidental data deletion in S3. The network access to your S3 buckets and objects can be controlled using the Amazon S3 access control lists. You can also create an S3 bucket policy for controlling external access to your bucket. With its Cross-Region Replication feature, your objects can be replicated to a different AWS Region automatically. Transferring data to and from your S3 bucket can also be improved by using the Amazon S3 Transfer Acceleration and the S3 Multipart Upload options.

S3 Standard vs S3 Standard-IA vs S3 One Zone-IA vs S3 Intelligent Tiering

	S3 Standard	S3 Standard-Infrequent Access (IA)	S3 One Zone-Infrequent Access (IA)	S3 Intelligent Tiering
Features	General-purpose storage of frequently accessed data	For long-lived, rapid but less frequently accessed data; data is stored redundantly in multiple AZs	For long-lived, rapid but less frequently accessed data; data is stored redundantly in only one AZ of your choice	For long-lived data that have unpredictable access patterns
Durability	99.999999999% (11 9's)	99.999999999% (11 9's)	99.999999999% (11 9's)	99.999999999% (11 9's)
Availability	99.99%	99.9%	99.5%	99.9%
Availability SLA	99.9%	99%	99%	99%
Number of Availability Zones	At least 3	At least 3	Only 1	At least 3
Minimum capacity charge per object	N/A	128KB	128KB	N/A
Minimum storage duration charge	N/A	30 days	30 days	30 days
Inserting data	Directly PUT into S3 Standard	Directly PUT into S3 Standard-IA or set Lifecycle policies to transition objects from the S3 Standard to the S3 Standard-IA storage class.	Directly PUT into S3 One Zone-IA or set Lifecycle policies to transition objects from the S3 Standard to the S3 One Zone-IA storage class.	Directly PUT into S3 Intelligent-Tiering or set Lifecycle policies to transition objects from the S3 Standard to the S3 Intelligent-Tiering storage class.
Retrieval fee	N/A	per GB retrieved	per GB retrieved	N/A
First byte latency	milliseconds	milliseconds	milliseconds	milliseconds
Storage transition	S3 Standard to all other S3 storage types including Glacier	S3 Standard-IA to S3 One Zone-IA or S3 Glacier	S3 One Zone-IA to S3 Glacier	S3 Intelligent to S3 One Zone-IA or S3 Glacier
Use Cases	Cloud applications, dynamic websites, content distribution, mobile and gaming applications, and big data analytics.	Ideally suited for long-term file storage, older sync and share storage, and other aging data.	For infrequently-accessed storage, like backup copies, disaster recovery copies, or other easily recreatable data.	Data with unknown or changing access patterns, optimize storage costs automatically, and unpredictable workloads

Additional Notes:

- Data stored in the S3 One Zone-IA storage class will be lost in the event of AZ destruction.
- S3 Standard-IA costs less than S3 Standard in terms of storage price, while still providing the same high durability, throughput, and low latency of S3 Standard.
- S3 One Zone-IA has 20% less cost than Standard-IA.
- It is recommended to use multipart upload for objects larger than 100MB.

Accessing S3 Buckets Publicly and Privately

By default, a newly created S3 bucket and the objects you upload in it will not be publicly accessible. Users who need access to your S3 bucket and objects will need to be granted explicit permissions from the bucket owner or from an administrator. To provide access to users and other services, you can create resource-based policies such as bucket policies and access control policies that define who has access to what. AWS users

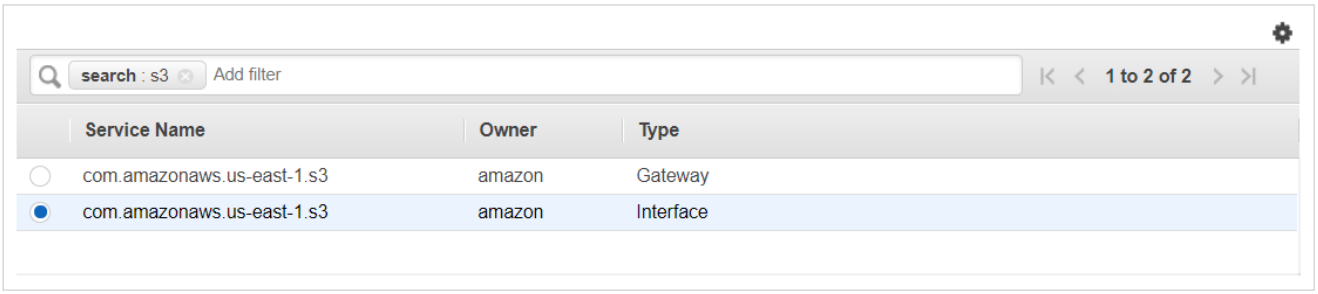
will also need the appropriate IAM permissions before they can perform any actions on your bucket and objects.

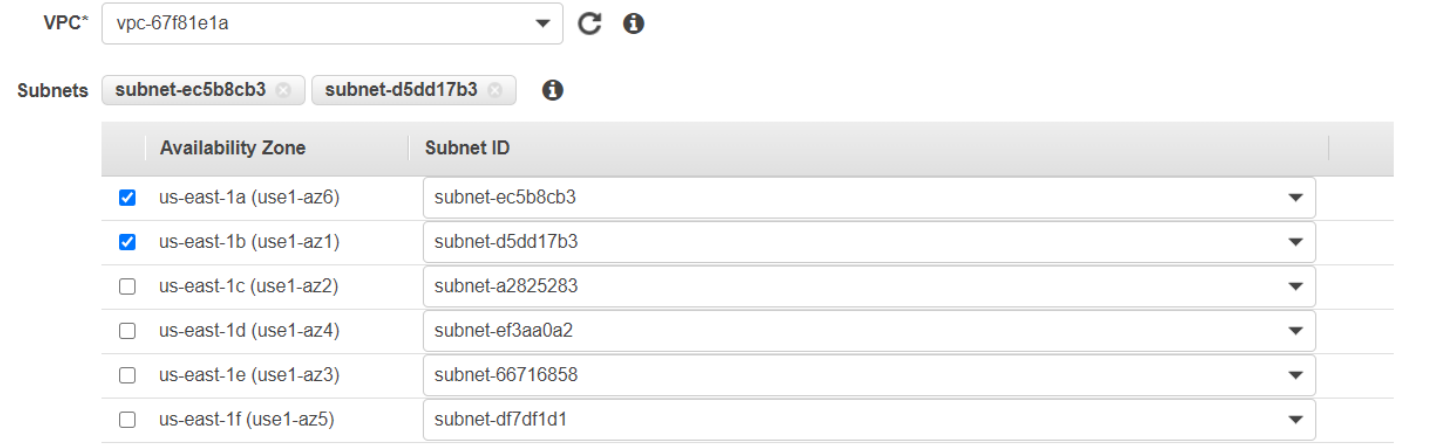
We know that once a user is provided access to an S3 bucket and its contents, all API activity on this bucket will pass through the public internet. This is true whether the request originates from within an AWS VPC or not. That is why your S3 bucket requires a unique name, to uniquely identify it with a publicly accessible S3 URL. But what if you prefer accessing S3 privately from within your VPC? What if you cannot afford having the data pass through the public internet? The first thing you'll need to do is create a VPC endpoint.

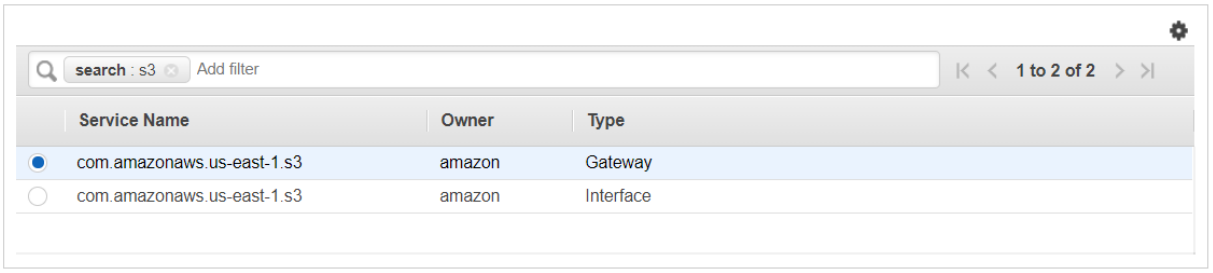
A VPC endpoint is a virtual device that allows your VPC resources to access AWS services directly without leaving the AWS network. VPC endpoints are powered by AWS PrivateLink, which enables you to privately access services by using their private IP addresses. Your VPC resources do not need to have public IP addresses to connect to Amazon S3 when using a VPC endpoint. To create a VPC endpoint, you first choose what type of endpoint you wish to use to access Amazon S3:

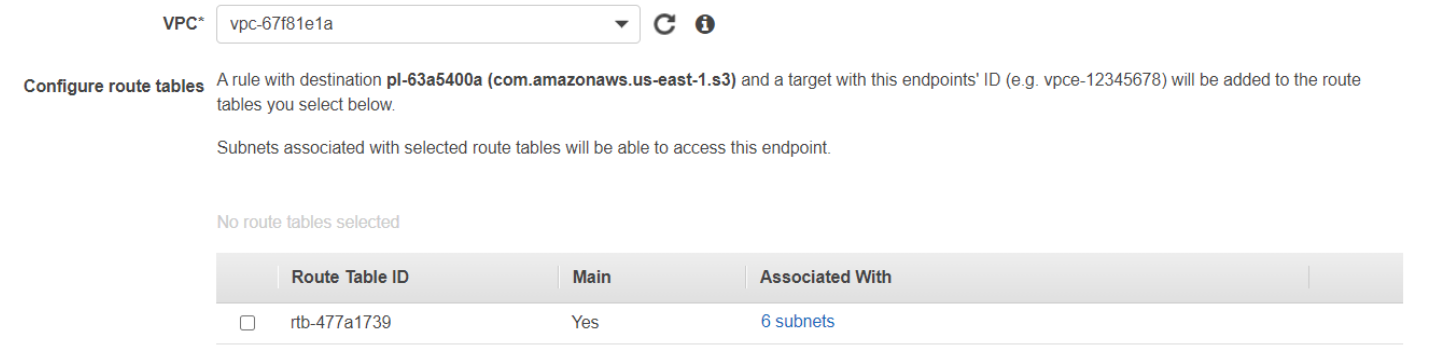
- An **interface endpoint** is an elastic network interface with a private IP address from the IP address range of the subnet(s) where you choose to deploy the ENI(s). Interface endpoints allow access from on-premises if it is connected to your VPC. It also allows access from resources that belong in a different region from your S3 bucket. You are billed for each interface endpoint you create.
- A **gateway endpoint** is a gateway that you specify in your route table(s) to direct traffic to S3. Gateway endpoints do not allow access from on-premises networks, and do not support cross-region access. Gateway endpoints are free of charge.

Next, you select the VPC you wish to associate your endpoint with. If you choose the interface endpoint option, you indicate which AZs and subnets to launch your endpoints in. You also select the security groups that are going to be attached to the ENIs. If you choose the gateway endpoint option, you indicate the route tables that will have a route to the endpoint.

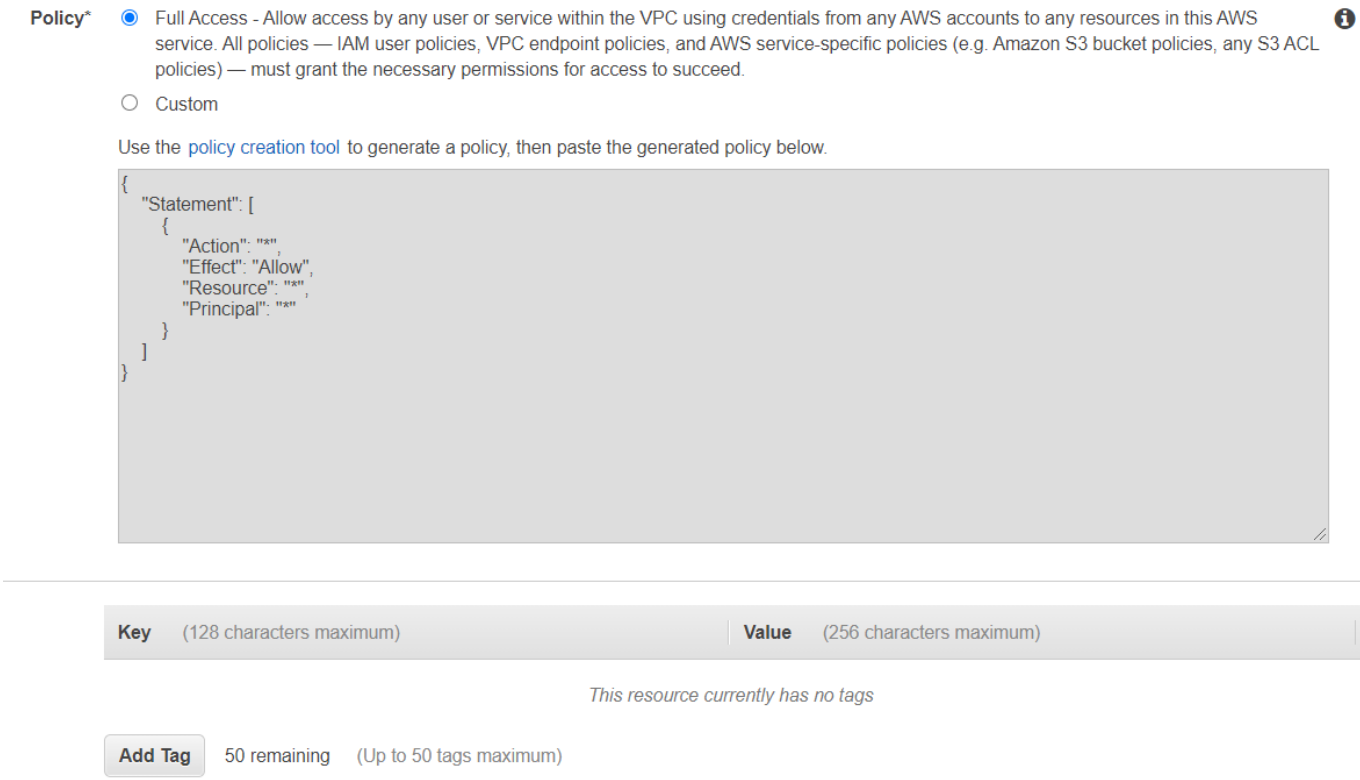








Optionally, you can create an access policy specifying the S3 buckets your endpoint will have access to, the principals that will be able to use your endpoint, and the actions they can make through your endpoint. You can also add tags to your endpoints.



Once you have created your endpoint, be sure to update your bucket policy with a condition that allows users to access the S3 bucket when the request is from the VPC endpoint.

References:
https://docs.aws.amazon.com/AmazonS3/latest/userguide/access-control-overview.html
https://docs.aws.amazon.com/AmazonS3/latest/userguide/privatelink-interface-endpoints.html
https://tutorialsdojo.com/amazon-s3/

Amazon S3 Bucket Features

In this section, we will tackle the features available in an S3 Bucket:

Lifecycle policies — These policies determine how your objects are stored in your S3 bucket. As you know, there are many S3 storage tiers to choose from. Lifecycle policies let you transition your objects from one storage tier to another, usually to reduce storage cost or to archive an object. Lifecycle policies are also used to

expire versioned objects and permanently delete them from your bucket. When creating a lifecycle policy, you configure two parameters for each transition or deletion action:

- Whether the policy should apply to all objects in the bucket or only a group of objects with matching prefix
- The number of days after object creation before the action is applied

S3 Bucket Policies and ACLs — S3 bucket policies are JSON-based policies used for access control. They work similarly to IAM policies, but are instead applied onto your S3 buckets rather than individual IAM users. You add a bucket policy to a bucket to grant other AWS accounts or IAM users access permissions for the bucket and the objects in it. Access control lists (ACLs), on the other hand, are preset options that you can enable to allow read and/or write access for other AWS accounts, users or the public.

Object Ownership — If you have external users uploading objects to a bucket you own, you can enable bucket-owner-full-control canned access control list (ACL) to automatically assume full ownership over the objects they upload.

Multipart Upload — For objects larger than 100MB, you can use S3's multipart upload feature to divide your file into parts and upload them individually. After all parts of your object are uploaded, S3 assembles these parts and creates the object. Multipart upload offers multiple benefits such as faster throughput thanks to parallel upload, retransmission for failed uploads, pause and resume upload capabilities, and better stability for uploading files with unknown file sizes.

S3 Transfer Acceleration — S3 TA leverages Amazon CloudFront's globally distributed edge locations to optimize long distance transfers from your client to Amazon S3. Although there is no guarantee that you will experience faster transfer speeds, S3 TA only bills you when there is an improvement compared to a regular S3 transfer. Using S3 TA is as simple as enabling it in your S3 bucket. S3 Transfer Acceleration also supports all bucket level features including multipart upload.

Static Web Hosting — An S3 bucket can be made to host static files such as images and webpages. Since an S3 bucket is public, you can configure it as a website, using the S3 URL as your domain name. This feature is convenient if you only need a simple and cost-effective webpage to get you going. When you configure your S3 bucket as a static website, make sure to set your objects as publicly available too. Amazon S3 website endpoints do not support HTTPS or access points. You will need to add a CloudFront to use HTTPS. You can also provide your static website a custom domain name using a DNS record in Route 53 pointing to your S3 bucket URL. For this matter, the domain name and the name of the S3 bucket must be an exact match.

Versioning — Versioning lets you keep a copy of an object whenever it is overwritten as its *versions*. You can preserve and restore back to a specific version of an object if you need to. This feature also protects your objects from accidental deletions, since versioning places deletion markers on an object version to mark it as removed, rather than permanently deleting it from your S3 bucket. By default, versioning is disabled on buckets, and you must explicitly enable it. Once it has been enabled, it cannot be disabled, but it can be suspended.

When you suspend versioning, any future updates on your objects will not create a new version, but existing versions will still be retained. Since a version of an object also takes up storage space, versioning will incur additional S3 costs, so only use this feature if you need it.

MFA Delete — MFA delete is a security feature that is used together with S3 Versioning to prevent unauthorized or accidental deletions in your S3 bucket. When enabled, the bucket owner must include two forms of authentication in any request to delete an object version or change the versioning state of the bucket. These two forms of authentication are his/her security credentials and the concatenation of a valid serial number, a space, and the six-digit MFA code.

Cross-Region Replication and Same-Region Replication — Replication is a feature that allows you to replicate objects from an S3 bucket in one region to another bucket in the same region or in another region. Buckets that are configured for object replication can be owned by the same AWS account or by different accounts. Objects can be replicated to multiple destination buckets. By default, S3 replication does not replicate existing objects, only objects that have been uploaded after replication was enabled. You must contact AWS Support Center if you intend to replicate existing objects.

Object Lock — Allows you to store objects using a write-once-read-many (WORM) model. Object lock prevents an object from being deleted or overwritten for a fixed amount of time or indefinitely.

S3 Access Points — A feature that simplifies data access for any AWS service or 3rd party client application that stores data in an Amazon S3 bucket. An Amazon S3 Access Point allows customers to create unique access control policies for each S3 access point to control access to shared datasets easily. These shared data sets can be in a form of media archives, user-generated content or data lakes. This feature can easily scale access for hundreds of applications by creating individualized access points with disctinct names and permissions customized for each application. An S3 Access Point can be restricted to a Virtual Private Cloud (VPC) to firewall Amazon S3 data access the private networks of the custome. Additionally, the AWS Service Control Policies can be used to ensure all access points are restricted within the specified VPC. The Amazon S3 Access Points are available in all regions for free.

S3 Event Notifications — This lets you receive notifications on certain events that occur in your S3 bucket. To enable notifications, you must first add a notification configuration that identifies the events you want S3 to publish and the destinations (SNS, SQS, Lambda) where you want the notifications to be sent. Amazon S3 can publish notifications for the following events:

- New object created events
- Object removal events
- Restore object events
- Replication events

Cross-origin Resource Sharing (CORS) — CORS is a way for client applications that are loaded in one domain to interact with resources in a different domain. When this feature is disabled, requests directed to a different domain will not work properly. If your S3 bucket is used for web hosting, verify if you need to enable CORS. To configure your bucket to allow cross-origin requests, you create a CORS configuration document. This is a document with rules that identify the origins that you will allow to access your bucket, the operations (HTTP methods) that will support each origin, and other operation-specific information.

Presigned URLs - By default, all S3 buckets and objects are private, and can only be accessed by the object owner. Object owners can share objects with other users or enable users to upload objects to their S3 buckets using a presigned URL. A presigned URL grants others time-limited permission to download or upload objects from and to the owner's S3 buckets. When object owners create presigned URLs, they need to specify their security credentials, the bucket name and object key, the HTTP method (GET to download the object), and expiration date and time. The bucket owner then shares these URLs to those who need access to the objects or to the buckets. A presigned URL can be used many times, as long as it has not expired.

References:
https://docs.aws.amazon.com/AmazonS3/latest/userguide/Welcome.html
https://tutorialsdojo.com/amazon-s3/

Amazon S3 Pricing Details

Some storage tiers in Amazon S3 have minimum usage requirements that may affect your billing if you are unaware of them.

Storage Tier	S3 Standard	S3 Intelligent Tiering	S3 Infrequent Access	S3 One Zone-IA	S3 Glacier Instant Retrieval	S3 Glacier Flexible Retrieval	S3 Glacier Deep Archive
Minimum capacity charge per object	None	None	128 KB	128 KB	40 KB	40 KB	40 KB
Minimum storage duration charge	None	None	30 days	30 days	90 days	90 days	180 days
Retrieval fee	None	None	per GB retrieved	per GB retrieved	per GB retrieved	per GB retrieved	per GB retrieved

Minimum capacity charge per object means that an object should meet the specified minimum size once stored in the corresponding storage tier. If the object is less than the specified minimum then the object is billed according to the minimum size requirement. For example, if the minimum capacity charge is 128KB and your object is 40KB only then it is billed as a 128KB object by Amazon S3.

Minimum storage duration charge is the amount of time that the object should be stored in the corresponding storage tier. If the object is deleted before the duration passes then the object is billed as if it was stored for the whole minimum duration. For example, if you have a 128KB object stored in S3 IA for 15 days and you delete it the next day, Amazon S3 will continue to charge you an equivalent of storing a 128KB file for the next 15 days.

References:
https://aws.amazon.com/s3/storage-classes/
https://tutorialsdojo.com/amazon-s3/

Amazon S3 Encryption Methods

When you are using Amazon S3, it is always important to know how you can protect your data, especially if it contains sensitive information. Amazon S3 offers both Server-Side encryption and Client-Side encryption to secure your objects at rest and in-transit.

- **With Server-Side encryption (SSE)**, Amazon S3 encrypts your object before saving it on disks in its data centers and then decrypts it when you download the objects. You have three different options on how you choose to manage the encryption keys.
    - **With Amazon S3-Managed Keys (SSE-S3)** — S3 uses AES-256 encryption keys to encrypt your objects, and each object is encrypted with a unique key.
    - **With Customer Master Keys (CMKs) stored in AWS Key Management Service (SSE-KMS)** — Similar to SSE-S3, but your key is managed in a different service, which is AWS KMS. SSE-KMS 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.
    - **With Customer-Provided Keys (SSE-C)** — You manage the encryption keys and S3 manages the encryption and decryption process.
- **With Client-Side encryption (CSE)**, data is first encrypted on the client-side before uploaded to Amazon S3. You manage the encryption process, the encryption keys, and related tools. The encryption key you use can be any of the following:
    - Customer master key (CMK) stored in AWS KMS.
    - Master key that you store within your application.

Amazon S3 Glacier

Amazon S3 Glacier is actually one of the storage classes in Amazon S3 but it has a web console of its own and a different set of APIs apart from S3. Historically, Amazon S3 Glacier is just called “Amazon Glacier” and it's totally separate from S3. These two storage services were eventually merged into one. The name of this service is based on the word "glacier" which means a cold, large mass of ice. If your data is rarely accessed, it is considered as "cold data" and if it is frequently accessed, it is referred to as “hot data”. This is also the same concept with Cold HDD volumes in Amazon EBS.

The Amazon S3 Glacier class is composed of two sub types which are the:

- Amazon S3 Glacier Instant Retrieval
- Amazon S3 Glacier Flexible Retrieval

Amazon S3 Glacier provides low-cost storage for data archiving and long-term backup. This is suitable for files that are rarely accessed and archived for a number of years to meet your regulatory requirements. The archived files are stored in a resource called, the "vault”. You can further lower down your storage costs by using the Amazon S3 Glacier Deep Archive class, which is the cheapest storage type in AWS.

The disadvantage of using Glacier is the time it takes to retrieve your archive data. Just like frozen meat that you have ‘thaw', you also have to wait for a while in order for your data in Glacier to become available. Amazon Glacier offers three types of Archive Retrieval Options. These are: Expedited, Standard, and Bulk.

Amazon S3 Glacier vs Amazon S3 Glacier Deep Archive

Amazon S3 Glacier Deep Archive is similar to Amazon S3 Glacier in that they are both storage classes built for archiving objects that you won't need again for a long time. Deep Archive offers a more competitive price point than S3 Glacier if your primary requirement is a durable and secure long-term storage for large amounts of data, but the tradeoff is that retrieval times take longer to finish. To make the comparison of these two storage classes simpler, we'll list down the key similarities and differences in two parts.

Similarities:

- Low cost storage option for archiving cold data that won't be retrieved often.
- Supports lifecycle policies to transition objects from S3 Standard, Standard-IA, OneZone-IA and Intelligent Tiering to Glacier and Glacier Deep Archive.
- Offers durability of 99.999999999% of objects across three or more Availability Zones with 99.99% availability.
- You may use the S3 API to directly upload objects to these storage classes.
- Objects that are stored in the S3 Glacier or S3 Glacier Deep Archive storage classes are not available in real time.
- When you initiate a restore request, a temporary copy of the object is made available for the duration that you specify in the request.

- Support for Object Lock and Cross-Region Replication features.
- Supports backing up tape drives through AWS Storage Gateway Tape Gateway and Amazon Snow devices.
- To maximize cost savings, objects to be archived should be at least 40 KB in size.
- You are billed for the number of retrieval requests you make and the size of your data retrievals per GB.
- Both are backed by Amazon S3 SLA.

Differences:

- You can transition objects from S3 Glacier to S3 Glacier Deep Archive but not the other way around.
- S3 Glacier offers three types of retrieval options: **Expedited** (takes 1–5 minutes to finish but only if AWS has enough retrieval capacity), **Standard** (3–5 hours) and **Bulk** (5–12 hours).
- S3 Glacier Deep Archive offers two types of retrieval options: **Standard** (finishes within 12 hours) and **Bulk** (within 48 hours).
- To maximize cost savings, you need to keep your objects archived in Glacier for at least 90 days, while Glacier Deep Archive requires at least 180 days.

References:

https://docs.amazonaws.cn/en_us/AmazonS3/latest/userguide/storage-class-intro.html
https://aws.amazon.com/s3/pricing/

AWS Storage Gateway

AWS Storage Gateway is a hybrid cloud storage service that connects your on-premises data storage to the AWS Cloud. It allows your on-premises applications to seamlessly use cloud storage services available in AWS. It is mainly used to integrate your local and cloud storage systems by using a gateway. A gateway is just a virtual machine, or a hardware appliance running on your on-premises data center.

This service has 3 types: File Gateway, Volume Gateway, and a Tape Gateway. A File Gateway provides the ability for you to store and retrieve objects in Amazon S3 using the NFS and SMB protocols. File Gateway can also work with your Microsoft Active Directory, either hosted on-premises, or in AWS. You can also access your data using S3 APIs. If you don't have virtualized compute resources, you can use a hardware appliance hosted on-premises, to replicate your local data to Amazon S3.

Volume Gateway provides block storage to your on-premises applications via the Internet Small Computer System Interface or iSCSI. It also uses Amazon S3 to store your data by taking point-in-time snapshots of your volumes. This allows you to generate Amazon EBS snapshots that you can use for your EC2 instances. A Volume Gateway can run in either a cached or stored mode. Cached volumes store the primary data in Amazon S3 while retaining copies of frequently accessed data subsets locally. Stored volumes, on the other hand, still store the entire dataset on-premises, and asynchronously back up the data to AWS.

The third type is the Tape Gateway. It is essentially a cloud-based Virtual Tape Library or VTL. Just like File Gateway and Volume Gateway, this one also uses Amazon S3. You also have the option to store the archived tapes in Amazon S3 Glacier or Amazon S3 Glacier Deep Archive. Your on-premises backup applications can connect to your tape gateway as iSCSI devices. It helps you reduce costs of eliminating the use of physical backup tapes while preserving the existing investment of your on-premises backup applications and workflows.

Storage Gateway is suitable for building a hybrid cloud storage infrastructure, where your on-premises storage systems are actively sending data to AWS. If you are planning to totally migrate your data to the cloud and decommission your on-premises storage, it is better to use the AWS DataSync service instead.

Moving Data From AWS Storage Gateway to Amazon S3 Glacier

We already know that you can transition objects in Amazon S3 to a different storage tier such as Amazon S3 Glacier using lifecycle policies. What you might not know is that you can also move data from AWS Storage Gateway to Amazon S3 Glacier. AWS Storage Gateway is a service that connects your on-premises access to virtually unlimited storage with S3. You just need the AWS Storage Gateway VM or physical device to act as a literal gateway. Data transfers are encrypted with SSL so you can rest assured that the transport is secure.

There are three types of Storage Gateway types that you can use: **File Gateway**, **Volume Gateway**, and **Tape Gateway**. File Gateway lets you access your S3 buckets via a file interface using SMB or NFS protocol, as if S3 was a file share you can mount. Volume Gateway provides an iSCSI target, which enables you to create block storage volumes and mount them as iSCSI devices. You can take snapshots of your volumes and use them to create new EBS volumes. Lastly, Tape Gateway is a cloud-based Virtual Tape Library. Your backup application can read data from or write data to virtual tapes by mounting them to virtual tape drives using the virtual media changer. Tape Gateway is usually used for archival purposes.

In this section, we'll be discussing File Gateway and Tape Gateway, which are the two services that can store data to Amazon Glacier.

Tape Gateway has the more obvious explanation. Since Tape Gateway is primarily used for archival, your archived tapes are sent to S3 Glacier or S3 Glacier Deep Archive, but not immediately. Data on your virtual tapes are first stored in a virtual tape library in S3 Standard while your backup application is writing data to tapes. After you eject the tapes from the backup application, they are then archived to S3 Glacier or S3 Glacier Deep Archive depending on what you choose. You can also store your tapes in S3 Glacier first then move them to Deep Archive later on.

File Gateway has an indirect approach to storing data in S3 Glacier. As mentioned earlier, File Gateway presents S3 via a file interface. You can move files between your application and S3 easily through this interface. File Gateway can use S3 Standard, S3 Standard-IA, or S3 One Zone-IA storage classes. Once you have stored your files in your S3 bucket, you can configure a bucket lifecycle policy to move your files to S3 Glacier or S3 Glacier Deep Archive. However, doing so will prevent you from retrieving the file through File Gateway again. You must restore the file from S3 Glacier first before you can retrieve it.

References:
https://aws.amazon.com/storagegateway/faqs/
https://tutorialsdojo.com/aws-storage-gateway/

Integrating AWS Storage Gateway to an Active Directory

AWS Storage Gateway File Gateway allows you to create an SMB file share that can be mounted on your Windows instances. You can configure either Microsoft Active Directory (AD) or guest access for authentication. To set up your SMB file share Microsoft AD access settings, perform the following:

1. Go to the Active Directory settings of your SMB file share.
2. Enter the Domain Name of the domain that you want the gateway to join. You can connect to your self-managed AD (running in the cloud or on-prem) or connect to AWS Directory Service.
3. Enter a set of domain credentials that has permissions to join a server to a domain.
4. You can optionally specify an organizational unit to place your SMB file share.
5. You can optionally indicate a set of domain controllers.
6. Finish the process by saving your changes.

Connecting your File Gateway file share to an Active Directory has many uses. First, the feature allows your users to authenticate with your AD before they can access the file share. Furthermore, you can create a list of AD users and groups that will have administrator rights to the file share. Lastly, you can provide a list of AD users or groups that you want to allow or deny file share access.

References:
https://docs.aws.amazon.com/storagegateway/latest/userguide/managing-gateway-file.html
https://tutorialsdojo.com/aws-storage-gateway/

Amazon Elastic Block Store (EBS)

Amazon EBS is a persistent block-level storage service that you usually attach to your Amazon EC2 instances. The acronym EBS stands for Amazon Elastic Block Store. Just like the EC2 Instance Store, this service is also a block storage type. However, the data in EBS is more persistent as it doesn't get lost easily. The files on your block store won't be affected even if your EC2 instance was stopped, restarted, or terminated.

A block store is also called an EBS volume, which you can mount or attach to your EC2 instances. An Amazon EBS volume is zonal in scope since technically, the underlying physical components of the volume only exist in a single Availability Zone. You can attach an EBS volume to any EC2 instances in the same Availability Zone only. This means you can't attach an EBS volume to an EC2 instance that is in a different AZ or region. Your data can also be encrypted at rest using AWS Key Management Service.

You can attach one or more EBS volumes in a single EC2 instance. Each instance has a root device volume that contains the system image used to boot the instance. The root EBS volume of a running EC2 instance can also be replaced. This allows you to restore your instance to its launch state or to a specific snapshot, without having to stop your instance.

Amazon EBS is suitable for a variety of workloads such as databases, enterprise applications, big data analytics engines, file systems, media workflows, and many other use cases. It allows you to store and retrieve your data with high throughput and low latency. Unlike Amazon S3, you don't have to send an API request over the public Internet or via a VPC Endpoint to fetch your data from Amazon EBS.

This is because your EC2 instance and your EBS volume are logically attached together and are both located within a single Availability Zone. It is possible that the underlying data center, which hosts your EC2 instance and EBS volume, is located within the same city or geographic area. Due to this proximity, Amazon EBS can provide low latency read or write access to your data.

Block storage is an integral technology that mainly operates on the hardware level. It's important to know the core concepts of block-level storage in Amazon EBS and how it is different from file-level storage or object storage. This will help you better understand the differences between the various AWS storage services available.

Basically, a block is a sequence of bytes or bits that represents your data. When you store a file, your system splits it into multiple data blocks, each with the same maximum length. The data length is also known as the block size. So if you have a block size of 4 kilobytes and your file is 8 kilobytes in size, then you'll have 2 blocks with 4 kilobytes each.

SSD vs HDD Type Volumes

On a given volume configuration, certain I/O characteristics drive the performance behavior for your EBS volumes. SSD-backed volumes, such as General Purpose SSD (gp2, gp3) and Provisioned IOPS SSD (io1,io2), deliver consistent performance whether an I/O operation is random or sequential. HDD-backed volumes like Throughput Optimized HDD (st1) and Cold HDD (sc1) deliver optimal performance only when I/O operations are large and sequential.

In the exam, always consider the difference between SSD and HDD as shown on the table below. This will allow you to easily eliminate specific EBS-types in the options which are not SSD or not HDD, depending on whether the question asks for a storage type which has ***small, random*** I/O operations or ***large, sequential*** I/O operations.

FEATURES	SSD Solid State Drive	HDD Hard Disk Drive
Best for workloads with:	***small, random*** I/O operations	***large, sequential*** I/O operations
Can be used as a bootable volume?	Yes	No
Suitable Use Cases	- Best for **transactional workloads** - Critical business applications that require sustained IOPS performance - Large database workloads such as MongoDB, Oracle, Microsoft SQL Server and many others…	- Best for ***large streaming workloads*** requiring consistent, fast throughput at a low price - Big data, Data warehouses, Log processing - Throughput-oriented storage for large volumes of data that is ***infrequently*** accessed
Cost	moderate / high 	low 
Dominant Performance Attribute	IOPS	Throughput (`MiB/s`)

Provisioned IOPS SSD (io1,io2) volumes are designed to meet the needs of I/O-intensive workloads, particularly database workloads, that are sensitive to storage performance and consistency. Unlike gp2, which uses a bucket and credit model to calculate performance, an io1 volume allows you to specify a consistent IOPS rate when you create the volume, and Amazon EBS delivers within 10 percent of the provisioned IOPS performance 99.9 percent of the time over a given year. Provisioned IOPS SSD io2 is an upgrade of Provisioned IOPS SSD io1. It offers higher 99.999% durability and higher IOPS per GiB ratio with 500 IOPS per GiB, all at the same cost as io1 volumes.

Volume Name	General Purpose SSD		Provisioned IOPS SSD	
Volume type	gp3	gp2	io2	io1
Description	General Purpose SSD volume that balances price performance for a wide variety of transactional workloads	General Purpose SSD volume that balances price performance for a wide variety of transactional workloads	High performance SSD volume designed for **business-critical** latency-sensitive applications	High performance SSD volume designed for latency-sensitive transactional workloads
Use Cases	Virtual desktops, medium sized single instance databases such as MSFT SQL Server and Oracle DB, low-latency interactive apps, dev & test, boot volumes	Boot volumes, low-latency interactive apps, dev & test	Workloads that require sub-millisecond latency, and sustained IOPS performance or more than 64,000 IOPS or 1,000 MiB/s of throughput	Workloads that require sustained IOPS performance or more than 16,000 IOPS and I/O-intensive database workloads
Volume Size	1 GB – 16 TB	1 GB – 16 TB	4 GB – 16 TB	4 GB – 16 TB
Durability	99.8% – 99.9% durability	99.8% – 99.9% durability	99.999%	99.8% – 99.9%

Max IOPS / Volume	16,000	16,000	64,000	64,000
Max Throughput / Volume	1000 MB/s	250 MB/s	1,000 MB/s	1,000 MB/s
Max IOPS / Instance	260,000	260,000	160,000	260,000
Max IOPS / GB	N/A	N/A	500 IOPS/GB	50 IOPS/GB
Max Throughput / Instance	7,500 MB/s	7,500 MB/s	4,750 MB/s	7,500 MB/s
Latency	single digit millisecond	single digit millisecond	single digit millisecond	single digit millisecond
Multi-Attach	No	No	Yes	Yes

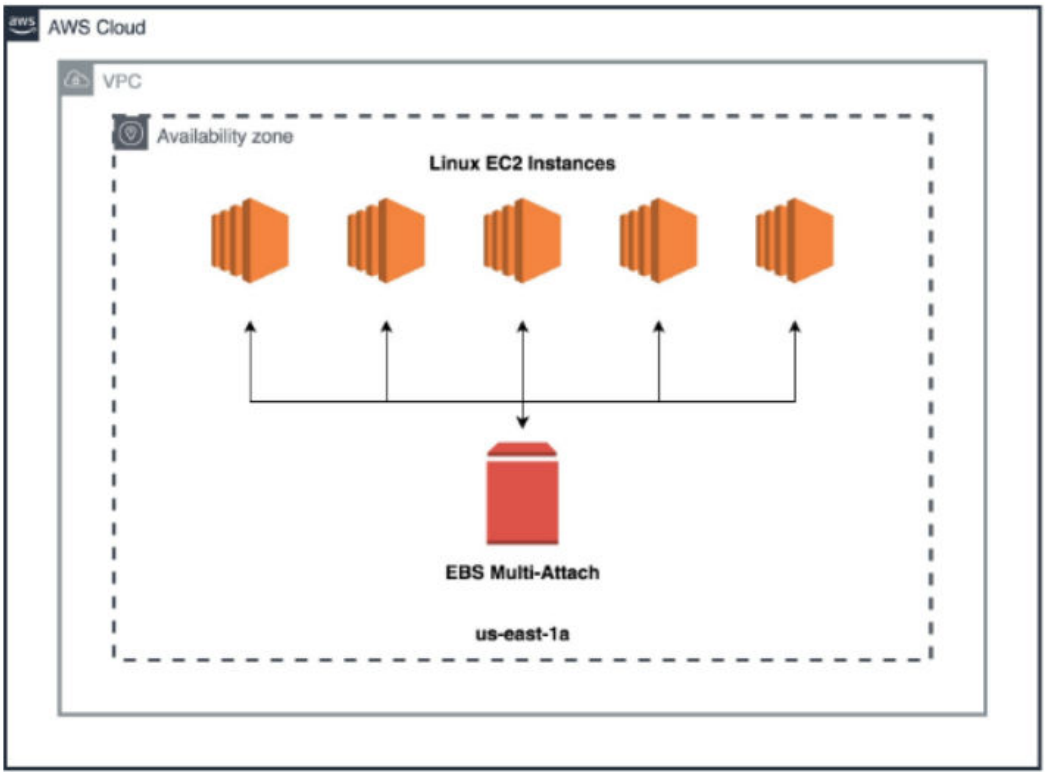
Volume Name	Throughput Optimized HDD	Cold HDD
Volume type	st1	sc1
Description	Low cost HDD volume designed for frequently accessed, throughput-intensive workloads	Throughput-oriented storage for data that is infrequently accessed Scenarios where the lowest storage cost is important
Use Cases	Big data, data warehouses, log processing	Colder data requiring fewer scans per day
Volume Size	125 GB – 16 TB	125 GB – 16 TB
Durability	99.8% – 99.9% durability	99.8% – 99.9% durability
Max IOPS / Volume	500	250
Max Throughput / Volume	500 MB/s	250 MB/s
Max IOPS / Instance	260,000	260,000
Max IOPS / GB	N/A	N/A
Max Throughput / Instance	7,500 MB/s	7,500 MB/s
Multi-Attach	No	No

Amazon EBS Multi-Attach Feature

Our understanding on Amazon EBS volumes is that they are virtual block devices that need to be attached to an Amazon EC2 instance before they can be used. While this is true, did you know that there is a type of EBS volume that you can attach to many EC2 instances simultaneously? Amazon EBS Provisioned IOPS (io1 and io2) volumes are currently the types that support EBS Multi-Attach. Multi-Attach lets you share access to an EBS data volume between up to 16 Nitro-based EC2 instances within the same Availability Zone (AZ). Each attached instance has full read and write permissions to the shared volume.

EBS Multi-Attach is primarily used with Amazon Linux instances. You may also use Multi-Attach with Windows instances, however, Windows does not recognize the data on the volume that is shared between the instances, which can result in data inconsistency. The Multi-Attach feature is not enabled by default. You will have to enable it during volume creation or modify your volume when it has been created already.

Multi-Attach volumes can't be created as boot volumes. Also, for io1 volumes, Multi-Attach can't be disabled once enabled. You can disable Multi-Attach for io2 volumes but only if it is attached to no more than one instance. If you'd like to modify the volume type of a Multi-Attach enabled volume, you must first disable the feature. Lastly, Multi-Attach enabled volumes are deleted on instance termination if the last attached instance is terminated and if that instance is configured to delete the volume on termination. If the volume is attached to multiple instances that have different delete on termination settings, the last attached instance's setting determines the delete on termination behavior.



AWS sometimes creates solutions that draw a fine line between one service and another to use for your needs. In this case, EBS Multi-Attach closely resembles Amazon EFS in that you can create shared file systems that multiple instances can use concurrently.

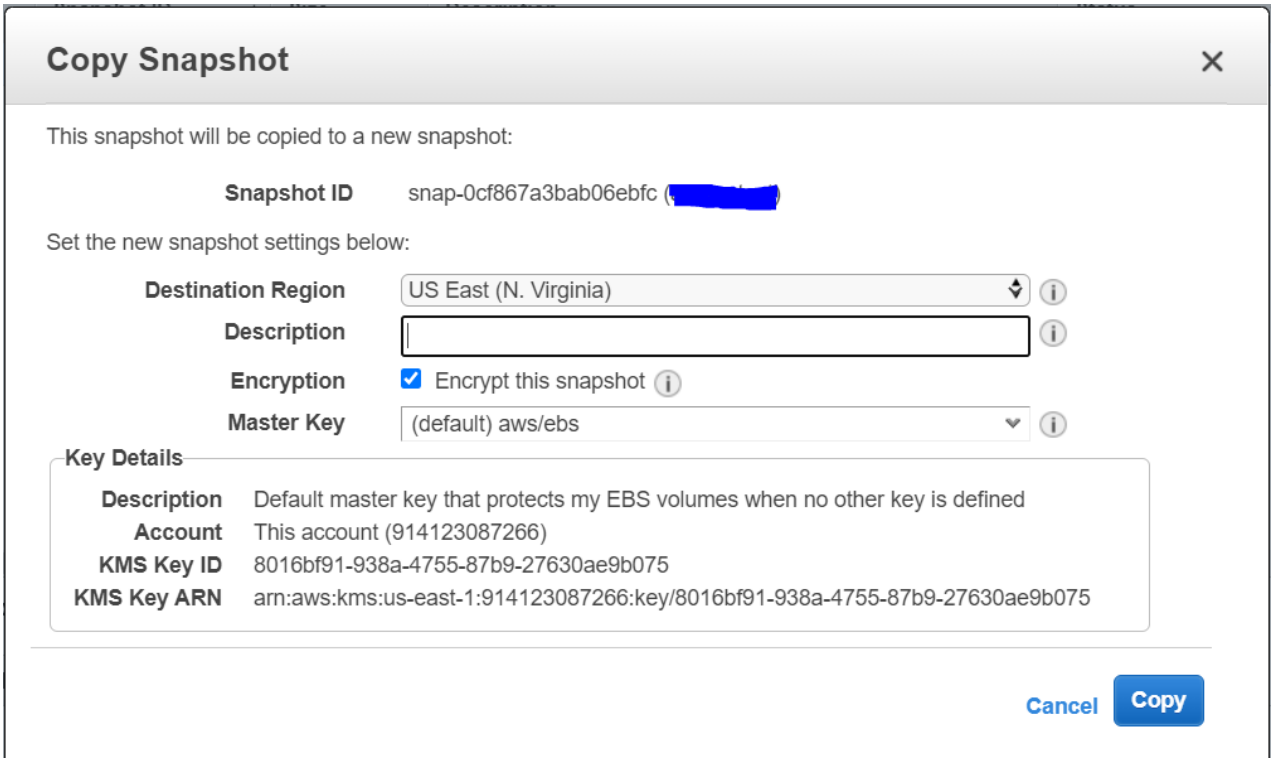
In the exams, whenever you are made to choose between EBS Multi-Attach and Amazon EFS, recall the limitations of EBS Multi-Attach. An example is that Multi-Attach enabled volumes do not support I/O fencing. Your applications must provide write ordering for the attached instances to maintain data consistency. Amazon EFS is more appropriate when you need a filesystem that needs to be concurrently accessed by hundreds to thousands of instances, and more so when these instances belong to different Availability Zones. There are also no limitations to the instance types that can mount EFS filesystems. EFS automatically scales in storage size and performance, unlike in EBS where manual intervention is required. Lastly, Amazon EFS by default provides traditional file permissions model, file locking capabilities, and hierarchical directory structure.

References:
https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-volumes-multi.html
https://tutorialsdojo.com/amazon-ebs-multi-attach/

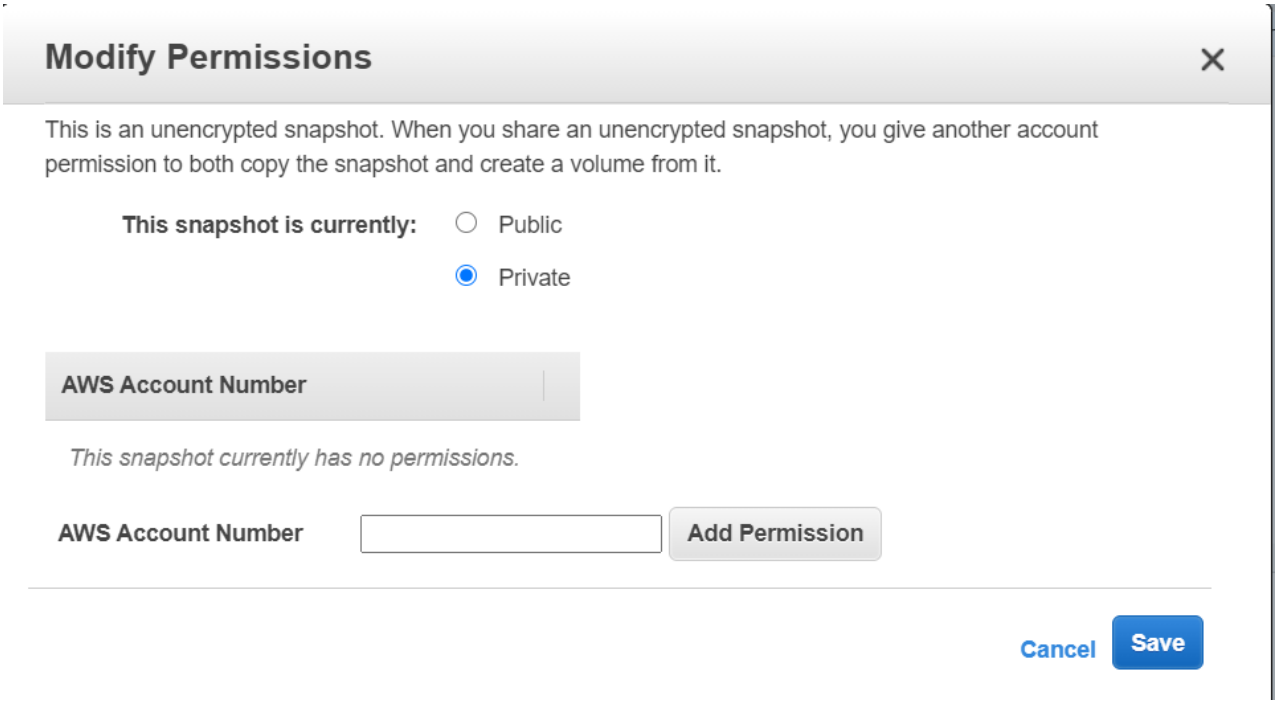
Amazon EBS Copy Snapshots

EBS Snapshots are a very simple but efficient way of taking backups of your EBS volumes in AWS. Snapshots are part of almost every disaster recovery plan, so making sure that they are available and usable when you need them is necessary. Your point-in-time snapshots are kept durably in Amazon S3, which we know is a service that's designed for durability. However, if one needed to restore a snapshot in another region or another AWS account, he/she would not be able to do so. An EBS snapshot is only available in the AWS Region it was created in, and only the account owner has access to the snapshot. If a regional disaster were to occur, you won't be able to use your EBS snapshots to rebuild your infrastructure in your DR region, not unless you copied them over previously.

Amazon EBS lets you copy snapshots from one region to another, or from within the same region. Amazon S3 server-side encryption protects a snapshot's data in transit during a copy operation. Copying snapshots lets you add or modify the encryption settings of that snapshot. This means that you can create copies of a backup with each having a different encryption key.



If you would like another account to be able to copy your snapshot, you can either modify the snapshot permissions to provide access to that account or make the snapshot public so that any AWS account can copy it.



Using snapshot copy within a single account and region does create a new copy of the data and therefore is cost-free as long as the encryption status of the snapshot copy does not change. Though if you copy a snapshot to a new region, or encrypt it with a new encryption key, the resulting snapshot is a complete, non-incremental copy of the original snapshot, which will incur additional storage costs. When you modify the encryption settings during your snapshot copy operation, you must ensure that the target account and/or target instance has permissions to use the encryption key.

Some use cases of copying snapshots include:

1. Regional disaster recovery
2. Data migration
3. Creating a base volume for different applications
4. Create a new volume with new encryption settings
5. Data retention and compliance requirements

References:

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-copy-snapshot.html
https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-modifying-snapshot-permissions.html

Amazon Elastic File System (EFS)

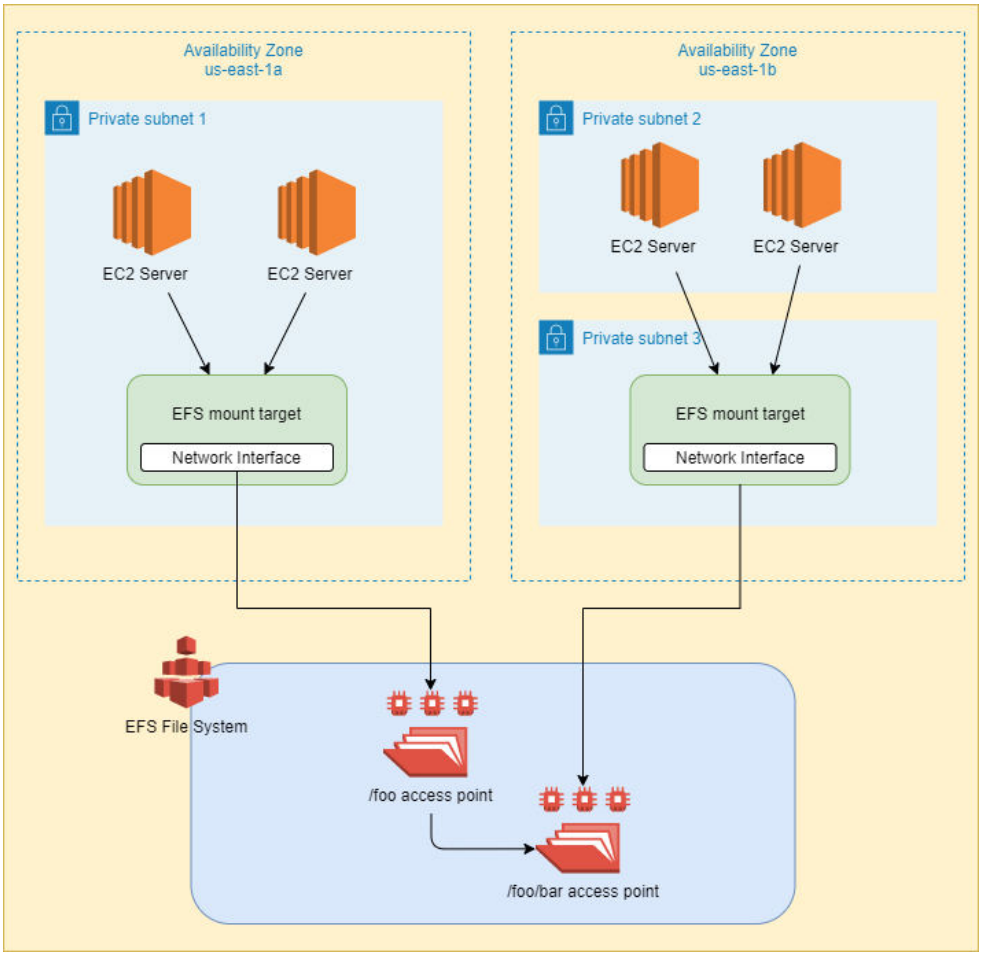
Amazon Elastic File System or Amazon EFS is a scalable shared file storage solution. It provides a POSIX-compliant shared file system that can be simultaneously accessed by multiple Amazon Linux EC2 instances in different Availability Zones. Amazon EFS is using the Network File System protocol or NFS, and as such, you have to mount the EFS file system to Linux EC2 instances or to your on-premises servers – just like a regular network file share. Amazon EFS only supports Linux servers and can't be used by Windows-based EC2 instances.

In Amazon EFS, you can choose between a Standard or an Infrequent Access storage class. The Standard storage class is best for active file system workloads while the latter is a cheaper storage class that's cost-optimized for infrequently accessed files. Similar to Amazon S3, you can also use a lifecycle policy to automatically move your data from the Standard class to the Infrequent Access storage class.

How To Mount An Amazon EFS File System

Before we dive in on how to mount an EFS file system, let's first go through what composes an EFS file system. Each file system has its own unique identifier, creation token, creation time, file system size in bytes, number of mount targets created for the file system, and the file system lifecycle state. To access your file system from a Linux EC2 instance, ECS container or a Lambda function, you must create mount targets in your VPC. When creating a mount target, you must indicate the Availability Zone at which the mount target will be created and add security groups to control access to your file system. Once done, you will be provided an IP address and a DNS name which you can use in your mount commands.

Another file system property you should know is your access point. An access point applies an operating system user, group, and file system path to any file system request made using the access point. Think of it as the directory where your requests are routed to, and this directory enforces specific access permissions similar to any Linux subdirectory. Access points ensure that an application always uses the correct operating system identity and the correct directory when reading from or writing to the file system.



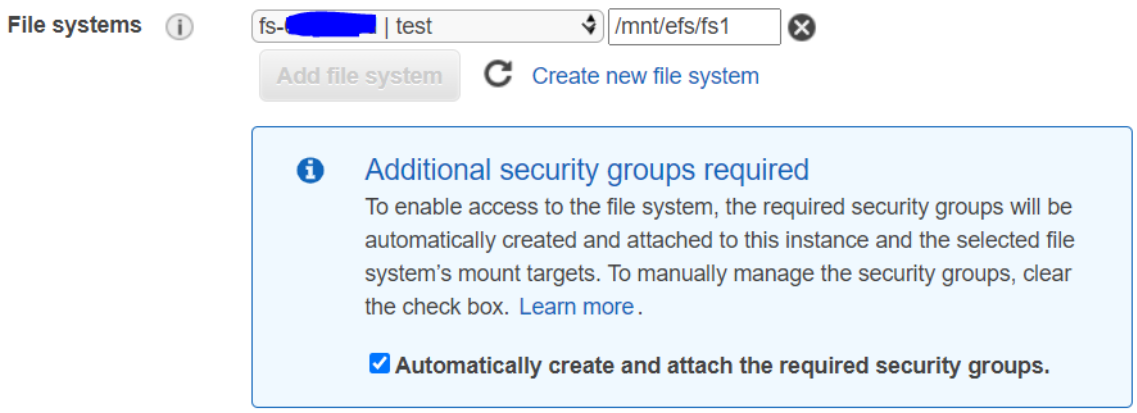
When mounting an EFS file system onto a Linux EC2 instance, the primary tool for this job is the **Amazon EFS mount helper**. To use the mount helper, you simply need to provide the following:

1. The file system ID of the EFS file system to mount
2. An Amazon EFS mount target

You may use any mount target, but if your EC2 instance is running in an AZ different from the mount target, you will incur data transfer charges. You might also experience increased latencies for file system operations. Furthermore, there are multiple ways to mount a mount target:

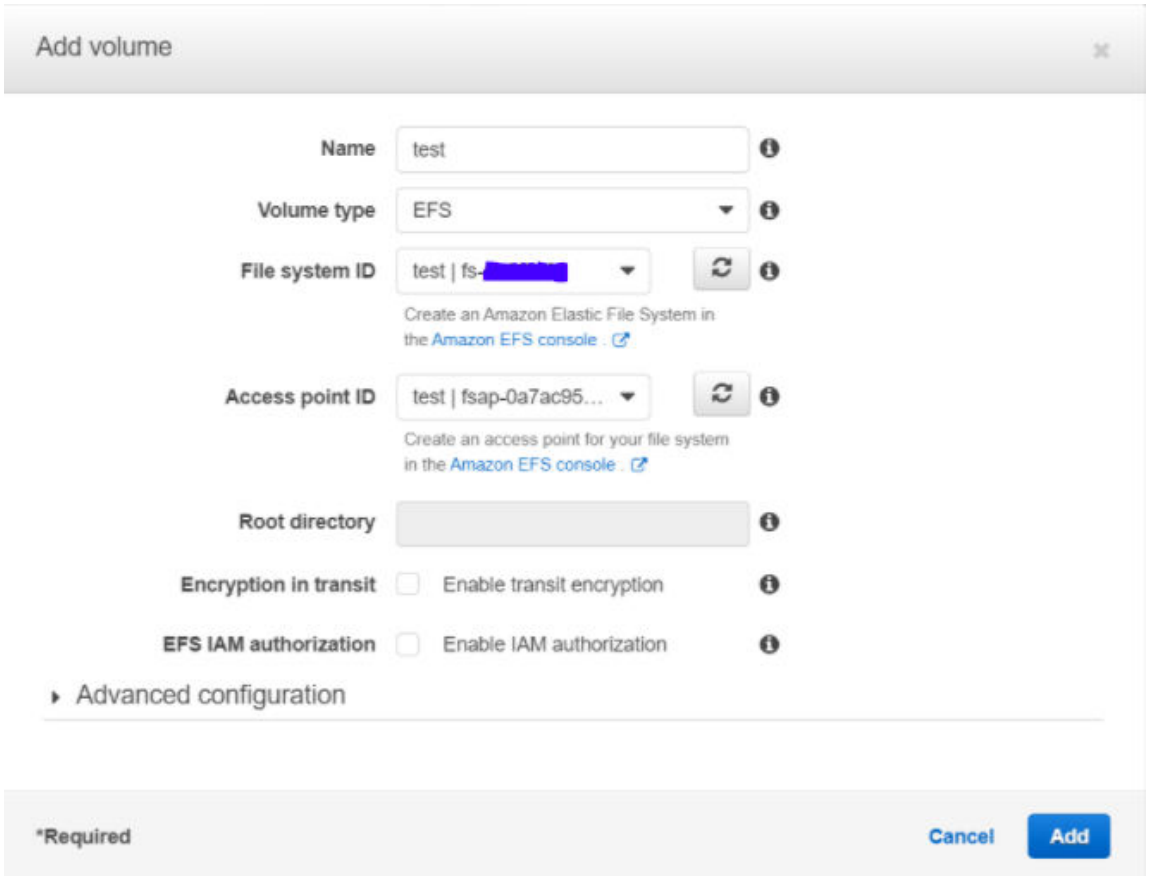
1. You can mount your target as is after you SSH into your instance using the mount command.
2. You can mount your target with a TLS parameter to enable encryption in-transit.
3. You can mount your target with IAM authorization (instance profile or named profile).
4. You can specify an EFS access point in your mount parameters.

If you prefer to mount your file system immediately at instance launch, you can specify in the configuration details the file system you wish to mount and the mount target that your EC2 instance will use. You can also automatically remount your filesystem after reboots by adding your mount command in /etc/fstab.



Lastly, if you would like to mount your file system without having to SSH into an instance or into multiple EC2 instances, you can use AWS Systems Manager Run Command to execute a shell script for you, and just specify the targets of the script.

For ECS containers and Lambda functions, mounting an EFS file system is as easy as specifying mount points in the ECS task definition's *Add volume* or Lambda function configuration.



File system

You can associate an existing Amazon Elastic File System (Amazon EFS) file system with your function. Visit the Amazon EFS console to create a new file system.

EFS file system

Choose an existing EFS file system to use with your Lambda function.

test fs-[redacted]
arn:aws:elasticfilesystem:us-east-1:[redacted]:file-system/fs-[redacted]
Owner: [redacted] Throughput Mode: bursting

Access point

An access point that is used to mount a network file system and integrates with IAM to control access.

test fsap-0a7ac952e4d13d2b3
arn:aws:elasticfilesystem:us-east-1:[redacted]:access-point/fsap-0a7ac952e4d13d2b3
POSIX uid: 0 POSIX gid: 0 Remote path: /foo/bar

Local mount path

Only absolute paths are supported.

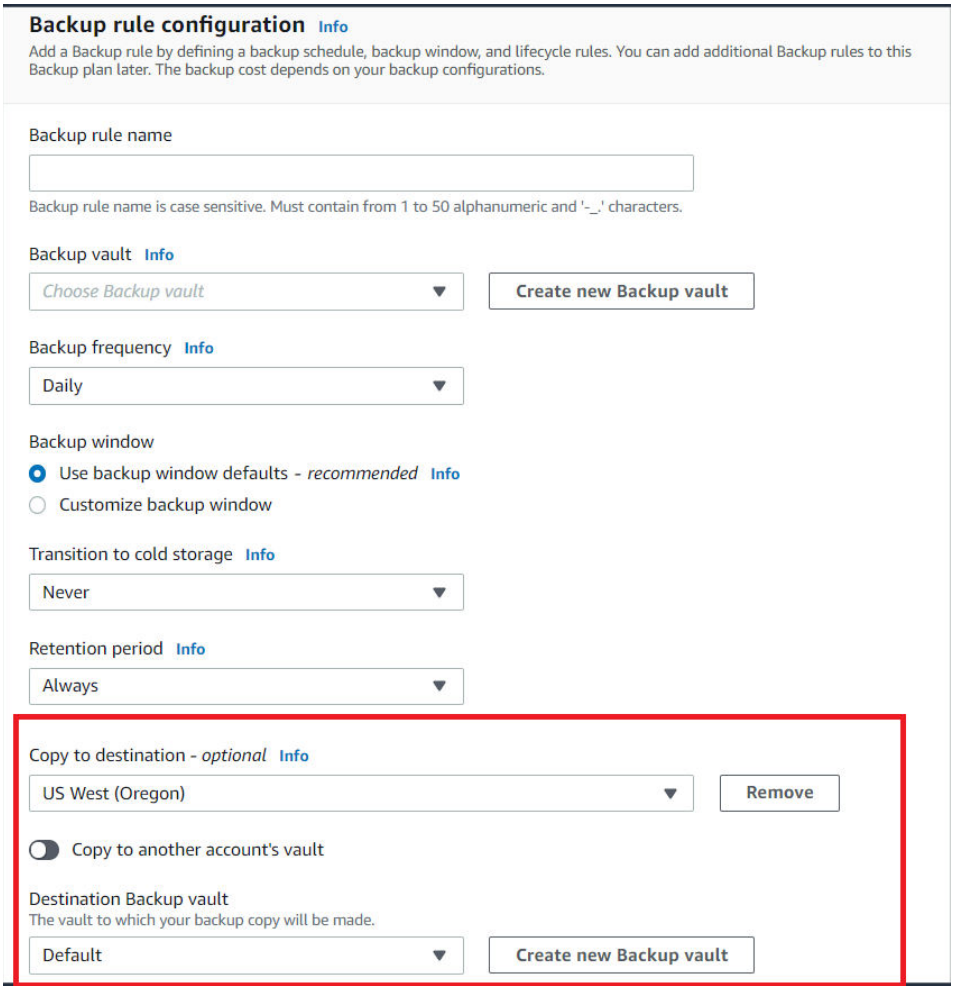
/mnt/

References:
https://docs.aws.amazon.com/efs/latest/ug/how-it-works.html#how-it-works-implementation
https://docs.aws.amazon.com/efs/latest/ug/mounting-fs.html
https://tutorialsdojo.com/amazon-efs/

EFS-to-EFS Regional Data Transfer

There are times when you need to copy over some data from one AWS Region to another. Your reasons may be for DR purposes or data retention policies imposed by your organization. Nevertheless, in AWS, there are usually straightforward ways to do so. For example, for EBS volumes, you can create a snapshot of your volume and copy it over to your destination region. For S3 objects, you simply create a new bucket in your destination region and configure replication in the origin bucket. But for Amazon EFS, there is no native feature to handle this process. You need the help of other AWS services to successfully migrate your EFS data from one region to another. In this deep dive, we'll be taking a look at the services that will help you do so.

If your goal is to recreate an entire file system in another region, you can use **AWS Backup** to take a backup of your EFS file system and have it copy the backup over to a destination region. During your initial backup, AWS Backup takes a full copy of your entire file system and stores it in a durable vault. Succeeding backups on your file system are incremental, meaning that only changes made after your latest backup will be taken. AWS Backup is able to backup your file system no matter the storage class you are using, but restoring a backup restores your files to the Standard storage class. If you've configured your backup plan to copy backup files to another region then AWS Backup copies your backups to a destination vault in the other region. Other settings you can define for your backup plan include whether to transition your backups to cold storage to lower storage costs, and the retention duration of your backups.



If your goal is to migrate or replicate data from one EFS file system to another, then you can use AWS DataSync for this purpose. AWS DataSync is able to copy files between two EFS file systems even if they belong to different regions and/or AWS accounts. To start copying data using AWS DataSync, first deploy the DataSync agent as an EC2 instance inside a VPC with access to your source file system. Once you activate the DataSync agent using a web browser, you select Amazon EFS as your destination AWS storage, enter your file system details, and start moving data. One advantage of using AWS DataSync is that you can copy your files over a private AWS network. To do so, simply follow these steps:

1. Create a VPC peering connection between your source EFS VPC and destination EFS VPC.
2. Add a rule in the security group of your source and destination EFS that would allow them to communicate with each other.
3. Create a VPC endpoint for AWS DataSync in the region of the destination EFS.
4. Initialize a DataSync Agent and choose the VPC endpoint as your service endpoint.
5. Start the agent and begin a transfer task.

References:

https://docs.aws.amazon.com/efs/latest/ug/awsbackup.html
https://aws.amazon.com/premiumsupport/knowledge-center/datasync-transfer-efs-cross-region/
https://aws.amazon.com/about-aws/whats-new/2019/05/aws-datasync-now-supports-efs-to-efs-transfer/
https://tutorialsdojo.com/amazon-efs/

Amazon EFS Storage Lifecycle

Amazon EFS is not exactly the cheapest storage service in AWS. If left unmanaged, it WILL hit you in the wallet. Although its price point is a reflection of its features and capabilities, we as Solutions Architects should always look for ways to lower cost. One such example is how you should optimize file storage in EFS. Amazon EFS has two storage classes: **Standard** (EFS-Standard) and **Infrequent Access** (EFS-IA). These storage classes are quite similar to the ones in Amazon S3. The Standard storage class offers a balance between cost and storage. This class is most suitable for storing frequently accessed files. You only need to pay for storage consumed by files in this class. The Infrequent Access storage class, on the other hand, brings you lower storage costs in exchange for retrieval fees. This class is most suited for files that you know won't be accessed very often. Although storage cost is lower in EFS-IA, overall costs can quickly ramp up if EFS-IA files are being accessed too often.

Lifecycle management policies control how your objects are stored in Amazon EFS. When enabled, lifecycle management migrates all your files that have not been accessed for a set period of time to the Infrequent Access storage class. You define the period of time from the selection below in your lifecycle policy:

- None
- 7 days since last access
- 14 days
- 30 days
- 60 days
- 90 days

Note that, as of the moment, you cannot set your own period. If in the exam there is a strict requirement that data should only be transitioned to IA storage after x number of days and x is not in the selection above, then consider your other options first.

To qualify for the transition to the IA storage class, files must at least be 128 KB in size. Files moved into the IA storage class remain there indefinitely. You can move files from the IA storage class back to the Standard storage class by copying them to another location on your file system. If you want your files to remain in the Standard storage class, disable Lifecycle Management by choosing None in the lifecycle policy and then copy your files to another location on your file system.

Amazon FSx

Amazon FSx is a family of file system services in AWS. It simplifies the process of launching, running, and scaling feature-rich, high-performance file systems in the AWS cloud in just a few clicks. Amazon FSx handles the manual task of hardware provisioning, patching, and creating backups – allowing you to focus on your applications and end users.

This service lets you choose what type of file system that you want to launch. You can create a cloud file system for NetApp ONTAP, OpenZFS, Windows File Server, and for Lustre. The file system that you choose should be based on your expertise and familiarity as well as the file system's feature sets, performance profiles, and data management capabilities to meet your demanding workloads. You can choose from the following Amazon FSx file system types to meet your requirements:

- Amazon FSx for Lustre
- Amazon FSx for Windows File Server
- Amazon FSx for NetApp ONTAP
- Amazon FSx for OpenZFS

Amazon FSx for Lustre is quite similar to Amazon EFS. It is also a POSIX-compliant shared file system that only supports Linux servers. The word “Lustre” actually refers to the open-source file system that this service is using. Lustre is a parallel file system used for large-scale cluster computing. The name Lustre is basically a combination of the word Linux and cluster. Amazon FSX for Lustre is primarily used for High-Performance Computing, machine learning, or HPC applications that need high-performance parallel storage for frequently accessed ‘hot' data. This service can provide a throughput of hundreds of gigabytes per second and millions of IOPS, to support your demanding workloads. You can mount an Amazon FSX for Lustre file share to your EC2 instances or your containers. Amazon FSx for Luster can also be connected to your Amazon EKS cluster using the Container Storage Interface (CSI).

Amazon FSx for Windows Server is essentially a fully managed Microsoft Windows file server. Unlike Lustre which is Linux-based, this service is backed by a fully native Windows file system. There are a lot of Microsoft-based technologies that you can integrate with this service. You can access this file share using the Server Message Block protocol or SMB. This protocol is commonly used by Windows servers. You can also integrate your existing Microsoft Active Directory to provision file system access to your users. The Amazon FSx for Windows Server can be used as shared file storage for your Microsoft SharePoint, Microsoft SQL Server database, Windows Container, or any other Windows-based applications.

Amazon FSx for Lustre vs Amazon FSx for Windows File Server

	Amazon FSx for Lustre	Amazon FSx for Windows File Server
Short description	A high-performance, scalable storage service powered by Lustre.	A fully managed, highly reliable, and scalable file storage that is accessible over the Server Message Block (SMB) protocol. Lowest cost SMB file server in AWS.
Use cases	Machine learning, high performance computing (HPC), video rendering, and financial simulations	For applications requiring use of Windows shared storage through SMB protocol and requiring support for other Windows features such as AD integration or a lift-and-shift replacement for Sharepoint for example.
Accessible from these sources	Intended for thousands of concurrent access from Linux-based instances and devices, whether in AWS or on-premises. FSx for Lustre integrates with Amazon EC2, AWS Batch, Amazon EKS, and Amazon Parallel Cluster.	Can be concurrently accessed by thousands of Windows, Linux, and MacOS compute instances and devices, whether in AWS or on-premises. Compute instances include Amazon EC2, Amazon ECS, VMware Cloud on AWS, Amazon WorkSpaces, and Amazon AppStream 2.0 instances.
Deployment options	Scratch file systems - designed for temporary storage and shorter-term processing of data. Data is not replicated and does not persist if a file server fails. Persistent file systems - designed for longer-term storage and workloads. The file servers are highly available, and data is automatically replicated within the Availability Zone (AZ) of the file system. The data volumes attached to the file servers are replicated independently from the file servers to which they are attached.	Only has persistent file systems. Can run in single AZ or multi-AZ.
Storage options	SSD storage for latency-sensitive workloads or workloads requiring the high IOPS/throughput. HDD storage for throughput-focused workloads that aren't latency-sensitive. Amazon FSx also provides a fast, in-memory cache on the file server.	
Managing	You can increase your file system's storage	Each file system can have up to 64 TB of data.

storage capacity	capacity every six hours. Throughput scales linearly as you increase storage.	Amazon FSx grows the storage capacity of your existing file system without any downtime impact to your applications and users.
How to mount	Install the open-source Lustre client on your Linux instance. Once it's installed, you can mount your file system using standard Linux commands.	In Windows, use the “Map Network Drive” feature to map a drive letter to a file share on your FSx file system. In Linux, use the cifs-utils tool to mount your file share.
Backups	Amazon FSx takes daily automatic backups of your file systems, and allows you to take manual backups at any point. Backups are incremental. Default backup retention is 7 days. You can only take a backup of a Lustre file system that has persistent storage and is not linked to an S3 bucket.	
Security	FSx for Lustre always encrypts your file system data and your backups using keys you manage through AWS KMS. Amazon FSx encrypts data-in-transit using SMB Kerberos session keys.	
	Encrypts data-in-transit when accessed from supported EC2 instances.	Encrypts data-in-transit using SMB Kerberos session keys.
Extra features	You can link your Lustre file system to an Amazon S3 bucket. You can also create multiple Lustre file systems linked to the same S3 bucket.	Amazon FSx for Windows File Server works with Microsoft Active Directory (AD) so you can easily integrate existing AD-based user identities. It also provides standard Windows permissions for files and folders. Data Deduplication is a feature in Windows Server that reduces costs by storing redundant data only once.

References:
https://aws.amazon.com/fsx/lustre/faqs
https://aws.amazon.com/fsx/windows/faqs/
https://tutorialsdojo.com/amazon-fsx/

Amazon Relational Database Service (RDS)

The Amazon Relational Database Service or Amazon RDS is a relational database that is managed by both you and AWS. It allows you to run various database engines such as Microsoft SQL Server, MySQL, MariaDB, PostgreSQL, Oracle, Aurora, and other DB engines. You can deploy your RDS databases using AWS CloudFormation, AWS Management Console, AWS CLI, or via the RDS APIs. This service removes the time-consuming tasks of hardware provisioning, patching, backups, and maintenance from you.

However, it is not completely managed by AWS, unlike DynamoDB, and other fully managed databases. There are certain things in your RDS database that you can control. For example, you can choose the underlying EC2 instance of your choice that will be used by your RDS database. You can select what type of DB instance size, DB instance type, and total storage your database will have. These attributes can be modified to meet your changing requirements. This makes it easy for you to upgrade or downgrade the underlying instance size and type of your RDS database based on your current workloads. You can even purchase a Reserved DB instance to further lower down your operating costs. On the networking side of things, you're the one who will choose the specific Availability Zone where your database will be hosted, as well as its associated security groups.

Actually, you can host your own database on your Amazon EC2 instance instead of using Amazon RDS. The issue here is the management overhead that it entails. You will be responsible for manually patching your database, scaling its storage, regularly taking database backups, ensuring high availability, and doing other maintenance tasks. Setting up master-slave database replication would be a manual process too as well as the regular task of monitoring the CPU, memory, and storage usage of your database. With Amazon RDS, all of these painstaking activities can be significantly reduced, or even eliminated, since AWS will handle these for you.

In a self-hosted database running in an EC2 instance, you can directly connect to the instance via SSH or RDP to fully control and configure your database. You are free to install and run different databases on your virtual machine and optimize it as you wish. Once logged in, you can manually open and modify the database configuration files to meet your requirements. A configuration file can be the my.cnf file of your MySQL database. This config file is usually located inside the `/etc/` or `/etc/mysql` folder in Linux. There are various parameters that you can define in your configuration file to customize your database. For example, you can set up a master-slave configuration with a read-only MySQL database by adding the "`read_only`" parameter in the my.cnf configuration file.

If you're using an MS SQL Server database, its config file is the `ConfigurationFile.ini` which is located in the DB directory that you define; and if you're using an Oracle database, then you would probably edit the ORA file to properly set up your system. In Amazon RDS, there's a different way of changing the configuration of your database.

Unlike a self-hosted database, you cannot directly access the underlying EC2 instance that your Amazon RDS database is using to change its configuration. You won't be able to establish an SSH or RDP connection to your

Amazon DB instance. However, you can use several features to accomplish this task, such as the DB Parameter Group and DB Options Group. A parameter group acts as a container for the engine configuration values that are applied to your DB instance. This group contains certain DB values that affect the behavior of your database, just as a regular, database configuration file does.

You also get to decide the actual time when RDS will apply the DB patches in its maintenance window. Again, RDS handles the actual installation of the recent DB patches automatically without any involvement on your part.

Using Amazon RDS is way better than running your self-managed database in an Amazon EC2 instance. Although you have full access to the virtual machine, you will be burdened in the actual implementation of a secure, highly available, and fault-tolerant database in your VPC. And of course, you are also responsible for applying the security patches of your operating system on a regular basis, which is used by your EC2 instance. The same goes for the latest patches needed for your database engine. These manual tasks entail a lot of time and effort to perform. Amazon RDS simplifies relational database administration tasks, so you don't have to do them yourself. It allows you to focus on optimizing your application rather than waste your time doing boring maintenance work. The process of patching and updating the underlying OS is simplified using RDS.

For example, you want to host a Microsoft SQL Server database in AWS. You can either use Amazon RDS or Amazon EC2 – but if you chose the latter, be prepared to manage your server and database engine yourself. You may even need to hire a Database Administrator to handle routine tasks – from provisioning, patching, backups, and optimization. With RDS, you don't have to worry about installing the latest security patches on your RDS instance. You just have to launch the database with the appropriate EC2 instance type for your workload and you're ready to go! And if the demand or the workload of your application changes, you can easily optimize your RDS database to meet your needs. Just as I mentioned earlier, there is only one drawback that you have to consider – unlike a self-hosted database in Amazon EC2, you cannot directly login or connect to the underlying EC2 instance that is used in your RDS database. You are also responsible for manually setting up or enabling the specific RDS features to optimize and scale your database.

You can deploy your RDS DB in two ways: either a Single-AZ or Multi-AZ deployment. A single-AZ is just one primary DB instance within a single Availability Zone. For Multi-AZ deployments, there are two DB instances launched – the primary instance in one AZ and the standby instance deployed in a different AZ. This multi-AZ design ensures high availability and it does so by writing the data on both the primary and the standby instance at the same time. This process is called: synchronous replication.

If an update SQL statement has been committed, the change will be done on both the primary and the standby instance simultaneously. The standby instance is also called a standby replica, and you can only deploy it in the same AWS Region where your primary DB instance is located. A standby instance is just on the standby in the event of a failure. So, if the primary DB instance fails, your database will still continue to run by failing over to the standby instance. This is also what happens in the event of an AZ outage. If you are planning to take a

backup or a snapshot of your database, you can use the standby instance instead of the primary instance. In this way, you don't have to suspend the database I/O activities that may affect your production environment.

Aside from using the standby replica, you can launch a Read Replica. A Read Replica is simply a DB instance that asynchronously replicates the data from the primary database. In asynchronous replication, RDS writes the data on the primary instance first before writing to the secondary instance. This is different from synchronous replication, in which the data is written on both the primary and secondary instance at the same time – which is why it's called synchronous in the first place!

Under the hood, an RDS Read Replica is just like setting the read_only parameter in the my.cnf configuration file of your MySQL database. Amazon RDS will set this specific parameter to 1, to ensure that the DB instance would not allow any changes to your database or to any of your tables. This read_only parameter can also be seen on the associated parameter group of your Read Replica.

You can create a Read Replica in the same or different AWS Region where your primary instance is hosted. The replication process usually takes about a few seconds to complete and it could be longer if your replica is placed in another AWS Region. A Read Replica is actually based on the native replication features of different database engines, like MySQL, MS SQL, Oracle, and the likes. This is perfect if you have an application with read-intensive workloads, like reporting tools or any business reporting processes. It allows you to offload the read requests from your primary DB instance to improve the database performance. You can also create globally redundant databases by launching cross-region read replicas to multiple AWS Regions.

Amazon RDS is suitable for applications that read or write constantly changing data, such as Online Transaction Processing applications or OLTP. An eCommerce website is a type of an OLTP application as it processes hundreds or thousands of online transactions per hour. A transaction can write or read data to and from the database. Amazon RDS is also ACID-compliant. ACID is actually an acronym for Atomic, Consistent, Isolated, and Durable. It ensures that each and every transaction is Atomic, Consistent, Isolated, and Durable to maintain the data integrity of your database. This allows Amazon RDS to process a large number of queries that involve multiple table-joins and complex queries.

It also has a feature called Amazon RDS Proxy that makes your applications more scalable, more secure, and more resilient to database failures. RDS Proxy is a fully managed, highly available database proxy for your RDS databases. It automatically connects your application to a new DB instance while preserving its application connections. In the event of a failover, RDS Proxy instantly routes the incoming requests directly to the new database instance; thus, minimizing application downtime.

Amazon RDS High Availability and Fault Tolerance

When it comes to production databases, architecting a highly available, fault tolerant database infrastructure is key in making sure that your operations continue to run smoothly in the event of a failure. Since we can easily launch new resources in the AWS cloud, and tear them down as easily too, it is always a good practice to create redundant infrastructure in every part of your system when applicable; and yes, that includes databases.

Amazon RDS is a managed relational database service that supports multiple database engines and versions. As you may know, different database engines have different ways of implementing high availability in a traditional sense. In Amazon RDS, these capabilities are further improved thanks to the innovations brought forth by AWS. Two concepts we'll touch on in relation to HA/FT are **Multi-AZ Deployments** and **Read Replicas**.

Amazon RDS Multi-AZ deployment creates and maintains a standby replica of your RDS DB instance in a different Availability Zone, effectively providing high availability and failover support for situations that would cause the primary database to go offline. Multi-AZ spans at least two Availability Zones within a single region. Your 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. Amazon RDS uses several different technologies to provide failover support. Multi-AZ deployments for MariaDB, MySQL, Oracle, and PostgreSQL DB instances use Amazon's failover technology. SQL Server DB instances use SQL Server Database Mirroring (DBM) or Always On Availability Groups (AGs). You should remember that you cannot use the standby replica to serve read traffic. For this purpose, you should use a read replica, which we'll discuss later on.

When converting a Single-AZ deployment to a Multi-AZ deployment, Amazon RDS takes a snapshot of the primary DB instance and then restores the snapshot into another AZ. RDS then sets up synchronous replication between your primary DB instance and the new instance. In the event of a planned or unplanned outage of your DB instance, RDS automatically switches to your standby replica. The time it takes for the failover to complete depends on the database activity and other conditions at the time the primary DB instance became unavailable. Also, the failover mechanism automatically changes the Domain Name System (DNS) record of the DB instance to point to the standby DB instance.

Amazon RDS Read Replicas let you scale out your DB instances across multiple AZs if you have a read-heavy database workload. You can create one or more replicas from the DB instance and use those replicas as a source for read operations. Read replicas can be created in the same AZ as the primary, in a different AZ but in the same region as the primary, or even in AZs in different regions if the RDS DB engine supports it. Data between your DB instance and read replicas are replicated asynchronously, so replicas might return stale data when you do a read on them. Another benefit of read replicas is that they store redundant copies of your data, so in the event of a failure on the primary DB instance, read replicas can be manually promoted to become standalone DB instances. When you promote a read replica, the DB instance is rebooted before it becomes available. Amazon RDS uses MariaDB, MySQL, Oracle, PostgreSQL, and Microsoft SQL Server DB engines'

built-in replication functionality to create the read replicas. MySQL and MariaDB perform logical replication, while Oracle, PostgreSQL and Microsoft SQL Server perform physical replication.

Similar to how Multi-AZ deployments are created, Amazon RDS takes a snapshot of your source DB instance and creates a read-only instance from the snapshot. RDS then uses asynchronous replication to update the read replica whenever there is a change to the primary DB instance. One requirement when creating read replicas is that automatic backups should be enabled. Take note that read replicas, by default, allow only read-only connections, but MySQL and MariaDB replicas can be made writable. Also, by default, a read replica is created with the same storage type as the source DB instance. However, you can create a read replica that has a different storage type from the source DB instance depending on the configuration. If you delete a source DB instance without deleting its read replicas in the same AWS Region, each read replica is promoted to a standalone DB instance.

Lastly, a few final reminders for RDS read replicas. You can't configure a DB instance to serve as a replication source for an existing DB instance. You can only create a new read replica from an existing DB instance. Read Replicas for MySQL and MariaDB support Multi-AZ deployments, so you can combine these two features to build a resilient disaster recovery strategy. Read Replicas DO NOT CACHE DATA. You'll need to add a caching layer using services such as Amazon Elasticache for example.

References:
https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Concepts.MultiAZ.html
https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_ReadRepl.html
https://tutorialsdojo.com/amazon-relational-database-service-amazon-rds/

Amazon RDS Security

Amazon RDS is a database service hosted in AWS, and it is always a given that you do everything you can to protect your databases and the data stored in them, no matter the platform. In this section, we'll discuss the many ways you can apply security for your Amazon RDS instances.

Network Isolation and VPC Security

Your RDS instances reside in a VPC, which is an isolated piece of network that you own and manage in AWS. No one can gain access to your VPC network unless you allow them to. Furthermore, there are many VPC security features available for you to use which are very important in securing your database network. It is a good practice to run your RDS instances in private subnets, and more to the fact that these subnets should be isolated from the rest of your system. This way, you can configure firewall rules (both security group and network acl) as well as routing rules that are dedicated for your databases. You can further secure your database access by using an IPsec VPN solution, and allow users to connect to the database through the VPN only. Lastly, you can set up intrusion detection systems to notify you immediately if there is a supposed threat

to your databases. Endpoint protection services such as AWS WAF may come in handy too since you can create WAF rules that mitigate SQL injection attempts.

Encryption At Rest

I'm sure this is a given, but you must encrypt your database to prevent others from easily reading your data. Amazon RDS encrypts your databases using keys you manage in the AWS Key Management Service (KMS). On a database instance running with Amazon RDS encryption, data stored at rest in the underlying storage is encrypted, as are its automated backups, read replicas, and snapshots. RDS encryption uses the industry standard AES-256 encryption algorithm to encrypt your data on the server that hosts your RDS instance. Amazon RDS also supports Transparent Data Encryption (TDE) for SQL Server (SQL Server Enterprise Edition) and Oracle (Oracle Advanced Security option in Oracle Enterprise Edition). With TDE, the database server automatically encrypts data before it is written to storage and automatically decrypts data when it is read from storage.

You can only enable encryption for an Amazon RDS DB instance when you create it, not after the DB instance is created. Once you have created an encrypted DB instance, you can't change the AWS KMS key used by that DB instance. If you'd like to encrypt an existing DB instance, take a snapshot of it and then create a copy of that snapshot, encrypt the copy, and restore it to have an encrypted version of your database. You also cannot disable encryption on RDS after you've enabled it on your DB instance. If you'd like to change encryption keys, export the data from your encrypted DB instance and import it to an unencrypted one.

Encryption In-Transit

Although you encrypt the data at-rest in your database, this is not enough as database traffic also contains your data. You should encrypt your network traffic to protect it from sniffers and malicious attacks. If someone were to get hold of your traffic data, who knows what they can do with them. They can attempt to intercept requests and send fake responses. Encrypt the communications between your application and your RDS DB instances using SSL/TLS. Amazon RDS creates an SSL certificate and installs the certificate on the DB instance when the instance is provisioned. Different DB engines have different ways for you to retrieve the SSL public key. Remember that in the network security section above, you can enforce HTTPS connections with security groups. You can also require your DB instance to only accept encrypted connections.

Access Controls

Amazon RDS is tightly integrated with AWS IAM which allows you to manage who can access and modify your RDS DB instances through IAM policies. In addition, you can tag your resources and control the actions that your IAM users and groups can do on your resources that have those tags. There is also the IAM database authentication feature which works with Aurora MySQL and Aurora PostgreSQL. With this authentication method, you don't need to use a password when you connect to a DB cluster. Instead, you use an authentication token.

When you first create a DB Instance, you need to enter the credentials of your master user account, which is used only within the context of Amazon RDS to control access to your DB Instances and will be provided database administrator privileges. Once you have created your DB Instance, you can connect to the database using the master user credentials and configure additional user accounts for your other users. You can also opt to disable the master account within the database settings (as a best practice), and use a separate account instead to perform administration work.

Logging and Monitoring

Although this is a given already, you should also enable logging for your database so you can monitor all activity that occurs within them. This will help you troubleshoot any security issues you might encounter in the future and prevent them from happening again. Logs that provide system activity are crucial in knowing the state of your databases and how well they are performing. Some users might even require them for auditing purposes, so be sure to store your logs somewhere durable such as Amazon S3 or Cloudwatch Logs.

References:
https://aws.amazon.com/rds/features/security/
https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/UsingWithRDS.html
https://tutorialsdojo.com/amazon-relational-database-service-amazon-rds/

Amazon Aurora

Amazon Aurora is a fully managed database service that is owned by AWS. It is also one of the database engine types available in Amazon RDS. It scales automatically, performs faster, and costs lower than other databases. Amazon Aurora is a type of relational database that is compatible with both MySQL and PostgreSQL DB engines. The applications and tools that you use with your MySQL or PostgreSQL databases are all compatible with Aurora, so you can use them interchangeably with little or no change.

Amazon Aurora is somehow similar to Amazon RDS in many ways but keep in mind that Amazon Aurora is technically within the Amazon RDS ecosystem. Aurora is not just a database engine since it has unique features that are only exclusive to Aurora databases.

It can automatically grow or scale its storage as needed. Unlike a regular RDS instance, Amazon Aurora is usually deployed as a database cluster. A cluster usually consists of a primary DB instance and multiple Aurora replicas or DB instances. These instances are like the standby or read replicas in Amazon RDS – but instead of calling it Amazon Aurora Multi-AZ deployments, it is simply called an Aurora “database cluster". By default, a cluster has a single-master configuration where applications can only write data to a single, master DB instance. In a multi-master cluster, all DB instances have read/write capability. There is even an option to launch a cluster with just a single primary DB instance, with no replicas.

Amazon Aurora performs faster than other databases. It can scale the computing components and storage of your database cluster automatically without any manual intervention. For data replication, the Aurora Replicas of your database cluster will typically lag behind the primary instance by a few milliseconds only – meaning, it can provide less than 1 second of read replication latency. The performance is almost the same even if the Aurora Replica is running on a different AWS Region, where the primary DB instance is located. In RDS, the replica lag takes a few seconds or even minutes. This is why Aurora is faster than Amazon RDS!

Amazon Aurora has a feature that allows you to create database endpoints. You can group the individual instances and associate them with a particular endpoint. You can launch a cluster, reader, custom, or instance endpoint and have the database autoscale based on your capacity requirements. You can also deploy an Aurora Serverless or an Aurora Global Database cluster.

Amazon Aurora Serverless is recommended for applications with sporadic usage workloads or with unpredictable usage. Since it's serverless, you only pay for the database resources that you consume, on a per-second basis. This provides you with a more cost-effective option to host the database tier of your applications.

You can use Amazon Aurora Serverless in various use cases. For instance, let's say you have a legacy application hosted on-premises that you want to migrate to AWS cloud. If there's a requirement to re-architect your application by using technologies that do not require any IT administration team to regularly manage your servers or clusters, then you can use a serverless stack in AWS. You can re-architect your application and turn

it into a microservices architecture. The application containers can be run using AWS Fargate and your database can be hosted in Amazon Aurora Serverless.

This is also helpful if your web application has sporadic usage patterns. For example, if your application has an extremely high usage at the beginning of each month, an unpredictable usage at the start of each week, and a moderate usage over the weekend. Although there's a certain pattern here, it's quite difficult to predict the demand and properly set a suitable instance size of your database with these changing usage. If you are looking for a cost-effective database platform that will not require any database modifications then you can use Amazon Aurora Serverless. This will allow your database to automatically scale the capacity up or down based on your application's needs. Using Amazon Aurora would be a perfect fit because doing this using a regular Amazon RDS would be quite difficult.

An Amazon Aurora Serverless database can also be used for applications with infrequent access patterns. It can scale down your database capacity if there's less incoming traffic coming in, without any manual intervention. If you need to migrate your on-premises database to AWS without having to worry about its particular database instance type, then you can migrate it to Amazon Aurora Serverless. This database type eliminates the need to manually modify your database instance type in anticipation of the changes in the number of your users or workloads in the future.

Let's now talk about the Amazon Aurora Global Database. This one is designed for globally distributed applications. It allows a single Aurora database to span multiple AWS regions that you define. It offers faster physical replication between Aurora clusters in various regions which eliminates the need to manually create cross-region Aurora Replicas yourself.

It spans at least two AWS Regions wherein the primary region supports an Aurora DB cluster that has one writer Aurora DB instance. The secondary region runs a read-only Aurora DB cluster that is made up entirely of Amazon Aurora Replicas. In addition, it can have up to five secondary regions to improve the availability of your database.

The Amazon Aurora Global Database is also suitable for implementing a multi-region disaster recovery plan for your enterprise application. It has a Recovery Point Objective, or RPO, of 1 second and a Recovery Time Objective, or RTO, of 1 minute. Let me repeat that – the Amazon Aurora Global Database provides an RPO of 1 second and an RTO of 1 minute! That's simply amazing! Those numbers are a game-changer in the industry, considering that most disaster recovery processes usually take a few minutes for their RTO at best. You will rarely see an RTO of 1 minute, let alone, 1 second! The same goes for its 1-minute RPO, which is an amazing number for avoiding data loss in your mission-critical applications.

Aurora Serverless Scaling

When you are using Amazon RDS or any relational database for your applications, and you notice that the database has varying usage patterns, wouldn't it be great having a database that automatically scales capacity based on demand? We already know that Amazon Aurora automatically scales its storage as your data grows, but how about CPU capacity and allowed number of connections? Amazon Aurora has a DB engine mode called Amazon Aurora Serverless, which is an on-demand, auto-scaling configuration for Amazon Aurora. You get most of the features and benefits that come with the standard Amazon Aurora, plus more. Amazon Aurora Serverless cluster automatically starts up, shuts down, and scales capacity up or down based on your application's needs. You do not need to keep monitoring and managing capacity yourself. And to prevent your Aurora Serverless from becoming too expensive, you can set a capacity range to prevent it from overscaling.

Amazon Aurora Serverless supports both MySQL and PostgreSQL, since it is just an extension of Amazon Aurora. If you'd like to move your data from Amazon Aurora to Amazon Aurora Serverless, simply take a snapshot from your existing Aurora provisioned cluster and restore it into an Aurora Serverless DB Cluster. One thing to note is that you can't give an Aurora Serverless DB cluster a public IP address, so you'll have to connect to it from within your VPC.

When configuring scaling options, 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 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 automatically creates scaling rules for thresholds for CPU utilization, connections, and available memory. A scaling point is a point in time at which the database can safely initiate the scaling operation.

Use Aurora Serverless for the following types of database workloads:

- Infrequently used applications
- Applications with variable workloads (high peaks and low dips)
- New applications with no benchmarked performance
- Applications with unpredictable workloads
- Development and test databases which can be shut down when not in use
- Multi-tenant applications

In Aurora Serverless, there are a few features that are not supported:

1. Aurora cloning
2. Aurora global databases
3. Aurora multi-master clusters
4. Aurora Replicas
5. AWS IAM database authentication
6. Backtracking in Aurora

7. Database activity streams
8. Performance Insights

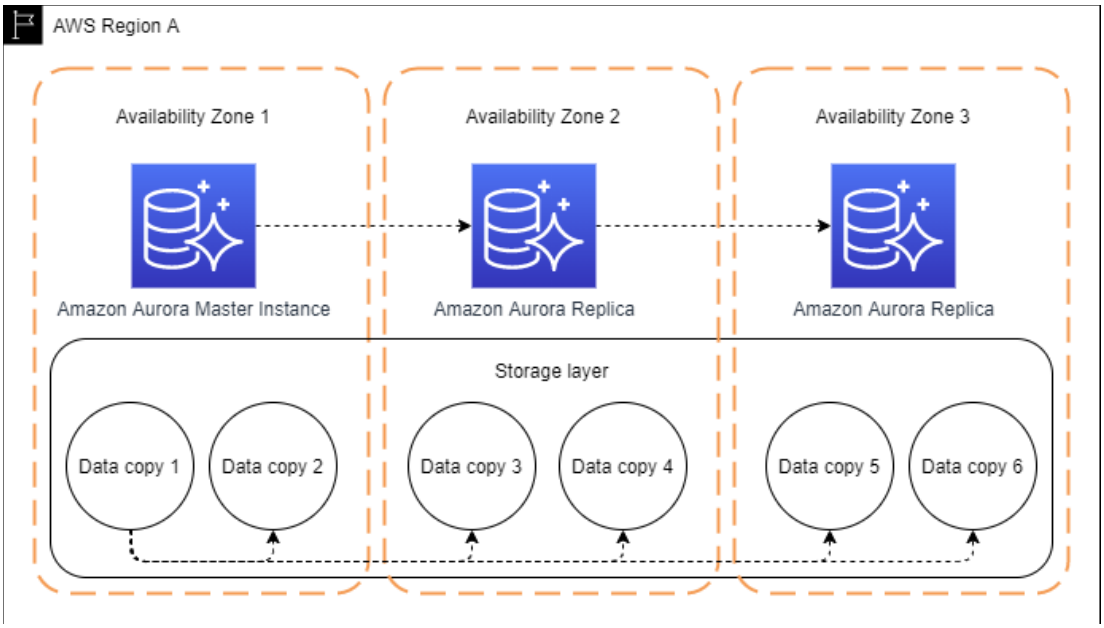
References:
https://aws.amazon.com/rds/aurora/serverless/
https://tutorialsdojo.com/aurora-serverless-tutorial-part-1/
https://tutorialsdojo.com/aurora-serverless-tutorial-part-2/

High Availability for Amazon Aurora

Although Amazon Aurora is a part of Amazon RDS, they do not share the same technology for implementing high availability and fault tolerance. The Amazon Aurora architecture separates storage hardware from compute hardware. Your data remains safe even if some or all of the DB instances in your Aurora cluster become unavailable. How Amazon Aurora achieves HA and FT are discussed below.

Amazon Aurora synchronously replicates your data six ways across three Availability Zones in a single AWS Region. Aurora stores these copies regardless of whether the instances in the DB cluster span multiple Availability Zones. For a cluster using single-master replication, after you create the primary instance, you can create up to 15 read-only Aurora Replicas in different AZs.

Aurora Replicas work similarly with Amazon RDS Read Replicas. You can offload your read operations to these replicas to reduce the burden on the primary database. When the primary instance encounters an issue and fails, one of the Aurora Replicas is promoted to primary via a failover. The cluster endpoint will then automatically point to this new primary database so you won't have to modify your connection strings. If you need multi-region DR, use Amazon Aurora Global Databases instead. Amazon Aurora Global Databases span multiple regions, and Amazon Aurora handles the replication between your DB instances with minimal replication lag. If you do not create Aurora Replicas nor Global Databases, in the event of a failure, Amazon Aurora recreates the primary instance using the data that is stored in other Availability Zones.



Reference:
https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/Concepts.AuroraHighAvailability.html
https://tutorialsdojo.com/amazon-aurora/

Amazon Aurora Global Database and Replicas

Perhaps you have an Amazon RDS Multi-AZ database with read replicas located in multiple regions, and you know that your database experiences read-heavy operations, especially in your secondary regions. If retrieving stale data is unacceptable due to the asynchronous replication of Amazon RDS then you should consider migrating your database cluster onto Amazon Aurora instead, if possible.

Amazon Aurora has a feature called “Global Database”, which is primarily designed for these globally distributed application scenarios. Enabling this feature allows Amazon Aurora to replicate your data across regions with no impact on database performance, with fast local reads and low latency in each region, and provides disaster recovery from region-wide outages.

An Aurora global database has a primary DB cluster in one Region, and up to five secondary DB clusters in different Regions. Global Database uses storage-based replication with typical latency of less than 1 second. With this, the chances of retrieving stale data is minimized. Furthermore, if your primary region suffers a performance degradation or outage, you can promote one of the secondary regions to become the new primary. An Aurora cluster can recover in less than 1 minute even in the event of a complete regional outage. This provides you with a Recovery Point Objective (RPO) of 1 second and a Recovery Time Objective (RTO) of less than 1 minute. You can further scale your secondary clusters by adding more read-only instances or

Aurora Replicas to a secondary region. The secondary cluster is read-only, so it can support up to 16 Aurora Replica instances rather than the usual limit of 15 for a single Aurora cluster.

When Aurora Global Database feels like a bit overkill, or you'd like to utilize MySQL/PostgreSQL's native replication features, you can scale your Aurora cluster by configuring Aurora Replicas to serve read-only transactions. Aurora Replicas also help to increase availability. If the primary instance becomes unavailable, Aurora automatically promotes one of the replicas. An Aurora DB cluster can contain up to 15 Aurora Replicas. The Aurora Replicas can be distributed across Availability Zones in your cluster's region. Additionally, Aurora Replicas return the same data for query results with minimal replica lag.

Aside from these benefits, one feature of an Aurora MySQL DB cluster is that you can create a Read Replica of it in a different region, by using MySQL binary log (binlog) replication. Each cluster can have up to five Read Replicas created this way, each in a different region. You can also replicate two Aurora MySQL DB clusters in the same region, by using MySQL binary log (binlog) replication. Same goes with two Aurora PostgreSQL DB clusters in the same region, by using PostgreSQL's logical replication feature. Aurora PostgreSQL does not currently support cross-region replicas. Since the logical replication process is handled by the database, it might have an effect on its performance, unlike Aurora Global Database where the replication happens in the storage layer.

References:
https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/aurora-global-database.html
https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/Aurora.Replication.html

Amazon DynamoDB

Amazon DynamoDB is a fully managed NoSQL database. It's highly scalable as it automatically scales your database without any manual intervention and can provide single-digit millisecond performance. DynamoDB is also serverless so you don't have to provision and manage any servers at all. It's a highly durable database with built-in security, backup, and in-memory caching for your websites, mobile apps, Internet-of-Things, and other use cases. Since its serverless, it eliminates manual database management tasks and provides the least amount of operational overhead. It is highly resilient and capable of automatically scaling its read and write capacity without the need for advanced capacity planning. DynamoDB can also be queried using simple key-value requests and can handle millions of requests per second. Yes! You heard that right! It can handle millions of requests per second! That level of performance is quite difficult to achieve if you are using a relational database!

You might be wondering what's the meaning of the word "Dynamo" in Amazon DynamoDB. Dynamo is actually the very first non-relational database developed at Amazon in the early 2000s. Amazon's eCommerce business back then requires a highly reliable, scalable, and distributed key/value database and that is where the development and the use of "Dynamo" started. Back then, Dynamo is just a database program that you can run locally but in 2012, AWS releases Amazon DynamoDB as a service. The current Amazon DynamoDB is still using some of the data patterns in the original Dynamo design but there have been major design changes over the years. You can even run a downloadable version of the current Amazon DynamoDB on your local computer! This is called DynamoDB Local and this is perfect if you want a self-contained database without using an AWS web service.

Given its history, Amazon DynamoDB is actually designed from the ground up to provide high scalability and ultra-fast performance that is measured in milliseconds or even in microseconds. This type of database is quite different from a relational database which usually has hundreds or even thousands of related tables. NoSQL databases are highly scalable which can accept millions of requests per second, unlike a relational database that has certain limits in terms of its read and write throughput.

A DynamoDB database is just composed of a single table by default, which is commonly referred to as a DynamoDB table. Since this is considered a non-relational database, a DynamoDB table does not have a relationship with other DynamoDB tables. It doesn't have a foreign key or any other SQL constraints.

The anatomy of a DynamoDB database is somehow similar to a single table in a relational database. Both of them have a primary key. A DynamoDB primary key is also called a partition key which can also include a sort key. In SQL, you have a table, row, column, primary key, index, view, nested table, array, and many other components. These parts have equivalent components in Amazon DynamoDB.

A "row" in SQL is called an "Item" in DynamoDB and a "column" is considered an "attribute". You can visualize a DynamoDB item as a JSON document with different types of attributes in it, where the item is the row and the attributes are the columns. A SQL index is a counterpart for the "secondary index" and the "View" is considered

as a “Global Secondary Index”. A secondary index is simply a data structure that contains a subset of attributes from a table, along with an alternate sort key to support the Query API operations. The main purpose of these secondary indexes is to make your DynamoDB queries faster. This is done by only querying a subset of items that are grouped by the secondary index, and not going through each and every item of the entire table.

There are two types of indexes in DynamoDB which are the local secondary index and global secondary index. A local secondary index lets you query over a single partition only, which is based on the partition key value of the query while a global secondary index allows you to query data across all partitions of the entire table. These indexes have their own type of Read Consistencies that you have to watch out for.

Let’s now discover the different features of Amazon DynamoDB that you can use to optimize your database and integrate with other AWS services.

You can configure DynamoDB as a multi-region database, automatically expire items based on their timestamp and modify its read and write throughput capacity. You can further reduce its response times from milliseconds to microseconds through caching or even add atomicity, consistency, isolation, and durability (ACID) properties to your DynamoDB table.

We will begin by exploring the different forms of a DynamoDB database. By default, you only launch a single DynamoDB table in an AWS Region that you choose. Alternatively, you can launch DynamoDB Global tables that provide multi-region replication to your data store. For Amazon DynamoDB Global tables, all of the necessary tasks of creating identical tables in different Regions are done automatically including the propagation of the ongoing data changes across all tables. Since this service is fully managed, the underlying servers that power your database are entirely managed by AWS themselves and not you, so you won’t see or access the EC2 instances or servers that your table uses. Unlike Amazon RDS, a DynamoDB table doesn’t reside within your custom VPC. Again, all the underlying infrastructure is handled exclusively by AWS.

The next feature is called DynamoDB Streams. Essentially, this is a stream of data that captures all of the data changes made to the items stored in your table. So if a new item is added, then that same data will also flow out of the DynamoDB stream at the point in time when the actual change occurred. You can also associate the stream’s Amazon Resource Name (ARN) with an AWS Lambda function which will then polls the stream and invokes the Lambda function synchronously when it detects new stream records. Item-level modifications can also be replicated to Kinesis Data Streams.

DynamoDB also has a feature to automatically expire the items based on their timestamp using TTL, or Time to Live. It allows you to define a per-item timestamp that can help you determine when an item is no longer needed. DynamoDB deletes the item from your table after the date and time of the specified timestamp. Using the TTL feature reduces the stored data in your table by retaining only the relevant items for your workloads.

It is also a feature called Amazon DynamoDB Transactions. By default, NoSQL databases like DynamoDB are not ACID-compliant. This provides an all-or-nothing change to multiple items both within and across tables. The consistency of data might vary if you have a distributed database where the stored data is scattered out of different servers or data centers. This is what the DynamoDB Transactions provide. You can use the DynamoDB transactional read and write API operations to manage complex business workflows that require adding, updating, or deleting multiple items as an atomic operation. When we say “atomic”, it usually means that the change is done via a single, all-or-nothing operation.

Amazon DynamoDB has also a caching feature called DynamoDB Accelerator or DAX for short. Most of the time, the response times of a DynamoDB table can be measured in single-digit milliseconds. However, if you have a use case that requires a response time in microseconds, then you can use its DAX feature. DAX delivers fast response times for accessing eventually consistent data. This is a type of in-memory cache that significantly reduces the response times of your table.

There are different options available in DynamoDB to increase or decrease its capacity for read and write operations. This is measured in terms of Read Capacity Unit or RCU, and Write Capacity Unit or WCU. Amazon DynamoDB has two types of Read/Write capacity mode, which are the Provisioned and On-Demand capacity modes.

As its name implies, the Provisioned Capacity mode allows you to provision, or manually set the RCU and WCU of your table. It also has an Auto Scaling feature where you can set the target utilization as well the minimum and maximum provisioned capacity. The minimum provisioned capacity ensures that the capacity won’t fall a certain threshold while the maximum provisioned capacity protects you from overspending. This mode is suitable if your application has predictable traffic that doesn’t vary over time.

The On-Demand Capacity mode on the other hand is commonly used if your application traffic is inconsistent or has varying access patterns. It is also suitable if you expect that there’ll be more traffic with sharp spikes in the future. This On-Demand Capacity mode does not have an explicit Auto Scaling setting that you can manage since the scaling activities are done automatically. In addition, you can also use this if your application has two types of access patterns. Say, for example, your application is expected to have clearly defined access patterns throughout the year but have variable amounts of traffic during certain times of the year when the company is having flash sales or product announcements.

In terms of security, there are several features that you can use to protect your data both in transit and at rest. You can ensure that the API calls from your private Amazon EC2 instances, which are residing in your VPC that go to your DynamoDB table, do not traverse the public Internet by creating a VPC Endpoint. You can do this in two easy steps: First, you have to launch a gateway endpoint, and second create a route table entry for the endpoint that you’ve just created.

DynamoDB also provides an Identity and Access Management feature to control access to your tables. You can create an IAM policy and specify the name of your DynamoDB table then include the allowed or restricted

actions that you want your IAM users to have. So if you have a table named Books, then you can create an IAM policy like this one and choose whether you allow GET, UPDATE, DELETE, SCAN, QUERY, or other API operations on the DynamoDB table.

Finally, let's cover the different backup options in DynamoDB which are the Point-in-time Recovery and the On-Demand Backup and Restore process. The Point-in-time Recovery option is also called PITR and it enables continuous backups to your table that allow you to restore your table at a specific point in time that you specify. So if you want to restore your table and its content last week, you just use the Point-In-Time Recovery option. You can restore your table to any point in time during the last 35 days.

The On-Demand Backup and Restore option is quite a manual process and does not continuously back up to your table, unlike PITR. You have to do the backup yourself for a particular time that you choose and restore it manually.

Amazon DynamoDB Transactions

DynamoDB transactions is a feature that lets you fulfill atomicity, consistency, isolation, and durability (ACID) across one or more tables within a single AWS account and region. Use DynamoDB transactional read and write APIs if your applications require adding, updating, or deleting multiple items as a single, all-or-nothing operation. A DynamoDB transaction can include up to 25 unique items or up to 4 MB of data.

- With the transaction write API, you can group multiple Put, Update, Delete, and ConditionCheck actions. You can then submit the actions as a single TransactWriteItems operation that either succeeds or fails as a unit. TransactWriteItems is supported in DynamoDB Accelerator but not in Global Tables.
- With the transaction read API, you can group and submit multiple Get actions as a single TransactGetItems operation. If a TransactGetItems request is submitted on an item that is part of an active write transaction, the read transaction is cancelled. TransactGetItems is supported in DynamoDB Accelerator but not in Global Tables.

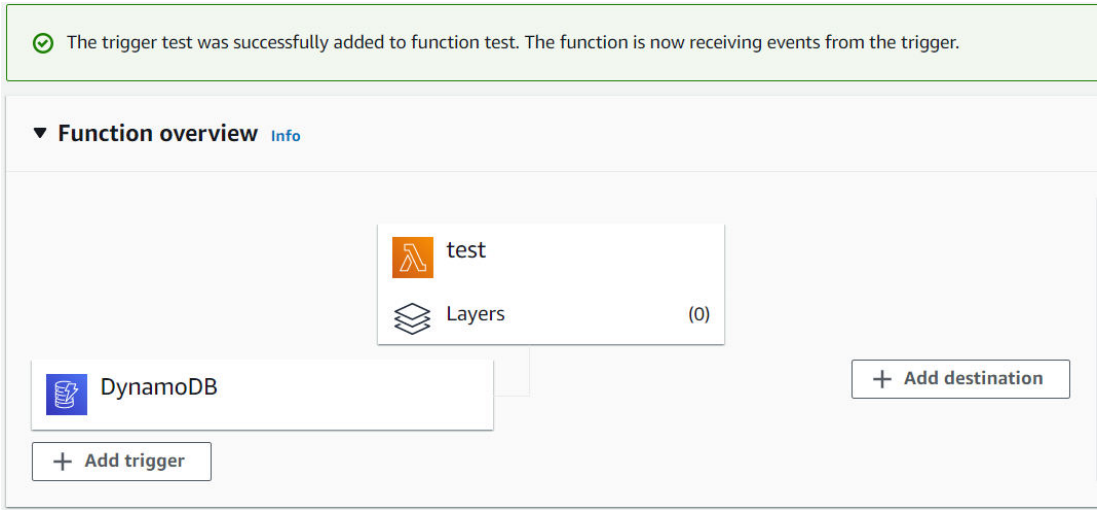
With the addition of DynamoDB transactions, you can choose among three options for read operations — eventual consistency, strong consistency, and transactional; and between two options for write operations — standard and transactional.

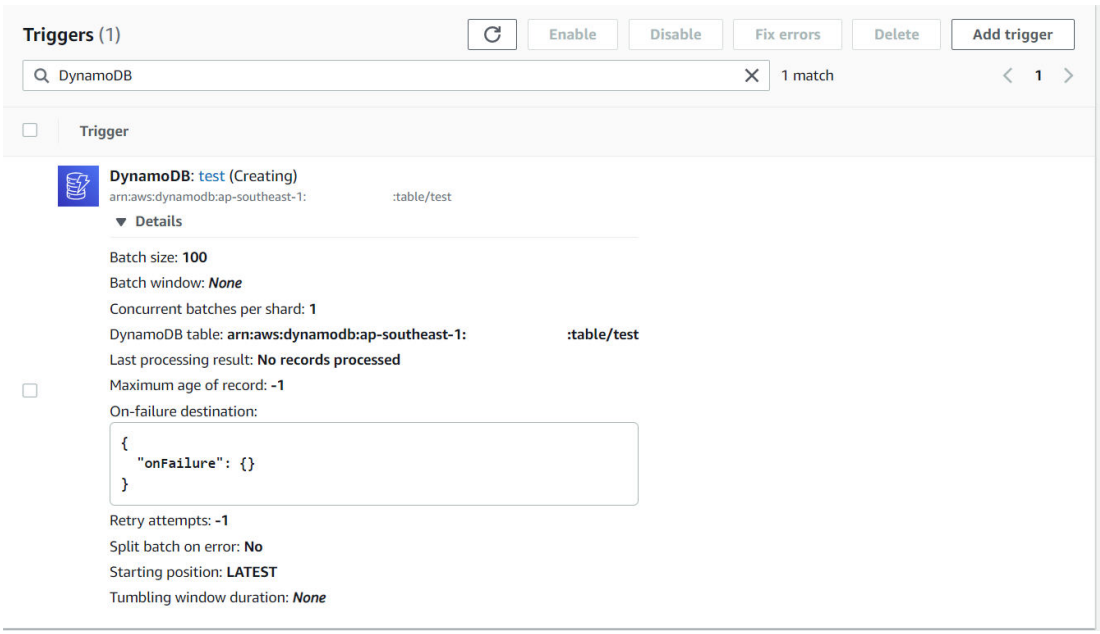
Know that transactional operations are different from batch operations. In batch operations, some queries may succeed while others do not. In transactional operations, it's all or nothing with your queries. You also can't target the same item with multiple operations within the same transaction.

References:
https://aws.amazon.com/blogs/aws/new-amazon-dynamodb-transactions/
https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/transactions.html
https://tutorialsdojo.com/amazon-dynamodb/

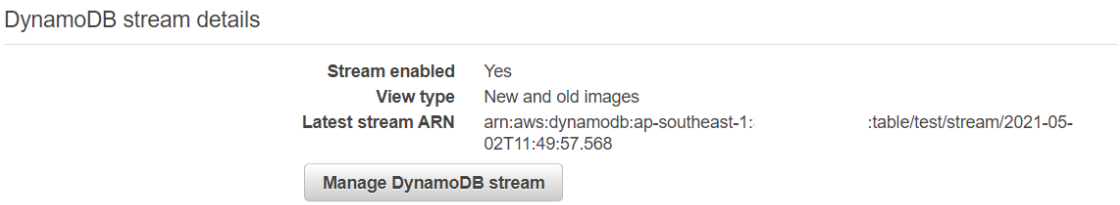
AWS Lambda Integration with Amazon DynamoDB Streams

Amazon DynamoDB is integrated with AWS Lambda so you can create *triggers*, which are pieces of code that automatically respond to events in DynamoDB Streams. With triggers, you can build applications that react to data modifications in DynamoDB tables.



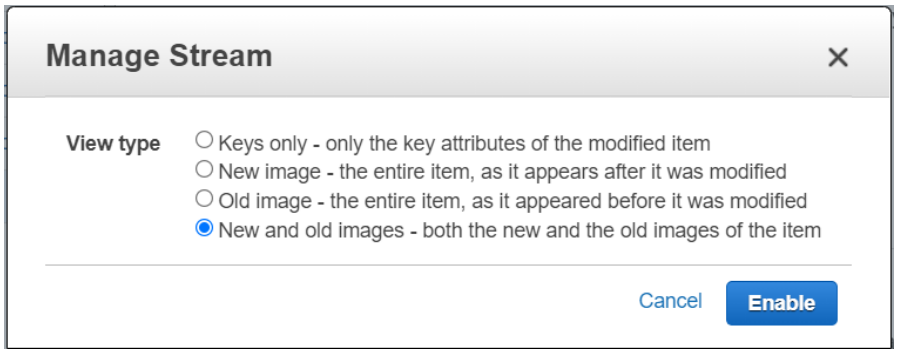


After you enable DynamoDB Streams on a table, associate the DynamoDB table with a Lambda function if AWS does not automatically associate it. AWS Lambda polls the stream and invokes your Lambda function synchronously when it detects new stream records.



Configure the StreamSpecification you want for your DynamoDB Streams:

- **StreamEnabled (Boolean)** – indicates whether DynamoDB Streams is enabled (true) or disabled (false) on the table.
- **StreamViewType (string)** – when an item in the table is modified, StreamViewType determines what information is written to the stream for this table. Valid values for StreamViewType are:
    - **KEYS_ONLY** – Only the key attributes of the modified items are written to the stream.
    - **NEW_IMAGE** – The entire item, as it appears after it was modified, is written to the stream.
    - **OLD_IMAGE** – The entire item, as it appeared before it was modified, is written to the stream.
    - **NEW_AND_OLD_IMAGES** – Both the new and the old item images of the items are written to the stream.



References:
https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/Streams.Lambda.html
https://docs.aws.amazon.com/amazondynamodb/latest/APIReference/API_StreamSpecification.html

Amazon DynamoDB Replication

In Amazon RDS, if you decided to replicate your databases to other AWS Regions, you would create Read Replicas in your desired region(s) and AWS will perform asynchronous replication between the primary instance and the read replicas. In Amazon DynamoDB, the concept of a read replica does not exist. Instead, to create copies of your DynamoDB tables across different regions, you will need to create a Global Table. A Global Table, in a basic sense, is just a collection of one or more DynamoDB replica tables. Each replica table

has the same table name, stores the same data, and uses the same primary key schema as the primary table. A global table can only have one replica table per region.

With RDS read replicas, applications can only read data from them, so no write operations can be performed. When an application writes data to any DynamoDB replica table in one region, DynamoDB propagates the write to the other replica tables in the other regions within the same global table automatically. Because of this, DynamoDB does not support strongly consistent reads across regions. To help ensure eventual consistency, DynamoDB global tables use a *last writer wins* reconciliation between concurrent updates.

When creating a global table, you first need to enable DynamoDB streams. DynamoDB streams will distribute the changes in one replica to all other replicas. Next, you select the region(s) where you would like to deploy a replica in. The *AWSServiceRoleForDynamoDBReplication* IAM role that is automatically created by DynamoDB allows the service to manage cross-region replication for global tables on your behalf.

References:
https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/V2globaltables_HowItWorks.html
https://aws.amazon.com/dynamodb/global-tables/
https://tutorialsdojo.com/amazon-dynamodb/

Caching with DynamoDB DAX

In most cases, the single digit millisecond performance of DynamoDB is sufficient for the user's needs. But for cases when single digit microsecond performance is required, you'll need to add a caching mechanism to your DynamoDB table. DynamoDB Accelerator (DAX) is a fully managed, write-through caching service that delivers fast response times for accessing eventually consistent data in DynamoDB. In the exam, unless there is a clear requirement to use Redis or Memcached, which in this case you'll use Amazon Elasticache instead, always choose DAX as your DynamoDB caching solution.

DAX is able to perform the following functions:

1. DAX reduces the response times of eventually consistent read workloads from single-digit milliseconds to microseconds.
2. DAX requires only minimal functional changes if your applications have already been using the DynamoDB API.
3. For read-heavy or bursty workloads, DAX provides increased throughput and potential cost savings by reducing the need to overprovision read capacity units.

If you need enhanced data security, DAX supports server-side encryption, but it does not support TLS. For high availability, configure a Multi-AZ DAX cluster. You can scale your DAX cluster by adding more nodes or by using larger node types. A DAX cluster in an AWS Region can only interact with DynamoDB tables that are in the same region. If you have tables in other regions, you must launch DAX clusters in those regions too.

DAX is not ideal for the following scenarios:

- Applications that require strongly consistent reads.
- Applications that do not require microsecond response times for reads, or that do not need to offload repeated read activity from underlying tables.
- Applications that are write-intensive, because the data in the cache will be frequently overwritten.

There are two caches available in DAX: **item cache** and **query cache**.

DAX maintains an item cache to store the results from GetItem and BatchGetItem operations. Cached items have a default cache TTL of 5 minutes. When a cache is full, DAX evicts older items (even if they haven't expired yet) to make room for new items.

DAX maintains a query cache to store the results from Query and Scan operations. These result sets are stored by their parameter values. You specify the TTL setting for the query cache when you create a new DAX cluster. If the query cache becomes full, DAX evicts older result sets (even if they haven't expired yet) to make room for new result sets.

References:
https://docs.amazonaws.cn/en_us/amazondynamodb/latest/developerguide/DAX.html
https://tutorialsdojo.com/amazon-dynamodb/

Amazon Redshift

Amazon Redshift is a fully managed data warehouse that allows you to analyze all your data using standard SQL and your existing Business Intelligence tools. A data warehouse is just a database optimized to analyze relational data coming from transactional systems, business applications, and other sources. The data stored here has a structure and is optimized for fast SQL queries. This is different from a data lake, where the data does not have a structure. Amazon Redshift has a concurrency scaling feature that supports virtually unlimited concurrent users and concurrent queries to achieve a consistently fast query performance It also has a feature called Redshift Spectrum that allows you to query and retrieve structured and semistructured data from files stored in Amazon S3 without having to manually load the data into your Redshift tables.

Amazon Redshift High Availability, Fault Tolerance and Disaster Recovery

Amazon Redshift is similar to Amazon RDS where it is also a fully managed RDBMS. But where Amazon RDS is for OLTP, database-type workloads, Amazon Redshift is designed for OLAP, data warehouse-type workloads. An Amazon Redshift data warehouse consists of your cluster of nodes which run a specific Redshift engine. In each cluster, there is one leader node and one or more compute nodes. The leader node receives queries from client applications, parses the queries, and creates query execution plans. It then coordinates the parallel execution of these plans with the compute nodes and collects the results from these nodes. Finally, it then returns the results of the query back to the client applications. Compute nodes do bulk of the query execution work based on the execution plans from the leader node and transmit data among themselves to serve these queries. Query results are then sent to the leader node for aggregation.

When launching your cluster, Amazon Redshift provisions your cluster in a randomly selected Availability Zone within the AWS Region you are in, though you can optionally use a specific Availability Zone if Amazon Redshift is available in that zone. All the cluster nodes are provisioned in the same Availability Zone. There is no option in Amazon Redshift to deploy a multi-AZ cluster. Amazon Redshift only supports Single-AZ deployments. If your cluster's Availability Zone experiences an outage, Amazon Redshift will automatically move your cluster to another AZ within the same region without any data loss or application changes, but you must enable the relocation capability beforehand in your cluster configuration settings.

If you need high availability for your Redshift cluster then you must create a new secondary cluster that will continuously receive new data from the primary cluster through some pipeline, such as Amazon Kinesis. However, if you only need high availability for nodes within a cluster, Amazon Redshift already automatically detects and replaces any failed node it finds. During this period, the data warehouse cluster will be unavailable for queries and updates until a replacement node is provisioned and added in. Additionally, if the leader node fails, inflight queries are dropped. Data for the replacement node is retrieved from the continuous backups in S3 and the most frequently queried data is prioritized during restoration. Single node clusters do not support data replication, so you will have to restore the cluster from a snapshot.

For disaster recovery, Amazon Redshift replicates all your data within your data warehouse cluster when it is loaded, and also continuously backs it up to Amazon S3. The service maintains at least three copies of your data – the original and replica on the compute nodes, and a backup in S3. You can also configure Redshift to asynchronously replicate your snapshots to S3 in another region. Automated backups are only kept up to a maximum of 35 days, but manual backups can be retained for a longer period.

References:
https://aws.amazon.com/redshift/faqs/
https://tutorialsdojo.com/amazon-redshift/

Amazon Redshift Spectrum

Amazon Redshift Spectrum is a feature of Amazon Redshift that allows you to query structured and semistructured data stored on Amazon S3 without having to load and transform the data into Amazon Redshift tables. If you have pools of data stored in Amazon S3 or you are using Amazon S3 as a data lake, Amazon Redshift Spectrum is capable of executing SQL queries on them, such as pull data, filter, project, aggregate, group, and sort. Best of all, Redshift Spectrum is serverless, so there is no infrastructure to maintain from your end. Redshift Spectrum runs on dedicated servers that are independent from those of Redshift clusters, and Redshift Spectrum automatically scales query compute capacity based on the size of the S3 data being retrieved. This means Redshift Spectrum is capable of massive parallel processing. You pay only for the queries you run against the data that you actually scan.

How Redshift Spectrum works is as follows:

1) You create Redshift Spectrum tables by defining the structure for your files and registering them as tables in an external data catalog. The external data catalog can be AWS Glue, the data catalog that comes with Amazon Athena, or your own Apache Hive metastore. You can also partition the external tables on one or more columns to optimize query performance.
2) Redshift Spectrum queries are sent to the leader node of your Redshift cluster. The leader node creates and distributes the execution plan to the compute nodes in your cluster.
3) Then, the compute nodes obtain the information describing the external tables from your data catalog. The compute nodes also examine the data available locally in your cluster and scans only the objects in Amazon S3 that are not present locally.
4) The compute nodes then generate multiple requests depending on the number of objects that need to be processed, and submit them concurrently to Redshift Spectrum. Redshift Spectrum worker nodes scan, filter, and aggregate your data from S3, and stream the required data for processing back to your Redshift cluster.
5) Final join and merge operations are performed locally in your cluster and the results are returned to your client applications.

When using Redshift Spectrum, your Redshift cluster and the S3 bucket data source must be in the same AWS Region. You also can't perform update or delete operations on external tables. You must recreate them if there are any changes that need to be made.

Comparison of Similar Analytics Service in AWS

Amazon Redshift Spectrum	Amazon Redshift	Amazon EMR	Amazon Athena
Use Amazon Redshift Spectrum if you are running complex queries on large amounts of data stored in Amazon S3 and Amazon Redshift, and you are planning on storing frequently accessed data in Amazon Redshift.	Use Amazon Redshift when you are pulling data from multiple different sources and joining them into one structured table for querying and analytics.	Use Amazon EMR if you use custom code to process and analyze extremely large datasets with big data processing frameworks such as Apache Spark, Hadoop, Presto, or Hbase	Use Amazon Athena if you only need a simple way to query data stored in Amazon S3. Data is returned in a table and can be exported into a csv file. Consecutive results are not stored in a structured format.

References:

https://aws.amazon.com/blogs/big-data/amazon-redshift-spectrum-extends-data-warehousing-out-to-exabytes-no-loading-required/

https://docs.aws.amazon.com/redshift/latest/dg/c-using-spectrum.html

https://tutorialsdojo.com/amazon-redshift/

Other AWS Databases

Amazon DocumentDB

Amazon DocumentDB is a fast, scalable, highly available MongoDB-compatible database service. MongoDB is a document-oriented database program that is cross-platform and is also a type of a NoSQL database. In a MongoDB database, you call a table a "collection"; a row a "document" and a column a "field". Each document contains fields and values in JSON format with no rigid schema enforced, unlike in traditional SQL databases. This is also the same concept in Amazon DocumentDB – it stores, queries, and index JSON documents. That's how DocumentDB got its name! All of the items are stored into JSON documents in Amazon DocumentDB.

Amazon Keyspaces

Amazon Keyspaces is a scalable, highly available, and managed Apache Cassandra–compatible database service. Apache Cassandra is an open-source, wide column data store that is designed to handle large amounts of data. With Amazon Keyspaces, you can run your Cassandra workloads on AWS without having to provision, patch, or manage servers.

Amazon DocumentDB

Amazon Neptune is a fast, reliable, fully-managed graph database service that makes it easy to build and run applications that work with highly connected datasets. It allows you to store billions of relationships and query your data graphs with milliseconds latency. Graph databases use nodes to store data entities and edges to store relationships between entities.

Amazon Timestream

Amazon Timestream is a fast, scalable, and serverless time series database service for IoT and operational applications. Basically, a time series is a sequence of data points recorded over a time interval for measuring events that have changed at that time frame. You can use this to track the change of stock prices, temperature measurements, and the CPU utilization of an EC2 instance over a specific amount of time.

Amazon QLDB

Amazon QLDB is a fully managed ledger database. It provides a transparent and immutable transaction log that is owned by a central trusted authority. These logs are cryptographically verifiable so you can provide an auditable history of all changes made to your application data. Amazon QLDB can be used to track each and every application data change.

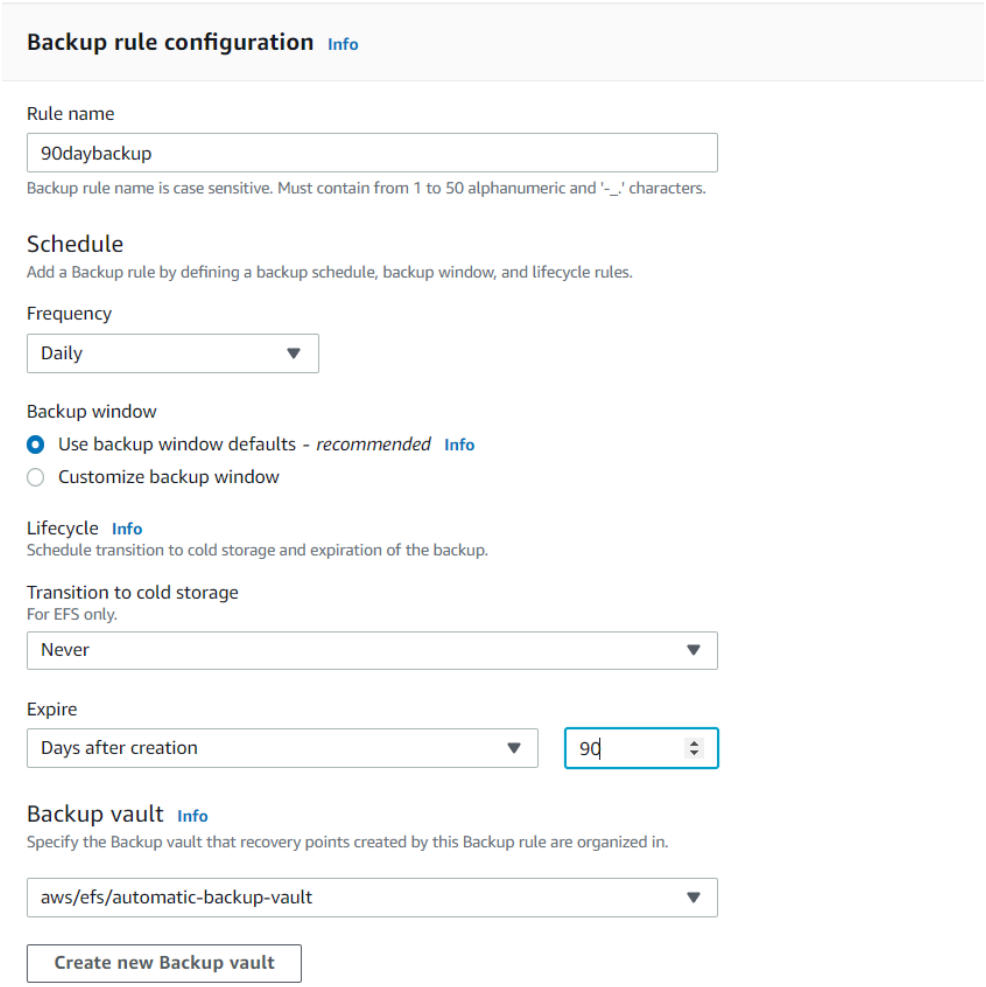
AWS Backup

Backups are a necessity for any storage device that contains critical data. They are a lifesaver when something goes wrong and you need to restore something back. Backups are a requirement for any production database and file system. Most companies develop their own backup strategies, such as deciding what types of backups to take and how long to keep them for.

In AWS, services such as Amazon RDS, Amazon Aurora, Amazon EFS, and Amazon DynamoDB support automated backups, so you never have to worry about not having a backup available. However, and you might not know this, automated backups or automated snapshots for these services have a maximum retention period of only 35 days. For some companies, this period is too short. To keep your backups for longer periods of time, you should create manual backups; but why would you do a task that repeats manually when you can automate it?

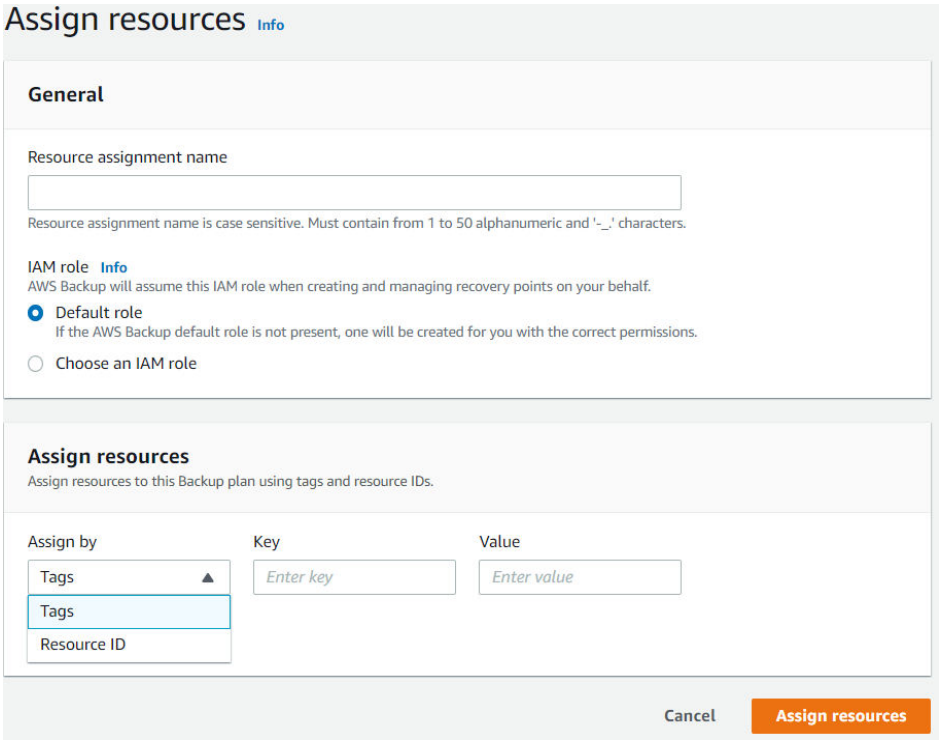
If you have a custom solution for taking manual backups programmatically because you need to process the backup, then there is nothing wrong with scripting your own automation. But if your only goal is to take recurring backups and keep them durably for an extended period of time, then you can use AWS Backup instead.

AWS Backup is a fully managed backup service that centralizes and automates backing up of data across different AWS services. With AWS Backup, you can create backup plans which define your backup requirements, such as how frequently to back up your data and how long to retain those backups. Your backups are then stored in what's called a backup vault. You can also specify in your backup plan if there should be a specific time window on when backups should run. Furthermore, AWS Backup supports on-demand backups if you only need to do a one-time backup.



To associate your AWS resources with your backup plans, simply list down the tags that would identify them or enter their resource IDs. In other words, every supported resource that has matching tags or resource IDs from those you entered will be included in the backup plan. You can choose which AWS services you'd like to opt-in with AWS Backup. Opting out a service means that even if a resource under that service matches a tag defined in one of your backup plans, AWS Backup will not take a backup of that resource. AWS Backup supports taking backups for the following services:

- Aurora
- DynamoDB
- EBS
- EC2
- EFS
- FSx
- RDS
- Storage Gateway



References:

https://docs.aws.amazon.com/aws-backup/latest/devguide/whatisbackup.html

Amazon VPC

Amazon VPC is a virtual network in a sense that these geographically distributed servers are virtually connected to each other to form a single network. This network is regional scope, meaning, your VPC can only exist within one AWS Region.

This virtual network works like a traditional network on your on-premises data center. It has also a CIDR block, a main route table, multiple subnets, and other external network connections. But unlike your traditional on-premises network, an Amazon VPC can leverage on the scalable global infrastructure of AWS at a fraction of the cost. A VPC is not just limited within a single data center. It can span to multiple data centers and Availability Zones within an AWS Region.

A virtual network can be subdivided by two or more subnetworks, or subnets for short. In AWS, a subnet must reside entirely within one Availability Zone only. One subnet cannot span to two or more AZs. However, you can have multiple subnets in the same Availability Zone.

As you know, an Availability Zone is basically composed of one or more data centers. The EC2 instances in the same subnet can freely communicate with each other as if they are located in a single building. In reality, these servers are scattered in one or more data centers.

If one server needs access to another server that resides in another subnet, you need to configure your main route table to connect these two subnets.

By default, all of the subnets of your Amazon VPC are interconnected. You can also create a custom route table that you can associate with your subnet.

You can create a public or private subnet for your VPC. A public subnet is perfect for your web servers that are meant to be publicly accessible over the Internet. Alternatively, you can place your backend systems like databases or application servers in a private subnet with no Internet access.

You can also set up security groups and network access control lists to manage the incoming traffic going in and out of your Amazon EC2 instances which are hosted in your VPC.

Physical Location and Resources in Amazon VPC

Amazon Web Services has its own global network of data centers spread around the world. AWS may use its own cloud data centers that it fully controls or use colocation data centers that are owned by other companies. An Amazon VPC is a regional resource which means that it can only exist in one particular AWS Region. It is a logical private network that is within the larger network of AWS. Every VPC can have one more or subnets where you can place your AWS resources. In reality, these subnets are mapped to certain Availability Zones that span across multiple data centers in a geographic region or cities.

For example, you launched a new VPC in the us-east-1, Northern Virginia region to better serve your customers located in New York, Washington DC, and other cities on the east coast. It's quite obvious where your VPC is located. Of course, it's somewhere in Northern Virginia as what the region name says, but where exactly is it? Where are the physical infrastructures that AWS use to host your EC2 instances, RDS databases, and other resources?

An Amazon VPC is regional in scope, which means that it can use different data centers that are in Northern Virginia. These facilities could be somewhere near its major cities, such as in Richmond, Ashburn, Sterling, Chantilly, or other nearby locations. AWS does not explicitly tell you the physical facilities that they're using in hosting your data or application, but you can intelligently guess where they are, based on the data centers available in that area.

Each of these cities can have one or more data centers. So for example, the city of Sterling can have 8 data centers, Ashburn may have 4, and Chantilly could have three or four centers. Each of these groups can represent an individual Availability Zone so the 8 centers in Sterling can be us-east-1a; the group in Ashburn may be us-east-1b and the one in Chantilly might be us-east-1c.

These Availability Zones are where you place the subnetworks or subnets of your VPC. You can place multiple subnets in a single Availability Zone. Again, you are mapping your subnet to a specific Availability Zone, which means that a subnet spans multiple data centers for that particular AZ. It's very important to note that a single AWS Region has a limited number of Availability Zones.

Say you launched a new EC2 instance in the us-east-1a, your instance will be hosted in one of the data centers that are located in the city of Sterling. If it's us-east-1b, it will be hosted in Ashburn and if it is in us-east-1c, the instance will be running in a rack server located in Chantilly. That's your Amazon VPC in the flesh! Take note that these are just estimations and the mapping of AZs and subnets differs from one account to another. AWS may or may not use these exact cities for your VPC but this gives you a general idea behind the underlying physical facilities and networking setup that powers your AWS resources.

Some AWS resources can be deployed within a VPC while others are not. For example, you cannot launch a new Amazon S3 bucket inside your VPC. That's definitely impossible since Amazon S3 is a regional resource that can't be contained or placed in a subnet. The same goes for Amazon DynamoDB tables and Lambda functions. However, what you can do is set up a direct connection for these services using a VPC Endpoint. This allows private connectivity between your EC2 instances and other AWS services without the traffic passing through the public Internet. With VPC Endpoints, the traffic between your VPC and the other service does not leave the Amazon network.

Different Gateways in Amazon VPC

There are different gateways that you can use in your VPC. You can choose between an internet gateway, customer gateway, virtual private gateway, carrier gateways, egress-only internet gateway, and others. These gates provide a way for your VPC to connect to the public Internet, your on-premises network, another VPC, or another external network.

A Virtual Private Cloud is private by default – meaning, it's a private cloud network that has no connection to any public networks. If you create a new VPC, it may not have a connection to the Internet if it doesn't have an Internet Gateway. As its name suggests, an Internet Gateway allows your EC2 instances to connect to the public Internet.

If you need to integrate your VPC with your on-premises data center, you can set up and configure a customer gateway and a virtual private gateway. Basically, a customer gateway device is a physical or software appliance that you own or manage in your on-premises network. A virtual private gateway is a component that you can attach to your Amazon VPC. You have to configure your customer gateway to work with the Site-to-Site VPN connection in AWS. The virtual private gateway is also used for establishing an AWS Direct Connect connection to your on-premises data center.

An egress-only gateway is primarily used for VPCs that use IP version 6. It functions as a Network Address Translation service and works as a NAT Gateway. Both a NAT instance or a NAT Gateway won't work in IPv6. To support this feature, you have to use the egress-only gateway instead. It is a horizontally scaled, redundant, and highly available VPC component that allows outbound communication over IPv6 from your Amazon EC2 instances in your Amazon VPC to the public internet. It also prevents various sources on the Internet from initiating an IPv6 connection with your EC2 instances.

A carrier gateway is primarily used for VPCs that use AWS Wavelength to deliver ultra-low latency applications for 5G devices. It allows incoming traffic from a carrier network in a specific location, and it also allows outgoing traffic to the carrier network and to the public Internet. A carrier gateway is only available for VPCs that contain subnets in a Wavelength Zone

Non-VPC Services

Not all compute, storage, and database services need to run in a VPC. It is important that you know these services so you can easily spot them out in the exam.

Services that do not require a VPC:

1) Amazon S3
2) Amazon DynamoDB
3) AWS Lambda (although you can configure Lambda to connect to a VPC to access resources in the VPC)

Security Group vs NACL

Security Group	Network Access Control List
Acts as a firewall for associated Amazon EC2 instances	Acts as a firewall for associated subnets
Controls both inbound and outbound traffic at the instance level	Controls both inbound and outbound traffic at the subnet level
You can secure your VPC instances using only security groups	Network ACLs are an additional layer of defense.
Supports allow rules only	Supports allow rules and deny rules
Stateful (Return traffic is automatically allowed, regardless of any rules)	Stateless (Return traffic must be explicitly allowed by rules)
Evaluates all rules before deciding whether to allow traffic	Evaluates rules in number order when deciding whether to allow traffic, starting with the lowest numbered rule.
Applies only to the instance that is associated to it	Applies to all instances in the subnet it is associated with
Has separate rules for inbound and outbound traffic	Has separate rules for inbound and outbound traffic
A newly created security group denies all inbound traffic by default	A newly created nACL denies all inbound traffic by default
A newly created security group has an outbound rule that allows all outbound traffic by default	A newly created nACL denies all outbound traffic default
Instances associated with a security group can't talk to each other unless you add rules allowing it	Each subnet in your VPC must be associated with a network ACL. If none is associated, the default nACL is selected.
Security groups are associated with network interfaces	You can associate a network ACL with multiple subnets; however, a subnet can be associated with only one network ACL at a time.

Your VPC has a default security group with the following rules:

1. Allow inbound traffic from instances assigned to the same security group.
2. Allow all outbound IPv4 traffic and IPv6 traffic if you have allocated an IPv6 CIDR block.

Your VPC has a default network ACL with the following rules:

1. Allows all inbound and outbound IPv4 traffic and, if applicable, IPv6 traffic.
2. Each network ACL also includes a non modifiable and non removable rule whose rule number is an asterisk. This rule ensures that if a packet doesn't match any of the other numbered rules, it's denied.

NAT Gateways and NAT Instances

NAT Gateways and NAT instances provide public internet connectivity to your private VPC resources without having to expose them to the public internet. NAT Gateways are managed NAT solutions, so you can easily provision and use them without having to maintain them. They also provide high bandwidth speeds and are highly available within a single subnet. NAT instances, on the other hand, give you more administrative control over your NAT workloads. They are EC2 instances that use a pre-configured AMI. NAT instances can be much cheaper if you do not totally need the benefits of a NAT Gateway.

Remember that when you launch a NAT Gateway or instance, you must place them in your public subnets and not your private subnets. They are literally a gateway between your public and private subnets, so mistakenly placing them in a private subnet will not provide you internet connectivity. Also note that a single NAT service can only run within a single subnet. For high availability and fault tolerance, you can use multiple public subnets and create a NAT service for each subnet. In this case, if one public subnet goes down, other private subnets would still have internet connectivity through their respective public subnets.

NAT Instance vs NAT Gateway

Attribute	NAT gateway	NAT instance
Availability	Highly available in the Availability Zone it is created in. But for true high availability, you should create a NAT gateway in a public subnet for each of your redundant private subnets or AZs.	Not highly available. You'll need a script to handle failover. For true high availability, you should launch a NAT instance in a public subnet for each of your redundant private subnets or AZs.
Bandwidth	Can scale up to 45 Gbps.	Depends on the bandwidth of the instance type you use.

Maintenance	Managed by AWS.	Managed by you, such as installing software updates or operating system patches on the instance.
Performance	Optimized for handling NAT traffic.	An Amazon Linux AMI that's configured to perform NAT.
Type and size	No available selection.	Select the instance type and size according to your predicted workload.
Cost	Charged on the number of NAT gateways you use, duration of usage, and amount of data that you send through the NAT gateways.	Charged on the number of NAT instances that you use, duration of usage, instance type and size, and storage. This option might be cheaper for some scenarios.
Public IP addresses	You need to associate an Elastic IP address to each NAT gateway at creation.	You may use an Elastic IP address or the automatically provided public IP address by AWS with the NAT instance.
Security groups	Cannot be associated with one. Control traffic using network ACLs.	Can be associated with one or more security groups.
Network ACLs	Use a network ACL to control the traffic to and from the subnet in which your NAT gateway resides.	Use a network ACL to control the traffic to and from the subnet in which your NAT instance resides.
Port forwarding	Not supported.	Manually customize the configuration to support port forwarding.
Bastion servers	Not supported.	Can be used as a bastion server.
Timeout behavior	When there is a connection timeout, a NAT gateway returns an RST packet to any resources behind the NAT gateway that attempt to continue the connection (it does not send a FIN packet).	When there is a connection timeout, a NAT instance sends a FIN packet to resources behind the NAT instance to close the connection.
IP fragmentation	Supports forwarding of IP fragmented packets for the UDP protocol. Does not support fragmentation for the TCP and ICMP protocols. Fragmented	Supports reassembly of IP fragmented packets for the UDP, TCP, and ICMP protocols

	packets for these protocols will get dropped.	

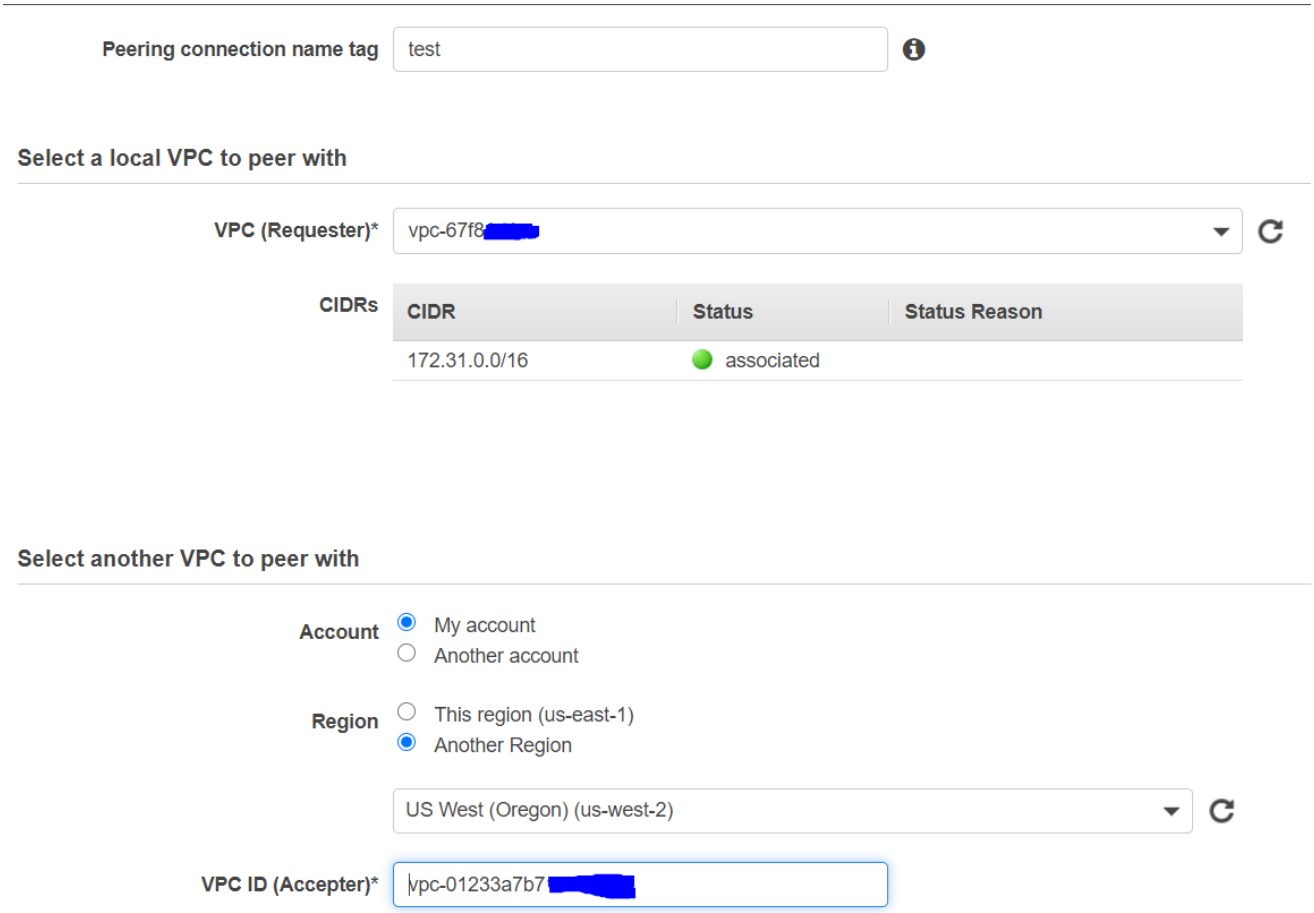
References:
https://docs.aws.amazon.com/vpc/latest/userguide/vpc-nat-comparison.html
https://docs.aws.amazon.com/vpc/latest/userguide/vpc-nat.html
https://tutorialsdojo.com/aws-cheat-sheet-amazon-vpc/

VPC Peering Setup

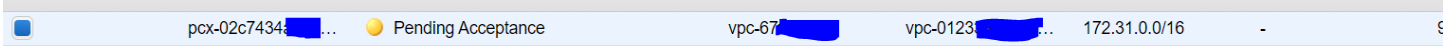
VPC peering is a common go-to solution for linking two VPC networks together. The solution is simple, effective, and does not cost anything to set up. Another advantage of VPC peering is that the connection is not a single point of failure and is not a bandwidth bottleneck unlike other VPC connection methods.

To create a VPC Peering connection with one of your VPCs, or another account's VPC, whether it be in the same region or another region, the steps are as follows:

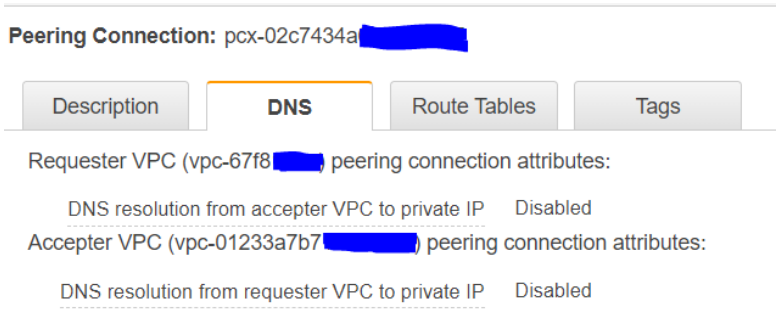
1) On your VPC console, create a peering request to your target VPC.
2) Indicate whether the target VPC is in the same account or another account, and whether in the same region or not.



3) Make sure that your target VPC CIDR does not overlap with your VPC.
4) Once the peering request is created, the target VPC will either accept or reject your peering request.



5) If you require DNS resolution between the two VPCs, you can enable them in your VPC peering settings.

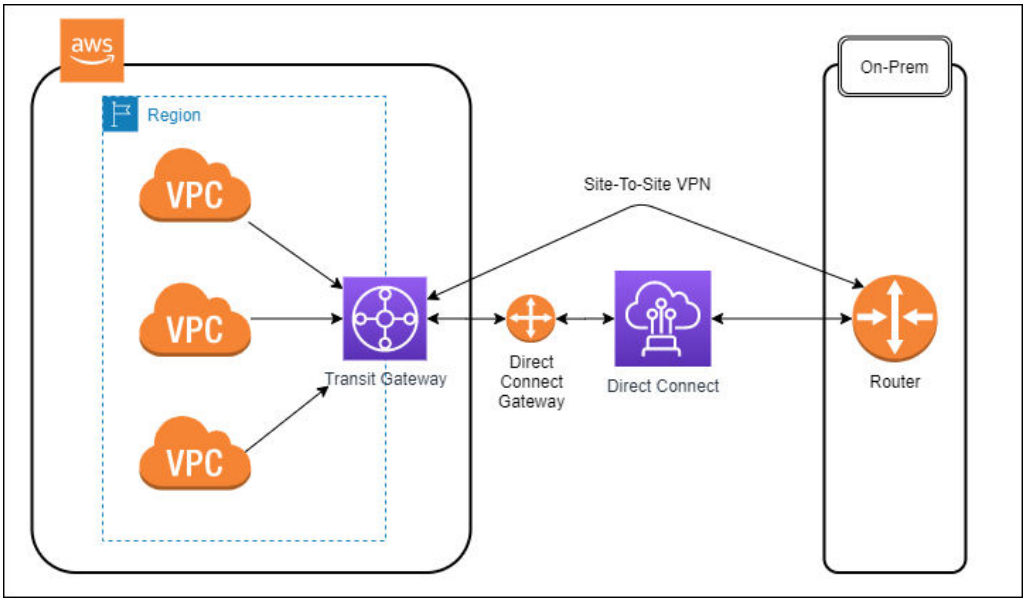


6) Once the target VPC accepts your peering request, you can now reference this connection in your route tables to specify which traffic needs to be routed over to the target VPC.

References:
https://docs.aws.amazon.com/vpc/latest/peering/create-vpc-peering-connection.html
https://tutorialsdojo.com/aws-cheat-sheet-amazon-vpc/

Utilizing Transit Gateway for Multi-VPC Connection

With VPC Peering, you can only connect two VPCs together. Managing multiple VPC Peering connections can be very troublesome when you have many interlinked VPCs. A better solution would be to use AWS Transit Gateway instead to handle these connections. AWS Transit Gateway requires little management overhead for managing multiple VPC connections. What's more, Transit Gateway lets you create Site-to-Site VPN solutions that are not possible with VPC Peering. Transit Gateway also works with Direct Connect line for hybrid environments, which would require a Direct Connect Gateway for it to work.



Adding CIDR Blocks to your VPC

When you create a VPC, you must provide a CIDR range that the VPC will use to allocate private IP addresses to your resources. In the event that you run out of IP addresses to allocate, you can expand your VPC by adding IPv4 CIDR blocks to it. When you associate a CIDR block with your VPC, a route is automatically added to your VPC route tables to enable routing within the VPC. Some restrictions to remember are:

- The CIDR block must not overlap with any existing CIDR block that's associated with the VPC.
- The allowed block size is between a /28 netmask and /16 netmask.
- You cannot increase or decrease the size of an existing CIDR block.
- You can disassociate secondary CIDR blocks that you've associated with your VPC; however, you cannot disassociate the primary CIDR block.

Reference:
https://docs.aws.amazon.com/vpc/latest/userguide/VPC_Subnets.html#vpc-resize

Amazon Route 53

Amazon Route 53 is the Domain Name System (DNS) web service that routes traffic to various targets. A domain name system, or DNS, is basically just a system that routes a domain name to a particular IP address. Route 53 allows you to map domain names to your Elastic IP addresses, EC2 instances, static S3 websites, Elastic Load Balancers, CloudFront distributions, and other AWS resources.

It also allows you to purchase and manage your custom web domains. So if you have a website called tutorialsdojo.com, you can route the traffic to an EC2 server, an S3 static website, a load balancer, or any AWS resource that you prefer. You can use a route policy, that allows you to customize how the traffic is routed to a specific domain. There are different route policies that you can choose, namely Simple, Failover, Geolocation, Geoproximity, Latency, Multivalue answer, or Weighted routing policy.

Route 53 for DNS and Domain Routing

Amazon Route 53 is a Domain Name System (DNS) web service that works similarly to other DNS providers out there such as CloudFlare and GoDaddy, with a few extra functionalities. You aren't required to use Route 53 as your DNS provider if you are using the AWS cloud, but since Route 53 is tightly integrated with other AWS services, you can always move from your current provider to enjoy these benefits. Route 53's primary functions can be summarized into four sections:

1. Domain registration
2. DNS management
3. Traffic management
4. Availability monitoring

Domain Registration

Since Route 53 is a domain registrar, you can certainly purchase and register your custom domain(s) through the service. Route 53 supports multiple top-level domains (TLD) with each having a corresponding price. You can also specify how many years you'd like to own the domain(s) before finalizing your purchase. Route 53 will then request for your contact details to keep you updated on the status of your domain purchase. Lastly, there is an option for some TLDs that allows you to automatically renew your domains before every expiration so you won't suddenly lose ownership of them. Once you've successfully purchased a domain, it should appear as a registered domain in Route 53.

If you have already purchased a domain before from another registrar, you can just transfer the ownership to Route 53. But when doing so, you should take note of the following:

- You might incur a transfer fee depending on the TLD being transferred.
- Expiration date may stay the same or may be extended depending on your TLD.
- Some registrars require you to have your domain registered with them for at least 60 days. If the registration for a domain name expired and had to be restored, it must have been restored at least 60 days ago.
- Make sure that the domain is transferable.
- Route 53 does not support all types of TLDs. Verify if the TLD is supported first before you initiate a transfer.

Similarly, if you can transfer domains into Route 53, then you can also transfer domains out of Route 53.

DNS Management

You may use Route 53 as your DNS service even if your domains are registered with a different domain registrar. It is able to resolve DNS queries to targets that are running inside and outside of AWS. In DNS management, everything starts at your hosted zones. A hosted zone is a container for DNS records, and these records contain information about how you want to route traffic for a specific domain. Hosted zones should have the same name as its associated domain. There are two types of hosted zones that you can create — **public hosted zone** and **private hosted zone**. The main difference between the two is, with public hosted zones, the records stored in them are publicly resolvable. On the other hand, private hosted zones contain records that are only resolvable within a VPC you associate, like if you want a record to resolve to a private EC2 instance for example.

In each public hosted zone, Route 53 automatically creates a name server (NS) record and a start of authority (SOA) record. Afterwards, you can create additional records in this hosted zone to point your domain and subdomains to their endpoints. If you are moving from an existing DNS service, you can also import a zone file instead to automatically populate your hosted zone. Be sure to modify the NS records of the DNS service to use the name servers of AWS. Once you've performed the actions above, just wait for DNS queries to come in (and wait for the DNS cache TTL to expire if the records were existing beforehand), and they should resolve to your designated targets.

For private hosted zones, DNS resolution is handled a bit differently. When you create a VPC, Route 53 Resolver automatically answers DNS queries for local VPC domain names of EC2 instances and records in private hosted zones. For all other domain names, Route 53 Resolver performs recursive lookups against public name servers. You can also integrate DNS resolution between Resolver and DNS resolvers on your network by configuring forwarding rules. Before you can start forwarding queries, you must create a Resolver inbound and/or outbound endpoint in the associated VPC.

- An inbound endpoint lets DNS resolvers on your network forward DNS queries to Route 53 Resolver via this endpoint.
- An outbound endpoint lets Route 53 Resolver conditionally forward queries to resolvers on your network via this endpoint.

There are multiple types of records that you can create in Route 53, but the most common ones you'll encounter are A record, AAAA record, and CNAME record. Furthermore, each of these records can be alias or non-alias records. A non-alias record means you just need to enter your targets' IP addresses or domain names and the TTL for the record. An alias record is a Route 53-specific feature that lets you specify your AWS resources as the target instead of an IP address or a domain name. When you use an alias record to route traffic to an AWS resource, there is no TTL to set; Route 53 automatically recognizes changes in the resource. Unlike a CNAME record, you can create an alias record at the zone apex. For example, an Alias A record can route traffic to the following targets:

1) Another A record in your hosted zone
2) API Gateway API
3) CloudFront distribution
4) Elastic Beanstalk environment
5) Application, Network and Classic Load Balancer
6) Global Accelerator
7) S3 web endpoint
8) VPC endpoint

Traffic Management

Each Route 53 DNS record also has its own routing policy. A routing policy determines how Route 53 responds to DNS queries. Different routing policies achieve different results:

- **Simple routing policy** – Resolves your DNS to a resource as is.
- **Failover routing policy** – Use for configuring active-passive routing failover. You can specify two DNS records with the same DNS name and have them point to two different targets. If your primary target becomes unavailable, Route 53 automatically routes succeeding incoming requests to your secondary target.
- **Geolocation routing policy** – Use when you want to route traffic based on the location of your users. This policy helps you serve geolocation-specific content to your users.
- **Geoproximity routing policy** – Use when you want to route traffic based on the location of your resources and, optionally, shift traffic from resources in one location to resources in another.
- **Latency routing policy** – Use when you have resources in multiple AWS Regions and you want to route traffic to the region that provides the best latency.
- **Weighted routing policy** – Use to route traffic to multiple resources in proportion to the weights you assign for each target. The greater the weight, the greater the traffic portion it receives. This policy can be used when you've deployed a new version of an application and you only want to route a percentage of your user traffic to it.

- **Multivalue answer routing policy** – Use when you want Route 53 to respond to DNS queries with up to eight healthy records selected at random. Users who query this type of record can choose a target from the DNS response to connect to.

Some of these routing policies can actually be used together, such as latency and weighted records, to produce a more complex routing system.

Availability Monitoring

The last primary feature of Route 53 is monitoring the health of your endpoints and taking the necessary steps in reducing DNS resolution downtime. A Route 53 health check can monitor any of the following:

- The health of a resource, such as a web server
- The status of other health checks
- The status of an Amazon CloudWatch alarm

Route 53 health check supports multiple types of network protocols for monitoring your targets. If you are familiar with the health check of an elastic load balancer, it's pretty much the same as a Route 53 health check. You indicate the network protocol, port, target and path of the health check, and optionally the check interval, failure threshold, and originating Regions of the health check requests.

You can use HTTP, HTTPS, or TCP for the network protocol, and even configure Route 53 to search for a specific string in the response body to determine if the response is good or not. Furthermore, you can invert the status of a health check, meaning Route 53 considers health checks to be unhealthy when the status is healthy and vice versa. After you create a health check, you can view the status of the health check, get notifications when the status changes via SNS and Cloudwatch Alarms, and configure DNS failover in response to a failed health check.

References:

https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/registrar.html
https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/dns-configuring.html
https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/dns-failover.html

Latency Routing vs Geoproximity Routing vs Geolocation Routing

Latency Routing	
Definition	Lets Route 53 serve user requests from the AWS Region that provides the lowest latency. It does not, however, guarantee that users in the same geographic region will be served from the same location. Latency-based routing is based on latency measurements performed over a period of time, and the measurements reflect changes in network connectivity and routing.
How it works	To use latency-based routing, you create **latency records** for your resources in multiple AWS Regions. When Route 53 receives a DNS query for your domain or subdomain, it determines which AWS Regions you've created latency records for, determines which region gives the user the lowest latency, and then selects a latency record for that region. Route 53 responds with the value from the selected record, such as the IP address for a web server. Record sets can be created using any record type supported by Route 53, except NS or SOA records.
Use Case	Use when you have **resources in multiple AWS Regions** and you want to route traffic to the **region that provides the best latency**

Geoproximity Routing

Definition	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.
How it works	To use geoproximity routing, you must use **Route 53 traffic flow**. You create traffic flow policies for your resources and specify one of the following values for each policy: - If you're using AWS resources, you can set the AWS Region where your resource is created - If you're using non-AWS resources, you can enter the latitude and longitude of the resource
Use Case	Use when you want to route traffic **based on the location of** your resources and, optionally, shift traffic from resources in one location to resources in another.

Geolocation Routing

Definition	Resources serve traffic based on the geographic location of your users, meaning the location that DNS queries originate from.
How it works	Geolocation works by **mapping IP addresses to locations.** Some IP addresses aren't mapped to geographic locations, so Amazon Route 53 will receive some DNS queries from locations that it can't identify. You can create a default record that handles both queries from IP addresses that aren't mapped to any location and queries that come from locations that you haven't created geolocation records for. If you don't create a **default record**, Route 53 returns a "no answer" response for queries from those locations. No two records should specify the same geographic location.
Use Case	Use when you want to route traffic **based on the location of your users.** - You can localize your content and present some or all of your website in the language of your users. - You can restrict distribution of content to only the locations in which you have distribution rights. - Useful for balancing load across endpoints in a predictable, easy-to-manage way, so that each user location is consistently routed to the same endpoint.

Active-Active Failover and Active-Passive Failover

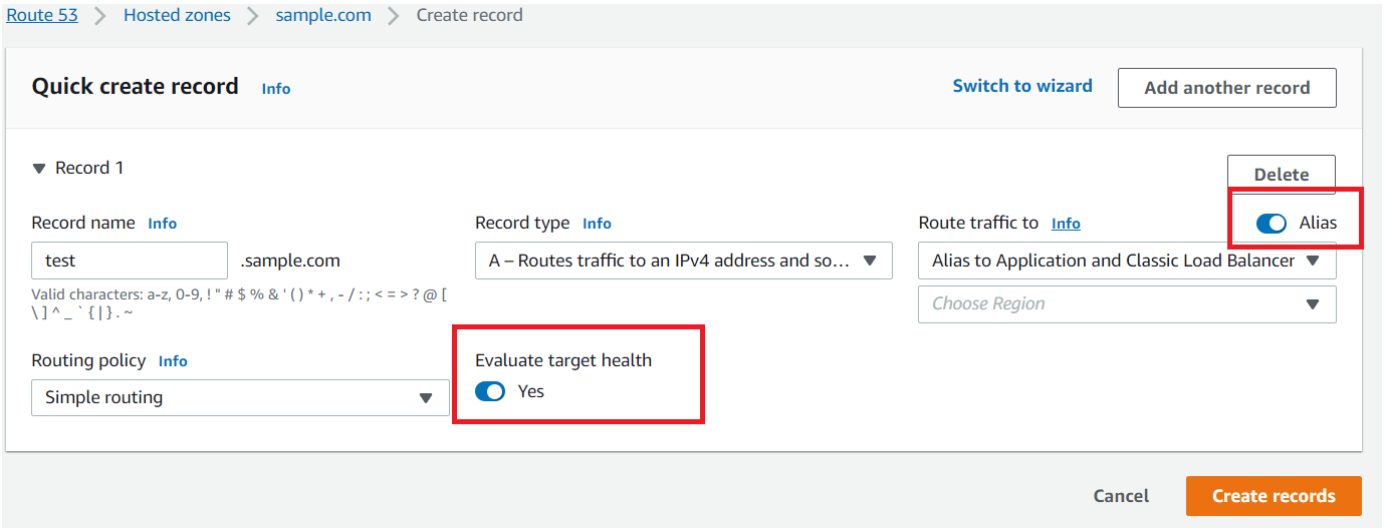
All types of systems nowadays need to implement some sort of redundancy and high availability to ensure business continuity. We'll never know when the next outage might occur, so by planning beforehand and developing solutions that consider the worst possible scenarios, we can create a highly resilient architecture that can achieve near 100% uptime.

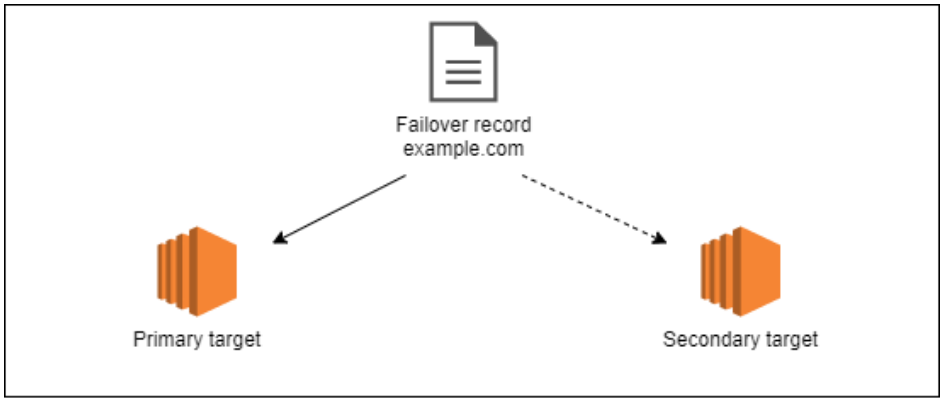
Hence, you should have a failover plan for every component of your system, and that includes your DNS services. AWS makes it very convenient for us to create solutions that focus on high availability and fault tolerance. In Route 53, AWS handles the availability of the service while you manage the policies that ensure your website's availability. Route 53 uses health checks to monitor the availability of your DNS targets. And there are two ways you can approach failovers in Route 53: active-active failover and active-passive failover.

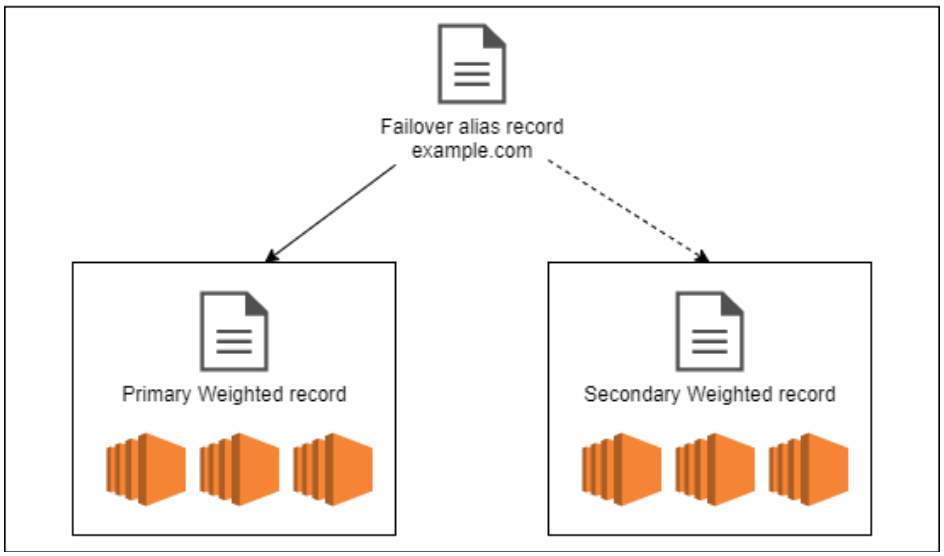
In an active-active failover setup, all DNS records that contain the same DNS name, the same record type (A, AAAA, CNAME, etc), and the same routing policy (simple, latency, weighted) are considered as active and queryable unless Route 53 marks them as unhealthy due to a health check. You can create multiple DNS records that have the same configuration but different targets in the same hosted zone. Route 53 will use any of these healthy records to respond to a DNS query.

Active-passive failover, on the other hand, uses the failover routing policy to handle DNS failovers. You'll be creating two failover alias records, one primary and one secondary, that are referencing your primary and secondary endpoints respectively. DNS queries are routed to your primary records for as long as their endpoints are healthy. In the event that your primary becomes unavailable, Route 53 will automatically respond to DNS queries using your secondary (failover). To create an active-passive failover configuration with one primary record and one secondary record, you just create the records and specify Failover for the routing policy. You can also associate multiple resources with the primary record, the secondary record, or both. Route 53 considers the primary failover record to be healthy as long as at least one of the associated resources is healthy.

If you are using Alias records for your primary and/or secondary records, there's no need for you to create manual health checks for those resources; just set Evaluate Target Health option in the record to Yes instead. For other record types, you will need to create manual health checks.







References:
https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/dns-failover-types.html

https://aws.amazon.com/premiumsupport/knowledge-center/route-53-dns-health-checks/
https://tutorialsdojo.com/amazon-route-53/

Route 53 DNSSEC

Domain Name System Security Extensions, or DNSSEC, is a protocol for securing DNS traffic. It prevents attackers from hijacking traffic to internet endpoints by intercepting DNS queries and returning their own IP addresses to DNS resolvers, known as DNS spoofing. When you configure DNSSEC for your domain, a DNS resolver establishes a chain of trust for responses from intermediate resolvers. The chain of trust begins with the top-level domain registry for the domain and ends with the authoritative name servers at your DNS service provider. To configure DNSSEC for a domain, your domain and DNS service provider must meet the following prerequisites:

1. The registry for the TLD must support DNSSEC.
2. The DNS service provider for the domain must support DNSSEC. Route 53 supports DNSSEC signing as well as DNSSEC for domain registration.
3. You must configure DNSSEC with the DNS service provider for your domain before you add public keys for the domain to Route 53. Configuring DNSSEC in Route 53 involves two steps:
    a. Enable DNSSEC signing for Route 53, and have Route 53 create a key signing key (KSK) based on a customer managed CMK in AWS KMS.
    b. Create a chain of trust for the hosted zone by adding a Delegation Signer (DS) record to the parent zone, so DNS responses can be authenticated with trusted cryptographic signatures.
4. If you've configured DNSSEC with a different DNS service provider for the domain, you must add the public encryption keys to Route 53.
    a. In Route 53, under *Registered domains*, choose the name of the domain that you want to add keys for.
    b. At the *DNSSEC* status field, choose *Manage keys*.
    c. Specify the key type - key-signing key (KSK) or zone-signing key (ZSK).
    d. Specify the algorithm that you used to sign the records for the hosted zone.
    e. Specify the public key of the key pair that you used to configure DNSSEC.
    f. Click on Add to finish.

References:
https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/domain-configure-dnssec.html
https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/dns-configuring-dnssec.html

AWS Elastic Load Balancing

Elastic Load Balancing or ELB, is a networking service that automatically distributes incoming traffic across multiple targets such as Amazon EC2 instances, containers, Lambda functions, and other components. As its name implies, this service is primarily used in "load balancing", or in the other words, distributing the incoming traffic load to multiple underlying servers. So instead of just having a single server to process the load, you can launch multiple servers to handle the application requests and to provide high availability. With an ELB, you can route the traffic to healthy resources running in different Availability Zones.

There are different types of Elastic Load Balancers that you can use namely: Application Load Balancer or ALB, Network Load Balancer or NLB, Gateway Load Balancer or GWLB, and Classic Load Balancer or CLB. An ALB is suitable for load balancing HTTP and HTTPS traffic to your application servers, microservices, and containers. NLB is recommended for load balancing TCP, UDP, and TLS traffic to your applications. It can also handle millions of requests per second while maintaining ultra-low latencies. GWLB is primarily used for deploying, scaling, and running third-party virtual appliances. These virtual appliances can be your custom firewalls, deep packet inspection systems or intrusion detection and prevention systems in AWS. A Gateway Load Balancer is mainly using the Internet Protocol to pass the OSI Layer 3 traffic to its registered targets. Finally, CLB is intended for applications that were built within the old EC2-Classic network. It is still being used today for applications with custom security policies and TCP passthrough configuration.

AWS ELB Request Routing Algorithms

You might have heard of a load balancer before, and you might already know what its purpose is, but are you familiar with how an AWS Elastic Load Balancer routes web requests to your targets?

We know that there are different variations of AWS ELBs, but for this section, we will just focus on these three types: Application Load Balancer, Network Load Balancer and Classic Load Balancer. Each of these types have their own routing procedures which we will elaborate below.

Application Load Balancer Routing	Network Load Balancer Routing	Classic Load Balancer Routing
1. When the load balancer receives a request, it first evaluates the listener rules in priority order to determine which rule to apply. Recall that listener rules specify how requests will be routed to appropriate targets. 2. Once a matching rule is found,	1. When the load balancer receives a request, it selects a target from the target group with a matching listener rule using **flow hash algorithm**. Flow hash algorithm checks on the following parameters: • The protocol • The source IP address and	1. This load balancer routes TCP requests to targets using round robin algorithm. 2. For HTTP and HTTPS requests, it uses the least outstanding requests algorithm.

the load balancer uses a routing algorithm to select a target from the target group for the rule action. The default routing algorithm is **round robin**. 3. Round robin algorithm attempts to distribute requests evenly to all targets by having each target take turns in receiving a request. 4. Another routing algorithm you can use for ALB is the **least outstanding requests algorithm**. Least outstanding requests algorithm is an algorithm that forwards incoming requests to targets with the lowest number of requests at that moment.	source port ● The destination IP address and destination port ● The TCP sequence number 2. The load balancer then routes each individual TCP connection to a single target for as long as the connection is alive, meaning once a TCP connection to a target has been established, NLB will keep using this connection for succeeding requests directed to this target.	

References:
https://docs.aws.amazon.com/elasticloadbalancing/latest/userguide/how-elastic-load-balancing-works.html#request-routing

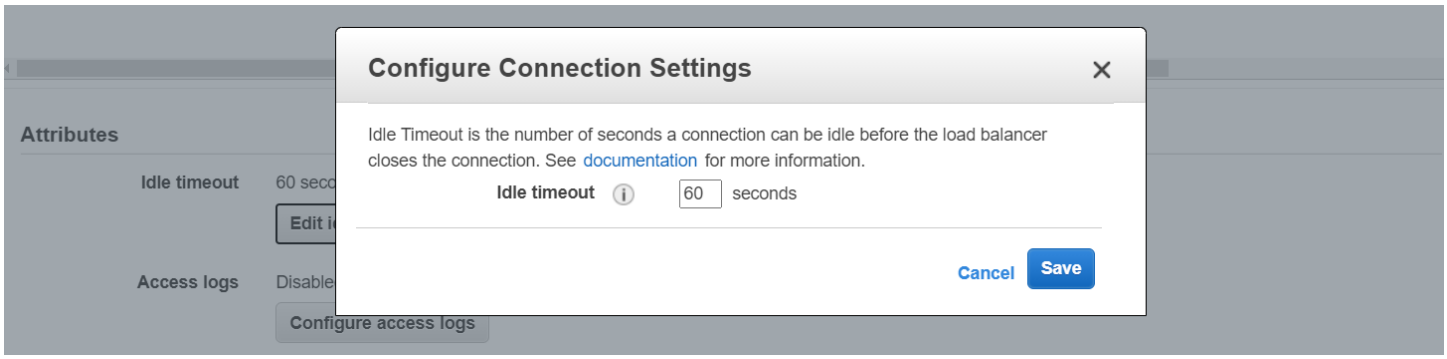
ELB Idle Timeout

For every request that arrives at an ELB, the load balancer establishes two connections: one with the client application, and another one with the target destination. To make sure that these connections are only kept alive for as long as they are in use, your load balancer has an idle timeout period that monitors the state of these connections. An ELB idle timeout is the number of seconds that a connection has to send new data to keep the connection alive. Once the period elapses and there has been no transfer of new data, the load balancer closes the connection. This allows new connections to be established without using up all your connection resources. For network operations that take a long time to complete, you should send at least one byte of new data before your idle timeout elapses to maintain the connection.

The default idle timeout for load balancers is set at 60 seconds. You can modify the idle timeout period of classic and application load balancers if you need a much longer period, but do note that having a longer idle timeout might make it easier to reach the maximum number of connections for your load balancer. The maximum timeout period you can configure is 4000 seconds or 1 hour 6 minutes and 40 seconds. Network

load balancers set the idle timeout value for TCP flows to 350 seconds. You cannot modify this value. Clients or targets can use TCP keepalive packets to reset the idle timeout.

Just to note. Setting the idle timeout to a higher number may be useful for some scenarios, but not all of them. When you are keeping a connection alive just to wait for a response from a long-running process, you should consider refactoring your applications to use asynchronous transmissions instead, or create a pipeline to decouple the response from the load balancer. Remember that, as a Solutions Architect, you should be designing the best solution for a given problem.

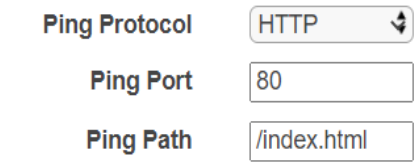
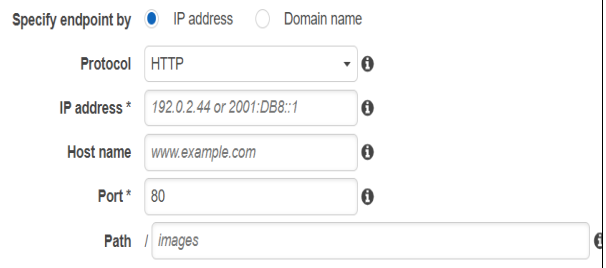
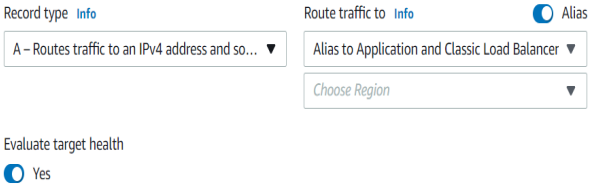
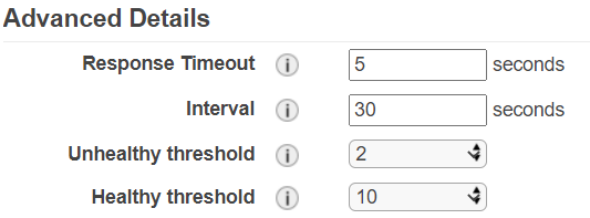


References:
https://docs.aws.amazon.com/elasticloadbalancing/latest/classic/config-idle-timeout.html
https://docs.aws.amazon.com/elasticloadbalancing/latest/application/application-load-balancers.html#connection-idle-timeout

ELB Health Checks vs Route 53 Health Checks For Target Health Monitoring

We all know that health checks are a very useful tool for making sure that AWS services such as AWS ELB and Amazon Route 53 know the state of their targets before forwarding traffic to them. In this section, we will take a look at ELB health checks and Route 53 health checks, and compare them with one another.

Health Check Service	AWS Elastic Load Balancing	Amazon Route 53
What is it for?	This health check periodically sends a request to a target instance, server or function to verify its status i.e. available to accept traffic requests.	This health check monitors the state of a record's target, which can be an EC2 instance, a server, or an AWS service that has an endpoint.
Target health check settings	You enter the port and common path of your targets that the load balancer will send the	You enter the domain name or the IP address, port, and path that Route 53 will use to send

	health check request to. 	the health check request to if the record is a non-alias record,  or by setting *Evaluate target health* to Yes if the record is an alias record. 
Area span	Load balancers can monitor targets that span multiple availability zones but not multiple regions.	Route 53 monitors your targets regardless of their location, as long as they are reachable by Route 53.
Health check frequency	You specify a value between 5 seconds and 300 seconds	Choose either every 10 seconds or every 30 seconds.
Response timeout	You can enter a value between 2 seconds and 60 seconds.	Cannot be configured.
Criteria to pass health check	You specify a threshold that a target should pass/fail a health check to determine its status. 	If more than 18% of health checkers report that an endpoint is healthy, Route 53 considers it healthy. If 18% of health checkers or fewer report that an endpoint is healthy, Route 53 considers it unhealthy. Route 53 health check servers are located in different locations worldwide.
Accessibility	Make sure targets are reachable by the load balancer. New targets can be easily added and removed from the load balancer.	Make sure endpoints are reachable and resolvable when users hit your URL. Due to DNS caching, it may take a while for new

		target endpoints to reflect to end users.
Primary purpose	High availability and fault tolerance for your services	DNS failover routing

There is no rule saying that you cannot use these two health checks together. In fact, it is a better practice to use them both! Amazon ELB will make sure that your traffic will only be handled by healthy targets, and Amazon Route 53 will make sure that your records have endpoints that are reachable and resolvable. Use different Route 53 record types and routing policies to perform an automatic DNS failover when an endpoint suddenly becomes unavailable, and control how the failover should occur.

References:
https://aws.amazon.com/blogs/aws/amazon-route-53-elb-integration-dns-failover/
https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/dns-failover.html
https://docs.aws.amazon.com/elasticloadbalancing/latest/classic/elb-healthchecks.html

Application Load Balancer vs Network Load Balancer vs Gateway Load Balancer

Feature	Application Load Balancer	Network Load Balancer	Gateway Load Balancer
Protocols	HTTP, HTTPS, gRPC	TCP, UDP, TLS	IP
Platforms	VPC	VPC	VPC
Health checks	HTTP, HTTPS, gRPC	TCP, HTTP, HTTPS	TCP, HTTP, HTTPS
Cloudwatch Metrics	✓	✓	✓
Logging	✓	✓	✓
Zonal Failover	✓	✓	✓
Connection Draining (deregistration delay)	✓	✓	✓
Load Balancing to multiple ports on the same instance	✓	✓	✓
IP addresses as targets	✓	✓ (TCP, TLS)	✓
Load balancer deletion protection	✓	✓	✓
Configuration idle connection timeout	✓		
Cross-zone load balancing	✓	✓	✓
Sticky sessions	✓	✓	✓
Static IP		✓	
Elastic IP address		✓	
Preserve Source IP address	✓	✓	✓
Resource-based IAM permissions	✓	✓	✓
Tag-based IAM permissions	✓	✓	✓
Slow start	✓		
Web sockets	✓	✓	✓
PrivateLink Support		✓ (TCP, TLS)	✓ (GWLBE)
Source IP address CIDR-based routing	✓		

Feature	Application Load Balancer	Network Load Balancer	Gateway Load Balancer
Layer 7			
Path-based routing	✓		
Host-based routing	✓		
Native HTTP/2	✓		
Redirects	✓		
Fixed response	✓		
Lambda functions as targets	✓		
HTTP header-based routing	✓		
HTTP method-based routing	✓		
Query string parameter-based routing	✓		
Security			
SSL offloading	✓	✓	
Server Name Indication (SNI)	✓	✓	
Back-end server encryption	✓	✓	
User authentication	✓		
Session Resumption	✓	✓	
Terminates flow/proxy behavior	✓	✓	✓

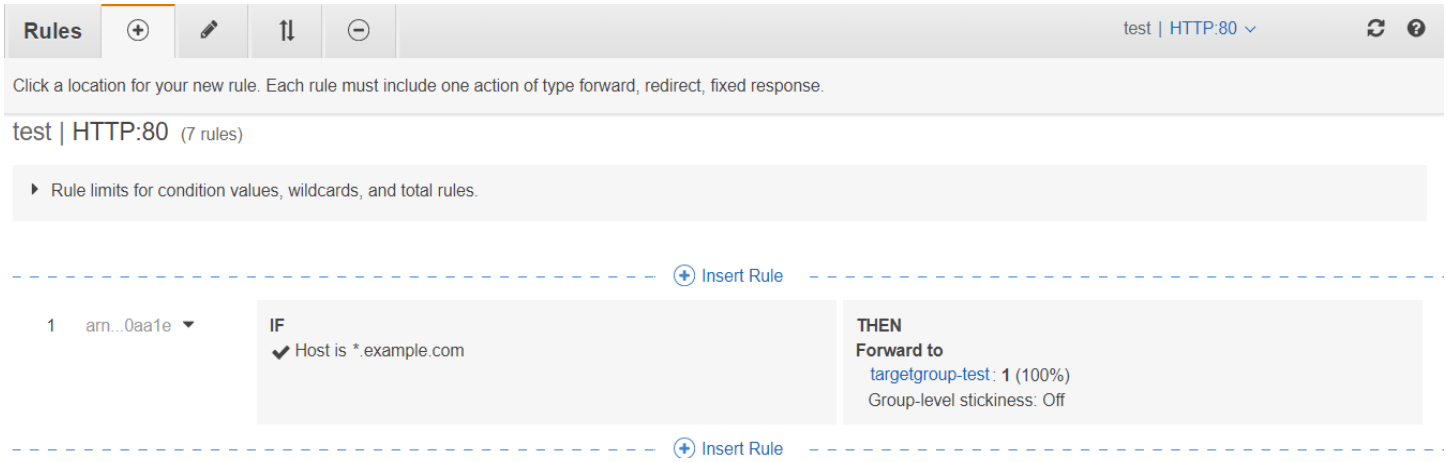
Application Load Balancer Listener Rule Conditions

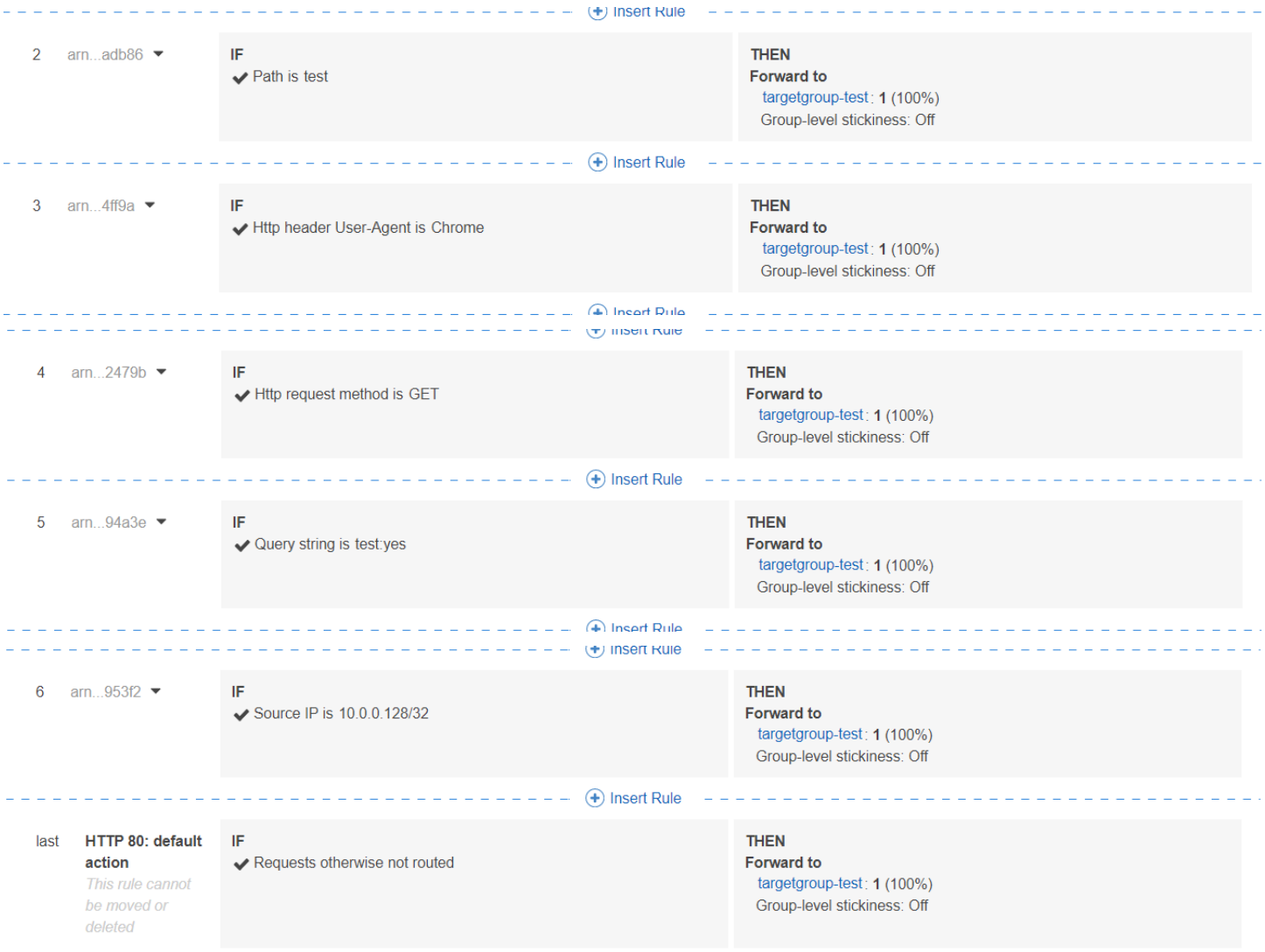
The AWS ELB Application Load Balancer is one of the most innovative services you can find in AWS. It offers many unique routing features that cannot be found in other types of elastic load balancers. But before we talk about listener rule conditions, let's first refresh ourselves with what listeners and listener rules are. A *listener* is a process that checks for incoming connection requests, 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.

You can add the following conditions to a listener rule to create multiple routing paths under a single load balancer:

- **host-header** — Route based on the host name of each request. Also known as host-based routing. This condition enables you to support multiple subdomains and different top-level domains using a single load balancer. Hostnames and match evaluations are not case-sensitive.
- **http-header** — Route based on the HTTP headers for each request. Standard and custom headers are supported. Header name and match evaluation are not case-sensitive.
- **http-request-method** — Route based on the HTTP request method of each request. You can specify standard or custom HTTP methods for the value. The match evaluation is case-sensitive, so to properly route requests to this condition, the request method must exactly match the value you've entered.
- **path-pattern** — Route based on path patterns in the request URLs. Also known as path-based routing. This condition allows you to route to multiple targets depending on the URL path supplied in the request. URL path does not include the query parameters. Path evaluation is case-sensitive.
- **query-string** — Route based on key/value pairs or values in the query strings. Match evaluation is not case-sensitive. This condition does not include the URL path in the evaluation.
- **source-ip** — Route based on the source IP address of each request. The IP address must be specified in CIDR format. Both IPv4 and IPv6 addresses are supported as values for this condition. If a client is behind a proxy, the condition evaluates the IP address of the proxy, not the IP address of the client.

A listener rule can include up to one of each of the following conditions: host-header, http-request-method, path-pattern, and source-ip; and include one or more of each of the following conditions: http-header and query-string. You can also specify up to three match evaluations per condition, but only up to five match evaluations per rule. This gives you more values to work with for each condition you create.





References:

https://docs.aws.amazon.com/elasticloadbalancing/latest/application/load-balancer-listeners.html#rule-condition-types

https://tutorialsdojo.com/aws-elastic-load-balancing-elb

Amazon CloudFront

Amazon CloudFront is a content delivery network service that quickly delivers static content and video stream to your clients. A content delivery network, or CDN, is a globally-distributed network of servers spread around the globe that stores or caches your files. It shortens the time to deliver your data which improves the response time of your application. Instead of fetching the data from one origin server that is hosted in a single location, your clients can retrieve your content from multiple edge locations located near them. You can set up a CloudFront distribution to cache your images, videos, media files, or software downloads of your websites.

CloudFront has three basic components which are the origin, the distribution, and the viewer. Basically, an origin is where the content originates or comes from. The distribution is an actual AWS resource that you can launch and configure, to control how to distribute your content to your users. Lastly, a viewer is the actual website visitors or users who view your content.

In your CloudFront distribution, you can set an Amazon S3 bucket, an Elastic Load Balancer, an AWS Elemental service, or an HTTP server running in an Amazon EC2 instance or in another external host as the origin. You can also set your on-premises server as the custom origin of your distribution.

Amazon CloudFront has a lot of caching and security features that you can use to distribute your content effectively and securely. For example, if your origin is an Amazon S3 bucket, you can restrict access to S3 content by using an origin access identity user or OAI. You can restrict access to your content based on the country or geographic location of your viewers. This can be done with its geo-restriction feature that lets you control the distribution of your content at the country level.

Amazon CloudFront has a feature called Lambda@Edge, which lets you do certain computations in proximity to your users, and not just deliver static content. Lambda@Edge allows you to run your custom code closer to the end-users of your application to improve your application performance and reduce latency. Since the traffic doesn't go to the underlying origin server, this saves you from high Data Transfer costs as well.

You can also improve the availability of your web applications with CloudFront by adding two origins, instead of one. You can set up an origin failover by creating an origin group that contains your primary and secondary resources. If your primary origin is unavailable, the traffic will failover to the secondary origin.

Serving private content to specific users is also possible with its signed URLs and signed cookies features.

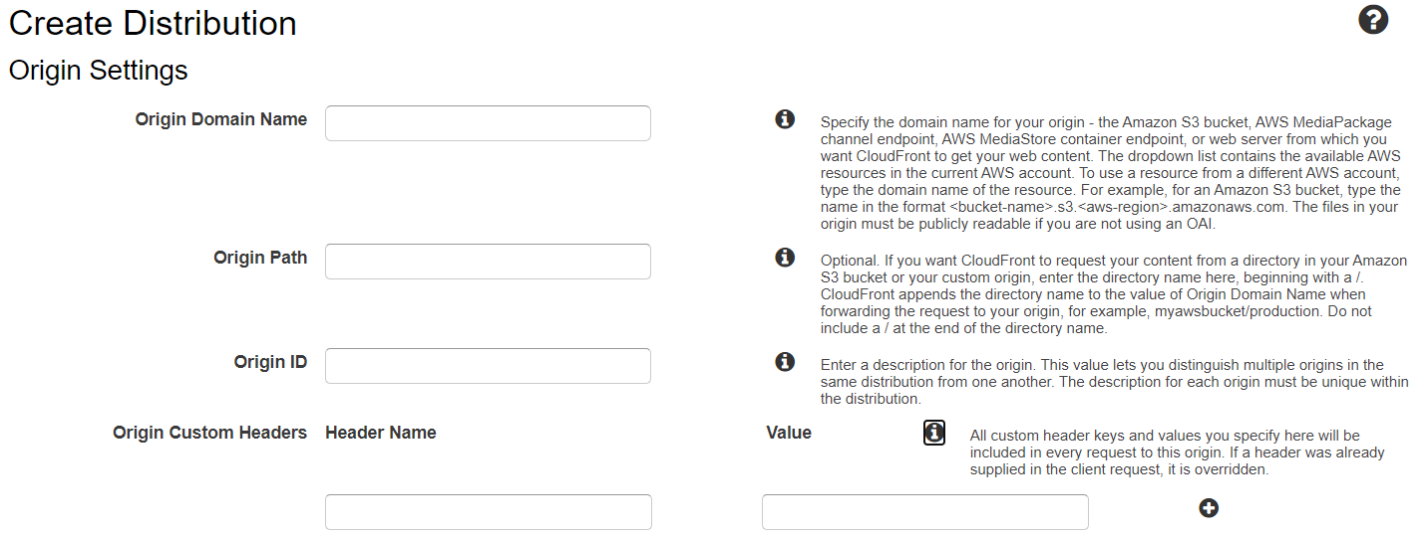
You can distribute your content securely via HTTPS by adding an SNI Custom SSL or a Dedicated IP Custom SSL. AWS WAF can be integrated as well with your CloudFront distribution to safeguard your application from common web vulnerabilities. CloudFront has a lot of other features that you can use for your workloads.

Custom DNS Names with Dedicated SSL Certificates for your CloudFront Distribution

Perhaps you have a set of EC2 web servers running behind an elastic load balancer serving your public website, and your website's DNS name is pointing directly to your load balancer in Route 53. This is the most common architecture you can build in the cloud. Although this architecture is absolutely fine as it is, there are still some areas you can improve upon. One of which is by placing a CDN (content delivery network) service such as Amazon CloudFront before your load balancer.

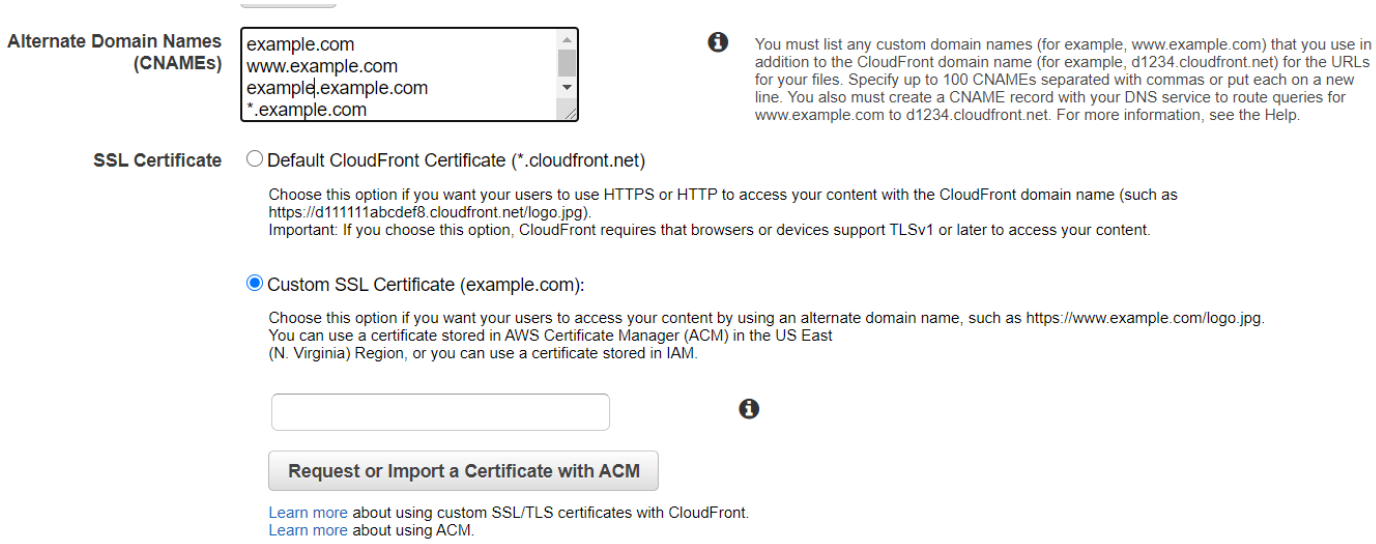
“Why?” you might ask. Amazon CloudFront is able to provide multiple benefits to your website. You can use CloudFront to have a better global reach since it's powered by AWS' global edge network. You can have CloudFront cache frequently requested objects from your website to speed up loading times for your users, while at the same time alleviating the burden from your web servers and databases from serving the same objects over and over again. It can also protect your website from security attacks such as DDoS since CloudFront introduces an extra layer before your actual architecture. You can also add in a WAF for additional security measures. These benefits sound great for any business that relies heavily on their website's performance. And here's how you can add a CloudFront to your architecture and repoint your domain name.

When you're creating a CloudFront distribution, you'll need to enter your origin domain name, which is the origin that CloudFront will use to serve requests. In this scenario, the origin domain name is the public DNS name of your elastic load balancer. You can also optionally provide an origin path if you want CloudFront to request your content from a specific directory in your custom origin. Next, you provide a custom origin ID so you can easily identify your custom origin. An origin ID is required since a single CloudFront distribution can support multiple origins and route requests to specific origins depending on the behavior that you define. For example, if the path pattern for a request includes */images/*.jpg*, you can tell CloudFront to route these requests to origin B and route everything else to origin A.



It is a good practice to always use HTTPS for your public websites, and you can enforce this in CloudFront, either by redirecting all HTTP requests to HTTPS or by allowing HTTPS requests only in the viewer protocol policy.

Each CloudFront distribution automatically generates a unique, publicly resolvable DNS endpoint for itself similar to an ELB. You can also list additional alternate domain names for your distribution. This enables your users to access your CloudFront using friendlier domain names. If you are enforcing HTTPS and you do not provide an alternate domain name for your CloudFront distribution, AWS lets you use the default CloudFront SSL certificate (*.cloudfront.net). But if you do provide alternate domain names for your CloudFront, you can utilize your own custom SSL certificates. The SSL certificate must be in AWS Certificate Manager (ACM) but doesn't necessarily have to be issued by ACM. You can import your own SSL certificate to ACM and it will work just fine.



For each origin, you can add multiple alternate domain names as long as they are supported by your custom SSL certificate. If you enter manilaph.com and manilaph1.com as alternate domain names, and manilaph1.com is not associated with your SSL certificate, the distribution will fail to launch. The domain names you enter can be parent domains, subdomains or wildcard domains.

Lastly, adding in your alternate domain names will not make them resolve automatically to your CloudFront distribution. You will also have to create the necessary DNS records for each of your alternate domain names in the appropriate hosted zones in Route 53 or any external DNS service you are using. If your hosted zone is in Route 53, you may create alias records to point the DNS records to your CloudFront. If you are using an external

DNS service, you may create CNAME records and point them to the CloudFront-generated public DNS endpoint (*.cloudfront.net). In our scenario, the custom domain name was already pointing to your load balancer beforehand. Simply modify the record's target to point to your CloudFront and wait for the DNS cache to refresh.

Once you've created your CloudFront distribution and made the necessary changes in Route 53, requests to your website will now be handled by CloudFront. CloudFront searches for the correct destination origin to route these requests, and optionally caches the origin's response if you've configured caching. You can monitor the status of your CloudFront and your website's performance in Amazon Cloudwatch. Furthermore, you can enable logging for your CloudFront which logs all the requests that it receives and stores the logs in an Amazon S3 bucket.

References:
https://aws.amazon.com/premiumsupport/knowledge-center/multiple-domains-https-cloudfront/
https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/using-https-alternate-domain-names.html
https://tutorialsdojo.com/amazon-cloudfront/

Restricting Content Access with Signed URLs and Signed Cookies

Sometimes, developers would like to add a CloudFront to their applications due to the benefits that the service provides, but these applications are not to be shared with the public. Take an S3 bucket for example. To prevent users from accessing your objects directly from the bucket, you'd place a CloudFront in front of the S3 bucket and have the users use CloudFront to access your objects. In this scenario, one potential security concern is that if your CloudFront URL got exposed to a third-party user, he or she will be able to access the same objects as well. To prevent this from happening, CloudFront has a neat feature that lets you securely serve private content to select users only. You can configure CloudFront to allow users to access your files using either *signed URLs* or *signed cookies* only.

When you create signed URLs or signed cookies to control access to your files, you can specify the following restrictions:

- An ending date and time, after which the URL is no longer valid.
- (Optional) The date and time that the URL becomes valid.
- (Optional) The IP address or range of addresses of the computers that can be used to access your content.

Part of a signed URL or a signed cookie is hashed using RSA-SHA1 algorithm and signed using the private key from an asymmetric key pair. When someone uses the signed URL or signed cookie, CloudFront compares the signed and unsigned portions of the URL or cookie. If they don't match, CloudFront doesn't serve the file.

Now what is the difference between signed URLs and signed cookies, and which one should you use? In a basic sense, they both provide the same functionality. Use signed URLs if you want to restrict access to individual files, or if your users are using a client that doesn't support cookies. Use signed cookies if you want to provide access to multiple restricted files, or if you don't want to change your current URLs. If your current URLs contain any of the following query string parameters, you cannot use either signed URLs or signed cookies:

- Expires
- Policy
- Signature
- Key-Pair-Id

CloudFront first checks your URLs for presence of any of the query parameters above. If any of them is present, CloudFront assumes that the URLs are signed URLs even if you haven't intended them as such, and therefore won't check for signed cookies.

Before you can create signed URLs or signed cookies, you need a signer. A signer is either a trusted key group that you create in CloudFront, or an AWS account that contains a CloudFront key pair. As soon as you add the signer to your CloudFront distribution, CloudFront starts requiring viewers to use signed URLs or signed cookies to access your files. There might be cases wherein you don't want all your content to be accessed this way. Hence, you can create multiple cache behaviors in your distribution and only associate the signer with some of them. This allows you to require signed URLs or signed cookies for some files and not for others in the same distribution.

References:
https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/PrivateContent.html
https://tutorialsdojo.com/amazon-cloudfront/
https://tutorialsdojo.com/s3-pre-signed-urls-vs-cloudfront-signed-urls-vs-origin-access-identity-oai/

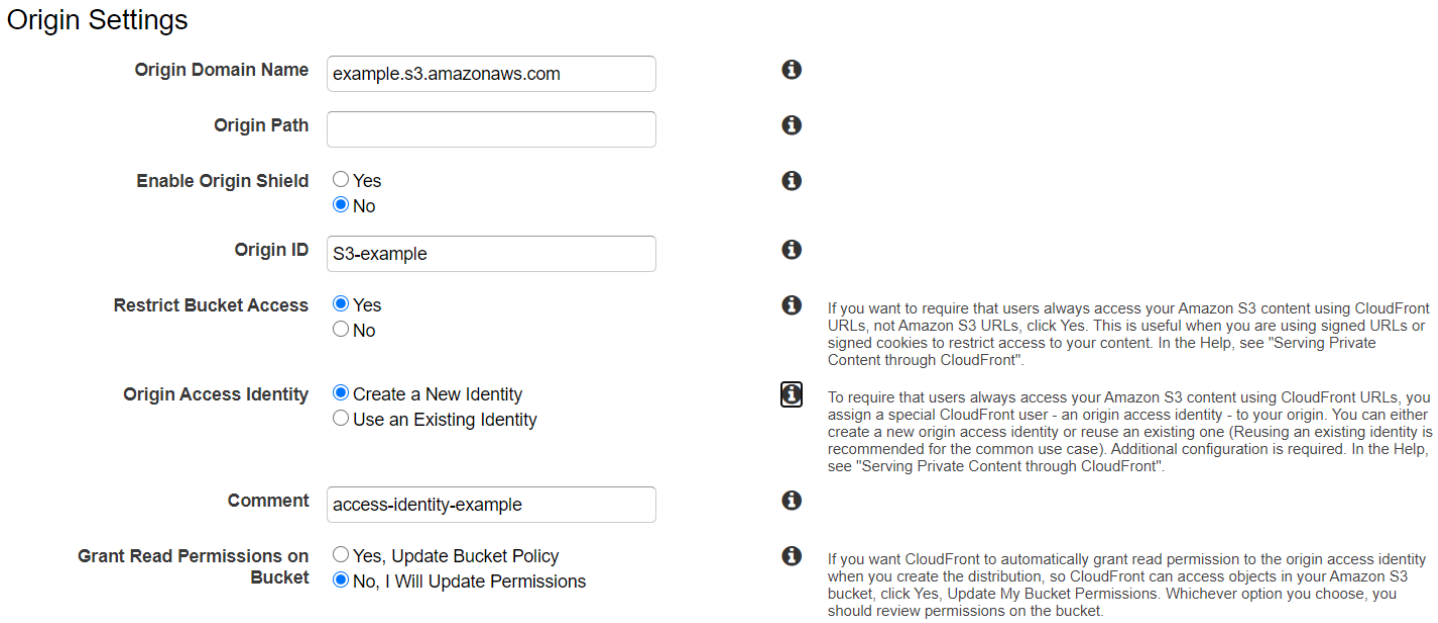
Origin Access Identity in CloudFront

When you first set up a publicly accessible S3 bucket as the origin of a CloudFront distribution, you grant everyone permission to read the files in your bucket. This allows anyone to access your files either through CloudFront or the Amazon S3 endpoint. This might be a security concern for you since you might want your objects to be accessible through CloudFront only. This is especially important if you have configured CloudFront signed URLs or signed cookies to restrict access to files in your S3 bucket, since they can bypass this by using the S3 file URL directly. Restricting access to content that you serve from S3 involves two steps:

1. Create a special CloudFront user called an origin access identity (OAI) and associate it with your distribution.
2. Configure your S3 bucket permissions so that CloudFront can use the OAI to access the files in your bucket and serve them to your users. Disable direct URL file access.

Origin access identity, or OAI, limits user access to your files only via CloudFront. So even if your S3 URL was exposed and a malicious attacker used it to try and access your files, the permissions you've set in your S3 bucket will prevent them from snooping around and retrieving anything. You can create an OAI while creating a CloudFront distribution or as an individual resource and associate it to a CloudFront distribution afterwards.

You can reuse existing OAIs since they are individual identities and are not directly tied to your origins. You can also have CloudFront immediately apply the necessary read permissions to your origin S3 bucket so that your OAI will be able to read your files. This saves you the time in writing your own S3 permissions (which might take you some time if you haven't done it before). An S3 bucket can have multiple OAIs as principals in its permission policy.



Here is an example of an S3 policy that allows an OAI to read all of its objects:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::cloudfront:user/CloudFront Origin Access Identity unique_identifier"
      },
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::tutorialsdojo/*"
```

```
    }
  ]
}
```

References:
https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/private-content-restricting-access-to-s3.html
https://aws.amazon.com/premiumsupport/knowledge-center/cloudfront-access-to-amazon-s3/
https://tutorialsdojo.com/amazon-cloudfront/
https://tutorialsdojo.com/s3-pre-signed-urls-vs-cloudfront-signed-urls-vs-origin-access-identity-oai/

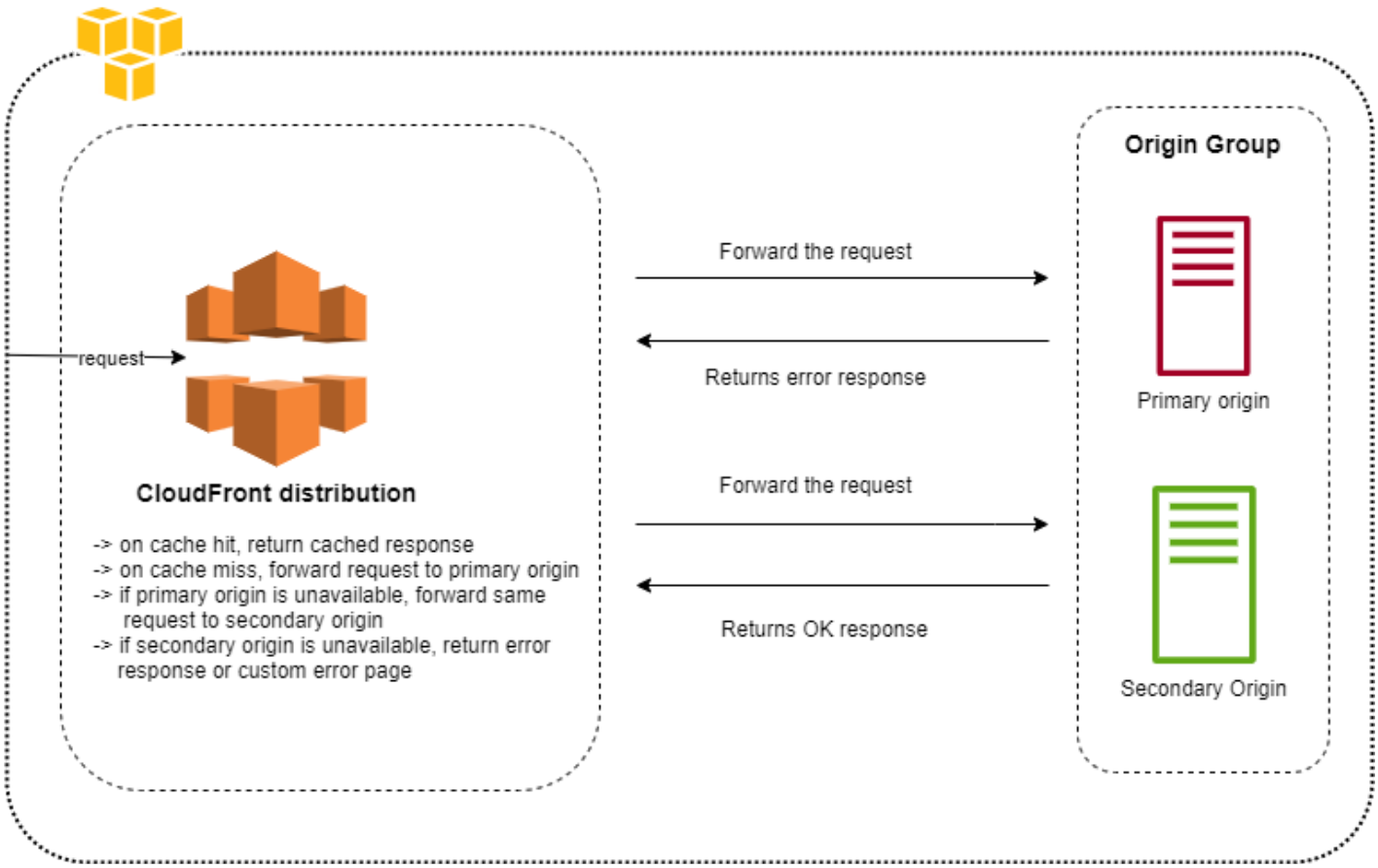
High Availability with CloudFront Origin Failover

Those that are using CloudFront must take into account the high availability of their origins. If it were to go down, your CloudFront should be able to automatically redirect traffic requests to a new origin. A CloudFront origin group lets you specify one primary origin and one secondary origin. If the primary origin becomes unavailable, or returns specific HTTP response status codes that indicate a failure, CloudFront automatically switches to the secondary origin. Origin failover requires your distribution to have at least two origins. Once you've created your origin group, you create or update a cache behavior to use the origin group.

After you configure origin failover for a cache behavior, CloudFront does the following for viewer requests:

1. When there's a cache hit, CloudFront returns the requested file.
2. When there's a cache miss, CloudFront routes the request to the primary origin in the origin group.
3. When the primary origin returns a status code that is not configured for failover, such as an HTTP 2xx or 3xx status code, CloudFront serves the requested content to the viewer.
4. CloudFront only routes the request to the secondary origin in the origin group when any of the following occur:
    a. The primary origin returns an HTTP status code that you've configured for failover
    b. CloudFront fails to connect to the primary origin
    c. The response from the primary origin times out

CloudFront fails over to the secondary origin only when the HTTP method of the viewer request is **GET**, **HEAD**, or **OPTIONS**. Other HTTP methods will not cause a failover. You can also create custom error pages for your primary and secondary origins in case they receive a request while they're unavailable.



References:

https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/high_availability_origin_failover.html

https://tutorialsdojo.com/amazon-cloudfront/

AWS Direct Connect

AWS Direct Connect is a cloud service solution that allows you to establish a dedicated network connection from your on-premises network to AWS. It provides a more consistent network experience over Internet-based connections. You can create a private virtual interface to enable your on-premises servers to connect to the virtual private gateway of your Amazon VPC. You can even group your virtual private gateways and private virtual interfaces using a Direct Connect Gateway. You can also use a public virtual interface to connect to your Amazon S3 buckets. Unlike a VPN, the data being sent over a Direct Connect connection does not pass through the public Internet.

Leveraging AWS Direct Connect

Some businesses have strict network and security requirements for their operations. For these cases, a dedicated and secure network to AWS is needed. If you need a dedicated network line for your traffic, provision an AWS Direct Connect from a provider and have it linked to your network. AWS Direct Connect provides many benefits compared to a VPN solution, such as a private connection to AWS, lower latency, and a higher network bandwidth. There are different ways to leverage Direct Connect:

1. If you need access to resources located inside a VPC, **create a private virtual interface (VIF) to a VGW attached to the VPC**. You can create 50 VIFs per Direct Connect connection, enabling you to connect to a maximum of 50 VPCs. Connectivity in this setup restricts you to the AWS Region that the Direct Connect location is homed to. This is not the best solution if you need to connect to a bunch of VPCs.
2. If your VPCs are located in different AWS Regions, **create a private VIF to a Direct Connect gateway associated with multiple VGWs**, where each VGW is attached to a VPC. You can attach multiple private virtual interfaces to your Direct Connect gateway from connections at any Direct Connect location. You have one BGP peering per Direct Connect Gateway per Direct Connect connection. This solution will not work if you need VPC-to-VPC connectivity.
3. You can associate a Transit Gateway to a Direct Connect gateway over a dedicated or hosted Direct Connect connection running at 1 Gbps or more. To do so, you need to create a **transit VIF to a Direct Connect gateway associated with Transit Gateway**. You can connect up to 3 transit gateways across different AWS Regions and AWS accounts over one VIF and BGP peering. This is the most scalable and manageable option if you have to connect to multiple VPCs in multiple locations.
4. If you need access to AWS public endpoints or services reachable from a public IP address (such as public EC2 instances, Amazon S3, and Amazon DynamoDB), **create a VPN connection to Transit Gateway over Direct Connect public VIF.** You can connect to any public AWS service and AWS Public IP in any AWS Region. When you create a VPN attachment on a Transit Gateway, you get two public IP addresses for VPN termination at the AWS end. These public IPs are reachable over the public VIF. You can create as many VPN connections to as many Transit Gateways as you want over public VIF. When you create a BGP peering over the public VIF, AWS advertises the entire AWS public IP range to your router.

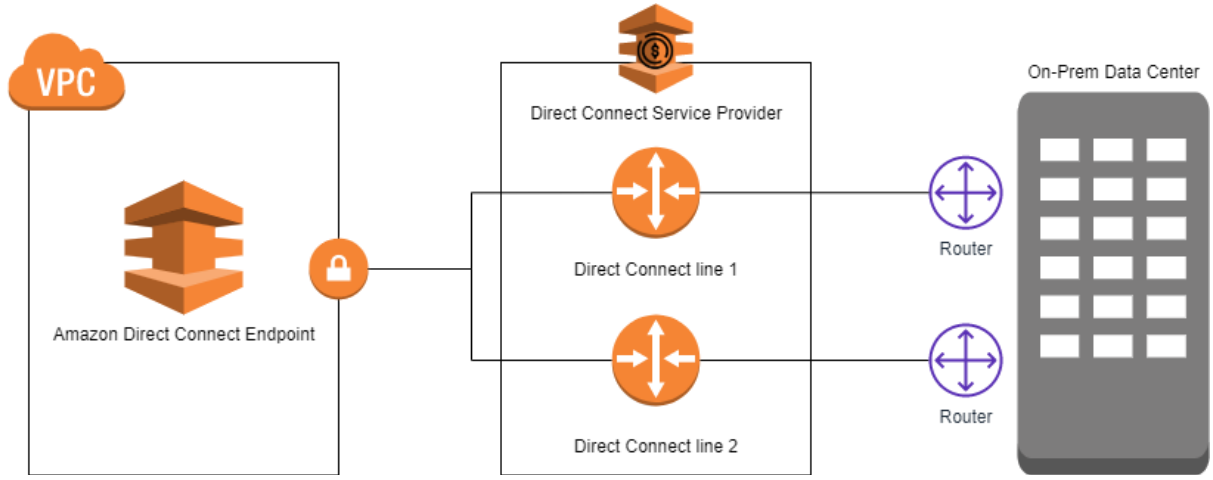
AWS Direct Connect supports both IPv4 and IPv6 on public and private VIFs. You will be able to add an IPv6 peering session to an existing VIF with IPv4 peering session (or vice versa). You can also create 2 separate VIFs – one for IPv4 and another one for IPv6.

References:
https://docs.aws.amazon.com/directconnect/latest/UserGuide/Welcome.html
https://tutorialsdojo.com/aws-direct-connect/

High Resiliency With AWS Direct Connect

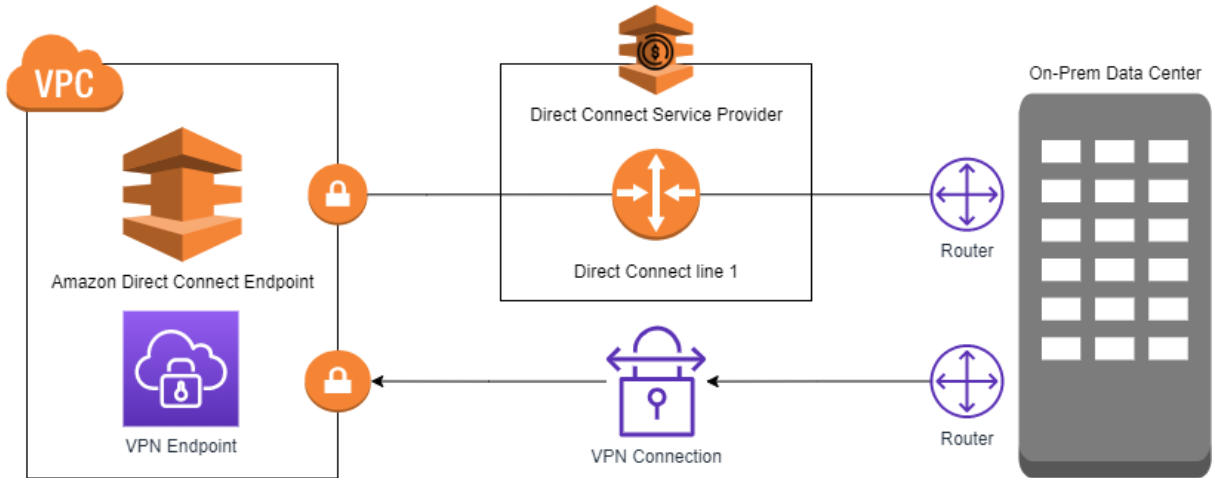
AWS Direct Connect, by default, is not a resilient network. The event of a line failure or network disruption can mean total downtime for you. There are approaches one can take to make an on-premises network connection to AWS more resilient, either by purchasing another Direct Connect line or by making use of the public internet and securing the connection with a VPN for example. Here we'll take a look at the different options in creating a resilient network with Direct Connect:

- Single on-premises data center having two Direct Connect lines (Development and Test)



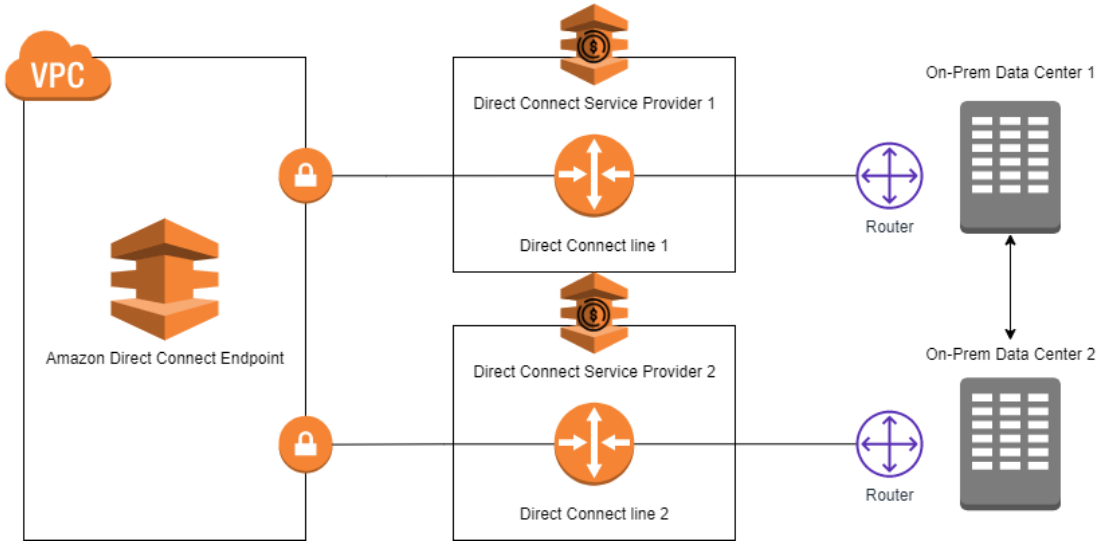
In this type of setup, if you only have a single on-premises data center connected to AWS, you may purchase two Direct Connect lines that are linked to two different devices or routers. If one of the connections were to fail, your network connection will automatically failover to the available Direct Connect line. You can also simulate a failover in AWS to verify if the setup meets your resiliency standards.

- Single on-premises data center having one Direct Connect line and a VPN solution as a secondary



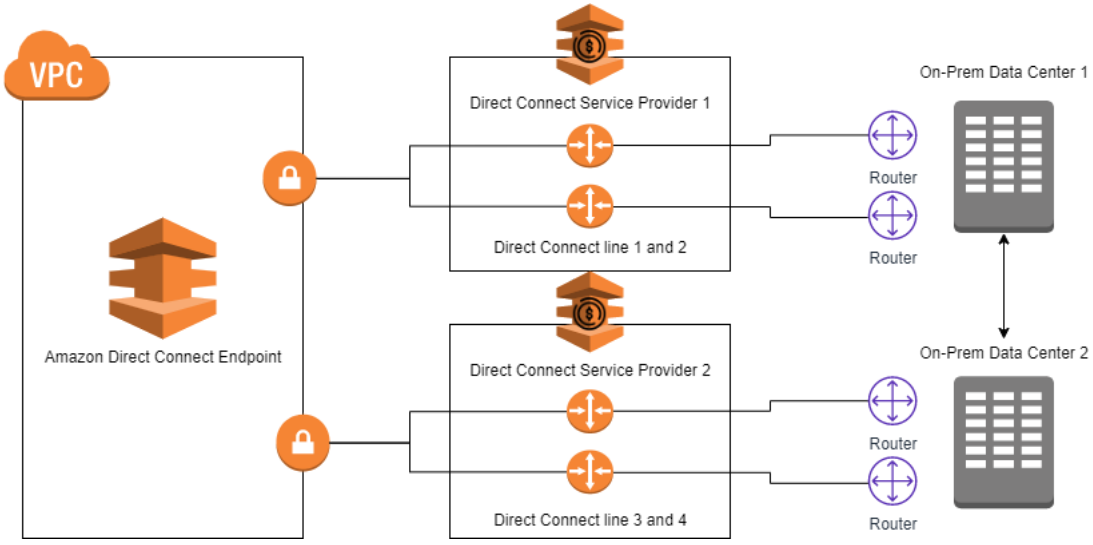
To save on cost, if a dedicated network is not a hard requirement, you may utilize an IPsec VPN connection as your failover solution instead. Do note that you will experience slower network speeds though with this approach.

- Two or more distinct on-premises data centers, each having its own Direct Connect line (High Resiliency)



The best way to make something resilient and highly available is to make it redundant. If you have multiple data centers in different locations connected to AWS, you can configure a Direct Connect line for each of them and link your data center networks together. If a data center's connection to AWS were to go offline, you can reroute the network to utilize the other active Direct Connect lines.

- Two or more distinct on-premises data centers with each having two Direct Connect lines (Max Resiliency)



If you truly, truly need that high uptime because you are running very critical workloads that cannot afford any kind of interruption, then you can set up redundant Direct Connect lines for each of your data centers. Think of this as the first resiliency solution, but applied for each of the critical data centers. This solution is very costly.

References:
https://aws.amazon.com/directconnect/resiliency-recommendation/
https://docs.aws.amazon.com/directconnect/latest/UserGuide/high_resiliency.html#high-resiliency-select-model
https://tutorialsdojo.com/aws-direct-connect/

AWS Global Accelerator

AWS Global Accelerator is a networking service that provides a set of static IP addresses that serves as single fixed entry points for your clients around the world. You can associate these static IP addresses to regional endpoints, such as Network Load Balancers, Application Load Balancers, EC2 instances, or Elastic IP addresses. With Global Accelerator, you can connect multiple AWS resources that are running in one or more AWS Regions using a single endpoint or an endpoint group. The static IP addresses accept incoming traffic onto the AWS global network that is in close proximity to your users. So if you have multiple ALBs located in multiple Regions, you can simply create one endpoint to access all of your load balancers.

Connecting Multiple ALBs in Various Regions

AWS Global Accelerator provides you two global static customer facing IP addresses that you can use as a common endpoint for your public facing endpoints. These static IP addresses can be BYOIP or can be taken from the Amazon IP address pool. One huge benefit of Global Accelerator is the ability to consolidate your public endpoints in different AWS Availability Zones and Regions, and provide a common entry point which are the two aforementioned IP addresses. Furthermore, Global Accelerator is able to support up to 10 different regions. With this feature, you can add or remove origins, Availability Zones or Regions without affecting your application availability. If an endpoint suddenly fails or becomes unavailable, Global Accelerator will automatically redirect your new connections to a healthy endpoint within seconds.

Global Accelerator can associate its IP addresses to regional AWS resources or endpoints such as Network Load Balancers, Application Load Balancers, EC2 Instances, and Elastic IP addresses. You control the proportion of traffic sent to each endpoint by assigning them different weights. Global Accelerator complements Elastic Load Balancers well for load balancing and traffic routing at a global scale. ELB handles load balancing within one region, while Global Accelerator manages the traffic across multiple regions. Once you have mapped the static IP addresses to your load balancer endpoints, you'll need to update your DNS configuration to direct traffic to the static IP addresses or DNS name of the accelerator.

To start using Global Accelerator with ELBs, simply do the following:

1. Create a standard accelerator.
2. Add a listener with the allowed reachable ports or port range, and the protocol to accept: TCP, UDP, or both.
3. Add one or more endpoint groups, one for each region in which you have a load balancer.
4. Add one or more ELB endpoints to endpoint groups.

References:
https://docs.aws.amazon.com/global-accelerator/latest/dg/work-with-standard-accelerators.html
https://turon.tutorialsdojo.com/aws-global-accelerator/

AWS IAM

The AWS Identity and Access Management service, or IAM for short, is a web service that helps you securely control and manage access to various AWS resources. The two primary functions of this service are quite obvious based on its name. It's for managing the identity of your users and for providing access management to the resources and services available in your account. In other words, it handles the authentication and authorization aspects of your cloud infrastructure.

The "identity" aspect of IAM helps you identify who are the validated users or entities in your organization. This is also known as authentication. An IAM Identity can be an IAM User, an IAM User Group, or an IAM Role. If you create a new AWS account, the very first user that will automatically exist will be the root user. A root user is accessed by signing in with the actual email address and password that you used to create the account. This user is quite powerful as it can change your account settings, access your billing details or even close down your AWS account. It also allows you how to create individual IAM users within your AWS account that corresponds to the users or the actual employees of your company. You can create an IAM user for your administrator, for your developers, for your network engineers, and so on. These IAM users are not separate AWS accounts but are just regular users that exist within your AWS account.

Let's now talk about the "Authorization" function of the IAM service. An IAM User can be authorized to access and control your AWS resources. You can create a policy that contains multiple permissions that authorize the IAM User to do a set of actions. This is the "access management" aspect of the IAM service. You can manage the type of access your users should have by attaching an AWS-managed policy, a customer-managed policy, or an inline policy, to different IAM identities. An AWS-managed policy is a standalone policy that is managed by AWS whereas a customer-managed policy is a standalone type that you administer in your own AWS account. An inline policy is a type that's just embedded in an IAM User. This inline policy can be embedded in an IAM Group or an IAM role. You can even attach multiple policies to a single IAM User.

The IAM service is where you apply the concept of granting the least privilege. This concept simply states that you should only grant a defined set of permissions necessary for the entity to perform its designated task. By doing so, it ensures that no resource can execute an operation that it is not meant to do – thereby, reducing the security risk of unauthorized actions.

For example, you have a new Solutions Architect in your company that only needs access to the AWS CloudFormation templates to create multiple AWS resources. After creating an IAM User, what permissions should you provide without compromising security? At the very least, you should add the IAM user to an IAM group with a policy that only allows AWS CloudFormation actions. If the user doesn't need this access anymore, you can then easily remove him or her from that IAM user group. You can also create an IAM Role that explicitly defines the CloudFormation permissions necessary for the new hire to properly do his or her job. This is a perfect example of following the best practice of granting the least privilege to your users. If you add this user to an IAM Group with PowerUserAccess or an AdministratorAccess IAM policy, then that's a security risk! This means that a DevOps engineer can do a lot of administrative actions that are out of scope and may

adversely affect your resources. What's worse is if you handed over the account root user credentials! The user can certainly perform the necessary CloudFormation stack operations but you're putting the security of your entire cloud infrastructure in imminent danger! Again, you should only grant the required permissions to your resources using the various policy types available in IAM.

Aside from granting access to an individual person, you can also provide access to different AWS resources such as your EC2 instances, S3 buckets, Lambda functions, and others. For Amazon EC2, you can use the instance profile to pass a specific IAM role to your EC2 instance for it to perform certain actions like uploading data to Amazon S3 or other operations. These IAM roles attached to your instance can also be viewed on your EC2 metadata. If you have a Linux virtual machine, just type curl `http://169.254.169.254/latest/meta-data/iam/info` to view the list of attached IAM roles.

For Amazon S3, you can set up a bucket policy to grant IAM users and other AWS accounts the access permissions for your bucket and its objects. Say you have multiple AWS accounts in your AWS Organizations that represent different departments. If one department wants to share an Amazon S3 bucket with all other departments, you can set up an S3 bucket policy that allows cross-account access to other departments.

You can also use IAM in your databases. For DynamoDB, you can design an IAM policy that allows access to put, update, and delete items in one specific table. This policy can be attached to an IAM role that can be used by a Lambda function or an ECS task to manage a particular DynamoDB table in your account. There's also a security feature called IAM DB Authentication for Amazon RDS and Aurora. This feature allows you to use IAM to centrally manage access to your database resources – eliminating the hassle of managing user access individually on each DB instance. It also improves the security of your application hosted in Amazon EC2 by not storing the database password and just using its instance profile to connect to Amazon RDS.

For Amazon SQS, you can set up an access policy to control external access to your SQS queue. This is helpful if you need to grant permissions to an external company to access your queue. An SQS access policy can be prepared to allow the other company to poll the queue without giving up the permissions of your own account.

In IAM, there are two primary types of policies that you can use. These are identity-based policy and resource-based policy. Basically, identity-based policies are attached to an IAM user, group, or role while resource-based policies are attached to a resource, like EC2 instances, S3 buckets, and others.

You can also use permissions boundaries for your IAM users and roles. A permissions boundary is a feature that allows you to set the maximum permissions that an identity-based policy can grant to an IAM entity. With this, an entity can only perform the actions that are allowed by both its identity-based policies and its permissions boundaries.

Identity-based Policies and Resource-based Policies

As you may already know, IAM policies are JSON documents that control what a principal can and cannot do in AWS. You explicitly state which permissions you'd like to grant and deny to a principal, and if they are only granted/denied permissions to specific resources. You can also add conditions to your policy statements, such as requiring the user to be MFA authenticated first before allowing any actions, for more granular controls. Below is an example of an IAM Policy:

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
            "ec2:AttachVolume",
            "ec2:DetachVolume"
         ],
         "Resource": [
            "arn:aws:ec2:*:*:volume/*",
            "arn:aws:ec2:*:*:instance/*"
         ],
         "Condition": {
            "ArnEquals": {"ec2:SourceInstanceARN": "arn:aws:ec2:*:*:instance/instance-id"}
         }
      }
   ]
}
```

There are two types of policies in IAM — **Identity-based** and **Resource-based**.

Identity-based policies are the ones you attach to IAM Users, Groups and Roles. Resource-based policies are ones that you attach to AWS services that support this type of policy, such as Amazon S3 buckets. Resource-based policies and resource-level permissions are two different things. Resource-based policies include a *Principal* element to specify which IAM identities can access that resource. Resource-level permissions refer to the ability to use ARNs to specify individual resources in a policy. Here is an example of a resource-based policy that allows principals with the *EC2RoleToAccessS3* role to retrieve objects from the sample S3 bucket, as long as the originating IP is not within 10.10.0.0/24 .

```
{
  "Version" : "2012-10-17",
  "Statement" : [
    {
      "Effect": "Allow",
      "Principal": {"AWS": "arn:aws:iam::123456789000:role/EC2RoleToAccessS3"},
      "Action": ["s3:GetObject","s3:GetObjectVersion"],
      "Resource": ["arn:aws:s3:::EXAMPLE-BUCKET/*"],
      "Condition": {
        "ForAnyValue:StringEquals": {
          "NotIpAddress": {"aws:SourceIp": "10.10.0.0/24"}
        }
      }
    }
  ]
}
```

Both identity-based policies and resource-based policies are evaluated to determine if a principal will have access or not. If both do not provide an explicit allow, or either one has an explicit deny, then the principal is denied access.

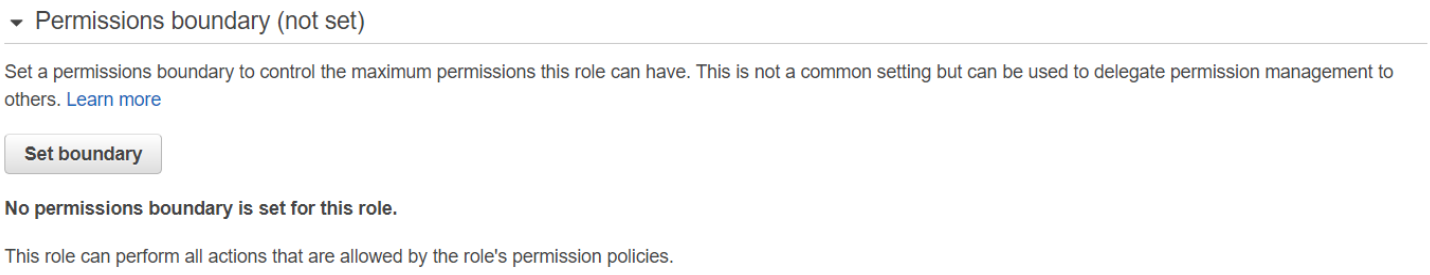
References:
https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies_identity-vs-resource.html
https://tutorialsdojo.com/aws-identity-and-access-management-iam/

IAM Permissions Boundary

When you have users working on different projects and in different environments, it can be difficult to keep track of what permissions they need to do their work. Sometimes, it would be quicker to just let the users attach the IAM policies they need to their IAM roles. This can cause security issues in your AWS account since you are not following the principle of least privilege. You should not provide that much freedom of access to your users, but you also do not want to hinder their work, so what should you do? You can set a middle ground by simply creating IAM permissions boundaries.

"A permissions boundary is an advanced feature for using a managed policy to set the maximum permissions that an identity-based policy can grant to an IAM entity. An entity's permissions boundary allows it to perform only the actions that are allowed by both its identity-based policies and its permissions boundaries." Simply put, a permissions boundary keeps IAM user permissions and IAM role permissions in check by limiting what they can do. A boundary permission takes precedence over an identity policy, so even if your users attach

Administrator privileges to their accounts, they will not be able to perform any actions that are beyond what is stated in their permissions boundary.



References:
https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies_boundaries.html
https://tutorialsdojo.com/aws-cheat-sheet-aws-identity-and-access-management-iam/

IAM Policy Structure and Conditions

We will be breaking down what constitutes an IAM Policy and what conditions you can add to your policies. The structure is as follows:

```
{
  "Statement":[{
    "Effect":"effect",
    "Action":"action",
    "Resource":"arn",
    "Condition":{
      "condition":{
        "key":"value"
        }
      }
    }
  ]
}
```

- **Effect** — The value can be either *Allow* or *Deny*. By default, IAM users don't have permission to do anything, so all requests are implicitly denied. **An explicit allow overrides the default. An explicit deny overrides any allows.**
- **Action** — The specific API action(s) that you are granting or denying permission.
- **Resource** — The resource that's affected by the action. You specify a resource using an Amazon Resource Name (ARN) or using the wildcard (*) to indicate that the statement applies to all resources.

- **Condition** — Conditions are optional. They can be used to control when your policy is in effect. Some conditions that you should be aware of are:
    - StringEquals - Exact string matching and case sensitive
    - StringNotEquals
    - StringLike - Exact matching but ignoring case
    - StringNotLike
    - Bool - Lets you construct Condition elements that restrict access based on true or false values.
    - IpAddress - Matching specified IP address or range.
    - NotIpAddress - All IP addresses except the specified IP address or range
    - ArnEquals, ArnLike
    - ArnNotEquals, ArnNotLike
    - Use a Null condition operator to check if a condition key is present at the time of authorization.
    - You can add IfExists to the end of any condition operator name (except the Null condition)—for example, *StringLikeIfExists*.

References:
https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/iam-policy-structure.html
https://tutorialsdojo.com/aws-cheat-sheet-aws-identity-and-access-management-iam/

IAM Policy Evaluation Logic

When a principal sends a request to AWS, the following events occur to determine if AWS will accept or deny your request:

1) AWS first authenticates the principal that makes the request.
2) AWS processes the information gathered in the request to determine which policies apply to the request.
3) AWS evaluates all of the policy types, which affect the order in which the policies are evaluated.
4) AWS then processes the policies to determine whether the request is allowed or denied.

There can be multiple policy types applied onto a single account. They are all evaluated by AWS following the evaluation logic:

1) If only identity-based policies apply to a request, then AWS checks all of those policies for at least one explicit Allow and does not have an explicit Deny.
2) If resource-based policies and identity-based policies both apply to a request, then AWS checks all the policies for at least one Allow and does not have an explicit Deny.
3) When you set a permissions boundary for an entity, the entity can perform only the actions that are allowed by both its identity-based policies and its permissions boundaries. An implicit deny in a permissions boundary does not limit the permissions granted by a resource-based policy.
4) If an AWS Organization SCP is present, identity-based and resource-based policies grant permissions to principals in member accounts only if those policies and the SCP allow the action. If both a

permissions boundary and an SCP are present, then the boundary, the SCP, and the identity-based policy must all allow the action with no explicit deny.

In summary, to know if a principal has permissions for an action or not, remember the behavior of each policy involved:

- By default, all requests are implicitly denied. Also, by default, the AWS account root user has full access.
- An explicit allow in an identity-based or resource-based policy overrides this default.
- If a permissions boundary, Organizations SCP, or session policy is present, it might override the allow with an implicit deny.
- An explicit deny in any policy overrides any allows.

References:
https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_evaluation-logic.html
https://tutorialsdojo.com/aws-cheat-sheet-aws-identity-and-access-management-iam/

AWS Key Management Service

AWS Key Management Service or KMS is a managed service that works almost like CloudHSM. Under the hood, KMS also uses hardware security modules that make it easy for you to create and control your encryption keys. But unlike CloudHSM, this service has multi-tenant access, which means you share the HSM with other tenants or AWS customers. You also cannot launch an HSM to Amazon VPC or EC2 instances that you own. The HSM is fully managed by the Amazon Web Services team themselves. AWS KMS can be integrated with other AWS services to help you protect the data you store with these services. For example, encrypting volumes or snapshots in Amazon EBS is powered by AWS KMS as well as Server-Side encryption (SSE-KMS) in Amazon S3 and database encryption in Amazon RDS.

AWS KMS uses envelope encryption, which is the practice of encrypting your plaintext data with a data key; and then encrypting that data key using another key, called the master key. The primary resources in KMS are called customer master key, or CMK. A CMK is basically is a representation of the master key that encrypts your data key. With AWS KMS, you can store your CMKs and automatically rotate them to meet your encryption requirements. You can also create a custom key store in AWS KMS with CloudHSM. This custom key store provides complete control over your encryption key lifecycle management and allows you to remove the key material of your encryption keys. You can also audit key usage independently of AWS CloudTrail or KMS itself.

AWS KMS Customer Master Key

The Customer Master Key or CMK is the most basic resource in AWS KMS. A CMK includes metadata, such as the key ID, creation date, description, and key state. The CMK also contains the key material used to encrypt and decrypt data. AWS KMS has two types of CMK encryption keys:

1) **Symmetric** - a 256-bit key that is used for encryption and decryption.
2) **Asymmetric** - an RSA key pair that is used for encryption and decryption or signing and verification (but not both), or an elliptic curve (ECC) key pair that is used for signing and verification.

Symmetric CMKs and the private keys of asymmetric CMKs never leave AWS KMS unencrypted.

Furthermore, there are three variations of CMKs in KMS:

1) **Customer managed** - These CMKs are what you have full control over. You handle establishing and maintaining their key policies, IAM policies, and grants, enabling and disabling them, rotating key material, adding tags, creating aliases that refer to the CMK, and scheduling the CMKs for deletion.
2) **AWS-managed** - These are CMKs in your account that are created, managed, and used on your behalf by an AWS service that is integrated with KMS. You cannot manage these CMKs, rotate them, or change their key policies. You also cannot use these CMKs in cryptographic operations directly; the service that creates them uses them on your behalf.
3) **AWS-owned** - These are CMKs that an AWS service creates, owns, and manages for use in multiple AWS accounts. You cannot view, use, track, or audit these CMKs.

By default, KMS creates the key material for all CMKs. You cannot extract, export, view, or manage this key material. Also, you cannot delete the key material alone; you must delete the whole CMK. However, you can import your own key material into a (customer-managed) CMK or create the key material for a (customer-managed) CMK in the AWS CloudHSM custom key store. Any type of CMK can be used for encryption and decryption. Data keys (symmetric data keys) and data key pairs (asymmetric data keys) can also be used for encryption and decryption. Only asymmetric CMKs and data key pairs can be used for signing and verification.

References:
https://docs.aws.amazon.com/kms/latest/developerguide/concepts.html#master_keys
https://tutorialsdojo.com/aws-key-management-service-aws-kms/

Custom Key Store

A custom key store for AWS KMS is a hardware security module (HSM) in a AWS CloudHSM cluster that you own and manage. You can create your CMKs in a custom key store, and KMS generates a 256-bit AES **symmetric key** material in the associated CloudHSM cluster that you can view and manage. This key material never leaves your HSM cluster unencrypted. You also have full control over the CloudHSM cluster, such as creating and deleting HSMs and managing backups. When you use a CMK stored in a custom key store, encryption and decryption happens in the hardware module in the cluster using this key material.

You should consider using a custom key store if you have any of the following requirements:

1. Key material cannot be stored in a shared environment.
2. Key material must be subject to a secondary, independent audit path. By independent, meaning AWS CloudHSM logs all API activity, local activity, user, and key management activity.
3. You need the ability to immediately remove key material from AWS KMS.
4. The HSMs that generate and store key material must be certified at FIPS 140-2 Level 3.

Custom key stores do not support creation of asymmetric CMKs, asymmetric data key pairs, or CMKs with imported key material, and you cannot enable automatic key rotation on a CMK in a custom key store. Key rotation must be performed manually by creating new keys and re-mapping AWS KMS key aliases. Each CloudHSM cluster can be associated with only one custom key store, and a cluster must contain at least two active HSMs in different Availability Zones. You can connect and disconnect your custom key store from a CloudHSM cluster at any time. When connected, you can create and use its CMKs. When it is disconnected, you can view and manage the custom key store and its CMKs, but not create new CMKs or use the CMKs in the custom key store for cryptographic operations.

References:
https://docs.aws.amazon.com/kms/latest/developerguide/custom-key-store-overview.html
https://tutorialsdojo.com/aws-key-management-service-aws-kms/

AWS KMS CMK Key Rotation

It is a security best practice to rotate encryption keys and passwords regularly, especially if these keys are used to protect very sensitive data. Key rotation lowers the risk of getting your key exposed and misused. AWS KMS is a service that lets you create and manage customer master keys. A customer master key is the primary resource in KMS. It is a logical representation of a master key.

The CMK includes metadata, such as the key ID, creation date, description, and key state, and it also contains the key material used for encrypting and decrypting data. When rotating your (customer-managed) CMKs in AWS KMS, you can create new CMKs and then modify your applications to use the new CMK. You can also enable automatic key rotation and let AWS KMS generate new cryptographic material for your CMKs every year.

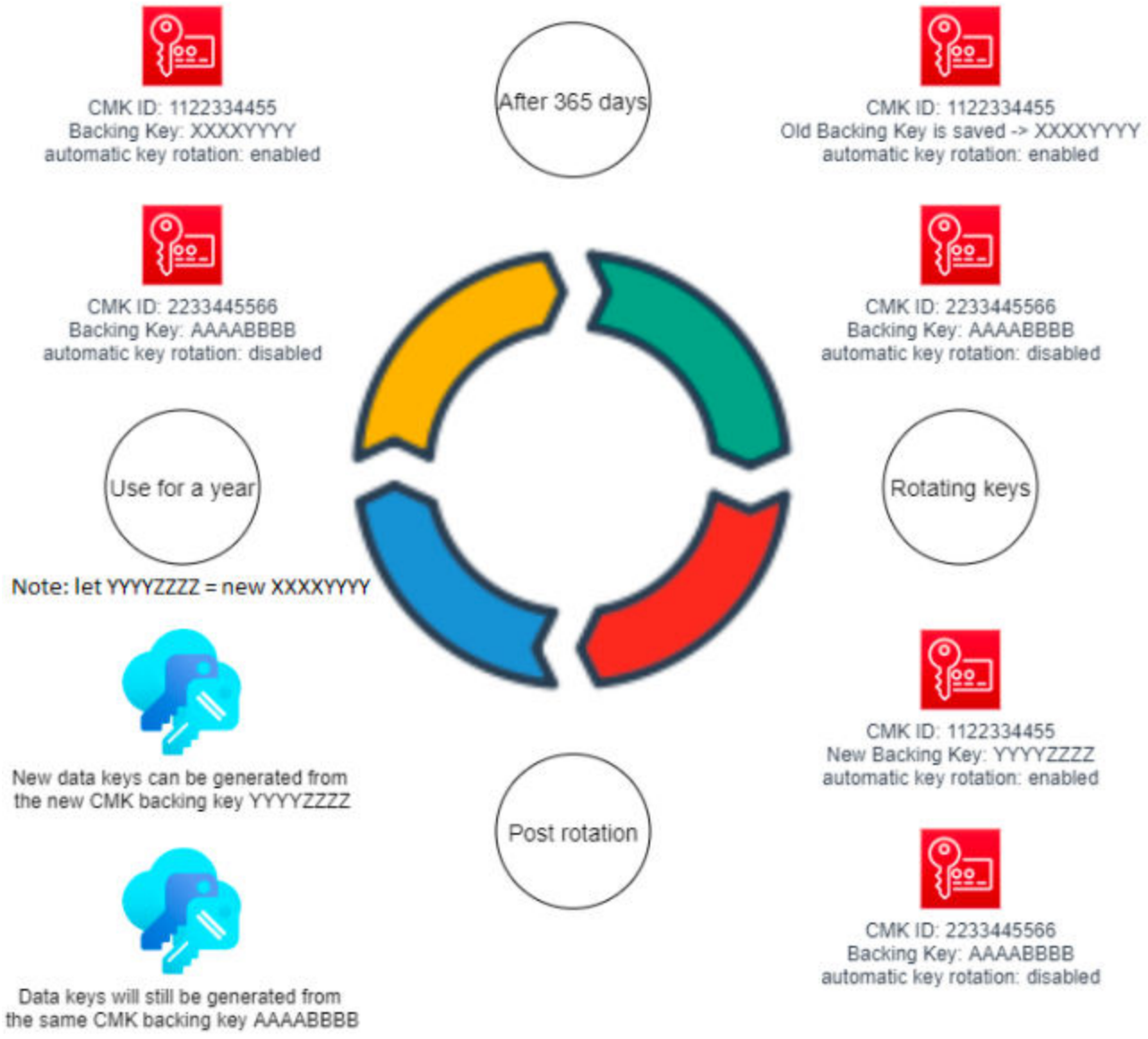
KMS also saves the older cryptographic material so it can be used to decrypt data that it has encrypted. KMS does not delete any rotated key material until you delete the CMK. There are limitations to automatic key rotation – asymmetric CMKs, CMKs in custom key stores, and CMKs with imported key material cannot be automatically rotated.

Automatic key rotation provides the following advantages:

1. The properties of the CMK, including its key ID, key ARN, region, policies, and permissions, do not change when the key is rotated.
2. You do not need to change applications or aliases that refer to the CMK ID or ARN.
3. AWS KMS rotates the CMK automatically every year. You don't need to remember or schedule the update.

However, automatic key rotation has no effect on the data that the CMK protects. It does not rotate the data keys that the CMK generated or re-encrypt any data protected by the CMK, and it will not mitigate the effect of a compromised data key. If you prefer having control over your rotation schedule and frequency, you should opt for manual key rotations instead.

How automatic key rotation works:



References:
https://docs.aws.amazon.com/kms/latest/developerguide/rotate-keys.html
https://tutorialsdojo.com/aws-key-management-service-aws-kms/

AWS Web Application Firewall

The AWS Web Application Firewall service or AWS WAF is basically just a web application firewall in AWS – just as its name implies. It protects your web applications from common web exploits that could affect application availability, compromise security, or consume excessive resources.

You can use AWS WAF to create custom rules that block common attack patterns, such as SQL injection and cross-site scripting attacks. You can integrate AWS WAF with Amazon CloudFront, Application Load Balancer (ALB), and Amazon API Gateway. With its IP Match condition feature, you can block malicious requests from a recurring set of IP addresses. AWS WAF can also protect your application from illegitimate requests sent by illegitimate external systems, through its rate-limiting rule. When these attacks occur, a rate-based rule can reduce the impact on the legitimate users of your websites. This can be achieved by creating rate-based web access control lists, or web ACLs, and attaching it to a CloudFront distribution, to help minimize the effects of a DDoS attack.

This service is also suitable if your application is authorized to be accessed from one specific country only. AWS WAF provides a geo match condition that blocks specific countries from accessing your site or to only allow access from certain countries that you define.

AWS WAF Rule Statements To Filter Web Traffic

AWS WAF is capable of protecting your public endpoints in CloudFront, Elastic Load Balancers, and API Gateway APIs from a multitude of web security threats. Rule statements tell AWS WAF how to filter out a web request. AWS WAF applies the corresponding action — allow, block or count — to a web request that matches a rule. Rule statements can be very simple (just one criteria to match) or complex (multiple statements combined using AND, OR, and NOT operators). You can use the following match statements to create a simple or complex rule statement:

Match Statement	Use Case
Geographic match	Allows you to allow or block web requests based on country of origin by creating one or more geographical, or geo, match statements. If you use the CloudFront geo restriction feature to block a country, requests from that country are blocked and are not forwarded to WAF.
IP set match	Inspects the IP address of a request against a set of IP addresses and address ranges that you want to allow through or block with your WAF.
Label match rule statement	Inspects the request for labels that have been added by other rules in the same web ACL.
Regex pattern set	Lets you compare regex patterns against a specified component of a web request.

Size constraint	Compares the size of a request component against a size constraint in bytes.
SQLi attack	Inspects for malicious SQL code in a web request.
String match	Searches for a matching string in a web request component. If a matching string is found, WAF allows/blocks the request.
XSS scripting attack	Inspects for cross-site scripting attacks in a web request.
Rate-based	Tracks the rate of requests of each originating IP addresses, and triggers a rule action on IPs with rates that go over a limit. You can use this type of rule to put a temporary block on requests from an IP address that's sending excessive requests.

References:
https://docs.aws.amazon.com/waf/latest/developerguide/waf-rules.html
https://tutorialsdojo.com/aws-waf/

Amazon Cloudwatch

Amazon CloudWatch is a suite of AWS services used in monitoring your systems on both AWS as well as in your on-premises network. CloudWatch is basically a metrics repository that collects system data from AWS services and the custom metrics that you publish. It monitors and analyzes these metrics and notifies you if a certain threshold has been reached. You also have the option to trigger a certain action based on a specific threshold or events that you define.

Amazon CloudWatch has several related services that you can choose from. They are: CloudWatch Metrics, CloudWatch Logs, CloudWatch Alarms, CloudWatch Events, CloudWatch Dashboards, and many more.

The primary task of the CloudWatch Metrics service is to collect the metrics from the AWS Services and your custom applications. You can also aggregate the metrics across multiple resources. Most AWS services send metric data to CloudWatch every 1 minute by default. However, for Amazon EC2, the metrics are only sent every 5 minutes. You can enable the detailed monitoring feature in Amazon EC2, to view the metrics in a 1-minute period.

CloudWatch Logs is a monitoring service for your logs. It allows you to monitor, store, and access your log files generated from your AWS Services or your custom applications. With this service, you can easily analyze, monitor, or even query your log data from various AWS services. You can also do the same thing for your custom applications. Just install a CloudWatch Logs agent to your EC2 instances and it will automatically collect and publish your application logs to CloudWatch.

CloudWatch Alarms is another service in the CloudWatch family. As its name implies, it is primarily used for setting alarms as part of your monitoring process. You can create a CloudWatch Alarm that performs one or more actions based on a system metric and a specific threshold. So if a metric breached the threshold that you define, the alarm will be triggered and it will notify you using Amazon SNS. You can trigger a custom action too, like Auto Scaling your EC2 instances, sending a billing alert, or invoking a Lambda function.

CloudWatch Events is a service that monitors and responds to the system events of your AWS services in near real-time. You can create a CloudWatch Event rule to track the changes or the state of your services and do a particular action. If a certain event occurred and it matches your event rule, the action that you specified will be invoked. You can also use this for a scheduled job that regularly runs for a period of time. If you need more advanced features, you can use the Amazon EventBridge service instead.

A CloudWatch Dashboard is just a customizable dashboard in the CloudWatch web console. You can use this board to monitor your resources in a single view, even if those resources are located across different AWS Regions. You can also publish and view your custom metrics using the CloudWatch Dashboard.

Monitoring Additional Metrics with the Cloudwatch Agent

We know that Amazon Cloudwatch is your default service for monitoring different performance, network, and statistics related metrics of your AWS services. Although Cloudwatch Metrics is able to collect different types of data from your resources, it does not capture everything. There are some system-level metrics and logs that we should also be monitoring but cannot be directly monitored by Cloudwatch. For such cases, you need to install a Cloudwatch agent into your servers (on-prem, EC2 instances, containers, etc) to be able to retrieve these system-level metrics and logs, and have them monitored by Cloudwatch metrics. Furthermore, you can configure Cloudwatch agent to use the StatsD and collectd protocols to collect custom application and service metrics. StatsD is supported on both Linux servers and servers running Windows Server. Collectd is supported only on Linux servers.

Once you've installed the agent in your server, you specify the configuration settings of the agent that will define what metrics and logs to collect and send to Cloudwatch. The default namespace for metrics collected by the CloudWatch agent is CWAgent, which means that the custom metrics will be stored under this folder. You can specify a different namespace in your configuration file.

When configuring the Cloudwatch agent in your server for the first time, you can simplify the configuration process by running the configuration wizard, which provides you with some predefined metric sets that you can start off with. In the exam, if you have a scenario wherein you need to monitor any of the following metrics in your servers, be sure to choose the option that uses Cloudwatch agent:

Windows Server Metrics	Linux Metrics
Paging: Paging File % Usage	Swap: swap_used_percent
LogicalDisk: LogicalDisk % Free Space	Disk: disk_used_percent, disk_inodes_free
PhysicalDisk: PhysicalDisk % Disk Time, PhysicalDisk Disk Write Bytes/sec, PhysicalDisk Disk Read Bytes/sec, PhysicalDisk Disk Writes/sec, PhysicalDisk Disk Reads/sec	Diskio: diskio_io_time, diskio_write_bytes, diskio_read_bytes, diskio_writes, diskio_reads
Memory: Memory % Committed Bytes In Use	Memory: mem_used_percent
Network Interface: Network Interface Bytes Sent/sec, Network Interface Bytes Received/sec, Network Interface Packets Sent/sec, Network Interface Packets Received/sec	Network: net_bytes_sent, net_bytes_recv, net_packets_sent, net_packets_recv
TCP: TCPv4 Connections Established, TCPv6 Connections Established	Netstat: netstat_tcp_established, netstat_tcp_time_wait

Processor: Processor % Processor Time, Processor % Idle Time, Processor % Interrupt Time, Processor % User Time	CPU: cpu_usage_guest, cpu_usage_idle, cpu_usage_iowait, cpu_usage_steal, cpu_usage_user, cpu_usage_system

References:
https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/Install-CloudWatch-Agent.html
https://tutorialsdojo.com/amazon-cloudwatch/

Cloudwatch Alarms for Triggering Actions

Cloudwatch Alarms is a useful, reactive automation tool for monitoring your AWS resources and making sure appropriate actions are made in response to certain situations. A metric alarm has three states:

- **OK** – The metric or expression is within the defined threshold.
- **ALARM** – The metric or expression is outside of the defined threshold.
- **INSUFFICIENT_DATA** – The alarm has just started, the metric is not available, or not enough data is available for the metric to determine the alarm state.

Each metric alarm consists of data points that inform Cloudwatch of the state of the metric that is being monitored. A data point reported to CloudWatch can fall under one of three categories:

- Not breaching (within the threshold)
- Breaching (violating the threshold)
- Missing

If the number of data points that are in a certain category meets your alarm threshold and changes the state of the alarm, you can define actions that Cloudwatch will perform for you in response to it. Examples of actions include:

1. Notifying a user or a group of users about the alarm by sending a message through Amazon SNS.
2. Stop, terminate, reboot, or recover an EC2 instance.
3. Scale an auto scaling group.
4. Create OpsItems in Systems Manager Ops Center to remediate the issue that triggered the alarm.

References:
https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/AlarmThatSendsEmail.html
https://tutorialsdojo.com/amazon-cloudwatch/

Cloudwatch Events (Amazon EventBridge) for Specific Events and Recurring Tasks

Another useful automation tool in AWS is Amazon Cloudwatch Events (Amazon EventBridge). Cloudwatch Events (Amazon EventBridge) lets you perform specific actions in response to an event or to a predefined schedule (cron). There are three ways to trigger a Cloudwatch Event (EventBridge Event):

1. Triggers on a matching event pattern emitted by an AWS service.
2. AWS API Call via CloudTrail.
3. Triggers on a regular schedule or regular rate (cron or rate expressions).

You can set up your AWS account to send events to other AWS accounts, or to receive events from other accounts. The sender account and receiver account must be using the same AWS Region in this case, since Cloudwatch is a regional service. You must also provide the required permissions to allow sending of events.

What's important to know is the supported targets of Amazon Cloudwatch Events (Amazon EventBridge) for processing events:

1. Amazon EC2 instances
2. AWS Lambda functions
3. Streams in Amazon Kinesis Data Streams
4. Delivery streams in Amazon Kinesis Data Firehose
5. Log groups in Amazon CloudWatch Logs
6. Amazon ECS tasks
7. Systems Manager Run Command, Automation, OpsItem and RunCommand
8. AWS Batch jobs
9. Step Functions state machines
10. Pipelines in CodePipeline
11. CodeBuild projects
12. Amazon Inspector assessment templates
13. Amazon SNS topics
14. Amazon SQS queues
15. EC2 CreateSnapshot, RebootInstances, StopInstances and TerminateInstances API calls.
16. The default event bus of another AWS account

And again, an event rule's target must be in the same region as the rule.

References:
https://docs.aws.amazon.com/AmazonCloudWatch/latest/events/WhatIsCloudWatchEvents.html
https://tutorialsdojo.com/amazon-cloudwatch/

AWS Audit Manager

The AWS Audit Manager is a service that you can use to continuously audit your AWS usage. It simplifies the process of doing risk and compliance assessment by utilizing a pre-built standard framework that is based on common compliance standards and AWS best practices. This service uses the data gathered from AWS Security Hub, AWS Config, AWS License Manager, and other AWS services to generate audit reports. The AWS Audit Manager can import license usage limits and the daily usage data from AWS License Manager to generate assessment reports accordingly.

Keep in mind that this is different from AWS Artifact, which only offers evidence to prove that the AWS Cloud infrastructure has met the compliance requirements. AWS Artifact does not check the AWS resources that you are actually using or even the way you use them. On the other hand, the AWS Audit Manager gathers evidence to prove that your usage of AWS services complies with the industry standards or regulations.

Amazon Inspector

Amazon Inspector is an automated security assessment service that helps improve the security and compliance of applications deployed on AWS. Amazon Inspector automatically assesses applications for vulnerabilities or deviations from best practices. After performing an assessment, Amazon Inspector produces a detailed list of security findings prioritized by level of severity. It provides an automated security assessment report that will identify unintended network access to your Amazon EC2 instances and vulnerabilities in those instances. These findings can be reviewed directly or as part of detailed assessment reports which are available via the Amazon Inspector console or API.

Amazon Detective

Amazon Detective makes it easy to analyze, investigate, and quickly identify the root cause of potential security issues or suspicious activities. Amazon Detective automatically collects log data from your AWS resources. What it basically does is to collect logs from AWS CloudTrail, Amazon VPC Flow Logs, Amazon GuardDuty findings, and other AWS services then use machine learning to analyze and conduct security investigations.

AWS Security Hub

AWS Security Hub is a service that provides a centralized and comprehensive view of the security posture of your cloud infrastructure across multiple AWS accounts. This service helps you to comply with specific security standards and best practices that you organization require. Basically, it works like a hub that collects security alerts and findings from multiple AWS services, such as Amazon GuardDuty, Amazon Inspector, Amazon Macie, AWS Identity and Access Management (IAM) Access Analyzer, AWS Firewall Manager and other sources. You can the AWS Security Hub to comply with the Payment Card Industry Data Security or PCI DSS Standard, Center for Internet Security or CIS Benchmarks, and many other security standards.

AWS Network Firewall

AWS Network Firewall is a managed network firewall service for your Amazon Virtual Private Clouds. This network security service is a managed network firewall that comes with intrusion prevention and detection capabilities. The AWS Network Firewall service allows you to filter traffic within the perimeter of your Amazon VPCs.

This service is commonly used in various network security use cases such as inspecting VPC-to-VPC traffic, filtering outbound traffic, securing both AWS Direct Connect connection and VPN traffic as well as filtering the Internet traffic. AWS Network Firewall also offers fine-grained network security controls for interconnected VPCs via the AWS Transit Gateway.

You can also use this to filter your outbound traffic to prevent unwanted data loss, block malware, and satisfy your strict network security compliance requirement. A single AWS Network Firewall can be configured with thousands of rules that can filter out network traffic routed to known bad IP addresses or suspicious domain names. It can also protect the AWS Direct Connect or VPN traffic that originates from client devices and your on-premises environments. The AWS Network Firewall can ensure that only authorized sources of traffic are granted access to your mission-critical VPC resources. It is also capable of performing the same activities as your Intrusion Detection Systems and Intrusion Prevention Systems or IDP/IPS. This is achieved by inspecting all inbound Internet traffic using features such as ACL rules, stateful inspection, protocol detection, intrusion prevention et cetera.

The AWS Network Firewall has 3 basic components, namely the Firewall, the Firewall Policy, and the Rule Group.

A firewall is a resource that you create in AWS Network Firewall. This firewall is connected to the Amazon VPC of your choice, where a network filter will be implemented based on the behavior defined in your firewall policy. Your firewall can have one firewall policy only, which contains rule groups that you define.

You can deploy the firewall in multiple Availability Zones and to one subnet per zone. Each subnet that is associated with your firewall must have at least one available IP address. Afterward, you must update your VPC route tables to send incoming and outgoing traffic through the firewall endpoints.

The second component is the firewall policy. This is a policy that defines the behavior of your firewall using a collection of stateless and stateful rule groups. A firewall policy can be associated with one or more firewalls. However, a firewall can only have one firewall policy. Changing a firewall policy affects all other firewalls that reference it.

The third component in AWS Network Firewall is called a rule group. This is basically a collection of stateless or stateful rules that define how to inspect and handle the network traffic in your VPC. A rules configuration

includes 5-tuple network values and domain name filtering. The 5-tuple format includes the source IP address, source port, destination IP address, destination port, and protocol.

AWS Network Firewall has two types of rules which could be stateless or stateful. The first one is stateless in the sense that it does not have any context of the packet's traffic flow. A stateless rule just checks the packet itself. On the contrary, a stateful rule knows the context or the state of the packet, including its direction flow and other information that is not provided by the packet itself. Each rule in your stateful rule group has an associated order for its evaluation sequence. This concept is similar to network access control lists and security groups. A network ACL is stateless, while a security group is stateful.

You can also choose how you want the AWS Network Firewall to handle the packets that match your rule criteria. You can either pass or drop a packet that was filtered by the network firewall. A packet can also be forwarded to another stateful rule group for re-evaluation. Setting up custom actions is possible as well. A custom action can be configured to publish data to CloudWatch metrics for future network analysis.

AWS CloudTrail

AWS CloudTrail is a service that's primarily used for IT audits. It tracks user activity and API usage in your AWS account. It stores the data in an Amazon S3 bucket owned by your AWS account or by other people. CloudTrail enables risk auditing by continuously monitoring and logging account activities, such as user actions in the AWS Management Console, AWS SDKs, APIs, or AWS CLI.

There are two types of events that can be logged in AWS CloudTrail: management events and data events. These events are also called control plane and data plane operations respectively. Management events provide information about the management operations performed on resources in your AWS account such as attaching an IAM Role, creating a VPC, or creating a subnet. Data events provide information about the resource operations performed on or in a resource, like Amazon S3 object-level API activities or invoking an AWS Lambda. You can also receive CloudTrail Log files from multiple AWS accounts by granting cross-account permissions to CloudTrail. Each account can deliver the log files to an S3 bucket in your central account.

What's Not Monitored By Default in CloudTrail and How To Start Monitoring Them

There are three types of events that you can log in AWS CloudTrail:

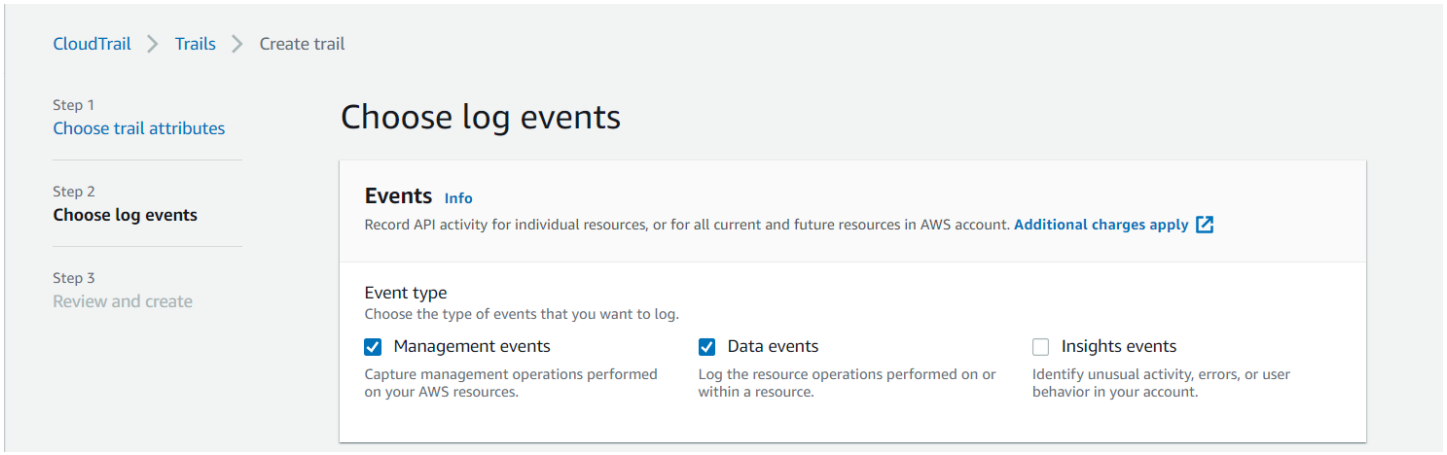
1. Management events which provide visibility into management operations that are performed on resources in your AWS account.
2. Data events which provide visibility into the resource operations performed on or within a resource.
3. Insights events which are logged when CloudTrail detects unusual write management API activity in your account.

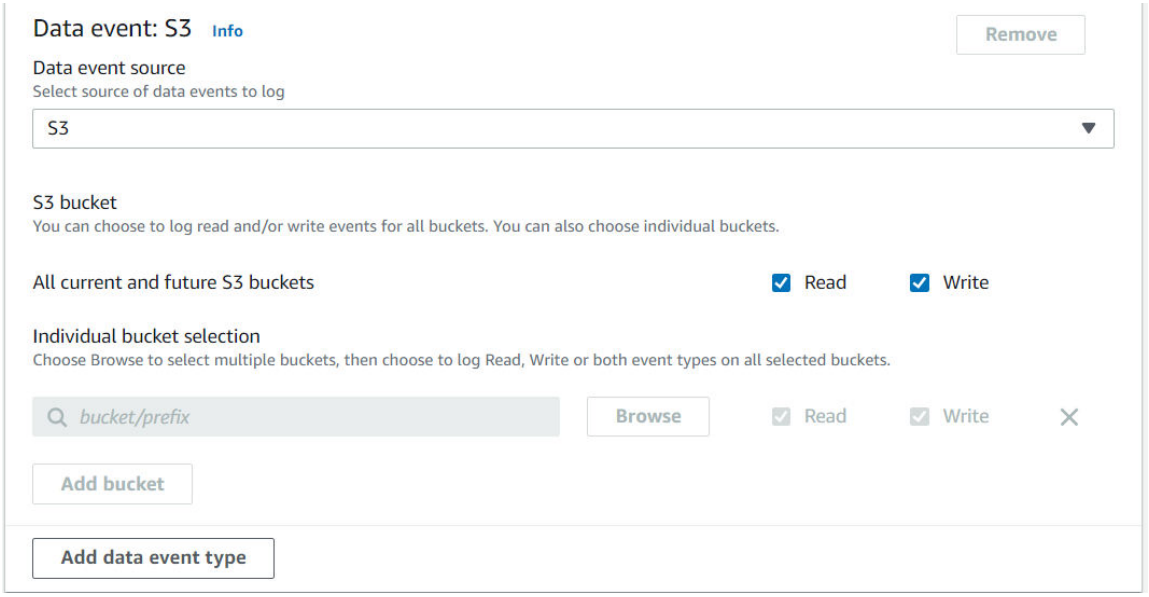
By default, AWS CloudTrail trails log all management events but don't include data or insights events.

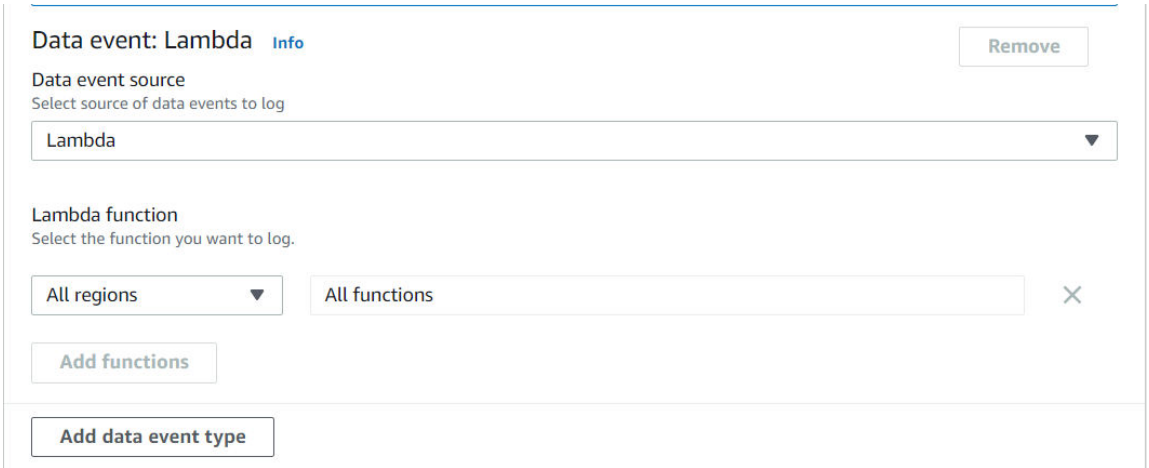
Data events are often high-volume activities, which is why they are not automatically logged. Events that belong under the data events include:

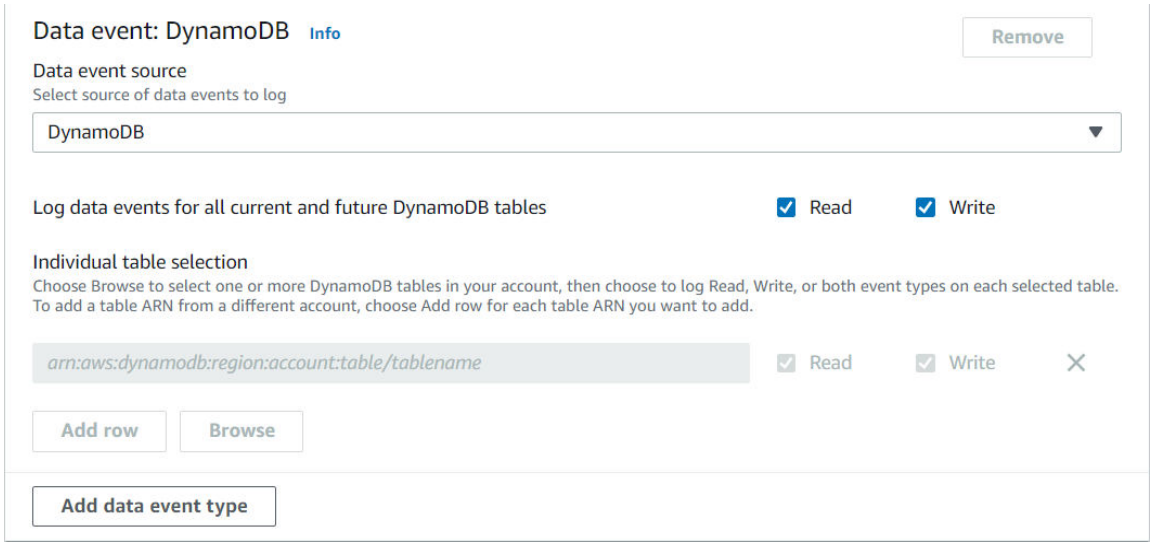
- Amazon S3 GetObject, DeleteObject, and PutObject API operations
- AWS Lambda function Invoke API
- Amazon DynamoDB PutItem, DeleteItem, and UpdateItem API operations.

To start recording CloudTrail data events, you must explicitly add the resources or resource types you want to collect activity to a trail. For single-region trails, you can log data events only for resources that you can access in that region. Though S3 buckets are global, Lambda functions and DynamoDB tables are regional. Note that you will incur additional charges for enabling data event logging.


CloudTrail > Trails > Create trail
Step 1
Choose trail attributes
Step 2
Choose log events
Step 3
Review and create
Choose log events
Events Info
Record API activity for individual resources, or for all current and future resources in AWS account. Additional charges apply
Event type
Choose the type of events that you want to log.
Management events
Capture management operations performed on your AWS resources.
Data events
Log the resource operations performed on or within a resource.
Insights events
Identify unusual activity, errors, or user behavior in your account.


Data event: S3 Info
Remove
Data event source
Select source of data events to log
S3
S3 bucket
You can choose to log read and/or write events for all buckets. You can also choose individual buckets.
All current and future S3 buckets
Read
Write
Individual bucket selection
Choose Browse to select multiple buckets, then choose to log Read, Write or both event types on all selected buckets.
bucket/prefix
Browse
Read
Write
Add bucket
Add data event type


Data event: Lambda Info
Remove
Data event source
Select source of data events to log
Lambda
Lambda function
Select the function you want to log.
All regions
All functions
Add functions
Add data event type



CloudTrail Insights is a feature that will log any unusual write API activity in your account which is then delivered to the destination S3 bucket for your trail. It uses machine learning to capture write management API usage that differs significantly from your account's typical usage patterns. And similar to data event logging, additional charges apply for logging Insights events.

References:
https://docs.aws.amazon.com/awscloudtrail/latest/userguide/cloudtrail-working-with-log-files.html
https://aws.amazon.com/premiumsupport/knowledge-center/cloudtrail-data-management-events/
https://tutorialsdojo.com/aws-cloudtrail/

Receiving CloudTrail Logs from Multiple Accounts and Sharing Logs To Other Accounts

There are occasions where one needs to monitor the CloudTrail of multiple AWS accounts, whether individually or as members of an AWS Organization. Consolidating the trails of each account into one will give you a centralized security viewpoint over the different accounts, and lets you store the trail logs in a single, secure location. To start receiving CloudTrail log files from multiple accounts, simply create an S3 bucket with cross-account write permissions for the target accounts in your master account, and configure the CloudTrail of the target accounts to publish their logs to the S3 bucket you created. After this, to make sure that audit logging does not get interrupted, you can create a policy in AWS Config that notifies you if any tampering was made to the CloudTrail configuration in the target accounts.

There are also situations when you need to share your CloudTrail logs to another AWS account, perhaps for auditing and investigation purposes. To share log files between multiple AWS accounts, you must perform the following steps:

1. Create an IAM role for each account that you want to share log files with.

2. For each of the IAM roles, create an access policy that grants read-only access to the account you want to share the log files with. For multiple account sharing, you can further restrict the policy to each account by granting read-only access to the logs that were generated by it.
3. Have an IAM user in each account assume the appropriate IAM role and retrieve the log files. Make sure that the IAM users in each account have the permission to assume their respective roles.

Once an account does not need to continue having access to the CloudTrail logs anymore, you can disable its access simply by deleting the IAM role you've created for it in the master account.

References:
https://docs.aws.amazon.com/awscloudtrail/latest/userguide/cloudtrail-receive-logs-from-multiple-accounts.html
https://docs.aws.amazon.com/awscloudtrail/latest/userguide/cloudtrail-sharing-logs.html
https://tutorialsdojo.com/aws-cloudtrail/

Amazon Simple Notification Service

The Amazon Simple Notification Service or Amazon SNS is a fully managed messaging and notification service. Amazon SNS enables you to communicate between systems through publish/subscribe patterns or "pub-sub". It enables messaging between decoupled microservice applications or directly to users via mobile push, email, or SMS. It has a resource called a "topic”. An Amazon SNS topic can be subscribed to by your application or other AWS services. For example, Amazon CloudWatch can use an SNS topic to send you an email or a Slack notification for any activities in your Auto Scaling group. You can configure one or more Amazon SQS queues to subscribe to a single SNS topic.

Amazon SNS also a feature called "message filtering” where a specific subscriber will only receive a subset of the messages. This can be done by assigning a filter policy to the topic subscription. With message filtering, you can configure one SNS Topic to publish specific messages to multiple Amazon SQS queues, based on the message type. You can use this to insurance quote type. Configure each backend application server to work its own SQS queue. This is also known as fanout event notification where messages are "pushed" to multiple subscribers, eliminating the need to periodically check or poll the queue for updates.

Amazon SNS Message Filtering

By default, an Amazon SNS topic subscriber receives every message published to the topic. There are cases when a subscriber should not be receiving every message published to a topic, or should only be receiving a subset of the messages relevant to the subscriber. To achieve this, a subscriber must assign a filter policy to the topic subscription.

A *filter policy* is a JSON object that defines the attributes to look for in a message before it is sent to a subscriber. When you publish a message to a topic, SNS first compares the message attributes to the attributes in the filter policy for each of the topic's subscriptions. If a match is found, the message is sent to the matching subscription's subscriber. If there are no filter policies in a topic, then all messages are sent to subscribers.

Since filter policies are written in JSON, the attributes are in a name: value format. A subscription accepts a message under the following conditions:

- Each attribute name in a filter policy matches an attribute name in the message.
- For each matching attribute name, at least one match exists between the values of the attribute name in the filter policy and the message attributes.

The way SNS evaluates a message against a filter policy for a match is that all policy attributes must match the message's attributes, but the message's attributes do not need to contain just the policy's attributes. Message attributes that aren't specified in the policy are just ignored by SNS.

Here is an example of an SNS subscription filter policy:

```
{
   "company": ["tutorialsdojo"],
   "platform": [{"anything-but": "Internet Explorer"}],
   "exams ": [
      "SAA",
      "SOA",
      "CDA"
   ],
   "fordiscount": [{"numeric": [">=", 5.99]}],
   "sale" : [{"exists": true}]
}
```

If we were to receive an SNS message that does not have all the attributes in the filter policy above, or if there is at least one matching attribute with a non-matching value, then the message is rejected. A filter policy can have a maximum of 5 attribute names.

In a filter policy, you can use the following conditionals to create more specific rules:

1. **Exact matching** — matches if a policy attribute value includes one or more message attribute values.
2. **Anything-but matching** — matches if a message attribute doesn't include any of the policy attribute values.
3. **Prefix matching** — matches any message attribute value that begins with the specified characters.
4. **Value range matching** — lets you use <, <=, >, and >= and = operators. Matches any message attribute that satisfies the policy attribute's operation.
5. **Attribute key matching** — uses the exists operator to check whether a message has an attribute whose key is listed in the filter policy.
6. **AND/OR logic** — You can apply AND logic using multiple attribute names. You can apply OR logic by assigning multiple values to an attribute name.

References:
https://docs.aws.amazon.com/sns/latest/dg/sns-message-filtering.html
https://tutorialsdojo.com/amazon-sns/

Amazon SNS Topic Types, Message Ordering and Deduplication

Amazon SNS has two types of topics that fulfill different requirements. We compare the two types below:

Amazon SNS Topic Type	Standard Topic	FIFO Topic
Throughput	Can support nearly unlimited number of messages per second	Can support up to 300 messages per second or 10 MB per second per FIFO topic
Ordering	Best effort; Does not guarantee that the messages are fanned out the order they come in	Guarantees the ordering of the messages. First in first out.
Message Deduplication (does not send duplicate)	Best effort; A message is delivered at least once, but occasionally more than one copy of a message is delivered.	Duplicate messages aren't delivered. Deduplication happens within a 5-minute interval, from the message publish time.
Delivery endpoints	Messages can be sent to Amazon SQS, to AWS Lambda, to Amazon Kinesis Data Firehose, through HTTP/S webhooks, through SMS, through mobile push notifications, and through email.	Messages can only be sent to SQS FIFO queue subscriptions.
Support for encryption	Messages sent to encrypted topics are immediately encrypted using a 256-bit AES-GCM algorithm and an AWS KMS CMK. Decryption occurs at the delivery endpoint.	
Fanout Limitations	Each account can have up to 100,000 Standard topics and each topic supports up to 12.5M subscriptions.	Each account can have up to 1000 FIFO topics and each topic supports up to 100 subscriptions.
Receive multiple messages in parallel	Yes	Yes, though to avoid any conflicts in the ordering, you need to consider adding another method to avoid messages arriving at the same time.

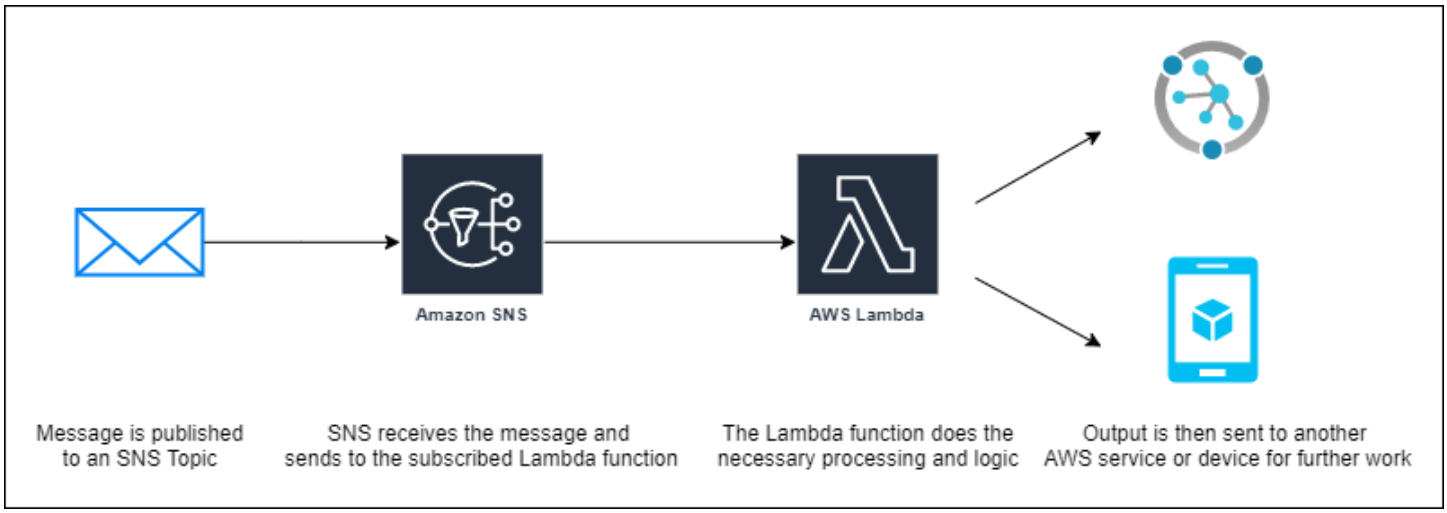
When you publish messages to an SNS FIFO topic, you set the message group ID. The group ID is a mandatory token that specifies that a message belongs to a specific message group. The SNS FIFO topic passes the group ID to the subscribed SQS FIFO queues. In the event that SNS FIFO loses access to the SQS FIFO queue (by some policy error for example), all messages are kept in SNS until the access is repaired and messages can be forwarded again.

You can avoid delivering duplicated messages by enabling content-based deduplication or by adding a deduplication ID to the messages being published. Each message published to a FIFO topic has its own sequence number. The sequence number is passed to the subscribed SQS FIFO queues as part of the message body.

References:
https://aws.amazon.com/sns/features/
https://tutorialsdojo.com/amazon-sns/

Invoke Lambda Functions Using SNS Subscription

There are many ways to invoke a Lambda function in and out of AWS; it can be invoked directly with the Lambda console, the Lambda API, the AWS SDK, the AWS CLI, and AWS toolkits. You can also configure other AWS services to invoke your function, or you can configure Lambda to read from a stream or queue and invoke your function. In this section, we'll take a look at how you can use Amazon SNS to invoke Lambda functions through subscriptions or in response to certain messages.



Amazon SNS supports Lambda functions as a target for messages sent to a topic. You can subscribe your function to topics in your account or in another AWS account. You can also choose target functions in your account or in another AWS account. For cross account subscriptions, you need to ensure that the AWS account with the target Lambda function authorizes your SNS topic to invoke their Lambda function. Additionally, you must create permissions to the target Lambda function to subscribe to your SNS topic.

To subscribe a function to a topic via the SNS console:

1) Go to your SNS console.

2) On the **Topics** page, choose a topic.
3) In the **Subscriptions** section, choose **Create subscription**.
4) On the **Create subscription** page, in the **Details** section, do the following:
    a) Verify the chosen **Topic ARN**
    b) **Protocol**: **AWS Lambda**
    c) **Endpoint**: Enter the ARN of a Lambda function.
5) Choose **Create subscription**.

You can also configure an SNS trigger in your Lambda function:

1) Go to the Lambda console and look for your function.
2) Under **Function Overview**, do the following
    a) Click **Add trigger**.
    b) Choose **SNS**.
    c) Choose the **SNS Topic** that will trigger your Lambda function.
    d) Click **Add**.
3) Save and verify your changes.

When a message is published to the SNS topic, SNS invokes the target function *asynchronously* with an event that contains the message and some metadata. The Lambda function receives the message payload as an input (event) parameter in JSON format, which you can manipulate and use however you like.

References:
https://docs.aws.amazon.com/lambda/latest/dg/with-sns.html
https://docs.aws.amazon.com/sns/latest/dg/sns-lambda-as-subscriber.html
https://tutorialsdojo.com/amazon-sns/

Amazon Simple Queue Service (Amazon SQS)

Amazon SQS is a message queueing service that uses a “polling” method, unlike Amazon SNS where messages are “pushed” to devices and targets. Amazon SQS is highly scalable and durable, and you don't need to set up any message brokers. In this section, we'll quickly take a look at the different queues that are available in Amazon SQS and the use cases of each one.

SQS Queues Types

Standard queue is your default, general purpose SQS queue. This type of queue can support a nearly unlimited number of API calls per second, per API action which are the following: SendMessage, ReceiveMessage, or DeleteMessage. Standard queues make sure to deliver your messages at least once, but because of its high throughput, there is a chance that more than one copy of a message might be delivered. Your applications should be idempotent to avoid any problems in consuming a copy of a previously consumed message. Also, standard queues do not ensure that your messages are queued in the same sequence they arrive in, so maintaining the ordering is a best effort. You can think of standard queues as the counterpart of standard topics in Amazon SNS.

Some use cases of a standard queue include:

- Decouple live user requests from intensive background work
- Allocate tasks to multiple worker nodes
- Batch messages for future processing

FIFO (first-in first-out) queue is a type of SQS queue that is designed for preserving the order of messages as they arrive, and that every message is delivered exactly once, but at the expense of some throughput speed. FIFO queues are best used for messaging when the order of messages is critical, or where duplicates can't be tolerated. Unlike standard queues where it can support a nearly unlimited number of API calls per second, FIFO queues can only support up to 300 API calls per second, per API method. If you use batching, which is grouping 10 messages into one API call, then FIFO queues can support up to 3,000 transactions per second, per batch API method (SendMessageBatch, ReceiveMessage, or DeleteMessageBatch). Similar to SNS FIFO, SQS FIFO queues use a message deduplication ID to identify sent messages. There is also the required message group ID which is a tag that indicates if a message belongs to a specific message group.

You can't convert an existing standard queue into a FIFO queue. You must either create a new FIFO queue for your application or delete your existing standard queue and recreate it as a FIFO queue.

Some use cases of a FIFO queue include:

- To make sure that user-entered commands are run in the right order.
- To display the correct product price by sending price modifications in the right order.
- To prevent a student from enrolling in a course before registering for an account.

Messages that can't be processed successfully in standard and FIFO queues are sent to a dead letter queue or DLQ.

Dead Letter Queues (DLQ)

Dead letter queues let you debug your application or messaging system to determine why some messages weren't processed successfully. The maxReceiveCount is a parameter that you specify in your queue to manage the number of times a message can fail processing. When the ReceiveCount for a message exceeds this max value, SQS moves the message to a dead-letter queue with its original message ID. Dead letter queues must be the same type as their source queues. You cannot use a standard dead letter queue for a FIFO source queue for example.

A dead letter queue lets you achieve the following:

- Configure an alarm for any messages delivered to a dead-letter queue.
- Examine logs for exceptions that might have caused messages to be delivered to a dead-letter queue.
- Analyze the contents of messages delivered to a dead-letter queue to diagnose software or the producer's or consumer's hardware issues.
- Determine whether you have given your consumer sufficient time to process messages.

Delay queues let you postpone the delivery of new messages to a queue for a short duration. If you create a delay queue, any messages that you send to the queue remain invisible to consumers for the duration of the delay period. The default and minimum delay for a queue is 0 seconds. The maximum is 15 minutes. Delay queues work similarly to visibility timeouts in that they make messages invisible from consumers for a specific period of time. The main difference between the two is that, for delay queue, a message is hidden when it is first added into the queue, whereas for visibility timeout, a message is hidden only after it is consumed from the queue.

Different queue types have different delay behaviors. For standard queues, changing the per-queue delay setting doesn't affect the delay of messages already in the queue. For FIFO queues, changing the per-queue delay setting affects the delay of messages already in the queue. You can set the delay on individual messages, rather than on an entire queue, using message timers.

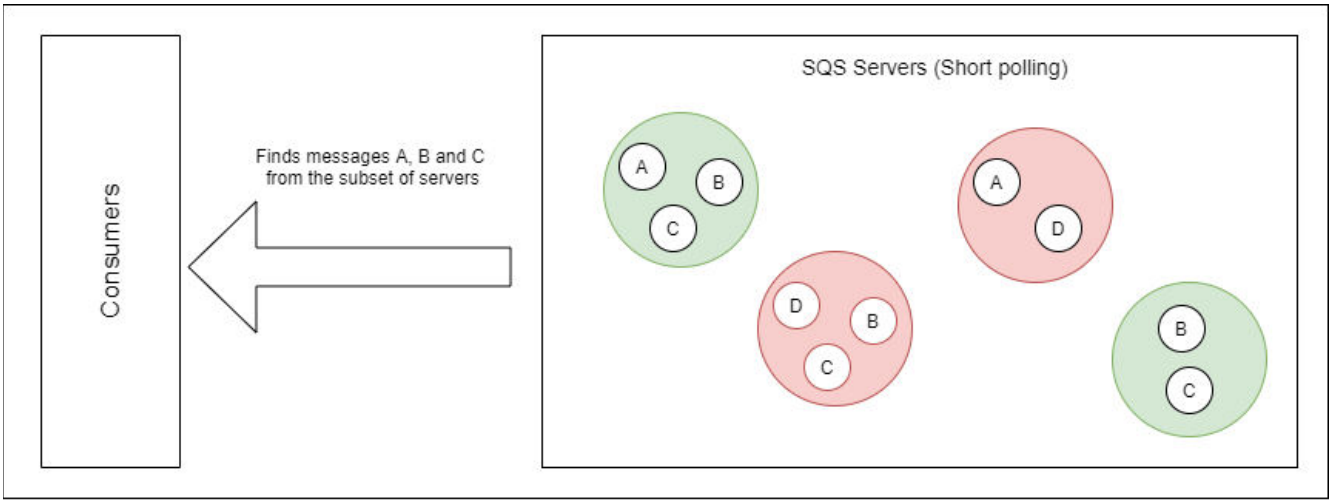
References:
https://docs.aws.amazon.com/AWSSimpleQueueService/latest/SQSDeveloperGuide/sqs-how-it-works.html
https://tutorialsdojo.com/amazon-sqs/

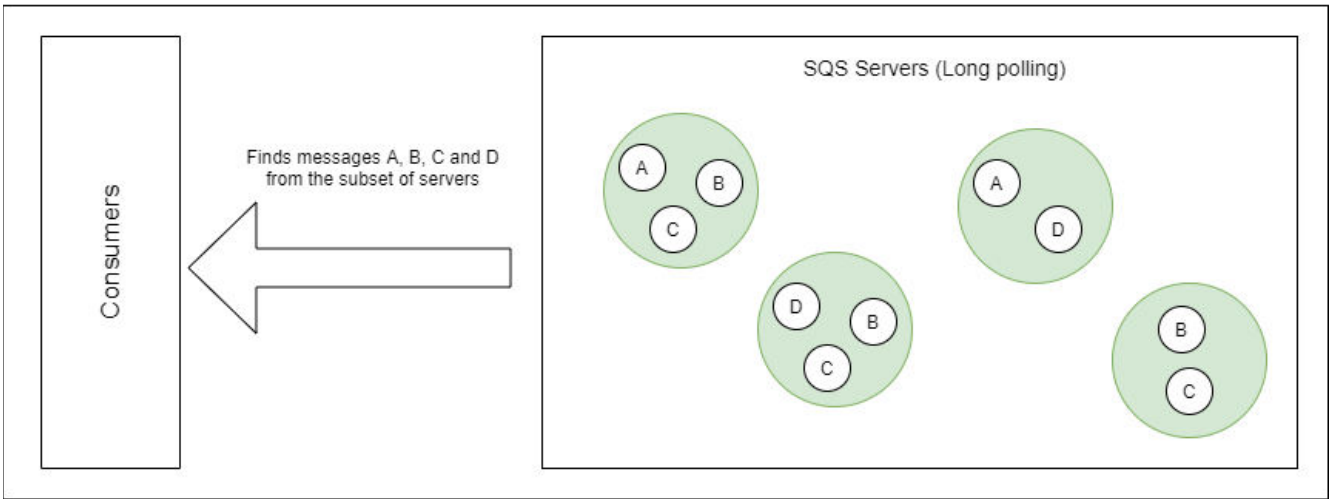
SQS Long Polling and Short Polling

Your SQS polling method determines the way SQS searches and returns your messages to you. There are two polling methods to choose from: **long polling** and **short polling**. Each polling method has its own advantages and disadvantages which we'll take a look at below.

Short polling is in effect when your wait time is 0. With short polling, the ReceiveMessage request searches only a subset of the SQS servers to find messages to include in the response. SQS sends the response right away, even if the query finds no messages. And since only a subset of servers are searched, a request might not return all of your applicable messages. Short polling is best for time-sensitive applications or batch applications that can send another query if it received an empty response previously.

Long polling is in effect when your wait time is greater than 0. With long polling, the ReceiveMessage request searches all of the SQS servers for messages. SQS returns a response after it collects at least one available message, up to the maximum number of messages specified in the request, and will only return an empty response if the polling wait time expires. The maximum long polling wait time is 20 seconds. Long polling helps reduce the cost of using SQS by eliminating the number of empty responses and false empty responses.





References:
https://docs.aws.amazon.com/AWSSimpleQueueService/latest/SQSDeveloperGuide/sqs-short-and-long-polling.html
https://tutorialsdojo.com/amazon-sqs/

Scaling Out EC2 Instances Based On SQS

Amazon SQS is able to support a high number of API calls for sending and receiving messages in a queue. You can have your applications run in an auto scaling group of EC2 instances to send and consume messages from an SQS queue in parallel to maximize work efficiency. Although, estimating the number of EC2 instances you'll need can be quite difficult if you do not use a proper metric for your auto scaling group. You'd be able to avoid this predicament if you had visibility on the number of messages in your SQS queue that needs to be processed.

There is an SQS metric in CloudWatch called ApproximateNumberOfMessagesVisible that tracks the number of messages in a queue. However, this metric might not be the most suitable for your target tracking policy since there are other factors besides the number of messages in a queue that should determine the number of auto scaling instances that you should have. You also have to consider the rate of messages processed by an auto scaling instance per unit of time and the latency between different components of your system.

Instead of tracking the number of backlog messages in a queue metric, it would be better to use *a backlog per instance* metric with the target value being the acceptable backlog per instance to maintain. To calculate your backlog per instance, get the *ApproximateNumberOfMessagesVisible* queue attribute to determine the length of the SQS queue, and divide that number by the number of auto scaling instances in the InService state. To calculate the acceptable backlog per instance, first determine how much your application can accept in terms of latency. Then, take the acceptable latency value and divide it by the average time that an EC2 instance takes to process a message.

References:
https://docs.aws.amazon.com/autoscaling/ec2/userguide/as-using-sqs-queue.html
https://tutorialsdojo.com/amazon-sqs/

Amazon Kinesis

Amazon Kinesis is a suite of services that you use to analyze data streams in real-time. Yup, you heard that right – you can analyze your data streams in real-time! The data is not just available within seconds but in a matter of milliseconds! The different Kinesis services allow you to collect, transform, process, load, and analyze the streaming data in real-time which empowers you to acquire data insights and respond to data changes quickly.

Basically, a data stream is a continuous stream, or flow, of data. For example, high-traffic websites can have a data analytics system in place that tracks each and every click on the site to analyze the users' behavior. This type of data stream is called a clickstream and there are many other kinds of data streams that need to be analyzed in real-time. The execution of data is done in a short time period, providing near-instantaneous output. This enables you to transform, process, and analyze data much faster instead of waiting until all your data is collected before the processing can begin.

Amazon Kinesis is comprised of different services, which are:

- Amazon Kinesis Data Streams
- Amazon Kinesis Data Firehose
- Amazon Kinesis Data Analytics
- Amazon Kinesis Video Streams

Amazon Kinesis Data Streams

Amazon Kinesis Data Streams is a massively scalable and durable real-time data streaming service. It can continuously capture gigabytes of data per second from thousands of different sources. The collected data is available in a matter of milliseconds for your real-time data analytics application. This can be used in real-time applications, website clickstreams, database event streams, Internet-of-Things telemetry, location-tracking events, predictive maintenance, mobile game data streams, online marketplaces with millions of financial transactions, real-time recommendations systems, and many more. It also provides ordering of records, including the ability to read and replay records in the same order to multiple Amazon Kinesis Applications. This is suitable if you have a requirement where the data events must be received in an ordered manner, or process the mobile game updates in order of receipt.

Whenever you see the keywords: “real-time” and maintaining the order of records, then the most likely service that you should use is Amazon Kinesis Data Streams. Multiple applications can consume records from the same Kinesis data stream concurrently while maintaining the order of records based on the time they are added to the stream. It also allows you to consume the records in the same order a few hours later. Amazon Kinesis Data Streams can be used to decouple your architecture. It is somehow similar to the Amazon Simple Queue Service but with notable differences. Both Kinesis Data Stream and Amazon SQS can accept data from your data sources and forward it to other AWS services for consumption. However, there are certain scenarios where it is better to use Kinesis than SQS.

First off, Amazon SQS can't process data in real-time unlike Kinesis and second, it doesn't maintain the order of data records by default. Amazon SQS has two types which are the Standard queue and First-In-First-Out or FIFO queue. The former doesn't preserve the exact order in which messages are sent and received, unlike an Amazon Kinesis Data Stream. Nevertheless, a Standard SQS Queue does provide the highest scalability and throughput. The SQS FIFO queue, on the other hand, maintains the order of records and provides exactly-once processing. The issue with this one is its speed and throughput as it only supports a limited number of transactions per second. Again, both of these SQS types don't perform in real-time, unlike Kinesis.

You can use an Amazon Kinesis Data Stream for various use cases. For example, if you need a solution that captures the clickstream data from multiple websites in real-time and analyzes it using batch processing. Amazon Kinesis Data Streams enables you to provide near-real-time recommendations for your users.

Amazon Kinesis Data Stream can also be used for a popular mobile game that streams score updates to a backend system which then posts the results on a leaderboard. A Kinesis Data Stream can be used to collect the mobile game scores in order of receipt which can then be processed by an AWS Lambda function. Finally, the processed data can be stored in a highly available database like DynamoDB, which also provides millisecond responsiveness.

You can also use this for your Internet-of-Things systems. For example, if you need to implement predictive maintenance on different types of machinery equipment using IoT sensors. These IoT sensors can send data to AWS in real-time wherein the data stream will receive events in an ordered manner for each machinery asset.

Kinesis can be used as a scalable, near-real-time solution in processing millions of financial transactions that can be consumed by your internal applications. The records in the data stream can be consumed by Amazon Kinesis Data Analytics which can be queried by your apps via SQL queries.

Let's now discuss the other services in the Amazon Kinesis family.

Amazon Kinesis Data Firehose

Amazon Kinesis Data Firehose is a fully managed service that provides an easy way to reliably transform and load your streaming data into data stores and analytics tools. It can deliver data to Amazon S3, Amazon Redshift, Amazon Elasticsearch Service, and any HTTP endpoint. You can also integrate this with your third-party service providers. Kinesis Data Firehose enables your data producers to directly send data to a specific destination or data store that you specify, without having to create custom applications or consumers. You can also configure Kinesis Data Firehose to transform your data before delivering it to the specified destination.

Amazon Kinesis Data Stream and Kinesis Data Firehose have a lot of similarities but they have certain differences too. Both of these services can accept your streaming data in real-time, however, Kinesis Data

Stream usually requires an external consumer to store the records into a data store while Kinesis Data Firehose does not. The primary use case for Kinesis Data Firehose is to act like a hose or a firehose. Imagine that your data is water and your data producer or data source is the faucet. If you want the water to continuously flow to a different location or to a bucket, then you have to use a hose. In the same way, that's what an Amazon Kinesis Data Firehose is doing. It just delivers your data stream directly to your Amazon S3 buckets, Redshift databases, Amazon ES clusters, and others without the need for a consumer.

It can transform the data before it is sent to its destination. This is internally done by invoking a Lambda function to transform the incoming source data and deliver the processed data to its destination. This is recommended if you need to parse the data stream to remove any sensitive data such as personal data or protected health information (PHI).

Amazon Kinesis Video Streams

This type of data stream in Kinesis that is used for videos. The Amazon Kinesis Video Streams is a service that securely streams video from connected devices or sources to AWS. This is used for data analytics, machine learning, video playback, and other types of media processing. It automatically provisions and scales all the required infrastructure to ingest streaming video data from millions of devices. An Amazon Kinesis Video Stream also stores, encrypts, and indexes video data in your streams to improve performance. In addition, it also provides access to your video data through a collection of easy-to-use APIs.

Amazon Kinesis Data Analytics

This is a serverless service that enables you to analyze your streaming data, acquire actionable insights, and respond to events in real-time. The Amazon Kinesis Data Analytics service reduces the complexity of building, managing, and integrating streaming applications with your custom applications and other AWS services. And since it's serverless, it is also cost-effective and highly scalable since you don't have to maintain any servers. Under the hood, it uses Apache Flink to process and analyze streaming data. Apache Flink is an open-source framework that acts as a distributed processing engine for stateful computations over unbounded and bounded data streams. The downside of using Apache Flink is the expensive overhead of setting up, managing, and maintaining your own server.

Kinesis Data Analytics also enables you to author and run code against streaming sources to provide real-time dashboards, perform time-series analytics, and create real-time metrics. With this, the data can be analyzed using SQL queries and the results can be delivered to Amazon S3, Amazon Redshift, and other data stores using Kinesis Data Firehose. You can use Java or Scala to process and analyze your streaming data.

You can also integrate this with other AWS services. For example, you can analyze the location data points of your GPS application that tracks the movement of people, bikes, automobiles, or any other moving object. The Amazon Kinesis Data Stream can send the streaming data to Kinesis Data Analytics for running near-real-time queries. You can also set up a REST API in API Gateway that can be used as an Amazon Kinesis proxy to allow access to your data points.

Kinesis Scaling, Resharding and Parallel Processing

- **Kinesis Resharding** enables you to increase or decrease the number of shards in a stream in order to adapt to changes in the rate of data flowing through the stream.
- Resharding is always pairwise. You cannot split into more than two shards in a single operation, and you cannot merge more than two shards in a single operation.
- The Kinesis Client Library (KCL) tracks the shards in the stream using an Amazon DynamoDB table, and adapts to changes in the number of shards that result from resharding. When new shards are created as a result of resharding, the KCL discovers the new shards and populates new rows in the table.
- The workers automatically discover the new shards and create processors to handle the data from them. The KCL also distributes the shards in the stream across all the available workers and record processors.
- When you use the KCL, you should ensure that **the number of instances does not exceed the number of shards** (except for failure standby purposes).
    - **Each shard is processed by exactly one KCL worker and has exactly one corresponding record processor**.
    - **One worker can process any number of shards**.
- You can scale your application to use more than one EC2 instance when processing a stream. By doing so, you allow the record processors in each instance to work in parallel. When the KCL worker starts up on the scaled instance, it load-balances with the existing instances, so now each instance handles the same amount of shards.
- To scale up processing in your application:
    - Increase the instance size (because all record processors run in parallel within a process)
    - Increase the number of instances up to the maximum number of open shards (because shards can be processed independently)
    - Increase the number of shards (which increases the level of parallelism)

Reference:
https://docs.aws.amazon.com/streams/latest/dev/kinesis-record-processor-scaling.html

Kinesis Data Streams vs Kinesis Data Firehose vs Kinesis Data Analytics vs Kinesis Video Streams

Given that there are four different variations of Amazon Kinesis, it's understandable that use cases between each of them can get confusing. Although there are definitely some scenarios where two or more Kinesis services can overlap, we have some pointers below that you can look out for to distinguish the correct service to use in the exam:

	Data Streams	Data Firehose	Data Analytics	Video Streams

Short definition	Scalable and durable real-time data streaming service.	Capture, transform, and deliver streaming data into data lakes, data stores, and analytics services.	Transform and analyze streaming data in real time with Apache Flink.	Stream video from connected devices to AWS for analytics, machine learning, playback, and other processing.
Data sources	Any data source (servers, mobile devices, IoT devices, etc) that can call the Kinesis API to send data.	Any data source (servers, mobile devices, IoT devices, etc) that can call the Kinesis API to send data.	Amazon MSK, Amazon Kinesis Data Streams, servers, mobile devices, IoT devices, etc.	Any streaming device that supports Kinesis Video Streams SDK.
Data consumers	Kinesis Data Analytics, Amazon EMR, Amazon EC2, AWS Lambda	Amazon S3, Amazon Redshift, Amazon Elasticsearch Service, generic HTTP endpoints, Datadog, New Relic, MongoDB, and Splunk	Analysis results can be sent to another Kinesis stream, a Kinesis Data Firehose delivery stream, or a Lambda function	Amazon Rekognition, Amazon SageMaker, MxNet, TensorFlow, HLS-based media playback, custom media processing application
Use cases	- Log and event data collection - Real-time analytics - Mobile data capture - Gaming data feed	- IoT Analytics - Clickstream Analytics - Log Analytics - Security monitoring	- Streaming ETL - Real-time analytics - Stateful event processing	- Smart technologies - Video-related AI/ML - Video processing

References:
https://aws.amazon.com/kinesis/
https://tutorialsdojo.com/amazon-kinesis/

AWS Glue

AWS Glue is simply a fully managed and serverless service that is primarily used for extract, transform, and load workloads or ETL. It simplifies the process of preparing and loading your data before running your data analytics workload. In AWS Glue, you can create a Data Catalog that allows you to specify and search your data that is stored on Amazon S3 and other AWS services. AWS Glue can automatically discover your data and store the associated metadata in the AWS Glue Data Catalog. Your data will be immediately searchable, queryable, and available for ETL once the metadata is stored.

AWS Glue ETL Process

AWS Glue simplifies a lot of the extract, transform, and load workloads you have because it reduces the manual processes and management tasks that you have to do. AWS Glue runs your ETL jobs in an Apache Spark serverless environment. The user has access to multiple tools under AWS Glue that provide visualizations and frameworks so you won't have to write your own code.

- AWS Glue Data Catalog lets users easily search and access data located in different data stores.
- AWS Glue Studio lets users visually create, run, and monitor ETL workflows.
- AWS Glue DataBrew lets users visually enrich, clean, and normalize data without writing code.
- AWS Glue Elastic Views lets users use SQL to combine and replicate data across different data stores.

Process:

- When initiating an ETL operation, AWS Glue Data Catalog will discover and search across your AWS data sets without moving the data. AWS Glue is able to collect both structured and semi-structured data from Amazon Redshift, Amazon S3, Amazon RDS, Amazon DynamoDB, and self-managed databases running on EC2 instances data stores. AWS Glue also supports data streams from Amazon MSK, Amazon Kinesis Data Streams, and Apache Kafka.
- If you have multiple data stores and you need to combine their data, you may use AWS Glue Elastic Views to do so and create materialized views. Views can be stored in Amazon Redshift, Amazon S3, Amazon Elasticsearch Service, Amazon DynamoDB, and Amazon RDS.
- Once the data is cataloged, it can be searched and queried using Amazon Athena, Amazon EMR, and Amazon Redshift Spectrum. AWS Glue Data Catalog stores metadata for all your data assets.
- You can compose visual workflows of ETL jobs in AWS Glue Studio and monitor their statuses there. You can also use AWS Glue Data Brew to clean and normalize your data.
- Output of the ETL jobs can be stored in AWS Lake Formation, Amazon Redshift, or Amazon S3. If further analytics is required, you may use Amazon Athena, Amazon Redshift Spectrum, Amazon EMR, Amazon Sagemaker and Amazon Quicksight to derive meaningful insights from the ETL outputs.
- Automate your succeeding ETL jobs by integrating AWS Lambda with AWS Glue.

AWS Developer Services

AWS has a lot of tools and services that can help you accelerate your application development – enabling you to create feature-rich apps in no time! These services can help you expedite the development of your web and mobile apps; easily add a user authentication capability or social logins to your applications; launch production-ready REST and GraphQL APIs in a matter of days instead of weeks or months, and deploy, manage, and connect to a data store easily.

We will cover the related development services in AWS for the AWS Certified Solutions Architect exam in this section.

AWS Amplify

AWS Amplify is one of the many development services in AWS that helps you build extensible, full-stack web and mobile apps faster. It allows you to easily start your application development and automatically scale your resources with less management overhead. In other words, AWS Amplify deploys your application in a serverless architecture without any manual intervention on your part. Plus, it also gives you the option to integrate Machine Learning capabilities into your applications!

Put simply, AWS Amplify is a set of purpose-built tools and features. It consists of the Amplify Studio, Amplify Libraries, Amplify CLI, Amplify Hosting, and many more. Let's quickly cover its different modules one by one.

Amplify Studio is a visual development environment that simplifies the development of your web and mobile apps. You can easily build your frontend UI with its set of ready-to-use UI components, which you can directly connect to your app backend. Since your UI components are already pre-built, the number of your boilerplate code will be reduced, allowing you more time to focus on implementing the application business logic.

You can define your data models, implement user authentication, and add file storage using the Amplify Studio without any backend expertise. In addition, you can import Figma prototypes built by your application designers into the Amplify Studio for more seamless collaboration.

The AWS Amplify Libraries are open-source JavaScript libraries in the Amplify Framework. These are specifically designed to aid you in building AWS-powered mobile and web apps. The Amplify JavaScript libraries are supported in React, React Native, Angular, Ionic, Vue, and different web and mobile frameworks.

The Amplify Command Line Interface is a CLI toolchain that you can use to configure and maintain your app backend straight from your local desktop. The Amplify CLI has an interactive workflow and intuitive use cases that you can leverage, such as authentication, storage, and API. It allows you to test your features locally and deploy your app to multiple environments. This tool provides infrastructure-as-code templates which are automatically loaded to CloudFormation or AWS SAM. With the Amplify CLI, you can have a much effective team collaboration and easy integration with Amplify's CI/CD workflow.

It also has a feature called Amplify Hosting, which allows you to host secure, reliable, fast web apps or websites via the AWS content delivery network. Under the hood, it deploys your applications to Amazon CloudFront's content delivery network, which comes with hundreds of points of presence globally. Amplify Hosting also allows you to set up your own custom domain for your applications and add custom alarms that send notifications if a certain metric exceeds a threshold that you specify.

These are some of the tools and features available in AWS Amplify that you can leverage with. Aside from Amplify Studio, Amplify Libraries, Amplify CLI, and Amplify Hosting, there are many other features and tools in AWS Amplify that you can try out, and AWS is adding more features soon to fast-track your software development experience!

AWS Device Farm

AWS Device Farm is an app testing service for testing and interacting with your Android, iOS, and web apps on real, physical phones and tablets. Instead of doing the manual testing yourself, you can just use this service so you can do a wide range of tests without having to provision and manage any testing infrastructure.

AWS Device Farm supports both native and hybrid Android, iOS, and progressive web apps. Cross-platform frameworks are also supported, such as PhoneGap, Titanium, Xamarin, Unity, and others.

This service is also capable of running Selenium tests on different desktop browsers and browser versions that are hosted in the AWS Cloud. AWS Device Farm can even execute your test suite on your own local machine and interact with browsers hosted on AWS Device Farm through the Selenium API.

This accelerates your mobile app development workflow by executing tests on multiple devices without the administrative overhead of maintaining a test environment. With AWS Device Farm, you can do integration tests that check and detect any customer issues before they are even released in production.

Amazon Managed Grafana

Amazon Managed Grafana is a fully managed service for Grafana. Grafana is an open-source analytics platform that is commonly used to query, visualize, observe, and make use of your system data that are gathered from multiple sources. When you hear the phrase "Amazon Managed", that means AWS is managing the underlying infrastructure required to run an open-source program or a particular tool. For Amazon Managed Grafana, AWS is the one that provisions and manages the required resources to run your Grafana dashboards, along with its other dependencies.

Let's take a good look at what Grafana really is. What you see here is an actual Grafana platform. It has tables, graphs, time series data, maps, statistics, and a range of other useful metrics. On its sidebar menu, you'll see the Dashboards, Explorer, Alert, Grafana Machine Learning, Kubernetes Monitoring, Synthetic Monitoring, and so much more. Grafana can collect metrics from different data sources so you can have a single place for all your data. In this way, all the metrics from your databases, servers, storage systems, APIs, custom data, logs, and other sources can be aggregated in one location. This is also the case in Amazon Managed Grafana.

In AWS, Amazon Managed Grafana can collect system metrics from multiple data sources in your observability stack, such as Amazon Managed Service for Prometheus, Amazon CloudWatch, and Amazon Elasticsearch Service. System alerts can also be automated by using different notification services in AWS. You can also integrate this with third-party vendors like Datadog, Splunk et cetera. In addition, you can set up your own self-managed data source like InfluxDB and integrate it with your Grafana workspace. You also don't have to worry about the infrastructure required to run your Grafana visualizations and dashboards since the necessary resources are all provisioned and managed by AWS itself.

Amazon Managed Grafana is somewhat similar to Amazon CloudWatch Dashboard since both of them have a dashboard where you can see your system metrics. However, Grafana has a much better user interface and can integrate easier with a range of other systems and data providers.

Amazon Managed Service for Prometheus

Amazon Managed Service for Prometheus is a fully managed service for the open-source monitoring tool called Prometheus. This is commonly used for monitoring modern cloud-native applications and Kubernetes clusters. Prometheus can enable you to securely ingest, store, and query metrics from different container environments.

In Amazon Managed Service for Prometheus, the resources required to run the open-source Prometheus tool are all provisioned and managed by the AWS team themselves. Scaling the underlying architecture is also handled by Amazon. You have the option to collect system metrics from your container clusters running in AWS, running on-premises, or even both.

Amazon Managed Service for Prometheus allows you to use the open-source Prometheus query language or PromQL. This query language helps you to monitor the performance of your containerized workloads that are running on the AWS Cloud or on-site. It can also scale automatically as your workloads increase or shrink and uses AWS security services to enable fast and secure access to data.

Amazon Managed Service for Prometheus has a feature called Alert Manager that handles all alerts that are sent from your workspace. These alerts are grouped and routed to downstream receivers that you specify. Duplicate alerts and messages can also be deduplicated by Alert Manager. Under the hood, it uses the Amazon Simple Notification Service as a receiver and routes messages to different SNS topics in the same account.

In this service, you create a workspace for your Prometheus metrics. A workspace isolates access control to a particular application suite for ingestion, storage, and querying of your metrics. The container metrics that are available in your Prometheus workspace can also be collected by the Amazon Managed Grafana service.

AWS Machine Learning Services

Machine learning is a branch of artificial intelligence that is all about imitating how humans learn. Just like humans, a machine can learn by watching, hearing, reading, and analyzing the things in its environment. This machine can be in the form of a simple computer program, an API, or an entire enterprise system. Once a machine consumes enough information, it can provide a range of helpful information and useful results that can save you time, effort, and even money.

Some people may find this topic to be quite difficult to understand, especially with all the complex math and algorithms it entails. To make it simple, let's start with a couple of real-world machine learning applications that you might be using already.

Most of us have a subscription to online streaming services like Netflix, Amazon Prime Video, Disney Plus, or HBO. In these streaming services, you might notice that they can recommend a particular movie or TV series based on shows that you have watched in the past. The recommendation engine will analyze your viewing history and predict the genre or type of shows you like. So if you are watching a lot of horror movies, you will be presented with different horror shows since this is the genre that you are always watching.

Let's look at another use case. Do you know that if you take a photo on your iPhone, it will automatically detect any text in the image? In this way, you can easily copy text and not manually type them on your phone. You can even try it right now! Take a picture of something that has a label, a word, or a number on it. Open that picture, then double-tap any text present in the image, and you will be able to copy them right away. This iPhone feature is called Live Text, which is a form of Optical Character Recognition that uses machine learning to transcribe text from images in your Photo Library.

Here's another example: if you're living in the US, you might have shopped at Amazon Go which is a convenience store with no checkout! Amazon Go uses computer vision and machine learning to automatically track all the products you grabbed from the store and bill you online without having to wait in line to check out your goods! This computer vision technology is also what powers self-driving cars, facial recognition systems, and many others. There is no doubt that we are already using Machine Learning in almost every moment of our lives.

In this section, we will learn about the various Machine Learning services that are available in Amazon Web Services. The primary machine learning platform in AWS is called Amazon SageMaker, which is followed by a lot of other ML services. Some services in AWS have machine learning features as well, like Amazon Aurora Machine Learning, Redshift ML, Deep Learning AMIs, and so much more. Take note that we won't cover them here as we will only focus on the dedicated ML services that are relevant to your upcoming AWS exam.

It is easy to get overwhelmed by the sheer number of services in AWS, so we will divide this lecture into several sections. The AWS Machine Learning services can be classified into these use cases: Computer Vision,

Language AI, Automated Data Extraction and Analysis, Customer Experience Improvement, Business Metrics, and DevOps.

For Computer Vision, we have:

- Amazon Rekognition
- Amazon Lookout for Vision
- AWS Panorama

For Automated data extraction and analysis, AWS has:

- Amazon Textract
- Amazon Augmented AI
- Amazon Comprehend

For Language AI, you have:

- Amazon Lex
- Amazon Transcribe
- Amazon Polly

For Customer Experience improvement, you can use the following services:

- Amazon Kendra
- Amazon Personalize
- Amazon Translate

For Business metrics, you can use:

- Amazon Forecast
- Amazon Fraud Detector
- Amazon Lookout for Metrics

And lastly, for DevOps, we have:

- Amazon DevOps Guru
- Amazon CodeGuru Reviewer
- Amazon CodeGuru Profiler
- Amazon CodeWhisperer

Let's discuss each of these services one by one.

Amazon SageMaker

First off, let's talk about Amazon SageMaker. Think of this as a full-fledged machine learning platform in AWS with tons of services, features, and components. Amazon SageMaker is not just a simple ML service but a fully managed cloud platform with lots of modules that you can use. With this, you can build, train, and deploy ML models for any use case with fully managed infrastructure, tools, and workflows. SageMaker removes the manual tasks from each step of the ML process to make it easier for you to develop high-quality models. This platform has so many modules that you choose from, namely: Amazon SageMaker Canvas, SageMaker Studio Lab, SageMaker Data Wrangler, SageMaker Autopilot, SageMaker JumpStart, SageMaker Clarify, and so much more!

Amazon Rekognition

Amazon Rekognition provides pre-trained and customizable computer vision capabilities to extract information and insights from your images and videos. Just as its name implies, it can recognize certain objects, faces, texts, scenes, labels, and other attributes from your media files or streaming videos. This service is perfect for facial recognition, where it can detect the face of a particular person or a well-known celebrity. It can also determine if someone is wearing a piece of Personal Protective Equipment like a mask, a helmet, or gloves. If you upload an image of you holding a guitar while sitting on your sofa, Amazon Rekognition can detect your face, your guitar, and the sofa where you sit. It also has a feature called Amazon Rekognition Custom Labels. With this, you can easily build a machine learning model to classify custom components or products from your dataset.

Amazon Lookout for Vision

Amazon Lookout is a suite of services that is comprised of Amazon Lookout for Equipment, Amazon Lookout for Metrics, and Amazon Lookout for Vision. The last one uses computer vision to detect defects on industrial products at scale. Amazon Lookout for Vision is primarily used in factories and manufacturing lines to quickly and accurately identify defects in each product. The dataset can be in a form of product images that are stored in an Amazon S3 bucket. You can provide a couple of baseline images to Amazon Lookout for Vision containing defect-free products, and this service will be able to automatically build a model for you within a few hours. From there, it can automatically detect anomalies in your product like dents, cracks, and scratches.

Amazon Textract

The name of this service is a combination of the words ”text” and “extract”. This will give you a hint that it is used to extract texts from scanned documents, notes, and images. Amazon Textract is a service that uses optical character recognition to automatically extract text from scanned files like PDFs, Word documents, hand-written notes, receipts, passports, IDs, and many others. What makes this service great is its capability to generate the results into a table form or a CSV file. It also has a query feature that allows you to extract a particular field using natural language questions. So if you upload your driver's license to Amazon Textract, you can submit a query like “What's the first name?” or “What's the driver's ID?“ and you'll get the value you asked for. You can also batch upload your documents to S3 and automate the text analysis process.

Amazon Augmented AI

Amazon Augmented AI or A2I is a service that provides human review workflows for common machine learning use cases. A human review literally means that a human being will review a certain output that your machine learning model generated before it can proceed to the next step of the workflow. This service augments your AI to ensure the accuracy of prediction results and helps provide continuous improvements to your machine learning model. You can directly integrate your workflows from Amazon Rekognition or Amazon Textract to Amazon Augmented AI. For example, you can do a human review of the key-value pairs that are extracted by Amazon Textract or implement image moderation by doing a human review of unsafe content, such as explicit adult or violent content from Amazon Rekognition. It is also possible to run a human review with a custom machine learning workflow of your choice.

Amazon Comprehend

Amazon Comprehend is a natural language processing service in AWS that can find insights and relationships in a text. It performs text analytics that can automatically extract key phrases, sentiment, language, syntax, topics, and even Personally Identifiable Information or PII from unstructured data. In essence, Amazon Comprehend is a service that comprehends or understands the information written in your text documents. This is different from Amazon Textract since Amazon Comprehend cannot read text from scanned documents. You need to have raw text data first in order to use Amazon Comprehend.

Amazon Lex

Amazon Lex is a machine learning service that allows you to develop chatbots. You can build Voice-based or Text-based chatbots with Amazon Lex easily. This is helpful if your company needs a self-service bot or a virtual agent for your conversational Interactive Voice Response (IVR) system, corporate website, or other customer-facing application. Amazon Lex can significantly reduce the costs of companies in maintaining its contact center.

Amazon Transcribe

Amazon Transcribe is simply a speech-to-text transcription service. The word transcribe means to make a written record of a speech, a phone call, or any spoken language, and this is exactly what Amazon Transcribe does. It is also helpful in contact centers as it can generate call transcripts and provide conversation insights to help improve customer experience and agent productivity. Amazon Transcribe also offers real-time transcription – where you can just talk to its endpoint, and it will immediately generate transcripts of your speech.

Amazon Polly

The other service relating to Language AI is called Amazon Polly. Essentially, Amazon Polly is the exact opposite of the Amazon Transcribe service. Instead of turning speech into text, it converts text into speech! If you input a text into Amazon Polly, it will generate a lifelike speech in different voices that you specify. So, for example, you typed: “Beautiful Philippine Islands” in the Amazon Polly console. You can hear the phrase: “Beautiful Philippine Islands“ in a male voice, a female voice, a kid's voice, or in any voice that you prefer. You can also customize the pronunciation of specific words and phrases by uploading your own lexicon files. A lexicon is simply a vocabulary of a particular language, and this is usually used if you have a non-English text that you want to turn into speech.

Amazon Kendra

Amazon Kendra is an intelligent search service in AWS. It is not just a typical search service that simply returns a match to your query from a single data source. Amazon Kendra can search from multiple data sources that can be structured or unstructured, then intelligently analyze the content before it sends a result. This service supports natural language processing, so you can ask questions using a language that you use in your everyday life. For instance, you can ask Amazon Kendra, “Who is the founder of the EdTech startup: Tutorials Dojo?” and it will search all of the documents in your S3 bucket, Amazon FSx file systems, Amazon RDS databases, Github repository, Jira, Slack, Sharepoint and other data sources for the answer. Again, it can search for information from a wide range of sources and not just from a traditional SQL database, then uses machine learning to provide context to your search results.

Amazon Personalize

Amazon Personalize is a service that provides personalized recommendations to your customers based on their past activity and behavior. It's just like the recommendation feature in Amazon Prime or Netflix, where new movies are automatically recommended for you based on your viewing history. If you watched a lot of Sci-Fi movies on their online streaming platform, they will automatically recommend more Sci-Fi shows on your profile. This is definitely a customer experience improvement since personalizing the user's content tends to convert more because they align with what the customers actually do and buy.

Amazon Translate

Amazon Translate is a real-time translation service in AWS. It works pretty much like Google Translate, where you input text in one language, and the service will translate it to a language that you choose. You can also create your own custom terminology. This allows you to customize the output of Amazon Translate based on a company-specific and domain-specific vocabulary. For example, I can set the acronym TD as “Tutorials Dojo” in English. The Amazon Translate service can accept input with my custom vocabulary and include it in the translation. In this case, I can enter the Tagalog phrase “Magandang Umaga TD” and Amazon Translate will return “Good morning Tutorials Dojo” as an output. The Filipino phrase “magandang umaga” means “good morning” in English, while “TD” is the custom term for “Tutorials Dojo” which we configured in Amazon Translate. You can also enable the Formality option, which controls whether the translation output uses a formal tone or not. The translation can also mask profane words or phrases, which is a very useful feature for customer-facing applications.

Amazon Forecast

Amazon Forecast is a machine learning service in AWS that helps you forecast a future outcome based on your historical records and other relevant data. You can either import or stream your time-series data to Amazon Forecast, and it can foretell your sales, web traffic, inventory, revenue, cloud resource capacity, or even the actual weather in the coming days or months ahead. It can also predict your future AWS bill! It has a range of built-in datasets too, like the Weather Index and the national holidays for various countries. Amazon Forecast uses a machine learning model called a Predictor. This “Predictor” uses an algorithm to consume all the time-series data that you provide and generate a prediction out of it.

Amazon Fraud Detector

Amazon Fraud Detector is yet another machine learning service that can automate fraud detection, just as its name suggests. It can identify potential fraudulent activity, fake reviews and spam account creation in neal-real-time. For instance, your website recently got a visitor whose IP address has a history of malicious activity such as spamming, hacking attempts, and DDoS attacks. Users with exactly the same IP address are posting spam on your website repeatedly. For this situation, you can use Amazon Fraud Detector to block any visitor who uses an offending IP address, an email domain, or any other attribute that you define.

Amazon Lookout for Metrics

Amazon Lookout for Metrics is one of the services of the Amazon Lookout family for detecting anomalies in your business metrics. An anomaly can be a sudden nosedive in your sales revenue or an unexpected drop in your customer acquisition rates. It can identify unusual variances in your business metrics and alert you immediately so you can take the proper course of action.

Amazon DevOps Guru

Amazon DevOps Guru detects abnormal behavior in your application or AWS resources that might cause unexpected downtimes or operational issues in the near future. It can monitor applications and AWS resources within your own account or on all accounts across your AWS Organization. It uses machine learning to identify operational defects long before they impact you and your customers. Amazon DevOps Guru can analyze your RDS databases and automatically determine an unusually high DB load that is more than three times or 5 times its normal value. It can also detect issues in your serverless stack like an extremely high number of invocations in your Lambda function that is beyond the currently provisioned concurrency or an overprovisioned write capacity on your DynamoDB tables.

Amazon CodeGuru

Amazon CodeGuru is a suite of development services in AWS. It contains different tools and features such as Amazon CodeGuru Reviewer, Amazon CodeGuru Profiler, BugBust, and many more. The primary function of Amazon CodeGuru Reviewer is to provide intelligent recommendations for improving your application performance, efficiency, and code quality. It can scan your code and detect a plethora of code defects like bad exception handling, insecure CORS policy, path traversal, hardcoded credentials, and many more. You can also integrate this with your CI/CD workflow so you can run code reviews and recommendations to improve your codebase. The other module for this service is called the Amazon CodeGuru Profiler. A profiler is basically a component that collects your CPU data and analyzes the runtime performance data from your live applications. This is helpful in identifying expensive lines of codes that inefficiently use the CPU, which causes CPU bottlenecks.

Amazon CodeWhisperer

Amazon CodeWhisperer is a coding tool that automatically generates code and functions in real-time. This tool is similar to Github CoPilot, which is an extension that you usually install in your visual studio IDE. The lines of codes are generated right from your IDE editor based on the comments that you write. For example, you can simply write a comment that outlines a specific task in plain English, such as “Upload a file to an Amazon S3 bucket with server-side encryption”. Amazon CodeWhisperer will take your comment as input and generate an entire function in the programming language that you define, which can upload a file to your S3 bucket with the required encryption and many more

AWS Deployment Services

There are different ways to build, test, and deploy your applications to your development or production environments. Back in the day, we used to just copy a ZIP file, a WAR file, or the binary files in the webapps directory of our web servers. Then, we have to manually configure and restart our Apache, NGINX, and IBM WebSphere servers to reflect the changes. This process is called a manual deployment and is prone to a lot of human errors. There are a lot of moving parts that you have to do by hand, which may take several hours to complete. Worse, provisioning the required virtual web server or a dedicated database cluster will take you weeks or even months to accomplish due to a lack of automation. Good thing that nowadays, you can do your deployment in a blink of an eye through the various deployment tools available at your disposal.

AWS offers a number of services that provide deployment and management capabilities for one or more aspects of your application lifecycle. These services enable organizations to build and deliver applications faster than their traditional CI/CD workflow. You and your team don't have to spend a lot of time manually provisioning, configuring, updating, monitoring, or securing your AWS resources anymore. These laborious tasks can be programmed into code and easily automated with the different deployment services available in AWS. Some of these services use the concept of "Infrastructure as Code" or IaC. An IaC is a process of managing and provisioning your servers, databases, CDNs, and other resources through machine-readable definition files. Simply put, you only need to provide a text-based definition file that will automatically provision the required resources for your application in just one click of a button.

The advent of cloud computing enables companies to deploy all their workloads entirely in the cloud. They also have the option to run a hybrid cloud architecture in which they utilize both their physical on-premises resources and cloud services in AWS at the same time. There's even an option now to do a multi-cloud deployment where you deploy your infrastructure to AWS, Azure, Google Cloud, and other public cloud providers simultaneously. You can run both your multitier applications and Kubernetes or Docker container cluster almost anywhere – whether it be on the cloud, on-premises, on multiple public clouds, or a combination of all three.

Let's discover the different deployment services in this lesson. These services have the capability to provision, configure, deploy, scale, and monitor your cloud architecture without any manual intervention on your part. They are:

- AWS CloudFormation
- AWS Elastic Beanstalk
- AWS CodeDeploy
- Amazon ECS Anywhere
- Amazon EKS Anywhere
- AWS OpsWorks
- AWS Proton

AWS CloudFormation

AWS CloudFormation is a service that enables you to provision and manage your AWS resources using a custom code template. You can create a custom template in YAML or JSON format that defines the AWS resources that you require, like Amazon EC2 instances, Amazon FSx filesystems, Amazon Aurora databases, CloudFront distributions, or any other resource, and the AWS CloudFormation service can deploy all of these automatically for you. AWS CloudFormation also comes with a graphic tool called the CloudFormation designer. This is a drag-n-drop online tool for creating, viewing, and modifying your AWS CloudFormation templates.

CloudFormation is the primary Infrastructure as Code service in AWS. It works like any other IaC tools in the market, like Terraform, Ansible, Chef, and Puppet. The key difference, however, is the additional features that it provides, which are fully compatible with the AWS Cloud.

A CloudFormation template deploys your cloud infrastructure resources in a group called a "stack". This stack can represent your entire cloud architecture or just its subset. If you have a large multi-tier architecture, you can create multiple templates to represent the different tiers of your enterprise application suite. You can create a CloudFormation template for your presentation layer stack, another template for your application layer stack, and one more for your data layer stack.

You can bundle your multiple stacks together into something called a nested stack. In CloudFormation, you can have a root stack with a hierarchy of nested stacks under it. This will effectively make the modules of your infrastructure code to be loosely coupled with each other, which makes the management of each individual stack much easier. Your application layer stack will only contain EC2, EKS, ECS, and other compute resources, while the data layer stack will have RDS, Aurora, DocumentDB, Amazon Neptune, Amazon Timestream, or any database services. Any change in one nested stack won't adversely affect the other stack.

Aside from provisioning your resources, Amazon CloudFormation allows you to change, modify, or scale your services that are already deployed in your AWS account. It even has a dry-run mode to check your upcoming changes before they are deployed in your cloud environment. This feature is called a "Change Set".

Essentially, a change set allows you to see how your changes might impact your running resources before finally implementing them. Say you want to change the name of your Amazon Aurora Serverless database. You can create a change set in CloudFormation that will create a new Aurora database and delete the old one. Of course, you will lose the data in the old database unless you've taken a DB snapshot for backup. The change set will show you the upcoming change so you can plan accordingly before you update your stack.

A CloudFormation stack is usually mapped in a single AWS account only, so if you are running your applications in two or more AWS accounts, you can use StackSets. A StackSet extends the capability of CloudFormation stacks by enabling you to create, update, or delete stacks across multiple accounts and AWS

Regions with a single operation. You can select an administrator account in your AWS Organization and then choose a particular CloudFormation template for your StackSet. This template will be the basis for provisioning the stacks into your selected target accounts.

The stack, change set, StackSet, and the graphical designer tool are the base features of the AWS CloudFormation service. There are different services in AWS that extend the capabilities in CloudFormation, namely the AWS Cloud Development Kit and AWS Serverless Application Model. The AWS Cloud Development Kit, or AWS CDK for short, is an open-source software development kit for Amazon Web Services. You can use this to programmatically model your AWS infrastructure using TypeScript, Python, Java, .NET, or any other programming languages that you prefer.

AWS Serverless Application Model (AWS SAM)

The AWS Serverless Application Model service, or AWS SAM, is an open-source framework that simplifies the development of your serverless applications on AWS. This is commonly used if your cloud stack is using AWS Lambda, Amazon DynamoDB, API Gateway, and other serverless services. AWS SAM can also have a SAM template that is essentially just an extension of the AWS CloudFormation template. A SAM template has some additional components that make it easier for you to work with serverless services in AWS.

Your apps can be stored in the AWS Serverless Application Repository. This repository service makes it easy for developers and companies to deploy, manage, and share their serverless applications in AWS and to the greater public. You can easily publish your serverless apps and share them with the community at large or privately within your organization. Each and every application in the AWS Serverless Application Repository is packaged with an AWS Serverless Application Model (SAM) template that defines the different AWS resources which the application uses.

AWS Elastic Beanstalk

AWS Elastic Beanstalk is a managed platform that allows you to upload your application code in AWS and provision the required cloud environment easily. You only need to upload your application package that is written in Java, .NET, PHP, Node.js, Python, Ruby, Go, or Docker, and then Elastic Beanstalk will deploy the necessary resources to run your application. You can either run a Web Server environment or a Worker environment. A Web Server environment runs a static website, a web app, or a web API that serves HTTP requests, while a worker environment, on the other hand, runs a worker application that processes long-running workloads on demand. The latter also performs tasks on a schedule that you define and can be integrated with the Amazon SQS queue. The AWS Elastic Beanstalk service also uses a configuration file like CloudFormation, to automatically deploy and configure your applications. These configuration files in Elastic Beanstalk are stored in the .ebextensions folder.

AWS CodeDeploy

AWS CodeDeploy is a fully managed deployment service that automates your application deployments in AWS. You can deploy your applications to Amazon EC2 instances, Amazon ECS clusters, AWS Lambda functions, and other computing services in AWS. You can even use this to deploy your application to the servers located on your on-premises network. This service is different from AWS CloudFormation, AWS SAM, and Elastic Beanstalk since you can only deploy your applications to existing compute resources. AWS CodeDeploy does not create AWS resources on your behalf and is intended for application deployments only.

Amazon ECS Deployment Options

Amazon ECS is a fully managed container orchestration service that supports Docker containers. This orchestration service allows you to easily run containerized applications on a managed cluster that you can control. Basically, container orchestration is an automation tool that reduces the operational effort needed to run your containerized workloads and services. Amazon ECS can automatically orchestrate or control the manual tasks of provisioning, deploying, scaling, networking, and many other tasks, so you don't have to.

Amazon ECS Anywhere is a feature of Amazon ECS that enables you to run and manage container workloads on your own on-premises infrastructure. You'll still have the same cluster management, workload scheduling, monitoring, and support features in Amazon ECS Anywhere, whether you are running your workload on-premises or in the cloud. This ubiquitous service can help you meet your compliance requirements and scale your hybrid architecture without undermining the previous investments you already have in your on-premises hardware.

When you create an Amazon ECS cluster, you can choose whether the compute resources will be deployed or launched in your own VPC, in AWS Fargate, or externally via Amazon ECS Anywhere. A cluster launched in a VPC uses Amazon EC2 instances that are orchestrated and controlled by Amazon ECS. You can use the Amazon CloudWatch Container Insights to monitor your container workloads.

If the launch type is AWS Fargate, then your cluster will be serverless, and the computing resources will be fully managed by AWS. Using AWS Fargate can significantly reduce the operating costs of running your containers. If you decide to launch your cluster externally, then your cluster will be run on your own server located on-site through the help of the Amazon ECS Anywhere service.

Amazon EKS Deployment Options

Amazon Elastic Kubernetes Service or Amazon EKS is a managed service that you can use to run Kubernetes on AWS. It's like Amazon ECS, but instead of Docker containers, this service is used for running Kubernetes clusters. Amazon EKS automates the installation, operation, and maintenance of your own Kubernetes control plane, pods, and nodes.

You can deploy your Kubernetes cluster in various ways in AWS and can include additional networking add-ons to improve your containerized architecture. A Kubernetes container can be deployed to an Amazon EKS cluster in your AWS account, to Amazon EKS on AWS Outposts, to Amazon EKS Anywhere, and through the Amazon EKS Distro. The first option allows you to launch a Kubernetes cluster using managed or self-managed Amazon EC2 nodes that you can customize and control. You can also choose to deploy your Kubernetes pods on AWS Fargate to make the cluster serverless and extremely cost-effective.

The second type is Amazon EKS on AWS Outposts which is a deployment option that uses a physical AWS Outpost rack on your on-premises network to run your Kubernetes workloads. The data plane is also located on-premises, so you can have more control compared with running it exclusively in AWS.

Using Amazon EKS Anywhere is another way to deploy your containers on-premises. It works like Amazon ECS Anywhere, which allows you to run your Kubernetes cluster entirely on your own. This means that the hardware, app deployment location, control plane, and data plane are all controlled on your own physical network. This gives you extensive control over all the components of your containerized application suite while maintaining official support from AWS.

The other deployment option that you can choose is Amazon EKS Distro. The word "distro" simply refers to the distribution of the same open-source Kubernetes software deployed by Amazon EKS in the AWS cloud. Amazon EKS Distro follows the same Kubernetes version release cycle as Amazon EKS and is provided to you as an open-source project that you can deploy on your own computer or on-site environment. It's similar to the Amazon EKS Anywhere option, except that it does not include support services offered by AWS.

AWS OpsWorks

AWS OpsWorks is a configuration management service that provides managed instances for your Chef and Puppet-based automation platforms. Chef and Puppet are IaC platforms that automate the manual task of provisioning and configuring your servers. AWS OpsWorks automates how your servers are provisioned, configured, and managed across your Amazon EC2 instances or your own physical servers situated in your corporate building. OpsWorks has three offerings, namely: AWS OpsWorks for Chef Automate, AWS OpsWorks for Puppet Enterprise, and AWS OpsWorks Stacks.

AWS Proton

AWS Proton is a service that automates container and serverless deployments in AWS. It empowers your platform teams and developers to have consistent development standards and best practices. This service is very useful if you have a large number of developers in your organization. AWS Proton enables your developers to deploy container and serverless applications using pre-approved stacks that your platform team manages. It balances control and flexibility in your organization by allowing developers to innovate within the set guardrails that you implement.

It also offers a self-service portal for your developers, which contains AWS Proton templates that they can use and deploy. A Proton template contains all the information required to deploy your custom environments and services. You can create an AWS Proton Component as well, which provides flexibility to your service templates. These components in AWS Proton provide platform teams with a way to extend core infrastructure patterns and define guardrails for your developers.

Comparison of AWS Services and Features

AWS CloudTrail vs Amazon CloudWatch

- **CloudWatch** is a monitoring service for AWS resources and applications. **CloudTrail** is a web service that records API activity in your AWS account. They are both useful monitoring tools in AWS.
- By default, **CloudWatch** offers free basic monitoring for your resources, such as EC2 instances, EBS volumes, and RDS DB instances. **CloudTrail** is also enabled by default when you create your AWS account.
- With **CloudWatch**, you can collect and track metrics, collect and monitor log files, and set alarms. **CloudTrail**, on the other hand, logs information on who made a request, the services used, the actions performed, parameters for the actions, and the response elements returned by the AWS service. CloudTrail Logs are then stored in an S3 bucket or a CloudWatch Logs log group that you specify.
- You can enable detailed monitoring from your AWS resources to send metric data to CloudWatch more frequently, with an additional cost.
- **CloudTrail** delivers one free copy of management event logs for each AWS region. Management events include management operations performed on resources in your AWS account, such as when a user logs in to your account. Logging data events are charged. Data events include resource operations performed on or within the resource itself, such as S3 object-level API activity or Lambda function execution activity.
- **CloudTrail** helps you ensure compliance and regulatory standards.
- **CloudWatch Logs** reports on application logs, while **CloudTrail Logs** provide you specific information on what occurred in your AWS account.
- **CloudWatch Events** is a near real time stream of system events describing changes to your AWS resources. **CloudTrail** focuses more on AWS API calls made in your AWS account.
- Typically, **CloudTrail** delivers an event within 15 minutes of the API call. **CloudWatch** delivers metric data in 5 minutes periods for basic monitoring and 1 minute periods for detailed monitoring. The CloudWatch Logs Agent will send log data every five seconds by default.

AWS DataSync vs Storage Gateway

	DataSync	Storage Gateway
Description	AWS DataSync is an online data transfer service that simplifies, automates, and accelerates the process of copying large amounts of data to and from AWS storage services over the Internet or over AWS Direct Connect.	AWS Storage Gateway is a hybrid cloud storage service that gives you on-premises access to virtually unlimited cloud storage by linking it to S3. Storage Gateway provides 3 types of storage interfaces for your on-premises applications: file, volume, and tape.
How it Works	Uses an **agent** which is a virtual machine (VM) that is owned by the user and is used to read or write data from your storage systems. You can activate the agent from the Management Console. The agent will then read from a source location, and sync your data to Amazon S3, Amazon EFS, or Amazon Fsx for Windows File Server.	Uses a **Storage Gateway Appliance** - a VM from Amazon - which is installed and hosted on your data center. After the setup, you can use the AWS console to provision your storage options: File Gateway, Cached Volumes, or Stored Volumes, in which data will be saved to Amazon S3. You can also purchase the hardware appliance to facilitate the transfer instead of installing the VM.
Protocol	DataSync connects to existing storage systems and data sources with standard storage protocols (NFS, SMB), or using the Amazon S3 API.	Storage Gateway provides a standard set of storage protocols such as iSCSI, SMB, and NFS.
Storage	AWS DataSync can copy data between Network File Systems (NFS), SMB file servers or self-managed object storages. It can also move data between your on-premises storage and AWS Snowcone, Amazon S3, Amazon EFS, or Amazon FSx.	File Gateway enables you to store and retrieve objects in Amazon S3 using file protocols such as NFS and SMB. Volume Gateway stores your data locally in the gateway and syncs them to Amazon S3. It also allows you to take point-in-time copies of your volumes with EBS snapshots which you can restore and mount to your appliances as iSCSI devices. Tape Gateway data is immediately stored in Amazon S3 and can be archived to Amazon S3 Glacier or Amazon S3 Glacier Deep Archive.
Pricing	You are charged standard request, storage, and data transfer rates to read from and write to AWS services, such as Amazon S3, Amazon EFS, Amazon FSx for Windows File Server, and AWS KMS.	You are charged based on the type and amount of storage you use, the requests you make, and the amount of data transferred out of AWS.
Combination	You can use a combination of DataSync and File Gateway to minimize your on-premises' operational costs while seamlessly connecting on-premises applications to your cloud storage. AWS DataSync enables you to automate and accelerate online data transfers to AWS storage services. File Gateway then provides your on-premises applications with low latency access to the migrated data. Tutorials Dojo	

S3 Transfer Acceleration vs Direct Connect vs VPN vs Snowball Edge vs Snowmobile

S3 Transfer Acceleration (TA)

- Amazon S3 Transfer Acceleration makes public Internet transfers to S3 faster, as it leverages Amazon CloudFront's globally distributed AWS Edge Locations.
- There is no guarantee that you will experience increased transfer speeds. If S3 Transfer Acceleration is not likely to be faster than a regular S3 transfer of the same object to the same destination AWS Region, AWS will not charge for the use of S3 TA for that transfer.
- This is not the best transfer service to use if transfer disruption is not tolerable.
- S3 TA provides the same security benefits as regular transfers to Amazon S3. This service also supports multi-part upload.
- ***S3 TA vs AWS Snow****
    - The AWS Snow* Migration Services are ideal for moving large batches of data at once. In general, if it will take more than a week to transfer over the Internet, or there are recurring transfer jobs and there is more than 25Mbps of available bandwidth, S3 Transfer Acceleration is a good option.
    - Another option is to use AWS Snowball Edge or Snowmobile to perform initial heavy lift moves and then transfer incremental ongoing changes with S3 Transfer Acceleration.
- ***S3 TA vs Direct Connect***
    - AWS Direct Connect is a good choice for customers who have a private networking requirement or who have access to AWS Direct Connect exchanges. S3 Transfer Acceleration is best for submitting data from distributed client locations over the public Internet, or where variable network conditions make throughput poor.
- ***S3 TA vs VPN***
    - You typically use (IPsec) VPN if you want your resources contained in a private network. VPN tools such as OpenVPN allow you to set up stricter access controls if you have a private S3 bucket. You can complement this further with the increased speeds from S3 TA.
- ***S3 TA vs Multipart Upload***
    - Use multipart upload if you are uploading large files and you want to handle failed uploads gracefully. With multipart upload, each part of your upload is a contiguous portion of the object's data. You can upload these object parts independently and in any order. If transmission of any part fails, you can retransmit that part without affecting other parts.
    - For S3 TA, as the name implies, accelerates your transfer speeds, not just for upload but also for download speed. There is no reason why you can't use S3 TA and multipart upload together, but if you are only handling small files, using multipart upload is not necessary.

AWS Direct Connect

- Using AWS Direct Connect, data that would have previously been transported over the Internet can now be delivered through a **private physical network connection** between AWS and your datacenter or

corporate network. Customers' traffic will remain in AWS global network backbone, after it enters AWS global network backbone.

- Benefits of Direct Connect vs internet-based connections
    - reduced costs
    - increased bandwidth
    - a more consistent network experience
- Each AWS Direct Connect connection can be configured with one or more **virtual interfaces**. Virtual interfaces may be configured to access AWS services such as Amazon EC2 and Amazon S3 using public IP space, or resources in a VPC using private IP space.
- You can run IPv4 and IPv6 on the same virtual interface.
- Direct Connect does not support multicast.
- A Direct Connect connection is **not redundant**. Therefore, a second line needs to be established if redundancy is required. Enable *Bidirectional Forwarding Detection* (BFD) when configuring your connections to ensure fast detection and failover.
- AWS Direct Connect offers SLA.
- Direct Connect vs IPsec VPN
    - A VPC VPN Connection utilizes IPSec to establish **encrypted network connectivity** between your intranet and Amazon VPC **over the Internet.** VPN Connections can be configured in minutes and are a good solution if you have an immediate need, have low to modest bandwidth requirements, and can tolerate the inherent variability in Internet-based connectivity. AWS Direct Connect **does not involve the public Internet**; instead, it uses **dedicated, private network connections** between your intranet and Amazon VPC.
- You can combine one or more Direct Connect dedicated network connections with the Amazon VPC VPN. This combination provides an IPsec-encrypted private connection that also includes the benefits of Direct Connect.

AWS VPN

- AWS VPN is comprised of two services:
    - AWS Site-to-Site VPN enables you to securely connect your on-premises network or branch office site to your Amazon VPC.
    - AWS Client VPN enables you to securely connect users to AWS or on-premises networks.
- Data transferred between your VPC and datacenter routes over an encrypted VPN connection to help maintain the confidentiality and integrity of data in transit.
- If data that passes through Direct Connect moves in a dedicated private network line, AWS VPN instead encrypts the data before passing it through the public Internet.
- VPN connection throughput can depend on multiple factors, such as the capability of your customer gateway, the capacity of your connection, average packet size, the protocol being used, TCP vs. UDP, and the network latency between your customer gateway and the virtual private gateway.
- All the VPN sessions are **full-tunnel VPN**. (cannot split tunnel)
- AWS Site-to-Site VPN enables you to create **failover** and CloudHub solutions **with AWS Direct Connect**.

- AWS Client VPN is designed to connect devices to your applications. It allows you to use an **OpenVPN-based client**.

Snowball Edge

- Snowball Edge is a **petabyte-scale data transport** solution that uses secure appliances to transfer large amounts of data into and out of AWS.
- Benefits of Snowball Edge include:
    - lower network costs,
    - Shorter transfer times,
    - and security using 256-bit encryption keys you manage through AWS Key Management Service (KMS)..
- Options for device configurations
    - **Storage optimized** – this option has the most storage capacity at up to 80 TB of usable storage space, 24 vCPUs, and 32 GiB of memory for compute functionality. You can transfer up to **100 TB** with a single Snowball Edge Storage Optimized device.
    - **Compute optimized** – this option has the most compute functionality with 52 vCPUs, 208 GiB of memory, and 7.68 TB of dedicated NVMe SSD storage for instance. This option also comes with 42 TB of additional storage space.
    - **Compute Optimized with GPU** – identical to the compute-optimized option, save for an installed GPU, equivalent to the one available in the P3 Amazon EC2 instance type.
- Similar to Direct Connect, AWS Snowball Edge is **physical hardware**. It includes a 10GBaseT network connection. You can order a device with either **50TB** or an **80TB** storage capacity.
- Data transported via Snowball Edge are stored in Amazon S3 once the device arrives at AWS centers.
- AWS Snowball Edge is not only for shipping data into AWS, but also out of AWS.
- AWS Snowball Edge can be used as a quick order for additional temporary petabyte storage.
- You can cluster Snowball Edge devices for local storage and compute jobs to achieve 99.999 percent data durability across 5–10 devices, and to locally grow and shrink storage on demand.
- For security purposes, data transfers must be completed **within 360 days of a Snowball Edge's preparation**.
- When the transfer is complete and the device is ready to be returned, the E Ink shipping label will automatically update to indicate the correct AWS facility to ship to, and you can track the job status by using Amazon Simple Notification Service (SNS), text messages, or directly in the console.
- Snowball Edge is the best choice if you need to more securely and quickly transfer terabytes to many petabytes of data to AWS. Snowball Edge can also be the right choice if you don't want to make expensive upgrades to your network infrastructure, if you frequently experience large backlogs of data, if you're located in a physically isolated environment, or if you're in an area where high-bandwidth Internet connections are not available or cost-prohibitive.
- For latency-sensitive applications such as machine learning, you can deploy a **performance-optimized SSD volume (sbp1)**. Performance optimized volumes on the Snowball Edge Compute Optimized device

use NVMe SSD, and on the Snowball Edge Storage Optimized device they use SATA SSD. Alternatively, you can use capacity-optimized **HDD volumes (sbg1)** on any Snowball Edge.
- If you will be transferring data to AWS on an ongoing basis, it is better to use AWS Direct Connect.
- If multiple users located in different locations are interacting with S3 continuously, it is better to use S3 TA.
- You **cannot** export data directly from S3 Glacier. It should be first restored to S3.

Snowmobile

- Snowmobile is Snowball Edge with larger storage capacity. Snowmobile is literally a mobile truck.
- Snowmobile is an **Exabyte-scale data transfer** service.
- You can transfer up to **100PB** per Snowmobile.
- Snowmobile uses multiple layers of security to help protect your data including dedicated security personnel, GPS tracking, alarm monitoring, 24/7 video surveillance, and an optional escort security vehicle while in transit. All data is encrypted with 256-bit encryption keys you manage through the AWS Key Management Service (KMS).
- After the data transfer is complete, the Snowmobile will be returned to your designated AWS region where your data will be uploaded into the AWS storage services such as S3 or Glacier.
- Snowball Edge vs Snowmobile
    - To migrate large datasets of 10PB or more in a single location, you should use Snowmobile. For datasets less than 10PB or distributed in multiple locations, you should use Snowball Edge.
    - If you have a high speed backbone with hundreds of Gb/s of spare throughput, then you can use Snowmobile to migrate the large datasets all at once. If you have limited bandwidth on your backbone, you should consider using multiple Snowball Edge to migrate the data incrementally.
    - Snowmobile **does not** support data export. Use Snowball Edge for this cause.
- When the data import has been processed and verified, AWS performs a software erasure based on NIST guidelines.

Amazon EBS vs EC2 Instance Store

	Amazon EBS volumes	EC2 instance store
Definition	Disk drives that you can virtually mount onto EC2 instances for persistent, block-level storage.	Physical disks mounted directly on the host computer of your EC2 instances that provide temporary block-level storage.
Lifespan	An EBS volume exists independently from EC2 instances. Even if your EC2 instances are terminated, you can retain your EBS volumes.	The instance store is deleted once you stop, reboot or terminate the EC2 instance.
Volume Types	1. General purpose SSD (gp2, gp3) 2. Provisioned IOPS SSD (io1, io2) 3. Throughput Optimized HDD (st1) 4. Cold HDD (sc1)	1. HDD 2. SSD 3. NVMe SSD
Availability	Only available in the AZ where it was launched, but snapshots can be copied to another AWS Region.	Only available on the instance where it was launched with.
Sizing constraints	Min of 1GiB and max of 16 TiB per volume. Size of volumes can be upgraded without downtime.	Storage size depends on the instance type you use. If it is used as a root volume, the maximum size is 10GB.
Remounting capabilities	Can be detached and reattached to another EC2 instance	No remounting capabilities since physical disks are directly attached to the host computer.
Multi-attach features	Lets you attach a single Provisioned IOPS SSD (io1 or io2) volume to multiple instances that are in the same Availability Zone.	Not supported
Backup and restore	Via EBS snapshots which are incremental backups of your EBS volumes. Backups are stored in S3 which you cannot directly access except through the EBS interface.	AMI backups

Native encryption support	AWS KMS encryption	AWS hardware encryption
Pricing	You are billed for the amount of storage provisioned, amount of IOPS provisioned, and/or amount of throughput provisioned. Pricing varies between AWS Regions and volume types.	Included as part of the EC2 instance's usage cost.
Use cases	• Boot volume • Persistent data store even after EC2 instance is stopped. • Backup and restore capabilities • Multi attach capabilities • High IO/Throughput volumes • Can be swapped between instances • Encryption via KMS	• Boot volume for some instance types • Very high IO/Throughput because directly attached to the physical machine • Temporary storage

Amazon S3 vs EBS vs EFS

Tutorials Dojo	S3	EBS	EFS
Type of storage	Object storage. You can store virtually any kind of data in any format.	Persistent block level storage for EC2 instances.	POSIX-compliant file storage for EC2 instances
Features	Accessible to anyone or any service with the right permissions	Deliver performance for workloads that require the lowest-latency access to data from a single EC2 instance	Has a file system interface, file system access semantics (such as strong consistency and file locking), and concurrently-accessible storage for multiple EC2 instances
Max Storage Size	Virtually unlimited	16 TiB for one volume	Unlimited system size
Max File Size	Individual Amazon S3 objects can range in size to a maximum of 5 terabytes.	Equivalent to the maximum size of your volumes	47.9 TiB for a single file
Performance (Latency)	Low, for mixed request types, and integration with CloudFront	Lowest, consistent; SSD-backed storages include the highest performance Provisioned OPS SSD and General Purpose SSD that balance price and performance.	Low, consistent; use Max I/O mode for higher performance
Performance (Throughput)	Multiple GBs per second; supports multi-part upload	Up to 2 GB per second. HDD-backed volumes include Throughput Optimized HDD for frequently accessed, throughput intensive workloads and Cold HDD for less frequently accessed data.	10+ GB per second. Bursting Throughput mode scales with the size of the file system. Provisioned Throughput mode offers higher dedicated throughput than burstring throughput.
Durability	Stored redundantly across multiple AZs; has 99.999999999% durability	Stored redundantly in a single AZ	Stored redundantly across multiple AZs
Availability	S3 Standard - 99.99% availability S3 Standard-IA - 99.9% availability S3 One Zone-IA - 99.5% availability. S3 Intelligent Tiering - 99.9%	Has 99.999% availability	99.9% SLA. Runs in multi-AZ

Tutorials Dojo	S3	EBS	EFS
Scalability	Highly scalable	Manually increase/decrease your memory size. Attach and detach additional volumes to and from your EC2 instance to scale.	EFS file systems are elastic, and automatically grow and shrink as you add and remove files.
Data Accessing	One to millions of connections over the web; S3 provides a REST web services interface	Single EC2 instance in a single AZ Amazon EBS Multi-Attach enables you to attach a single Provisioned IOPS SSD (io1 or io2) volume to up to 16 Nitro-based instances that are in the same Availability Zone.	One to thousands of EC2 instances or on-premises servers, from multiple AZs, regions, VPCs, and accounts concurrently
Access Control	Uses bucket policies and IAM user policies. Has *Block Public Access* settings to help manage public access to resources.	IAM Policies, Roles, and Security Groups	Only resources that can access endpoints in your VPC, called a *mount target*, can access your file system; POSIX-compliant user and group-level permissions
Encryption Methods	Supports SSL endpoints using the HTTPS protocol, Client-Side and Server-Side Encryption (SSE-S3, SSE-C, SSE-KMS)	Encrypts both data-at-rest and data-in-transit through EBS encryption that uses AWS KMS CMKs.	Encrypt data at rest and in transit. Data at rest encryption uses AWS KMS. Data in-transit uses TLS.
Backup and Restoration	Use versioning or cross-region replication	All EBS volume types offer durable snapshot capabilities.	EFS to EFS replication through third party tools or AWS DataSync
Pricing	Billing prices are based on the location of your bucket. Lower costs equals lower prices. You get cheaper prices the more you use S3 storage.	You pay GB-month of provisioned storage, provisioned IOPS-month, GB-month of snapshot data stored in S3	You pay for the amount of file system storage used per month. When using the Provisioned Throughput mode you pay for the throughput you provision per month.
Use Cases	Web serving and content management, media and entertainment, backups, big data analytics, data lake	Boot volumes, transactional and NoSQL databases, data warehousing & ETL	Web serving and content management, enterprise applications, media and entertainment, home directories, database backups, developer tools, container storage, big data analytics
Service endpoint	Can be accessed within and outside a VPC (via S3 bucket URL)	Accessed within one's VPC	Accessed within one's VPC

AWS Global Accelerator vs Amazon CloudFront

- CloudFront uses multiple sets of dynamically changing IP addresses while Global Accelerator will provide you a set of static IP addresses as a fixed entry point to your applications.
- CloudFront pricing is mainly based on data transfer out and HTTP requests while Global Accelerator charges a fixed hourly fee and an incremental charge over your standard Data Transfer rates, also called a Data Transfer-Premium fee (DT-Premium).
- CloudFront uses Edge Locations to cache content while Global Accelerator uses Edge Locations to find an optimal pathway to the nearest regional endpoint.
- CloudFront is designed to handle HTTP protocol meanwhile Global Accelerator is best used for both HTTP and non-HTTP protocols such as TCP and UDP.

Interface Endpoint vs Gateway Endpoint vs Gateway Load Balancer Endpoint

Interface Endpoint	Gateway Endpoint	Gateway Load Balancer Endpoint
• An elastic network interface with a private IP address that serves as an entry point for traffic destined to a supported AWS service, endpoint service, or AWS Marketplace service. • For each interface endpoint, you can choose only one subnet per Availability Zone. Endpoints are regional, which means they are only usable within the same region they are created in. • Since interface endpoints use ENIs, they also use security groups to control traffic. • Can be accessed through AWS VPN connections or AWS Direct Connect connections, through intra-region VPC peering connections from Nitro instances, and through inter-region VPC peering connections from any type of instance. • An endpoint only returns responses to traffic that is initiated from resources in your VPC. • An interface endpoint supports IPv4 TCP traffic only.	• A gateway that is a target for a specific route in your route table, used for traffic destined to a supported AWS service which is either DynamoDB or S3. • You can create multiple gateway endpoints in a single VPC, for example, to multiple services. You can also create multiple endpoints for a single service, and use different route tables to enforce different access policies from different subnets to the same service. But you cannot have multiple endpoint routes to the same service in a single route table. • You can modify the endpoint policy that's attached to your gateway endpoint, and add or remove the route tables that are used by the endpoint. • Gateway endpoints are supported within the same region only. You cannot create an endpoint between a VPC and a service in a different region. • Gateway endpoints support IPv4 traffic only. • You must enable DNS resolution in your VPC, or if	• Enables you to intercept traffic and route it to a service that you've configured using Gateway Load Balancers. • You choose the VPC and subnet that your endpoint should be created in. An endpoint network interface is assigned a private IP address from the IP address range of your subnet. You cannot change the subnet later. • After you create the Gateway Load Balancer endpoint, it's available to use when it's accepted by the service provider. The service provider can configure the service to accept requests automatically or manually. • Security groups and endpoint policies are not supported. • Endpoints support IPv4 traffic only. • You cannot transfer an endpoint from one VPC to another, or from one service to another.

• You can add endpoint policies to interface endpoints. The Amazon VPC endpoint policy defines which principal can perform which actions on which resources. An endpoint policy does not override or replace IAM user policies or service-specific policies. It is a separate policy for controlling access from the endpoint to the specified service. • After you create an interface endpoint, it's available to use when it's accepted by the service provider. The service provider must configure the service to accept requests automatically or manually. AWS services and AWS Marketplace services generally accept all endpoint requests automatically. • An interface endpoint (except S3 interface endpoint) has corresponding private DNS hostnames.	you're using your own DNS server, ensure that DNS requests to the required service are resolved correctly to the IP addresses maintained by AWS. • When you associate a route to your gateway endpoint, all instances in subnets associated with this route table automatically use the endpoint to access the service. • A gateway endpoint cannot be used beyond the scope of the VPC it is linked to.	

Amazon Kinesis vs Amazon SQS

Amazon Kinesis is a real-time data streaming service that can handle any amount of streaming data and process data from hundreds of thousands of sources with very low latencies. Amazon SQS is a message queueing service that decouples your applications, and although it provides high message throughput, it is not as fast as Kinesis. Consumer applications both poll data from these two services. Multiple consumers can process Kinesis stream data at the same time, while only a single consumer can process a single message from SQS.

There are four types of Kinesis streams:

1. Kinesis Data Streams
2. Kinesis Video Streams
3. Kinesis Data Firehose
4. Kinesis Data Analytics

There are two types of SQS queues:

1. Standard queue
2. FIFO queue

In Kinesis streams, data records are stored in the order they arrive in. SQS standard queue does a best effort in maintaining message ordering, while SQS FIFO queue stores messages in the order they arrive in. You need to use Kinesis libraries to interact with your Kinesis streams. For SQS, you only need to use AWS API or AWS SDK to handle your messages.

In Kinesis, data is kept in the stream for as long as the retention period is not up, and consumers can choose which chunks of data they will consume. This also means that consumers can replay messages in Kinesis Data Streams in the same exact order they arrived in. In SQS, the message after polling becomes invisible from other consumers for a set amount of time, and you need to manually delete the message from the queue for it to be completely removed.

In Kinesis Data Streams, to handle a large amount of streaming data, you must make sure that you have enough shards in your stream. In SQS, you must make sure that your producers do not go over the API throughput limit for sending messages.

Kinesis has many built in big data, analytics, & ETL features and integrations. For example, Kinesis Data Streams enables real-time processing of streaming big data. Kinesis Data Analytics lets you run SQL queries immediately on the streamed data. Kinesis Firehose immediately captures, transforms, and loads streaming data into your target consumers. SQS Standard queue provides at-least-once delivery. SQS FIFO queue provides exactly-once processing, which means that each message is delivered once and remains available until a consumer processes it and deletes it. Duplicates are not introduced into the queue.

Latency Based Routing vs Amazon CloudFront

The goal of using Route 53 latency based routing and/or Amazon CloudFront is to speed up delivery of content to your users. The difference between the two technologies depends on a few factors:

1. Your infrastructure setup
2. The content you wish to deliver
3. Your goal in using the technology

For infrastructure setup, if you are currently using multiple AWS regions to deliver content to your users around the globe, then Route 53 latency based routing makes sure that your users are redirected to the application endpoint that provides them the best latency. With CloudFront, you don't necessarily need to deploy your applications in multiple regions. Instead, you just deploy your application in a single region and configure the locations where you want CloudFront to cache and serve your content. This setup can save you huge amounts of money if you don't require using multiple AWS regions.

For the content you wish to deliver, latency based routing always delivers the latest content that your application has. This might be important for you if for example you are serving real time data. CloudFront, on the other hand, lets you cache static and dynamic content that match the caching rules you specify (e.g. matching headers). If you do not enable caching, then CloudFront does not help reduce the latency of content delivery to your global customers. There are also instances wherein you'd only want to cache specific objects, which in this case, CloudFront will be useful.

Aside from reducing the latency for content delivery to your customers, you might have other reasons why you would use latency based routing or CloudFront. For example, you can combine latency based routing with weighted routing to create a highly available global infrastructure. Or you might want to customize your content depending on the region that the content originates from. You might also want to run some analytics on your global customers and which region is accessed the most.

Perhaps you want to integrate Route 53 routing records with some endpoints health checks. For CloudFront, you might want to put some geo restriction rules. You might want to control how your cached content is served to customers. Or you might like to run Lambda@Edge to perform some edge location computing. Perhaps you are not only using CloudFront to reduce network latency, but also as an anti-DDoS solution for your web applications, since CloudFront integrates with AWS WAF. CloudFront can also let you serve custom error pages if you need to. There are many other features that you can use along with Route 53 latency based routing or CloudFront depending on your needs. There is also no rule saying that you can't use both technologies together.

Amazon EFS vs. Amazon FSx for Windows File Server vs. Amazon FSx for Lustre

Amazon EFS	Amazon FSx for Windows File Server	Amazon FSx for Lustre
• Amazon EFS is a serverless, scalable, high-performance file system in the cloud. • EFS file systems can be accessed by Amazon EC2 Linux instances, Amazon ECS, Amazon EKS, AWS Fargate, and AWS Lambda functions via a file system interface such as NFS protocol. • Amazon EFS supports file system access semantics such as strong consistency and file locking. • EFS file systems can automatically scale in storage to handle petabytes of data. With Bursting mode, the throughput available to a file system scales as a file system grows. Provisioned Throughput mode allows you to provision a constant file system throughput independent of the amount of data stored. • EFS file systems can be concurrently accessed by thousands of compute services without sacrificing performance.	• Amazon FSx for Windows File Server is a fully managed, scalable file storage that is accessible over SMB protocol. • Since it is built on Windows Server, it natively supports administrative features such as user quotas, end-user file restore, and Microsoft Active Directory integration. • FSx for WFS is accessible from Windows, Linux, and MacOS compute instances and devices. Thousands of compute instances and devices can access a file system concurrently. • FSx for WFS can connect your file system to Amazon EC2, Amazon ECS, VMware Cloud on AWS, Amazon WorkSpaces, and Amazon AppStream 2.0 instances. • Every file system comes with a default Windows file share, named “share”. • Common use cases for FSx for WFS include CRM, ERP, custom or .NET applications, home directories, data analytics, media and entertainment workflows, software build environments, and Microsoft	• Amazon FSx for Lustre is a serverless file system that runs on Lustre — an open-source, high-performance file system. • The Lustre file system is designed for applications that require fast storage. FSx for Lustre file systems can scale to hundreds of GB/s of throughput and millions of IOPS. FSx for Lustre also supports concurrent access to the same file or directory from thousands of compute instances. • Unlike EFS, storage capacity needs to be manually increased, and only every six hours can you do so. • Amazon FSx for Lustre also integrates with Amazon S3, which lets you process cloud data sets with the Lustre high-performance file system. • Common use cases for Lustre include machine learning, high-performance computing (HPC), video processing, financial modeling, genome sequencing, and electronic design automation (EDA). • FSx for Lustre can only be used by Linux-based instances. To access your file system, you first

• Common use cases for EFS file systems include big data and analytics workloads, media processing workflows, content management, web serving, and home directories. • Amazon EFS has four storage classes: Standard, Standard Infrequent Access, One Zone, and One Zone Infrequent Access • You can create lifecycle management rules to move your data from standard storage classes to infrequent access storage classes. • Every EFS file system object of Standard storage is redundantly stored across multiple AZs. • EFS offers the ability to encrypt data at rest and in transit. Data encrypted at rest using AWS KMS for encryption keys. Data encryption in transit uses TLS 1.2 • To access EFS file systems from on-premises, you must have an AWS Direct Connect or AWS VPN connection between your on-premises datacenter and your Amazon VPC.	SQL Server. • You can access FSx file systems from your on-premises environment using an AWS Direct Connect or AWS VPN connection between your on-premises datacenter and your Amazon VPC. • You can choose the storage type for your file system: SSD storage for latency-sensitive workloads or workloads requiring the highest levels of IOPS/throughput. HDD storage for throughput-focused workloads that aren't latency-sensitive. • Every FSx for WFS file system has a throughput capacity that you configure when the file system is created and that you can change at any time. • Each Windows File Server file system can store up to 64 TB of data. You can only manually increase the storage capacity. • Your file system can be deployed in multiple AZs or a single AZ only. Multi-AZ file systems provide automatic failover. • FSx for Windows File Server always encrypts your file system data and your backups at-rest using keys you manage through AWS KMS. Data-in-transit encryption uses SMB Kerberos session keys.	install the open-source Lustre client on that instance. Then you mount your file system using standard Linux commands. Lustre file systems can also be used with Amazon EKS and AWS Batch. • FSx for Lustre provides two deployment options: 1) Scratch file systems are for temporary storage and shorter-term processing of data. Data is not replicated and does not persist if a file server fails. 2) Persistent file systems are for longer-term storage and workloads. The file servers are highly available, and data is automatically replicated within the AZ that is associated with the file system. • You can choose the storage type for your file system: SSD storage for latency-sensitive workloads or workloads requiring the highest levels of IOPS/throughput. HDD storage for throughput-focused workloads that aren't latency-sensitive. • FSx for Lustre always encrypts your file system data and your backups at-rest using keys you manage through AWS KMS. FSx encrypts data-in-transit when accessed from supported EC2 instances.

Amazon RDS vs DynamoDB

Tutorials Dojo	RDS	DynamoDB
Type of database	Managed relational (SQL) database	Fully managed key-value and document (NoSQL) database
Features	Has several database instance types for different kinds of workloads and supports six database engines - Amazon Aurora, PostgreSQL, MySQL, MariaDB, Oracle Database, and SQL Server.	Delivers single-digit millisecond performance at any scale.
Storage Size	- 128 TB for Aurora engine. - 64 TB for MySQL, MariaDB, Oracle, and PostgreSQL engines. - 16 TB for SQL Server engine.	Supports tables of virtually any size.
Number of tables per unit	Depends on the database engine	256
Performance	General Purpose Storage is an SSD-backed storage option that delivers a consistent baseline of 3 IOPS per provisioned GB with the ability to burst up to 3,000 IOPS. Provisioned IOPS Storage is an SSD-backed storage option designed to deliver a consistent IOPS rate that you specify when creating a database instance, up to 40,000 IOPS per database instance. Amazon RDS provisions that IOPS rate for the lifetime of the database instance. Optimized for OLTP database workloads. Magnetic – Amazon RDS also supports magnetic storage for backward compatibility.	Single-digit millisecond read and write performance. Can handle more than 10 trillion requests per day with peaks greater than 20 million requests per second, over petabytes of storage. DynamoDB Accelerator (DAX) is an in-memory cache that can improve the read performance of your DynamoDB tables by up to 10 times—taking the time required for reads from milliseconds to microseconds, even at millions of requests per second. You specify the read and write throughput for each of your tables.
Availability and durability	Amazon RDS Multi-AZ deployments synchronously replicates your data to a standby instance in a different Availability Zone Amazon RDS will automatically replace the compute instance powering your deployment in the event of a hardware failure..	DynamoDB global tables replicate your data automatically across 3 Availability Zones of your choice of AWS Regions and automatically scale capacity to accommodate your workloads.
Backups	The automated backup feature enables point-in-time recovery for your database instance. Database snapshots are user-initiated backups of your instance stored in Amazon S3 that are kept until you explicitly delete them.	Point-in-time recovery (PITR) provides continuous backups of your DynamoDB table data, and you can restore that table to any point in time up to the second during the preceding 35 days. On-demand backup and restore allows you to create full backups of your DynamoDB tables' data for data archiving.

	RDS	DynamoDB
Scalability	The Amazon Aurora engine will automatically grow the size of your database volume. The MySQL, MariaDB, SQL Server, Oracle, and PostgreSQL engines allow you to scale on-the-fly with zero downtime. RDS also supports storage auto scaling Read replicas are available in Amazon RDS for MySQL, MariaDB, and PostgreSQL as well as Amazon Aurora.	Support tables of virtually any size with horizontal scaling. For tables using on-demand capacity mode, DynamoDB instantly accommodates your workloads as they ramp up or down to any previously reached traffic level. For tables using provisioned capacity, DynamoDB delivers automatic scaling of throughput and storage based on your previously set capacity.
Security	Isolate your database in your own virtual network. Connect to your on-premises IT infrastructure using industry-standard encrypted IPsec VPNs. You can configure firewall settings and control network access to your database instances. Integrates with IAM.	Integrates with IAM.
Encryption	Encrypt your databases using keys you manage through AWS KMS. With encryption enabled, data stored at rest is encrypted, as are its automated backups, read replicas, and snapshots. Supports Transparent Data Encryption in SQL Server and Oracle. Supports the use of SSL to secure data in transit.	DynamoDB encrypts data at rest by default using encryption keys stored in AWS KMS.
Maintenance	Amazon RDS will update databases with the latest patches. You can exert optional control over when and if your database instance is patched.	No maintenance since DynamoDB is serverless.
Pricing	A monthly charge for each database instance that you launch. Option to reserve a DB instance for a one or three year term and receive discounts in pricing, compared to On-Demand instance pricing.	Charges for reading, writing, and storing data in your DynamoDB tables, along with any optional features you choose to enable. There are specific billing options for each of DynamoDB's capacity modes.
Use Cases	Traditional applications, ERP, CRM, and e-commerce.	Internet-scale applications, real-time bidding, shopping carts, and customer preferences, content management, personalization, and mobile applications. Tutorials Dojo

Redis (cluster mode enabled vs disabled) vs Memcached

	Redis (cluster mode enabled)	Redis (cluster mode disabled)	Memcached
Data Types	string, sets, sorted sets, lists, hashes, bitmaps, hyperloglog, geospatial indexes	string, sets, sorted sets, lists, hashes, bitmaps, hyperloglog, geospatial indexes	string, objects (like databases)
Data Partitioning (distribute your data among multiple nodes)	Supported	Unsupported	Supported
Modifiable cluster	Only versions 3.2.10 and later	Yes	Yes
Online resharding	Only versions 3.2.10 and later	No	No
Encryption	3.2.6, 4.0.10 and later	3.2.6, 4.0.10 and later	Unsupported
Sub-millisecond latency	Yes	Yes	Yes
FedRAMP, PCI DSS and HIPAA compliant	3.2.6, 4.0.10 and later	3.2.6, 4.0.10 and later	No
Multi-threaded (make use of multiple processing cores	No	No	Yes
Node type upgrading	No	Yes	No
Engine upgrading	Yes		
Cluster replication (create multiple copies of a primary cluster)	Supported	Supported	Unsupported
Multi-AZ for automatic failover	Required	Optional	Unsupported
Transactions (execute a group of commands as an isolated and atomic operation)	Supported	Supported	Unsupported
Pub/Sub capability	Yes	Yes	No
Backup and restore (keep your data on disk with a point in time snapshot)	Supported	Supported	Unsupported
Lua Scripting (execute transactional Lua scripts)	Supported	Supported	Unsupported
Use Case	• You need to partition your data across two to 250 or 500 nodes if the Redis engine version is 5.0.6 or higher. (clustered mode only). • You need geospatial indexing (clustered mode or non-clustered mode). • You don't need to support multiple databases. • Plus features of non-clustered mode	• You need complex data types, such as strings, hashes, lists, sets, sorted sets, and bitmaps. • You need to sort or rank in-memory datasets. • You need persistence of your key store. • You need to replicate your data from the primary to one or more read replicas for read intensive applications. • You need automatic failover if your primary node fails. • You need pub/sub capabilities. • You need backup and restore capabilities. • You need to support multiple databases.	• You need the simplest model possible. • You need to run large nodes with multiple cores or threads. • You need the ability to scale out and in, adding and removing nodes as demand on your system increases and decreases. • You need to cache objects, such as a database. • Needs Auto Discovery to simplify the way an application connects to a cluster.

AWS WAF vs AWS Shield Basic vs AWS Shield Advanced

	AWS WAF	AWS Shield Basic	AWS Shield Advanced
Security Features	AWS WAF can monitor web requests transmitted over HTTP or HTTPS. AWS WAF helps protect web applications from attacks by allowing you to configure rules that allow, block, rate-limit, or monitor web requests based on conditions that you define. These conditions include IP addresses, HTTP headers, HTTP body, URI strings, SQL injection, and cross-site scripting. Rate-based rules also help you from web-layer DDoS attacks, brute force login attempts, and bad bots.	AWS Shield provides protection against common and most frequently occurring OSI layer 3 and 4 attacks like SYN/UDP floods, reflection attacks, and DDoS attacks for applications running on AWS. AWS Shield's detection and mitigations work with IPv4 and IPv6 traffic.	AWS Shield Advanced provides additional protections against more sophisticated and larger attacks for your applications running in AWS. Provides near real-time notifications of suspected DDoS incidents. Also employs advanced attack mitigation and routing techniques for automatically mitigating attacks. Having a Business or Enterprise support plan lets you engage with the AWS DDoS Response Team.
Integration	AWS WAF is tightly integrated with Amazon CloudFront, Application Load Balancer, Amazon API Gateway, and AWS AppSync	Most of the AWS resources are automatically integrated and protected from common and frequently occurring network and transport layer DDoS attacks.	Can be integrated with Amazon EC2, Elastic Load Balancing, Amazon CloudFront, AWS Global Accelerator, and Route 53 for a higher level of DDoS attack mitigation.
Pricing	You are charged based on the number of web access control lists (web ACLs) that you create, the number of rules that you add per web ACL, and the number of web requests that you receive.	AWS Shield Standard is automatically enabled to all AWS customers at no cost.	You pay a monthly fee of $3,000 per month per organization. In addition, you also pay for AWS Shield Advanced Data Transfer usage fees for AWS resources enabled for advanced protection.

AWS KMS vs AWS CloudHSM

Many AWS services provide native encryption support for data in-transit and data at rest. Knowing what you need to protect and how to protect it will let you determine which AWS encryption service you should use.

When to use KMS:
When you encrypt data, you need to protect your encryption key. To further secure your data, you should also encrypt your encryption key. The final encryption key, or master key, is the most crucial segment in your encryption process, since it can decipher all the data keys that you used to encrypt your data. AWS Key Management Service, or AWS KMS, lets you create, store, and manage customer master keys (CMKs) securely. Your CMKs never leave AWS KMS unencrypted, and CMKs can only be used through AWS KMS to decrypt objects. AWS KMS has key policies that let you specify who has access to your CMKs and what they can do with it.

A CMK can be used to encrypt small amounts of data (up to 4096 bytes). If you need to encrypt larger content, use the CMK to generate, encrypt, and decrypt the data keys that are then used to encrypt your data, in place of the CMK. Data keys can encrypt data of any size and format, including streamed data. However, do keep in mind that AWS KMS does not store or manage data keys, and you cannot use KMS to encrypt or decrypt with data keys. AWS KMS only manages the CMKs.

With AWS KMS, you can create symmetric and asymmetric keys and data key pairs, as well as import your own symmetric key material. Keys generated by AWS KMS can be scheduled to automatically rotate on an annual basis. When creating a CMK, you must specify whether the key will be used for encryption/decryption or sign/verify operations.

When to use CloudHSM:
AWS KMS CMKs are stored in FIPS-validated hardware service modules (HSMs) that KMS manages (shared tenancy among AWS customers). A hardware security module (HSM) is a specialized security device that generates and stores cryptographic keys. If you prefer to manage your own HSMs to store your keys in KMS, or you require FIPS 140-2 type 3, you may use AWS CloudHSM. Once you've created your own HSM, you can have the HSM generate and store your encryption keys, and create users and set their permissions for your HSM. For security and isolation from other AWS customers, CloudHSM must be provisioned inside an Amazon VPC.

Additionally, you can offload SSL/TLS cryptographic processing for HTTPS sessions to your CloudHSM module, which cannot be done on AWS KMS. Offloading the process lessens the computational burden on your servers. Some other uses for CloudHSM include securing the private keys for an issuing Certificate Authority (CA), and enabling Transparent Data Encryption for Oracle databases.

RDS Read Replica vs RDS Multi-AZ vs Vertical Scaling vs Elasticache

There are many ways to increase the performance, availability and scalability of an Amazon RDS instance. However, some implementations overlap each other in use cases and may seem redundant. Choosing the correct implementation for a certain situation may not necessarily be as obvious as it seems, but there are definitely some nuances that you can make note of.

Amazon RDS Read Replicas provide enhanced performance and durability for your DB instances. They provide horizontal scaling for read-heavy databases. Read replicas can also be manually promoted to master DB instances if the master instance starts failing. Data between the master instance and read replicas are replicated asynchronously. Remember that read replicas can only read-only connections; write connections will not go through. Read replicas provide scaling on read capacity while reducing the burden on your master instance.

Amazon RDS Multi-AZ is a solution that increases the availability of your RDS master instance. In the event of an outage, RDS will do an automatic failover to your backup DB instance in the other AZ. RDS Aurora uses asynchronous data replication to keep the master and standby instances updated. Non-Aurora engines use synchronous replication. With Multi-AZ enabled, your database will always span at least two Availability Zones within a single region. Your standby replica cannot handle read and write queries.

When you need more resources for your master DB instance, you can always **scale up the instance size** to gain more CPU, memory, network throughput, and dedicated EBS bandwidth. You usually scale up your DB instance if you need more read and write capacity, and that read replicas are unnecessary for your needs. Oftentimes, the initial size you choose for your RDS instance is incorrect or inadequate. An Amazon RDS performance best practice is to allocate enough RAM so that your working set resides almost completely in memory. The working set is the data and indexes that are frequently in use on your instance. There is minimal downtime when you are scaling up on a Multi-AZ environment because the standby database gets upgraded first, then a failover will occur to the newly sized database. A Single-AZ instance will be unavailable during the scale operation.

Adding an Elasticache in front of your RDS instance increases the read performance for your application since the data resides in memory. If you have items that are frequently accessed, you can cache them in Elasticache and reduce the burden on your DB instance. Elasticache is not a good option if your database is more write-heavy than read-heavy, unless you really need that extra bump in read performance. Comparing a cache to a read replica, a cache is better suited if the application queries the same items over and over again or the results are static. If you have been previously using Redis or Memcached already, Elasticache also allows you to lift and shift your solution over. If the items that are being read vary way too much, a read replica might be a better choice instead.

Scaling DynamoDB RCU vs DynamoDB Accelerator (DAX) vs Secondary Indexes vs ElastiCache

Similar to Amazon RDS, there are also multiple options available to DynamoDB when you want to increase the performance of your tables. Each option has its own use case, pros, and cons that you should consider all together when choosing for the best solution.

Scaling DynamoDB Read Capacity can be achieved in two ways, depending on your capacity mode. For On-Demand Mode, you do not need to perform capacity planning. DynamoDB automatically scales your read and write capacity to meet demands. However, if your workloads spike very often, On-Demand mode might become very costly for you if you do not manage your capacity limits properly. For Provisioned Mode, you specify the number of reads and writes per second that you require for your application to meet all the time. You can use auto scaling to adjust your table's provisioned capacity automatically in response to traffic changes. This helps you manage your usage to stay at or below a defined request rate in order to make cost more predictable. DynamoDB auto scaling will actively manage the throughput capacity for your tables and global secondary indexes. You just define an upper and lower limit for the read and write capacity units. You also define a target utilization percentage within that range. You should scale your read capacity units when your DynamoDB tables and indexes experience high read operations and the items being read are not suited for cache.

DynamoDB DAX is a fully managed, in-memory cache for DynamoDB. You use DynamoDB DAX if you wish to achieve microsecond response time. With DynamoDB DAX, there is no need to change your code. You can continue using DynamoDB SDKs and APIs as is. If you have very strict performance requirements, or if you have common table items that are being queried repeatedly, DynamoDB DAX is the solution for you. You also avoid having to overprovision read capacity for your DynamoDB. You only pay for the capacity you provision in DynamoDB DAX. Since DAX is a cache, it is possible that your applications might query stale data. If your applications require strongly consistent reads or have write-intensive workloads, then you should not use DAX.

Secondary Indexes can speed up read operations by helping you avoid scanning your whole table when querying non-primary key attributes. You can retrieve data from the index using a *Query* operation, in much the same way as you use *Query* with a table. You can also *Scan* an index, in much the same way as you would *Scan* a table. A table can have multiple secondary indexes, allowing you to have multiple query patterns. Every secondary index is also 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. Do note that the read performance of your secondary indexes are still bound by the read capacity units of your DynamoDB table. Also, rather than boosting the performance of your table, indexes are more like optimizing your data structure to help you query the results you need faster.

For caching requirements, you would usually go with DynamoDB Accelerator, since it does not require any code modification if you've been using DynamoDB already. You'll only prefer Amazon Elasticache as your caching

service if you're specifically required to use Redis or Memcached, or if you have a feature in Elasticache that is not currently supported in DAX. Some of the unsupported features for example are:

- DAX does not support Transport Layer Security (TLS).
- DAX only supports applications written in Go, Java, Node.js, Python, and .NET.
- DAX may not be available in your desired region.
- You want to manage the cache invalidation logic.

FINAL REMARKS AND TIPS

That's a wrap! Thank you once again for choosing our Study Guide and Cheat Sheets for the AWS Certified Solutions Architect Associate (SAA-C03) exam. The Tutorials Dojo team spent considerable time and effort to produce this content to help you pass the AWS exam.

We also recommend that before you take the actual SAA-C03 exam, allocate some time to check your readiness first by taking our AWS practice test course in the Tutorials Dojo Portal. You can also try the free sampler version of our full practice test course here. This will help you identify the topics that you need to improve on and help reinforce the concepts that you need to fully understand in order to pass the SAA-C03 exam. It also has different training modes that you can choose from such as Timed mode, Review mode, Section-Based tests, Topic-based tests, and Final test plus bonus flashcards. In addition, you can read the technical discussions in our forums or post your queries if you have one. If you have any issues, concerns or constructive feedback on our eBook, feel free to contact us at support@tutorialsdojo.com.

On behalf of the Tutorials Dojo team, I wish you all the best in your upcoming AWS Certified Solutions Architect - Associate exam. May it help advance your career, as well as increase your earning potential.

With the right strategy, hard work, and unrelenting persistence, you can definitely make your dreams a reality! You can make it!

Sincerely,
Jon Bonso and the Tutorials Dojo Team

ABOUT THE AUTHOR

Jon Bonso (10x AWS Certified)

Born and raised in the Philippines, Jon is the Co-Founder of **Tutorials Dojo**. Now based in Sydney, Australia, he has over a decade of diversified experience in Banking, Financial Services, and Telecommunications. He's 10x AWS Certified, an AWS Community Builder, and has worked with various cloud services such as Google Cloud, and Microsoft Azure. Jon is passionate about what he does and dedicates a lot of time creating educational courses. He has given IT seminars to different universities in the Philippines for free and has launched educational websites using his own money and without any external funding.